2021

A Novel User Oriented Network Forensic Analysis Tool

Dany Joy

Recommended Citation

Joy, D. (2021) *A Novel User Oriented Network Forensic Analysis Tool.* Thesis. University of Plymouth. Retrieved from https://pearl.plymouth.ac.uk/secam-theses/316

PHD

A Novel User Oriented Network Forensic Analysis Tool

Joy, Dany

Award date:
2021

Awarding institution:
University of Plymouth

Copyright Statement

UNIVERSITY OF PLYMOUTH

A Novel User Oriented Network Forensic Analysis Tool

By

Dany Joy

A thesis submitted to the University of Plymouth in partial fulfilment for the degree of

DOCTOR OF PHILOSOPHY

School of Engineering, Computing and Mathematics

June 2021

Acknowledgement

First and foremost, I would like to express my heartfelt gratitude to my Director of Studies, Professor Nathan L Clarke, for his valuable guidance and generous support throughout my PhD journey. His valuable insight has always helped in improving the quality of my research. I will never be able to thank him enough for not giving up on me, in spite of me letting him down as a research student.

I thank my second supervisor, Professor Steven Furnell, who, despite his busy schedule, found time to offer guidance and suggestions. I would also like to thank Dr Fudong Li for joining the supervisory team as my third supervisor and for overseeing my research tasks.

Last, but not least, I would like to thank my parents and my sister for the unconditional love, support and encouragement that they have given me throughout my life so far.

Author’s Declaration

At no time during the registration for the degree of *Doctor of Philosophy* has the author been registered for any other University award without prior agreement of the Doctoral College Quality Sub-Committee.

Work submitted for this research degree at the University of Plymouth has not formed part of any other degree either at the University of Plymouth or at another establishment.

Relevant seminars and conferences were regularly attended at which work was often presented and papers were published during the course of this research project. Other research skills development courses were also attended.

Word count of the main body of the thesis: 71,799

Signed ……..…Dany Joy…..………

Date ….……..24/06/2021………...

Abstract

A Novel User Oriented Network Forensic Analysis Tool

Dany Joy

In the event of a cybercrime, it is necessary to examine the suspect's digital device(s) in a forensic fashion so that the culprit can be presented in court along with the extracted evidence(s). But, factors such as existence and availability of anti-forensic tools/techniques and increasing replacement of hard disk drives with solid state disks have the ability to eradicate critical evidences and/or ruin their integrity. Therefore, having an alternative source of evidence with a lesser chance of being tampered with can be beneficial for the investigation. The organisational network traffic can fit into this role as it is an independent source of evidence and will contain a copy of all online user activities. Limitations of prevailing network traffic analysis techniques – packet based and flow based – are reflected as certain challenges in the investigation. The enormous volume and increasing encrypted nature of traffic, the dynamic nature of IP addresses of users' devices, and the difficulty in extracting meaningful information from raw traffic are among those challenges. Furthermore, current network forensic tools, unlike the sophisticated computer forensic tools, are limited in their capability to exhibit functionalities such as collaborative working, visualisation, reporting and extracting meaningful user-level information. These factors increase the complexity of the analysis, and the time and effort required from the investigator.

The research goal was set to design a system that can assist in the investigation by minimising the effects of the aforementioned challenges, thereby reducing the cognitive load on the investigator, which, the researcher thinks, can take the investigator one step closer to the culprit. The novelty of this system comes from a newly proposed interaction based analysis approach, which will extract online user activities from raw network metadata. Practicality of the novel interaction-based approach was tested by designing an experimental methodology, which involved an initial phase of the researcher looking to identify unique signatures for activities performed on popular Internet applications (BBC, Dropbox, Facebook, Hotmail, Google Docs, Google Search, Skype, Twitter, Wikipedia, and YouTube) from the researcher's own network metadata. With signatures obtained, the project moved towards the second phase of the experiment in which a much larger dataset (network traffic collected from 27 users for over 2 months) was analysed. Results showed that it is possible to

extract unique signature of online user activities from raw network metadata. However, due to the complexities of the applications, signatures were not found for some activities. The interaction-based approach was able to reduce the data volume by eliminating the noise (machine to machine communication packets) and to find a way around the encryption issue by using only the network metadata.

A set of system requirements were generated, based on which a web based, client-server architecture for the proposed system (i.e. the User-Oriented Network Forensic Analysis Tool) was designed. The system functions in a case management premise while minimising the challenges that were identified earlier. The system architecture led to the development of a functional prototype. An evaluation of the system by academic experts from the field acted as a feedback mechanism. While the evaluators were satisfied with the system's capability to assist in the investigation and meet the requirements, drawbacks such as inability to analyse real-time traffic and meeting the HCI standards were pointed out. The future work of the project will involve automated signature extraction, real-time processing and facilitation of integrated visualisation.

Contents

List of Figures

List of Tables

1 Cybercrime

1.1 Introduction

The Internet plays a crucial role in human lives by assisting individuals to accomplish daily activities such as communication, work, education, social interaction, banking and shopping. According to the 2017 Cybercrime Report published by Cybersecurity Ventures, there were 3.8 billion Internet users in the world in 2017 and the number of users is expected to reach 6 billion by 2022 and 7.5 billion by 2030 (Cybersecurity Ventures, 2017). Via their personal computers, smartphones and tablets, users use numerous websites and apps. For instance, there are more than 1.7 billion websites and 3.9 million apps (Google Play and Apple App Store combined) available as of the first quarter of 2019 (Dadax, 2019; Statista, 2019). These websites and apps offer different services such as news, email, information search, online streaming, social media, chat and VoIP to the users, and the frequency with which people use those services is big. For instance, as of April 2018, there are 2,686,911 E-mails sent, 66,673 Google searches carried out, 73,619 YouTube videos viewed, 8,091 Tweets sent and 3,112 Skype calls made on average in each second (Dadax, 2018). Such large numbers of users, devices, websites/apps and the services they offer, and the frequency with which users interact with the services will result in enormous volumes of data that traverse the Internet. In 2018, the volume of global IP traffic was at a rate of 156 Exabyte per month and it is estimated to become 396 Exabyte per month by 2022 (Cisco, 2018). This not only shows the amount of Internet-based interactions that the users are engaged in but also provides an indication of the wealth of information that is being shared online. These information can be associated with an individual or an organisation. Meanwhile, a different group of people target these individuals and organisations, devices that are connected to the Internet and/or the information via crimes committed in the cyberspace. These crimes are categorised as ‘cybercrimes’.

‘Cybercrime’ has become an inevitable term today, and is described as an offense that targets a computer or a digital device that compromises the confidentiality, integrity or availability of data, or an offense that targets a person or a group of people by using a computer or a digital device (Council of Europe, 2001). Computer crimes are divided into three categories: crimes where the computer is – a target (where the criminal’s motive is to compromise the data within the victim’s computer), or a tool (where computer is used as a tool to commit conventional crimes), or incidental (where a computer is not required to commit the crime, but is somehow connected to the criminal activity) (Goodman, 1997). Cybercrimes are global crimes and they differ from traditional crimes. “Computer-related crimes are committed across space and do not stop at the conventional state-borders; they can be perpetrated anywhere and against any computer user in the world” (Commission of the European Communities, 2001).

Cybercrimes can have a major scale of impact whether it is the number of people affected by the crime, the financial or non-financial damages caused or the geographical area where the targeted victims are located (United Nations, 1994). For instance, the Internet Crime Complaint Centre receives approximately 300,000 complaints per year on average, and the crimes reported in 2018 have resulted in a total loss of $2.7 billion for the victims (Internet Crime Complaint Centre, 2018). Perhaps, the impact of cybercrimes is seen more on the organisations that fall victim to the attacks than on individuals. It is estimated that by the year 2021, cybercrime will cost the world $6 trillion per annum (Cybersecurity Ventures, 2017).

Organisations are hugely affected by data breaches and security incidents resulting from cyber-attacks. Based on the Verizon’s Data Breach Investigation Report, 2016 saw financial and healthcare industries as the top two victims of data breaches that resulted in confirmed disclosure of data to an unauthorised party. Financial and Insurance sector was at the top of the list that faced a total of 471 data breaches followed by the healthcare industry that accounted for a total of 296 data breaches; covering a combined proportion of 39% of the total figure. In comparison, in terms of the number of security incidents, the public sector was in the front with 21,239 reported incidents (Verizon’s DBIR, 2017). With the intention of providing valuable lessons in improving cyber and privacy risk management, a survey named Global State of Information Security Survey (GSISS) was conducted by PricewaterhouseCoopers (PwC). Responses from 9,500 business and technology executives in 122 countries and more than 75 industries were collected during the survey. The results revealed that cyber-attacks had compromised 35% of the customer records, 30% of the employee records and 29% of the internal records in 2017 (PwC GSISS, 2018). In terms of expenditure, the cybersecurity industry is expected to grow to $93 billion in 2018 from $86.4 billion in 2017. Also, it is predicted that organisations will experience a ransomware attack in every 14 seconds by 2019 (Cybersecurity Ventures, 2017). The current and projected statistics show how severe the damage caused by cyber-attacks can be and how the situation could possibly rupture the economic stability of the organisations.

Cybercrimes that the organisational systems and information are susceptible to can be external attacks as well as insider threats. Different surveys that are carried out on a yearly basis were able to show what proportion of the data breaches within the organisations have resulted from these two types of threats. One survey showed that 75% of the total data breaches that occurred in 2017 were perpetrated by outsiders whereas 25% of the breaches involved internal actors (Verizon's DBIR, 2017). According to the GSISS including 560 UK respondents, employees are responsible for 27% of all cyber security incidents that occurred in the United Kingdom in 2017 in comparison with the 20% of all incidents occurred in 2016. The same survey found that, on a global scale, 30% of the current employees are likely to be the source of cyber security incidents (PwC GSISS, 2018). Based on the statistics, it is evident that majority of the data breaches in organisations are caused by outside attackers.

Even though not as big in number as external attacks, surveys suggest that insider threats, if left unresolved, can be equally detrimental. A threat caused by an insider occurs due to the intentional or unintentional misuse of organisation's resources such as their systems or their critical information. Compared to the preparations taken by the organisations in dealing with external attacks, the counter measures taken against insider threats is undervalued. The employees are physically situated within the perimeters of the organisation and are already inside the defence mechanisms put in place by the organisations against attacks. Also, depending on the respective designations, employees will possess legitimate access to the organisational resources. Such factors can make insider and privilege misuse hard to detect in comparison with the external attacks making it more dangerous. Insider misuse incidents can occur either due to premeditated actions carried out by insiders with the intention of sabotaging the organisation or due to accidental misuse of employee designation resulting in the leakage of organisational data. Examples for accidental misuse are clicking on an unsuspicious link received via e-mail or downloading an attachment received from a colleague, and an example for intentional insider and privilege misuse is legitimately accessing the organisation's data server and retrieving confidential information with the intention of leaking it to the outside world. During the survey, it was found out that most of the insider misuse incidents take months or years to be discovered and the impact of such insider & privilege misuse incidents would be severely damaging (Verizon's DBIR, 2017). In 2017, a total of 7,743 incidents were reported that involved insider & privilege misuse among which 277 incidents confirmed data disclosure. Public, healthcare and finance industries were the top industries that were affected due to the insider misuse (Verizon's DBIR, 2017).

Given the amount and complexity of the cybercrime attacks that are caused by both inside and outside entities and the severity of the damage caused by such attacks, protecting against cyberattacks can be a challenging task. Organisations take security measures and put policies in place in order to fight cybercrimes. Network monitoring/analysis tools, firewalls, intrusion detection/prevention systems, and antivirus software are among the components that are utilised by the organisations for detecting and preventing cyberattacks.

These precautions may help in detecting threats but may not be sufficient to find the culprit(s) that caused the incident. An investigation that will analyse the suspects' digital devices in a forensic and timely fashion will be required to extract the evidence(s) and to reveal the individual(s) behind the incident. Even though established computer forensic tools exist that can forensically examine the devices, increasing usage of solid state disks and existence of anti-forensic tools/techniques can challenge or prevent the investigator from achieving the goal. The investigator may be left with partial or no evidence and with an incomplete picture of the crime. Therefore, it is going to be beneficial to consider alternative sources of evidence such as the organisational network traffic. The network traffic is useful candidate because it will contain a copy of all online user activities, which the culprit will find difficult to tamper with. Extraction of these online user activities can provide information that may help the investigator to have better understanding of the situation.

Existing network forensic tools that are developed to assist the investigators are designed to analyse the traffic based on either deep packet inspection or flow based analysis technique. Limitations of the aforementioned network analysis techniques such as the massive volume and increasing encrypted nature of the network traffic, and the unreliability in devices having a static IP address can increase the complexity of the investigation. Furthermore, even though existing tools can provide information about the online services that were used, they do not identify or reveal the specific activities that the users were engaged in while interacting with those services. Also, unlike the sophisticated computer forensics tools, existing network forensics analysis tools are limited in exhibiting features such as case management, collaborative working, visualisation, and reporting. These factors will demand more time and effort, thereby putting increased pressure on the investigator. Therefore, a system that can minimise the aforementioned limitations, extract meaningful information about the users, and incorporate the listed functionalities of sophisticated computer forensic tools can be of assistance in the investigation process, can reduce the cognitive load on the investigator and take the investigator one step closer to identifying the culprit.

1.2 Aim and Objectives

The research aims to develop a novel web-based user-oriented network forensic analysis tool (UO-NFAT) that can reduce the cognitive pressure exerted on the investigator by extracting online user interactions from the analysis of low-level network traffic metadata. The proposed tool also provides the investigator with the ability to perform case management, to visualise the identified user interactions rather than trawling through low-level packet data and to generate a case report containing the extracted valuable information.

The objectives of the research are:

- To review the domains of digital forensics and analyse the current state of the art within the field of network forensics in order to perform a requirement analysis and extract the system requirements.
- To investigate a novel approach for analysing and visualising network traffic in order to reduce the cognitive burden placed on investigators.
- To design a novel architecture for the proposed user oriented network forensic analysis tool that is built on industry accepted case management principles.
- To implement a functional prototype that will incorporate the core features and functionalities exhibited by the proposed UO-NFAT.
- To evaluate the proposed UO-FAT.

1.3 Thesis Overview

Chapter 2 begins with presenting some important survey findings regarding cybercrime trends followed by presenting the different branches of digital forensics – computer forensics, embedded forensics and cloud forensics – to evidence the depth and breadth of the field itself. The chapter also presents an isolated study of computer forensic capabilities and different anti-forensic techniques used by the attackers.

Chapter 3 analyses different network forensic methodologies and different network forensic tools that are currently being used where the tools are compared based on their capabilities and also comparisons are made with the established computer forensic tools. The chapter also discusses different methodologies supporting intelligent analysis of network data. A variety of research challenges are identified in this chapter followed by generating a requirement list for a novel network forensic analysis tool.

Chapter 4 begins by analysing the existing approaches – packet based and flow based approach – to justify the unsuitability of the aforementioned approaches. The chapter proposes a novel approach (i.e. the interaction based approach) for the purpose of analysing the network traffic metadata to extract the signatures for identifying the online user activities. Following this, the chapter discusses an experimental methodology for manually analysing the captured network traffic to identify and extract signatures of the online user interactions for the selected Internet based applications and then discuss the obtained results.

Chapter 5 begins by recalling the system requirements that was devised for the proposed user oriented network forensic analysis tool (UO-NFAT). The following section of the chapter presents the novel UO-NFAT architecture followed by the detailed discussion of the structure and behaviour of each and every component and the relationship between those components.

Chapter 6 discusses the functional prototype that was implemented based on the designed UO-NFAT architecture. The chapter takes the reader through different user interface windows of the prototype in order to get familiarised with the tool and to understand the way around the tool while conducting an investigation utilising the fully developed version of the tool. The next section of the chapter discusses a set of three use cases in order to demonstrate how the tool can be utilised to assist in the investigation of different types of cases.

Chapter 7 begins by discussing a methodology for evaluating the research work done along with the phases involved (i.e. the questionnaire design phase, the video podcast production phase, the selection phase and the interview phase). The next section of the chapter introduces the academic experts that participated in the expert evaluation process followed by the feedback provided by the evaluators during the interview sessions. The chapter ends by a detailed discussion of the feedback obtained from the expert evaluators.

Finally, the Chapter 8 presents the achievements of the research and the limitations followed by a brief discussion of the future research directions for network forensics.

2 Digital Forensics

2.1 Introduction

Organisations worldwide became part of the digital revolution by embracing utilisation of digital devices that store and transfer information in digital formats, both locally and over networks. This, eventually, triggered the attention of criminals to enter the cyberspace and resulted in those cybercriminals establishing ways to exploit the devices as well as ways to tamper with or steal the digitised information in order to commit various crimes (Information Security and Forensics Society, 2004). In order to uncover such criminals and to serve justice, it is essential to have a substantial amount of evidence to back up the criminal or civil charges and to have an appropriate method for handling the evidence properly. But extracting digital evidence without losing its value and integrity is critical and requires a systematic procedure for acquiring and preserving the data, followed by the analysis of that evidence (Information Security and Forensics Society, 2004), which results in the need for digital forensics.

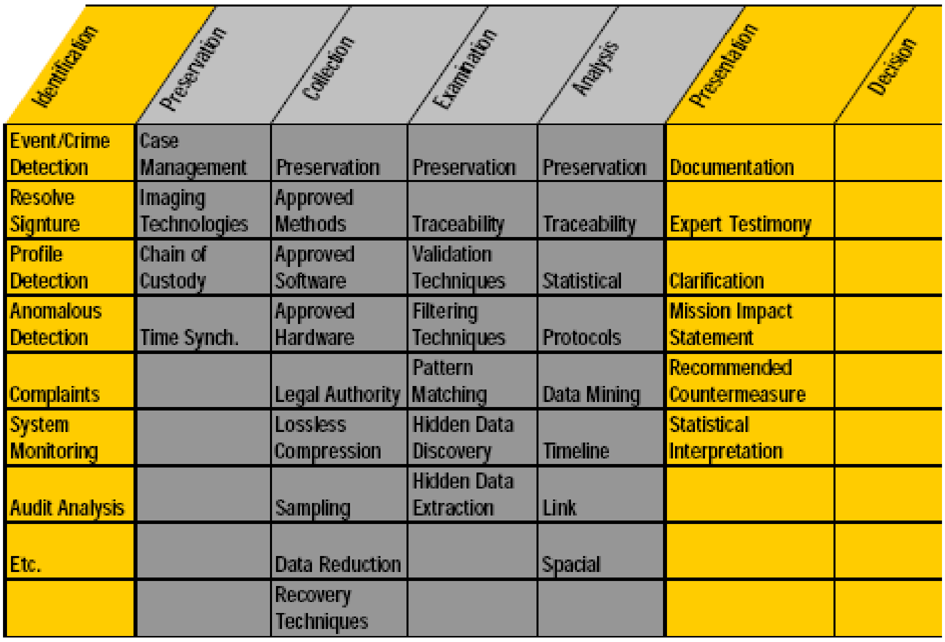
Digital forensics emerged in the 1980s as techniques for recovering data, but was limited in its growth by factors such as the variety of hardware, software and applications, dependency on time-sharing and centralised computing systems and lack of processes, tools and training for recovery that were formal in nature. Due to these factors, data recovery was a simpler and easier task to accomplish compared to the present-day. However, the aforementioned limiting factors evolved over time causing the digital forensics domain to grow, which led to a ‘golden age’ of digital forensics from 1999 to 2007 (Garfinkel, 2010). But, in recent times, due to factors such as the diversity of file formats, usage of different encryption/decryption techniques, entry of new types and increase in the size of storage devices, increase in the volume of network traffic and increase in application platforms, recovery and analysis of any form of intrusion or illegal activities has become significantly more challenging. Now, digital forensics focuses not only on data recovery but has become a useful tool in any form of cybercrime investigation.

Section 2.2 of this Chapter begins with an overview of the digital forensics domain followed by brief reviews of different branches of the domain – computer, mobile, network, and cloud forensics in its subsections. Section 2.3 includes a review of the popular computer forensic tools in use in order to show the sophisticated nature and maturity of the computer forensic domain. The section also identifies the challenges and limitations associated with this domain.

Section 2.4 dives into a detail discussion of anti-forensics – an area that creates a major challenge for the computer forensic investigation. Different techniques utilised by the culprits to sabotage the investigation are reviewed in this Section. Section 2.5 concludes the Chapter by confirming that network data is an alternative source of evidence in conducting cybercrime investigation.

2.2 Digital Forensics

An event to be mentioned that played a vital role in the entire history of digital forensics is the Digital Forensic Research Workshop (DFRWS) that took place in 2001 with the intention of developing a better foundation for the field. *"Digital Forensic Research Workshop (DFRWS) was one of the first large scale consortiums lead by academia rather than law enforcement with the intention of providing directions to the scientific community dealing with the challenges of digital forensics"* (Reith *et al.*, 2002). As a result, a generalised process model for conducting digital forensic investigation was emerged in this consortium. The DFRWS process model contains seven phases – identification, preservation, collection, examination, analysis, presentation and decision (Palmer, 2001). Figure 2.1 is the diagrammatic representation of this process model that was proposed in DFRWS:



Identification	Preservation	Collection	Examination	Analysis	Presentation	Decision
Event/Crime Detection	Case Management	Preservation	Preservation	Preservation	Documentation	
Resolve Signture	Imaging Technologies	Approved Methods	Traceability	Traceability	Expert Testimony	
Profile Detection	Chain of Custody	Approved Software	Validation Techniques	Statistical	Clarification	
Anomalous Detection	Time Synch.	Approved Hardware	Filtering Techniques	Protocols	Mission Impact Statement	
Complaints		Legal Authority	Pattern Matching	Data Mining	Recommended Countermeasure	
System Monitoring		Lossless Compression	Hidden Data Discovery	Timeline	Statistical Interpretation	
Audit Analysis		Sampling	Hidden Data Extraction	Link		
Etc.		Data Reduction		Spacial		
		Recovery Techniques				

Figure 2.1 DFRWS investigation process model (Palmer, 2001)

The topmost raw of the table shows the different phases of the process which follows a linear direction of execution and columns below each of the phases contains the techniques or methods used in each of those phases (Palmer, 2001).

- Identification phase is in charge of detecting suspicious events or crimes utilising profile detection, anomalous detection, system monitoring techniques etc.
- Preservation phase contains assembling the data in order to perform the digital forensic investigation using appropriate imaging technologies, following case management guidelines and ensuring the custody of devices from which the data are being preserved.
- Collection phase is the extraction of only the relevant data from the data gathered in the preservation phase using approved hardware and software tools and lossless compression, sampling, data reduction and recovery techniques.
- Examination phase uses filtering techniques, pattern matching techniques, validation techniques and hidden data discovery and extraction techniques in order to find any traces of crime.
- Any traces of crime obtained from the examination phase are analysed in the analysis phase using different methods such as statistical, protocol, timeline analysis and data mining.
- Presentation phase consists of documenting and presenting the results (in a legally understandable manner) obtained in the analysis phase using statistical interpretation methods, clarification methods and expert testimonies and also recommending any countermeasures. Appropriate decisions are made in the decision phase based on the facts presented.

DFRWS was a milestone in the path of digital forensics during which everyone accepted the need of having a standardised digital forensic investigation procedure and resulted in the creation of the aforementioned well accepted process model which became a foundation upon which various investigation process models were developed for different branches of digital forensics. During this golden era, digital forensics branched itself into different fields such as computer forensics, mobile phone forensics, network forensics and cloud forensics each focusing on investigation in their respective fields. Summary definitions for these fields are given below:

- Computer forensics is defined as "the discipline that combines elements of law and computer science to collect and analyse data from computer systems, networks, wireless communications and storage devices in a way that is admissible as evidence in a court of law" (US-CERT, 2008). Because the term 'computer forensics' is sometimes interchangeable with 'digital forensics', this definition can be taken to summarise computer forensics when the techniques are applied to collect and analyse data from computer systems.
- Mobile phone forensics can be summarised as using digital forensics techniques to collect and analyse the data from mobile phones in order to extract digital evidence by using tools and techniques that are tailored for mobile phone devices. Mobile phone forensic analysis is defined as "the science of recovering digital evidence from a mobile phone under forensically sound conditions using accepted methods" (Curran *et al.*, 2010).
- Network forensics is defined as "use of scientifically proven techniques to collect, fuse, identify, examine, correlate, analyse and document digital evidence from multiple, actively processing and transmitting digital sources for the purpose of uncovering facts related to the planned intent or measured success of unauthorized activities meant to disrupt, corrupt and/or compromise system components as well as providing information to assist in response to recovery from these activities" (Palmer, 2001).
- Cloud forensics, being a cross discipline of cloud computing and digital forensics and being considered as a subset of network forensics, can be summarised "as using the main phases that are used in network forensics with the techniques that are custom-made for cloud computing environments taking the technical, organisational and legal aspects of the cloud into consideration" (Ruan *et al.*, 2011).

Even though the field of investigation is different, the basis for developing an investigation process model is still the one proposed in DFRWS. A discussion of each of these fields is made in the following sub-sections.

2.2.1 Computer Forensics

Securing the data residing in computers, whether they belong to individuals or organisations, from both outside and inside threats is a task that requires reactive as well as proactive measures. As the numbers of traditional crimes conducted with the help of computers as well as the crimes conducted on computer data are on an increase, the motives for these attacks have also become diverse. While some attacks are done to display the technical competency of the attacker, others are performed to cause financial or personal losses or even bigger issues such as terrorism to individuals, organisations and even countries. A systematic investigation procedure, known as computer forensics, is essential for collecting, preserving, examining, analysing all the evidences in order to present the final findings in an admissible manner in a court of justice (Casey, 2011).

Even though computer security is extremely important, the concept is different from computer forensics and they tend to complement each other. Computer security attempts to resist the unauthorized access and maintain the confidentiality, integrity and availability of computer systems whereas computer forensics comes into the scene in order to acquire, preserve and analyse the digital evidence if and when the security is breached. Computer forensics induces the significance of security when the counterpart, computer security, fails to achieve the same and controlling the access and usage of computers, networks and other devices (Information Security and Forensics Society, 2004).

On a broad perspective, the important steps taken by a computer forensic investigator during the process are data acquisition involving factors such as selecting the appropriate data source and deciding the location on that source to choose data from, imaging involving creating compressed or non-compressed disk images, creating a digital fingerprint for the hard disk to verify the integrity using MD5 or SHA1 algorithm, custody of suspected disk, forensic analysis of data that includes dead or live analysis followed by the investigation and presentation of results. Within a suspected machine volatile and non-volatile memories are the main sources of evidence, from a computer forensic point of view. Because of this reason, any computer forensic investigator approaching a victim or attacker machine will have special interest in volatile memory sources such as RAM, cache memory and register memory and non-volatile memory such as the hard disks, USB drives, SD cards etc. Evidence might also be residing anywhere within the file system or sources such as slack spaces, virtual memory and registry files. In computer forensic investigation, it is significant that the investigator has the knowledge of how different file systems operate so that they can cope with the analysis tools (e.g.: - EnCase, FTK) that are being used in the investigation (Clarke, 2010).

The branch of computer forensics emerged and different methodologies for conducting the investigation were proposed by law-enforcement as well as academic researchers from time to time. A computer forensic investigation procedure was developed by Farmer and Venema in 1999 containing phases – secure and isolate, record the scene, conduct a systematic search for evidence, collect and package evidence and maintain chain of custody and it also resulted in development of 'The Coroner's Toolkit' (Farmer and Venema, 2005). However, the proposal was a forensic procedure that was only suitable for systems based on UNIX platform and was not a generalised one.

Reith et al. proposed a more generalised digital forensic process model in 2002 which was an enhancement of the model proposed in Digital Forensic Research Workshop (DFRWS) including phases – identification, preparation, approach strategy, preservation, collection, examination, analysis, presentation and returning evidence. Although not tested, this model was proposed to be able to overcome some of the obstacles existed in previous models such as lack of digital forensic standardisation by being able to deal with past, present and future digital devices and at the same time by providing a common framework for judicial system and law enforcement to work within a court of law (Reith *et al.*, 2002).

Another paper proposed a forensic process model with the purpose of investigating computer crimes that could be applied to general computer systems (including platforms such as Windows NT/2000, UNIX and Cisco routers) which contained phases – pre-incident preparation, detection of incidents, initial response, response strategy formulation, duplication, investigation, security measure implementation, network monitoring, recovery, reporting and follow-up (Mandia and Procise, 2003). This model was an example for a much more generalised computer forensic process model even though it was not suitable to address the forensic investigation related to devices such as PDAs, cell phones and to adapt future digital technology (Reith *et al.*, 2002).

Different investigation process models were proposed over the years with each model introducing solutions to problem of investigating the crime scene and the evidences. Table 2.1 discusses the some of the important computer forensic investigation methodologies proposed over the years, both before and during the development of other branches of digital forensics:

Table 2.1: Computer Forensic Investigation Methodologies

CF Investigation methodologies	Proposed by/ Year	Phases	Brief discussion
Computer Forensic Investigative Process	Pollitt/ 1995	Acquisition Identification Evaluation Admission	This model offered a simple and basic structure for forensic investigation taking both technical (physical & logical) and legal aspects of the case into consideration.
An Integrated Digital Investigation Process (IDIP)	Carrier & Spafford/ 2003	Readiness (operations readiness, infrastructure readiness) Deployment (detection & notification, confirmation & authorisation) Physical Crime Scene Investigation (preservation, survey, documentation, search & collection, reconstruction, presentation) Digital Crime Scene Investigation (preservation, survey, documentation, search & collection, reconstruction, presentation) Review	This model views the digital crime scene is connected to the physical crime scene and thus invites the practices used in physical crime investigation into the model and looks for hidden and corrupted files to find evidences.
End to end Digital Investigation	Stephenson/ 2003	Collecting evidence Analysis of individual events Preliminary correlation Event normalizing Event de-confliction Second level correlation Timeline analysis Chain of evidence construction Corroboration	Model proposed that incorporates computing technology and traditional investigative methods in order to find and relate the event data and evidences that form the final product of the investigation process. This model uses Digital Investigation Process Language (DIPL) with which the investigator can create different investigation models and compare them with benchmark models allowing the investigator to choose the optimum model.
Enhance Integrated Digital Investigation Process (EDIP)	Baryamureeba, Tushabe/ 2004	Readiness Deployment Trace Back Dynamite Review	In this model, each of the phases had sub-phases and was connected back to the previous phase until the output is better making the model more flexible. This model was an enhancement of the IDIP model and contained both physical and digital crime scene investigation as sub-phases of the 'dynamite phase'.
Extended Model of Cybercrime Investigation	Ciardhuain/ 2004	Awareness Authorisation Planning Notification Search/ Identify Collection Transport	This model was proposed to resolve the imperfections in the previous models and focused equally on phases other than collection and analysis and on the information flow among its phases which required a good amount of

CF Investigation methodologies	Proposed by/ Year	Phases	Brief discussion
		Storage Examination Hypothesis Presentation Proof/ Defence Dissemination	details for each case.
Computer Forensic Field Triage Process	Roger, Goldman, Mislan, Wedge & Debrota/ 2006	Planning Triage Usage Timeline Internet Evidence	CFFTP model was developed based on the triage concept that focuses on finding and prioritising the evidences and getting rid of the irrelevant ones. It identifies the timeline of important events from the crime perspective along with finding the online activities combining them to form the final evidences. However, this model is limited by the type of crime scenarios it can be applied to.
Common Process Model for Incident Response & Computer Forensics	Freiling, Schwittay/ 2007	Pre-incident preparation Pre-analysis phase (detection of incidents, initial response & formulation of response strategy) Analysis phase (live response, forensic duplication, data recovery, harvesting & reduction and organisation) Post-analysis phase (report & resolution)	A model proposed combining both management part (incident response) and investigation part (computer forensics) in a flexible manner. Even though this model applies the management procedures alongside the investigation procedures, this model is not tested and hence its workability is ambiguous. Also, factors such as the organisation's response posture and legal policies can become constraints for a full scale forensic analysis.
Digital Forensic Model based on Malaysian Investigation Process	Perumal/ 2009	Planning Identification Reconnaissance Analysis Results Proof & Defence Archive storage	This model was proposed to fulfil what previous models lacked by giving importance to steps such as acquisition of evidences, information flow and chain of custody.
Systematic Digital Forensic Investigation Model	Agarwal, Gupta M, Gupta S & Gupta S.C/ 2011	Preparation Securing the scene Survey & Recognition Documentation of scene Communication shielding Evidence collection Preservation Examination Analysis Presentation Result	An 11-phase model, proposed to fill the voids in previously proposed models providing the stakeholders an efficient of executing their cybercrime policies. Examination phase contained techniques such as keyword searching, pattern matching etc. and the model also took the country's digital forensic law into consideration.

Researchers have led their attention into different areas of computer forensic investigation such as automation, privacy infringement, litigation, evidence mapping, and issues related to the enormous volume and raw state of computer data, and implementation of artificial intelligence techniques and correlation of forensic artefacts. Continued effort was exerted on designing new frameworks as well as developing tools with better capabilities, which resulted in making computer forensics a fully-fledged branch. Table 2.2 lists the research papers and the key ideas proposed, which relates to the aforementioned areas of focus.

Table 2.2: Research papers published in the field of computer forensics

Author(s)/ Year/ Paper	Key Content of Research
James & Gladyshev/ 2013/ Challenges with Automation in Digital Forensic Investigations	Discussion of some challenges with implementation and acceptance of automation in digital forensic investigation.
Aminnezhad *et al.*/ 2012/ A survey on privacy issues in Digital Forensics	Discussion of balance between retrieving key evidences and infringing user privacy in CF.
Agarwal *et al.*/ 2011/ Systematic Digital Forensic Investigation Model	Proposed an investigation process model (11 phases) in order to help *"forensic practitioners and organizations for setting up appropriate policies and procedures in a systematic manner"*.
Mercuri/ 2010/ Criminal Defence Challenges in Computer Forensics	Recognition of challenges that criminal defence has to face due to the flaws in CF investigation process in order to provide fairness to both defence and prosecution.
Hoelz *et al.*/ 2009/ Artificial intelligence applied to computer forensics	Proposed application of AI (using multi intelligent agents system and case based reasoning) to reduce the large volume of data to be analysed and to link the evidences (using data correlation).
Arthur and Olivier/ 2009/ Processing Algorithms for Components within a Forensic Evidence Management System	Proposed the architecture (FEMS) to automate the analysis phase of the investigation in order to *"preserve the integrity of digital evidence and thereby improving the quality of investigative inferences"*.
Waits *et al.*/ 2008/ Computer Forensics: results of live response inquiry vs. memory image analysis	Describes a test scenario into which a live response approach and volatile memory image analysis approach are applied to decide which one is the better approach.
Ieong/ 2006/ FORZA – Digital forensics investigation framework that incorporate legal issues	Proposed a technical-independent framework (FORZA) (by incorporating questions – what, why, how, who, where and when into Zachman's framework) to dissolve the barricade between information technologists, investigators and legal practitioners and the tasks performed by each party; tested in a web hacking example.
Rogers *et al.*/ 2006/ Computer Forensics Field Triage Process Model	Proposed an investigation process model (CFFTPM) for "*providing onsite identification, analysis & interpretation of digital evidence in a short time frame*"; it has been tested in real world cases.

The investigation models discussed in this subsection and the research papers listed in Table 2.2 indicate the depth and breadth of the research conducted in the field of computer forensics. Starting from DFRWS, computer forensics has been the only branch of digital forensics that had undergone expansion to a much farther extend making itself, by far, the most mature branch of digital forensic science.

2.2.2 Mobile Forensics

Along with the technological developments in the field of computers, mobile devices such as cell phones, Personal Digital Assistants (PDAs), smartphones, GPS devices and game consoles have flourished and have become an inseparable part of daily life.

For instance, in the last decade, mobile-cellular/smartphone devices have displayed an enormous growth in their usage as well as in the development of technologies used in them. The number of mobile-cellular subscriptions has grown from 3.368 billion in 2007 to 7.740 billion in 2017. With the world population being 7.6 billion, the mobile-cellular subscriptions have become 101.84% of world's population (ITU, 2017). The number of smartphone users was 2.8 billion in 2017 and was estimated to reach 3 billion in 2018 and 3.8 billion in 2021 (Newzoo, 2018). A combination of powerful hardware components such as the system-on-a-chip (SoC), memory and storage, and efficient operating systems such as Android and iOS have made these devices powerful computers in their own right. As a result, smartphones are not only a medium for making phone calls and sending/receiving text messages, but are also devices that facilitate the user to engage in activities such as e-mailing, web browsing, online shopping, gaming, video calling, and using social media. As of January 2019, there were 3.9 billion active users that were using smartphones and tablets to access the Internet. Furthermore, mobile devices were accountable for 45% of the total time (6 hours and 42 minutes) of the daily Internet usage worldwide (DataReportal, 2019). As a part of providing a platform for conducting such activities, smartphones store sensitive user-data such as text messages, call details, photos, videos, passwords, contact information, bank details, personal documents and data associated with social media apps. Since being resourceful in the data sector, mobile/smartphone devices are no exception to different types of exploitations. According to the McAfee Labs' Threat Report published in December 2017, while more than 2.5 million new mobile phone malwares were detected, the total number of mobile malwares reached over 20 million in the third quarter of 2017. The number of new mobile malwares declined and reached around 1.75 million in the third quarter of 2018 (McAfee Labs, 2017; McAfee Labs, 2018). The numbers of mobile banking and mobile ransomware Trojans detected in the third quarter of 2017 were 19,748 and 108,073. A year later, in the third quarter of 2018, these numbers have become 55,101 and 13,075 respectively. Browser apps (35%), Office app (27.8%) and Android apps (22.71%) were the most exploited applications during the attacks detected in the third quarter of 2017, and the same applications (Office app

– 69.9%, Browser apps – 13.7% and Android apps – 12.4%) maintained the top three spots in the third quarter of 2018. (Kaspersky Labs, 2017; Kaspersky Labs, 2018).

The aforementioned statistics show the growing popularity of mobile/smartphone devices and the increased attention that these devices received from cybercriminals. Attacks that are targeted at these devices can cause a serious threat to the security of data stored on those devices. Alternatively, if a mobile/smartphone was retrieved from a crime scene, then it is possible that the device may have been used for communication while committing the crime or that it may contain information about the criminal or the victim (Lutes and Mislan, 2008). In both cases, the device is going to be a useful source of evidence, which needs to be analysed in a forensic fashion. Investigation that involves the extraction and analysis of data residing in different categories of mobile devices such as smartphones, tablets, game consoles and GPS devices is known as mobile forensics – a comparatively young yet developing branch of digital forensics. Due to popularity, the category of smartphone devices is chosen as an example to showcase the nature of investigation conducted within this domain.

In smartphones, the data can reside in locations such as the Subscriber Identity Module (SIM) card, the internal or external storage or the internal memory. Characteristically a SIM card can store both user and network data (Curran *et al.*, 2010; Ibrahim *et al.*, 2016). However, the gradual increase in the internal and external storage capacities of smartphones have reduced the requirement to store user-data on the SIM card. For instance, Android devices may or may not store user-data on the SIM card, which will depend on whether or not the device manufacturer allows it. Storage of user-data associated with smartphone apps can vary depending on the operating system that they run on. For instance, in Android devices, files that are specific to an app will be stored in a private directory created for that app by the file system and will be stored within the internal storage. Such files will not be accessible to other apps or the user and will be removed when the app is uninstalled. Files such as photos, videos or music that are shared between different apps and that can be accessed/modified by the user will be stored in a public directory or an app-specific directory in the external storage (Google Android Developer Docs, 2019).

Extracting data from mobile phone devices is a delicate process considering the chances of sensitive information getting lost. During the acquisition phase of the investigation, data can be extracted from the mobile device using a manual, logical or physical extraction technique. In manual extraction, the investigator will carefully browse through the contents of the device similar to how the user would and the information that appears on the screen in each step of the process will be recorded for proof. During logical extraction, the device will be connected to a forensic workstation via a cable, Bluetooth or infra-red interface and the data will be extracted in a request/response manner. The extracted data will be a copy of the logical content (i.e. files and folders) that are managed by the device's operating system. The main drawback of this technique is that it will not be able to extract any of the hidden or deleted files. Also, since the device needs to remain switched on during the extraction, possibility of the operating system altering the data is high. Physical extraction will create a raw image of the physical storage by copying data in a bit-by-bit fashion, and this type of acquisition is performed when the device is locked or broken. Joint Test Action Group (JTAG) interface, physical chip extraction and bootloader are the commonly used techniques for physical data extraction. The physical extraction technique, unlike the aforementioned methods, will be able to extract deleted files from the physical storage (Curran *et al.*, 2010; Anobah *et al.*, 2014; Cahyani *et al.*, 2017). Analysis of the extracted data can reveal information about the system, applications and the associated user, which may be useful for the investigation.

During the imaging process, it has to be made sure that there is constant power supply to avoid the risk of losing data as the internal memory is volatile in nature. The removable storage, due to its non-volatile nature, can be removed and analysed separately. Although the removable storage does not have any operating system installed in them, they have their own file system and this gives the examiner the freedom to use established computer forensic tools for the analysis purposes (Curran *et al.*, 2010; Barmpatsalou, 2018). Using non-forensic software for examination purposes is considered only as a last alternative and the interface suggested to connect the mobile device to the forensic system is cable while other interfaces such as infra-red, Bluetooth and Wi-Fi come with the risk of being insecure (Curran *et al.*, 2010). Making sure that the SIM card is not permitted to connect to the mobile phone network during the analysis process is essential so that any data present on the card does not get overwritten. A similar step used in other branches of digital forensics is used here by copying contents of the SIM card onto a new blank SIM card and for this purpose a special holder is used which will stop the original card from enabling any network connections before copying the data and the analysis can be performed on the duplicated SIM card data (Curran *et al.*, 2010). Mobile forensics tools such as UFED Touch provides built-in SIM card reader allowing the investigator to read the contents.

While conducting a forensic investigation on mobile devices, the investigators may run into different challenges. For instance, during the acquisition phase, compatibility between smartphones and the investigation tools can be a hurdle to overcome due to the availability of a wide range of operating systems. Unlike in the desktop/laptop domain, newer versions of smartphone operating systems are released within a short span of time. This will demand tools that are capable of investigating devices that run on a variety of operating systems. In order to deliver data security, manufacturers may incorporate encryption functionality into their devices, which can create another challenge for the investigators during the acquisition. Moreover, the device user can intentionally install apps that are capable of hiding, encrypting, forging or wiping the data, thereby adding another layer of complexity to the investigation (Das, 2017). Invasive physical extraction of data from the devices is a complex and costly process requiring hardware equipment tailored for the job and the expertise to carry-out the task. This could result in the investigators not being able to extract any deleted data from the devices. The growing practice of extending the devices' storage into the cloud, which allows the users to back up their data, can pose another challenge. Acquiring data that is stored in a cloud storage may require overcoming lengthy legal procedures adding an extra layer of difficulty to the investigation process. Lack of common standards for checking the quality of data extracted by different mobile device forensic tools can make the process difficult for the investigators (Anobah *et al.*, 2014).

2.2.3 Network Forensics

Wireless or wired, almost all computers and other digital devices are connected through the Internet. World Internet usage and population statistics shows that the estimated number of Internet users around the world in 2000 was over 360 million and, as of December 31st, 2017, the number has grown to over 4 billion. Asia and Europe are leading in the number of Internet users with 1.99 billion and 700.15 million users respectively (InternetWorldStats, 2018). These figures point to the fact that the number of users connected to a network via computers and other digital devices is enormous; so is the volume of data travelling through those networked devices. For instance, the annual global business IP traffic that consists of Internet traffic, managed IP traffic and mobile data was measured at a rate of 27.29 Exabyte per month in 2018. Estimates also show that the number will reach 63.31 Exabyte per month by the end of 2022. Values for the global consumer IP traffic were 129 and 333 Exabyte per month respectively (Cisco, 2018). Data of such an increasing magnitude is a clear indication of the growing number of user activities that take place within the network. This, in turn, will invite the attention of more cybercriminals followed by increasing the probability of crimes that target the network data, thereby costing the organisations their time, money and reputation.

A cybercrime that takes place within an organisational network can originate from an outsider or can be an inside job. Attacks committed by an outsider will require exploiting the vulnerabilities within the network. General types of attacks that take place in a network environment can be classified in to four main categories – probing, denial of service attacks (DoS), user to root attacks and remote to local attacks (Microsoft, 2014).

- In probing, the attacker will be making a scan of an entire network of computers, which will provide information about the machines and services that are available on the network and about the known vulnerabilities. The information gained can be utilised to administer the attack.
- Denial of Service is the type of attack in which the attacker will manage to make the resources (computing or memory resources) busy as a result of which they will become unavailable for legitimate users.
- In user to root attack category, the attacker will log in to a system within the network as a legitimate user. Upon gaining entry to the network, the attacker will have the ability to exploit the vulnerabilities and to gain root access.
- In remote to local attacks, the attacker will not be having a user account. The attacker will first exploit any vulnerability within the system in order to gain access and then will start attacking the machines.

Threats that originate from the insiders (i.e. the employees) can appear more genuine in nature as the perpetrator can gain access to the network using their legitimate user credentials, which can make insider threats hard to detect.

Organisations employ network security products such as firewalls and intrusion detection and prevention systems in order to protect their systems and data from attacks. This will result in a constant monitoring of the network traffic, and in giving alerts if anomalous activities are detected and in preventing known attacks. However, networks can still be prone to attacks that can break the security barrier set by such security products fulfilling the criminals' intentions. Hence, a systematic and time bound procedure for investigating such crimes in a forensically sound manner, known as network forensics, is essential. Network forensics can be considered as an extension of computer forensics and will include similar phases in the investigation process. However, the domain is not as matured as the computer forensics domain and the investigators may have to depend on more than one network monitoring/analysis tools for assistance during the investigation. Also, the source of evidence will be the network traffic data that was logged via firewalls, intrusion detection systems or networking devices such as routers and switches and the investigation will include the analysis of this data (Joshi *et al.*, 2010). Since the raw data that flow into and out of the network are generated as a result of the communications that took place between the users, an efficient analysis of the data can deliver useful insights into the user activities. Issues such as the increasing volume, the raw state and the encrypted nature of the network traffic, and lack of sophisticated tools that can reduce the overall effort needed from the investigators have created challenges for the domain.

2.2.4 Cloud Forensics

Cloud has become a popular term in the recent years and there are multiple service providers that offer different cloud storage solutions, both free and commercial. Cloud solutions benefit individuals as well as organisations to store, access and share their data. Dropbox, Google Drive, Amazon's Cloud, Sky Drive, iCloud are some of the most popular cloud storage services.

National Institute of Standards and Technology (NIST) defines cloud computing as *"a model which provides a convenient way of on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications, and services), that can be rapidly provisioned and released with minimal management effort or service provider interaction"* (Mell and Grance, 2009). Statistics show that an organisation can save up to 37% of their estimated cost by using the cloud storage services provided by Amazon Cloud instead of outsourcing them to a data centre (Khajeh-Hosseini *et al.*, 2010). Based on the figures projected by 'Market Research Media', the cloud computing market is estimated to grow at 30% compound annual growth rate (CAGR) attaining $270 billion in 2020 (Market Research Media, 2018). The total size of the public cloud computing market was worth $126 billion in 2017 and the projected values for 2018 and 2020 are $141 billion and $163 billion respectively (Statista, 2018). Based upon these statistics, it can be pointed out that cloud computing is one of the fast emerging markets.

Depending on the nature of service provided by the cloud service provider (CSP), three different groups of cloud computing are Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS). Table 2.3 contains the three models and their features along with some examples.

Table 2.3: Service models of Cloud Computing (Zawoad and Hasan, 2013; Ashraf, 2014)

Cloud computing model	Features	Example
Software as a Service (SaaS)	- Subscribed users have the permission to use the cloud service provider's application that is running on the cloud infrastructure that saves the users and organisations from the hassle of distributing the software. - Users will not be having control over the network, servers, operating systems, storage or on the application itself. - Partial allowance of access control management for multi-user application. - Monthly subscription fees.	Google Drive, Google calendar
Platform as a Service (PaaS)	- Subscribed users can set up their own application or SaaS application in the cloud infrastructure. - Users do not have any control over the network, servers, operating systems or storage. - Users can control the deployed application and some application hosting environment configurations. - Fee depends on the bandwidth and database usage.	Google App Engine, Windows Azure
Infrastructure as a Service (IaaS)	- By subscribing processing power and storage, users can set up their own virtual machine and install their own operating system and applications. - Users have full control over operating system, deployed applications, storage and limited control to select networking components. - Users can increase storage and processing power based on requirements.	Amazon EC2

From deployment perspective, cloud computing models can be of four categories namely private cloud, public cloud, community cloud and hybrid cloud. In a private cloud, acting as an internal data centre, the entire cloud infrastructure will be owned by a particular organisation or group. In a public cloud, however, the entire infrastructure is owned by the cloud service provider and the services can be used by the public or a large industry group. When different organisations share a cloud infrastructure because of their intersecting requirements, it forms a community cloud whereas a hybrid cloud is a combination of two or more of the aforementioned cloud infrastructures (Huth and Cebula, 2013).

The composition of the cloud computing model will only be complete with the inclusion of five essential characteristics along with the aforementioned service models and deployment models. 'On-demand self-service' is one of the characteristics that allows the consumer to provision server time and network storage automatically without having to interact with the CSPs. Another aspect of the model, known as 'broad network access', allows its consumers to access server time and network storage on different client platforms such as mobile phones, tablets, laptops and work stations. Cloud systems offer a pool of physical and virtual resources such as network storage, processing, memory and network bandwidth in order to serve multiple clients in a dynamic nature while meeting all of the client requirements. This particular property of the system is known as 'resource pooling'. 'Rapid elasticity' is another essential aspect that allows its consumers to allocate and deallocate cloud space, according to their needs, both rapidly and automatically. Cloud systems exhibit another characteristic, known as 'measured service', that provides transparency to the client as well the provider by automatically monitoring, controlling and reporting the resource usage for the services that are being used (NIST, 2011).

Perhaps the major concern for the individuals and organisations that depend on cloud service providers to store data will be about the risk of their valuable personal and organisational information being exposed to someone that has illegitimate intentions and about the security measures taken by the service providers

Cloud forensics is defined "as the application of computer forensic principles and procedures in a cloud environment" (Zawoad and Hasan, 2013). Another definition states that cloud forensics is a subset of network forensics as cloud computing is based on broad network access and hence cloud forensics follows phases of network forensics with techniques that are custom-made for cloud computing environments (Ruan *et al.*, 2011).

Resources used in cloud computing, such as networks, servers, storage, applications and services, are a collection of configurable networked resources used collectively in order to meet various client requirements. Because of this, crimes committed targeting those resources in a cloud environment will have a nature similar to those committed in a network environment. Hence, the main phases used in network forensics are followed in cloud forensics as well. However, investigation of a crime committed in a cloud environment is more complicated considering the different aspects involved. There are technical, organizational and legal aspects that need to be taken into account while carrying out a cloud forensic investigation making it an issue existing in more than one dimension (Ruan *et al.*, 2011). Some factors that elevates the challenges in conducting cloud forensic investigation are:

- Data collection for the purpose of investigation will be controlled by factors such as the location of data (user-side or provider-side), integrity of data that are being collected, maintaining the confidentiality of other tenants (in case of a public cloud), tools and techniques needed to collect the data based on the cloud service model etc. and these factors can raise challenges to the investigation process.
- Due to the elastic nature of cloud, investigation phases such as data acquisition, data recovery, evidence examination and analysis will require both static and live forensic tools on a large scale which raises challenges.
- Cloud environments in which multiple tenants are sharing the resources will contain a large pool of data which includes the potential evidence as well and segregating the data of interest from the pool is challenging and will require new tools and procedures.
- Proactive steps taken can help cloud forensic investigation (such as preserving regular snapshots of storage, continually tracking authentication and access control). However, this can be a challenging task considering the large volume and nature of cloud data.
- Even though the service agreement is between the user and the cloud service provider (CSP), a cloud environment can include more CSPs as part of outsourcing requiring in and extended level of investigation of all the links involved which can raise challenge. In the process of an investigation, the chain of CSPs involved must be working together with law enforcement, academia (for technical expertise) and third parties (for auditing and compliance).
- Depending on the location of forensic data, conducting the forensic investigation without breaking the law will require generating regulations and agreements and can be challenging.

In 2014, an extensive study was carried out by NIST in order to identify the challenges faced while conducting forensic investigation in a cloud environment, which also indicates the complex nature and magnitude of the challenges. Participants of this study were individuals from the government, private industries and academia that fall into the stakeholder community. The methodology involved an in-depth study of the available literature, gathering input information from the participants and conducting group discussions among the participants. 65 challenges were identified during the study, which are shown in Figure 2.2Figure 2.2. For a meaningful representation, the challenges were grouped into primary categories such as architecture, data collection, analysis and anti-forensics, and logical subcategories such as data recovery, data integration and metadata logs (NIST, 2014).

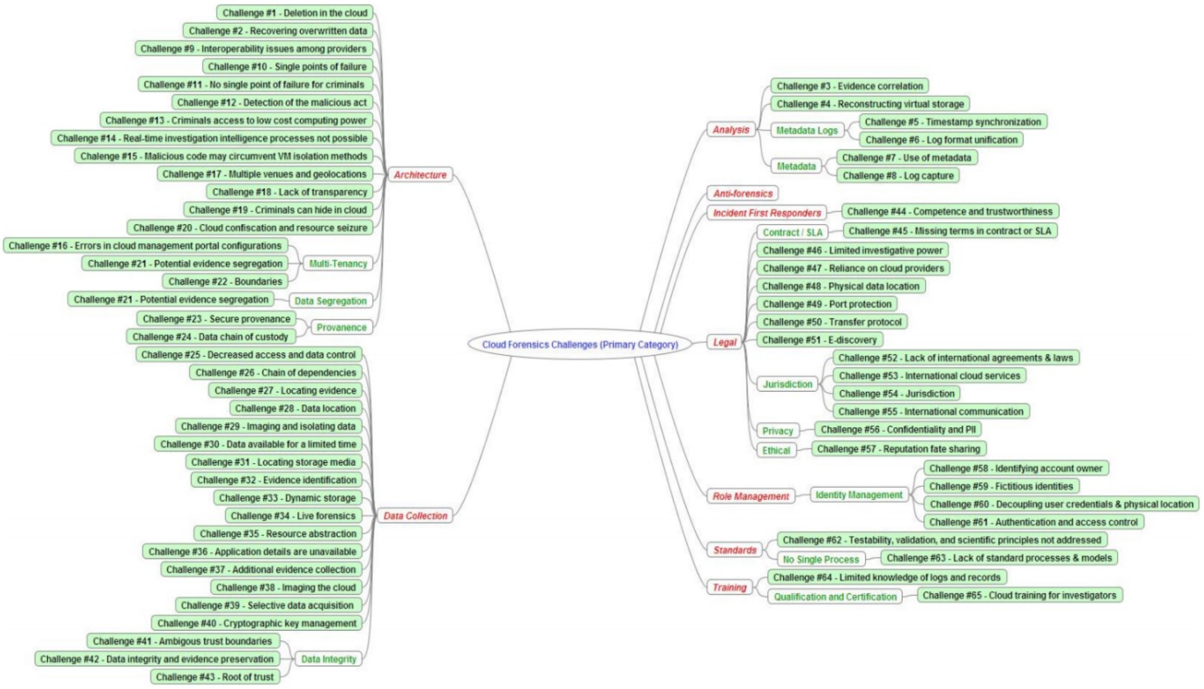

Challenge #1 - Deletion in the cloud
Challenge #2 - Recovering overwritten data
Challenge #9 - Interoperability issues among providers
Challenge #10 - Single points of failure
Challenge #11 - No single point of failure for criminals
Challenge #12 - Detection of the malicious act
Challenge #13 - Criminals access to low cost computing power
Challenge #14 - Real-time investigation intelligence processes not possible
Chalenge #15 - Malicious code may circumvent VM isolation methods
Challenge #17 - Multiple venues and geolocations
Challenge #18 - Lack of transparency
Challenge #19 - Criminals can hide in cloud
Challenge #20 - Cloud confiscation and resource seizure
Architecture
Challenge #16 - Errors in cloud management portal configurations
Challenge #21 - Potential evidence segregation
Challenge #22 - Boundaries
Multi-Tenancy
Challenge #21 - Potential evidence segregation
Data Segregation
Challenge #23 - Secure provenance
Challenge #24 - Data chain of custody
Provanence
Challenge #25 - Decreased access and data control
Challenge #26 - Chain of dependencies
Challenge #27 - Locating evidence
Challenge #28 - Data location
Challenge #29 - Imaging and isolating data
Challenge #30 - Data available for a limited time
Challenge #31 - Locating storage media
Challenge #32 - Evidence identification
Challenge #33 - Dynamic storage
Challenge #34 - Live forensics
Challenge #35 - Resource abstraction
Challenge #36 - Application details are unavailable
Challenge #37 - Additional evidence collection
Challenge #38 - Imaging the cloud
Challenge #39 - Selective data acquisition
Challenge #40 - Cryptographic key management
Data Collection
Challenge #41 - Ambigous trust boundaries
Challenge #42 - Data integrity and evidence preservation
Challenge #43 - Root of trust
Data Integrity
Cloud Forensics Challenges (Primary Category)
Analysis
Challenge #3 - Evidence correlation
Challenge #4 - Reconstructing virtual storage
Metadata Logs
Challenge #5 - Timestamp synchronization
Challenge #6 - Log format unification
Metadata
Challenge #7 - Use of metadata
Challenge #8 - Log capture
Anti-forensics
Incident First Responders
Challenge #44 - Competence and trustworthiness
Legal
Contract / SLA
Challenge #45 - Missing terms in contract or SLA
Challenge #46 - Limited investigative power
Challenge #47 - Reliance on cloud providers
Challenge #48 - Physical data location
Challenge #49 - Port protection
Challenge #50 - Transfer protocol
Challenge #51 - E-discovery
Jurisdiction
Challenge #52 - Lack of international agreements & laws
Challenge #53 - International cloud services
Challenge #54 - Jurisdiction
Challenge #55 - International communication
Privacy
Challenge #56 - Confidentiality and PII
Ethical
Challenge #57 - Reputation fate sharing
Role Management
Identity Management
Challenge #58 - Identifying account owner
Challenge #59 - Fictitious identities
Challenge #60 - Decoupling user credentials & physical location
Challenge #61 - Authentication and access control
Standards
Challenge #62 - Testability, validation, and scientific principles not addressed
No Single Process
Challenge #63 - Lack of standard processes & models
Training
Challenge #64 - Limited knowledge of logs and records
Qualification and Certification
Challenge #65 - Cloud training for investigators

Figure 2.2: Cloud Computing Forensic Science Challenges (NIST, 2014)

Findings of this study form basis for the aforementioned argument about the complex nature and magnitude of the challenges within the cloud forensic domain, thereby making it a field that demands more research.

2.3 Computer Forensic Capability

Among the branches of digital forensics, computer forensics is, by far, the most mature branch. Since the ‘golden age’, researchers have been focused on identifying and resolving different issues existed within the computer forensic domain, which increased the depth and breadth of the field. Such rigorous research has also led to the development of computer forensic tools that are capable of meeting many of the investigation requirements. Even though computer forensic tools still needs development in certain aspects, their level of capability is high compared to that of the tools available for network, mobile and cloud forensics.

In order to support the above statement, following is the representation of some of the commonly used commercial computer forensic (CF) tools based on various performance criterions. Tools and features used were selected and filtered from a comparison made available by Paraben Corporation (specialising in the developing computer forensic and mobile phone forensic tools) in January 2013 and were grouped into four categories. Computer forensic tools used for comparison are Guidance Software EnCase Forensic 7, AccessData FTK 5, X-Ways Forensic, Belkasoft Evidence Center, TechPathways ProDoscover and Paraben's P2 Commander 3.0. The four categories are:

- Capability to perform file analysis
- Capability to search
- Capability to perform forensic analysis
- Capability to report and export the results of forensic analysis

Table 2.4 lists the file analysis features, such as view, search and export the analysed files, of the computer forensics tools. FTK and EnCase can analyse variety of file types. TechPathway ProDiscover could analyse limited file types in comparison with some of the best tools available.

Table 2.4: CF tools' capability to perform file analysis (Paraben Corp., 2013; Paraben Corp., 2016)

	Guidance Software EnCase Forensic 7	**AccessData FTK 5**	**X-Ways Forensic**	**Belkasoft Evidence Center**	**TechPathways ProDiscover**	**Paraben's P2 Commander 3.0**
Multimedia files	✓	✓	✓	✓	✓	✓
Documents	✓	✓	✓	✓	✓	✓
Text files	✓	✓	✓	✓		✓
Archive files (.zip, .rar, .tar)	✓	✓	✓			✓
Windows Registry Hive files	✓	✓	✓	✓	✓	✓
E-mail files	✓	✓	✓			✓
E-mail client (e.g. MS Outlook Express)	✓	✓	✓	✓	✓	✓
E-mail client (e.g. Windows Mail E-mail)	✓	✓	✓	✓		✓
Chat history (e.g. Skype	✓	✓	✓	✓		✓

	Guidance Software EnCase Forensic 7	**AccessData FTK 5**	**X-Ways Forensic**	**Belkasoft Evidence Center**	**TechPathways ProDiscover**	**Paraben's P2 Commander 3.0**
chat/call)						
Chat history (e.g. Yahoo! Messenger)	✓	✓		✓		✓
Browser history (e.g. Internet Explorer)	✓	✓	✓	✓	✓	✓
Browser history (e.g. Firefox)	✓	✓	✓	✓	✓	✓
Browser history (e.g. Chrome)		✓	✓	✓	✓	✓

Table 2.5 lists different search features exhibited by computer forensic tools. FTK can perform the major search criterions like Boolean search, hexadecimal search, search for regular expressions, malwares, indexed files etc. Evidence Centre is the least capable among the major tools available.

Table 2.5: CF tools' capability to search (Paraben Corp., 2013; Paraben Corp., 2016)

	Guidance Software EnCase Forensic 7	**AccessData FTK 5**	**X-Ways Forensic**	**Belkasoft Evidence Center**	**TechPathways ProDiscover**	**Paraben's P2 Commander 3.0**
Boolean searches	✓	✓	✓		✓	✓
Hex searches	✓	✓	✓		✓	✓
Regular Expression searches	✓	✓	✓	✓	✓	✓
Pornographic image detection		✓	✓	✓		✓
Malware detection	✓	✓			✓	
Indexed file quick searches	✓	✓	✓		✓	✓

Table 2.6 lists different features exhibited by Capability to perform forensic analysis. FTK and EnCase can forensically analyse data from a variety of sources and are capable of calculating hash values, detecting file signatures and provides support for NIST database. Belkasoft EvidenceCentre can forensically analyse limited data sources.

Table 2.6: CF tools' capability to perform forensic analysis (Paraben Corp., 2013; Paraben Corp., 2016)

	Guidance Software EnCase Forensic 7	**AccessData FTK 5**	**X-Ways Forensic**	**Belkasoft Evidence Center**	**TechPathways ProDiscover**	**Paraben's P2 Commander 3.0**
Physical disk & disk image	✓	✓	✓	✓	✓	✓

	Guidance Software EnCase Forensic 7	AccessData FTK 5	X-Ways Forensic	Belkasoft Evidence Center	TechPathways ProDiscover	Paraben's P2 Commander 3.0
browsing						
Un-partitioned space browsing	✓	✓	✓	✓	✓	✓
Unallocated space browsing	✓	✓	✓	✓	✓	✓
Deleted file recovery	✓	✓	✓	✓	✓	✓
File slack browsing	✓	✓	✓		✓	✓
File attributes browsing	✓	✓	✓	✓	✓	✓
File indexing by signature	✓	✓	✓		✓	✓
NIST database support	✓	✓			✓	✓
File Hash code calculation (MD5, SHA-1)	✓	✓	✓	✓	✓	✓
Detecting file signature & extension mismatches	✓	✓	✓	✓	✓	✓
Custom Hash databases	✓	✓	✓		✓	✓
Memory dump analyser	✓	✓	✓	✓	✓	✓
RAM analysis	✓	✓	✓	✓		

Table 2.7 represents the capability of CF tool to report and export the results of forensic analysis. While FTK and EnCase can report and export the analysis results in different formats, tools like X-ways Forensic and ProDiscover lack to support most of these formats.

Table 2.7: CF tools' capability to report and export the results (Paraben Corp., 2013; Paraben Corp., 2016)

	Guidance Software EnCase Forensic 7	AccessData FTK 5	X-Ways Forensic	Belkasoft Evidence Center	TechPathways ProDiscover	Paraben's P2 Commander 3.0
XML report	✓	✓		✓	✓	✓
HTML report	✓	✓	✓	✓		✓
CVS report	✓	✓		✓		✓
Text report	✓	✓		✓		✓
File & Folder	✓	✓				✓

export						
E-mail export to MSG format	✓	✓				✓
E-mail export to EML format		✓		✓		✓
E-mail attachments export	✓	✓				✓

The tools listed here are a representation of the entire computer forensic tool population. While some of the tools successfully exhibit different features and meet the investigation requirements, others are limited in their capabilities. Still, due to the magnitude and effectiveness of the achievements made within the field, computer forensics prevails as the most mature branch of digital forensics.

However, challenges persists in certain areas of this domain. Increasing volume of data due to the advancements in storage device technology and time taken to carry out the investigation process are important challenges in computer forensics that needs more research. More accurate and efficient automation techniques are still needed. Similarly, legal constraints involved and steganography are other issues that need more attention and research. Lastly, existence of anti-forensic tools can aid the culprit to delete any evidences from the hard drive which can raise a huge problem for the investigation, which is being discussed in Section 2.4.

2.4 Anti-Forensics

Anti-forensics can be thought of as a growing collection of tools and techniques that upset forensic tools, investigations and investigators, intend to avoid detection of any unusual event that has taken place, affect the order of information collection, make the investigator spend more time on the case and cast doubt on the forensic report generated at the end of investigation (Garfinkel, 2007).

The oldest and the most common anti-forensic technique used is overwriting data and metadata, known as sanitization. Utilities that allow sanitization are easily distributed with most of the operating systems and require little training to run. Anti-forensic tools can perform the task of sanitization by overwriting the entire media or individual files on the media or files that were “deleted” but still exist on the drive. Apple’s disk utility and Microsoft’s ‘cipher.exe’ are two examples for anti-forensic tools that are capable of sanitizing the drive using 35 passes of data or random passes of data. However, the challenge for such tools was overwriting selective data from the drive as deciding the importance of data from a forensic perspective became difficult. The attacker can use different tools such as ‘Timestomp’ that will overwrite the ‘create’, ‘modify’, ‘access’ and ‘change’ timestamps in an NTFS file system and ‘Defiler’s Toolkit’ that is capable of overwriting the inode timestamps and delete the directory entries in Unix systems in order to overwrite the metadata residing in the targeted machine as the metadata can be crucial in giving indications about the attack.

Traditional anti-forensic techniques used by the attackers include encryption, steganography and other data hiding methods as well. Encryption techniques can be applied to encrypt a particular application or the entire file system itself, which can then be protected using strong passwords. For example, Microsoft Office Word 2016 uses a 256-bit long cipher key and the Advanced Encryption Standard (AES) encryption algorithm to encrypt the Word-documents (Microsoft, 2016), which can be really difficult to crack and the task can only become more complex if those applications are using password protection. One possible way to overcome this barrier of encryption is by waiting for the system to run those applications as the system will contain the required passwords while in RAM. But this leaves the investigator with the task of examining gigabytes of RAM data. Steganography is another technique that is being utilised by the attacker that can raise the level of challenge faced by the computer forensic investigator. In this technique, data get hid inside other forms of data such as text, image, music or video. Besides, if the data that is being hidden is encrypted, then it increases the difficulty of revealing any information. The main advantage of steganography is that it does not give a clue about whether or not any data is kept hidden. There are tools capable of hiding data within the unused blocks of a file system (e.g.: - StegFS in Linux ext2) or hiding an encrypted file system within a file system itself (e.g.: - TrueCrypt). Counter measures taken by the investigators are not strong enough to reveal the hidden information as tools such as ‘StegAlyzerSS’ and ‘StegSecret’ can only detect and uncover steganography if the methods used for the process are known in the first place. Hiding the data in locations such as slack spaces, bad blocks which the forensic tools may not consider as the initial sources for their suspicions. Tools such as Metasploit’s Slacker, FragFS, RuneFS, Waffen FS and KY FS are a few examples of the anti-forensic tools that can assist the attacker community.

Booting the operating system from a live CD or bootable USB tokens allows the attacker to use the tools of his/her choice to place the attack and then remove those sources (live CDs, bootable USBs) in order to reduce the footprint of any attack. Likewise, using virtual machines to install operating systems and to place the attacks and then carefully deleting all files related to the virtual machine aids the attacker in avoiding any attack signs. Buffer-overflow exploitations and utilising the increased storage space provided by different e-mail services to store the attack tools are also among the anti-forensic techniques used by the attackers.

Another technique that the attacker applies in order to win the war against the computer forensic tools (CFTs) is exploiting the weaknesses of forensic tools by making use of the situation where a forensic tool fails to validate its input data, placing a denial of service attack against the CFT or by converting the file formats (containing potential forensic data) into those that the CFTs may not scan for evidences. Vulnerabilities identified in tools such as tcpdump (in its decoding routines for AFS RPC packets), Snort and Ethereal can be taken advantage of by the attacker by subverting them and running the arbitrary code as the CFTs fail to validate the input data. Attacker can make the computer forensic tools hang by placing denial of service (DoS) attacks on the system resources (memory, CPU, log file entries, zip files) that the CFTs will be analysing (e.g.: - compression bombs; small compressed data files that take up enormous amount of storage space once uncompressed known as ‘compression bombs’, carefully placed regular expressions in log files etc.). Metasploit Project’s Transmogrify program can change a ‘.txt’ file-extension into ‘.exe’ and add ‘MZ’ at the beginning of the file that will trick ‘EnCase’ which will assume it as a binary file and will not scan it (Garfinkel, 2007).

Anti-forensic tools (AFTs) are capable of detecting the active presence of computer forensic tools within a system and changing their behaviour. For example, if the hard drive is running ‘Self-Monitoring Analysis and Reporting Technology’ (SMART) then the ‘power on minutes’ counter will be increasing intensely, which will be sensed by the AFT and be understood that the hard drive is being imaged.

Table 2.8 lists different anti-forensic tools and techniques along with their main features, which was created after reviewing research conducted within the field of anti-forensics.

Table 2.8: Different Anti-Forensics techniques and their features

AF tool/techniques	Examples	Main Features
Overwriting data/ metadata	Apple disk utility, Microsoft’s cipher.exe, Timestomp, Defiler’s Toolkit	Partial/Full sanitization of drive, can overwrite the timestamps, easily distributed with OS.
Cryptography, password protection	Mac, Linux, Windows OS file systems that offer encryption	Strong encryption key and passphrase can make the encrypted data difficult to crack.

AF tool/techniques	Examples	Main Features
Steganography	StegFS, TrueCrypt	Capable of hiding data inside data (encrypted or otherwise).
Minimizing the footprints of attacks	Using Live CDs, bootable USBs, VMs, memory injection	Place the attack reducing the footprints
Exploit the CFTs	Metasploit Project's Transmogrify, compression bombs	AFTs that can exploit the vulnerabilities in CFTs
Detecting presence of CFTs	Reading SMART counters	AFTs detect if the CFTs are monitoring

2.5 The SSD Challenge

Traditional data storage devices that are used in computers (i.e. hard disk drives – HDDs) have known limitations such as reliability, the speed of accessing data and high power consumption. These limitations have led to the adoption of a new type of storage device, known as Solid State Disks (SSDs). When compared with HDDs, SSDs are compact, consumes less power, and are quicker in reading data, which have made them popular.

Figure 2.3 shows gradual increase and decrease of the worldwide shipments of HDD since 1976. Statistics show that there has been an almost-steady decline in the sales of HDDs since 2010 and this trend is expected to continue for the upcoming years (Statista, 2019).

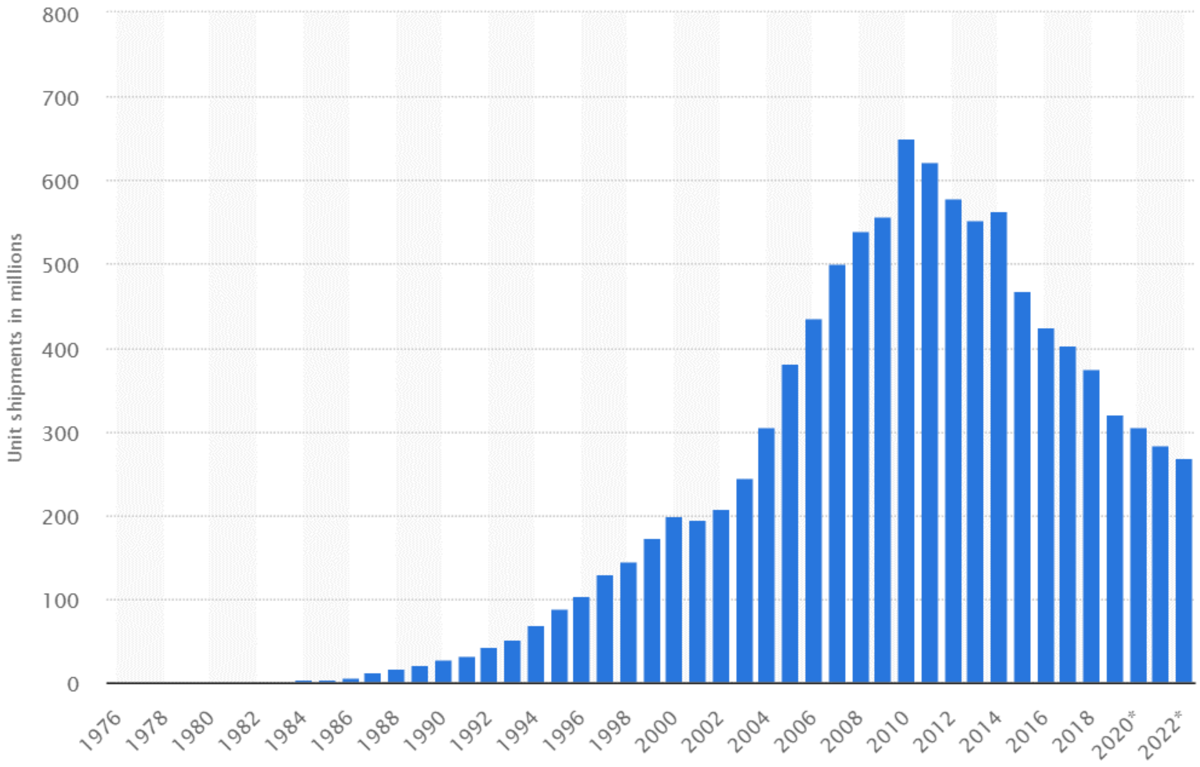


Figure 2.3: Worldwide shipments of Hard Disk Drives (Statista, 2019)

The overall revenue obtained from the worldwide sales of enterprise HDDs/SSDs for each year from 2012 till 2014, and the estimated revenue for the following years till 2019 is shown in Figure 2.4. Figures show that the revenue made via the sales of enterprise SSDs has been on a path of increase since 2012 and was estimated to surpass the corresponding values of the enterprise HDDs in 2017 (Statista, 2016). Also, the rate of increase of the revenue is higher compared to that of the enterprise HDDs.

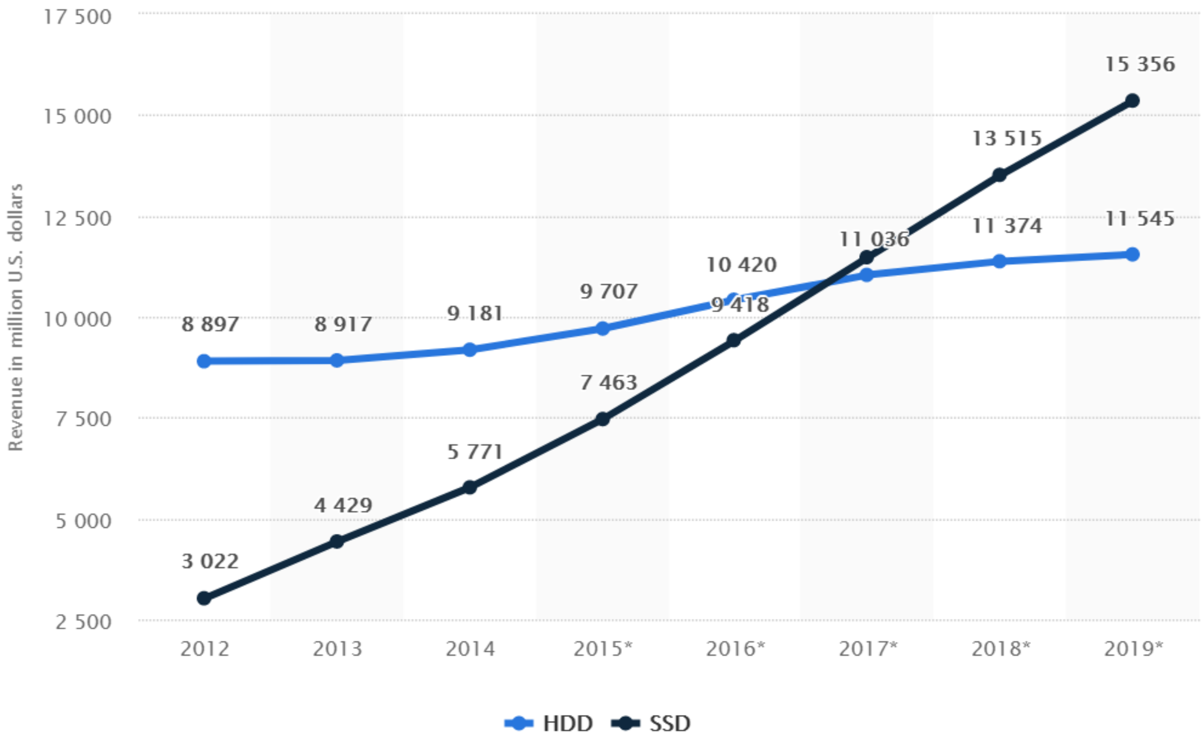


Figure 2.4: Worldwide revenue from enterprise HDDs/SSDs sale (Statista, 2016)

The aforementioned statistics imply that HDDs are being replaced by SSDs at a faster rate due to their capability to overcome the limitations that exist in HDDs. However, SSDs have introduced certain challenges for the computer forensic community. Prior to explaining those challenges, it is important to understand the composition, and the methods of operations of an SSD, because they are connected.

The underlying working mechanisms of HDDs and SSDs are completely different from each other. While HDDs consists of magnetic plates that rotates on a spindle, SSDs consists of flash memory cells that are built from semiconductor chips. A memory cell will contain an electron trapped inside, which is interpreted as '1' in the binary code; and the cell will release the electron when supplied with a charge, which is interpreted as '0'. SSDs can have different variants depending on the number of bits that can be stored in a memory cell – a Single-level cell (SLC), a Multi-level cell (MLC), a Triple-level cell (TLC) and a Quad-level cell (QLC) that can store 1bit, 2bits, 3bits and 4bits per cell respectively. The memory cells are grouped to form a single unit known as a page (e.g., 8KB or 16KB in size), and up to 512 pages are grouped to form a block. SSDs can perform operations such as reading, writing and erasure of data, and the lifespan of SSDs will decrease as the number of read/write/erase cycles increases. While data can be read from/written into an individual page, the erasure of data is carried out at a block level in SSD. This approach is adopted because erasure requires a higher voltage and therefore erasing data at a page level can stress the neighbouring memory cells. In order to increase their efficiency and longevity, modern SSDs incorporated features such as garbage collection and TRIM that focus on reducing the number of read/write/erase cycles as much as possible (Bell and Boddington, 2010; Bednar and Katos, 2011; Joshi and Habbard, 2016; Hruska, 2019).

A peculiarity of SSDs is that it will always have to erase a block before being rewritten with new data. This can cause delay for the write operation as the SSDs get filled in with more data thereby slowing down their performance in the long run. Modern SSDs solve this problem by introducing a process known as ‘garbage collection’, which will identify redundant data and will erase them in the background during its downtime. Garbage collection will occur only periodically, and will be controlled and carried out by the SSD controller that was built into the device at the time of manufacture. The SSD controller will have access to the file system’s metadata with which the controller will be able to identify the redundant memory cells (Bell and Boddington, 2010; Bednar and Katos, 2011; Joshi and Habbard, 2016; Hruska, 2019). For instance, a particular block may end up having both good and redundant pages when the operating system modifies a file. In this case, the SSD controller will read data from the good page(s) of that block and write the data into new page(s) in another block, and then will erase the old block entirely.

In the case of HDDs, a file deleted by the user is removed only logically. Even though the file is not be visible to the user, it will still be existing physically on the magnetic disk. The operating system will label the specific sectors on the disk that belong to the deleted file as available for rewriting. Contents of the deleted file can be recovered, unless those specific sectors were overwritten by new data. The procedure is completely different for SSDs. SSDs had to overcome another problem of not knowing when the user deletes a file. Unless the SSD controller can identify the pages that are associated with a deleted file, it will continue to preserve the data resulting in unnecessary read and write cycles. In order to resolve this issue, a TRIM command was introduced. Whenever the user deletes a file, the operating system will send the TRIM command to notify the SSD controller (Bell and Boddington, 2010; Bednar and Katos, 2011; Joshi and Habbard, 2016; Hruska, 2019). The SSD controller will now know that it does not need to rewrite the contents of the pages of the file in question, which will save the controller from performing any unnecessary read/write operations. Later, the SSD controller can permanently erase the contents of the block that contains the deleted file, making the block ready for reuse.

Despite being a better choice than HDDs, features such as garbage collection and TRIM can cause the SSDs to introduce new challenges for a digital forensic investigator. A computer forensic tool may or may not be able to recover deleted files (that can be of evidential value) during the creation of a forensic image of the seized traditional HDD. Also, both the evidence and the forensic image(s) will/should be having matching hash values maintaining the integrity. Conversely, this may not be the case with SSDs. SSD controllers are capable of permanently erasing redundant data and user-deleted files from the disk by employing garbage collection and TRIM, which may or may not occur prior to the disk being seized. As a result, the investigator may never get hold of data that can be considered as the smoking gun evidence. Also, it is possible that multiple forensic images created from the same SSD have different hash values as the garbage collection/TRIM operations can take place in the background. This can result in the investigator failing to maintain the integrity of the evidence, when presented in a court. Both of these scenarios cause the SSDs to add more uncertainty into the process thereby jeopardising the investigation.

2.6 Conclusion

In comparison with the other branches (i.e., network, mobile and cloud forensics), computer forensics is, by far, the most mature and popular branch of digital forensics. Research carried out within the domain of computer forensics has produced numerous well-established tools – both commercial and open-source – that are capable of performing various tasks associated with each and every phase of the investigation. While computer forensic tools are capable of dealing with case management including presentation and documentation of the analysed results, the investigator may require assistance from more than one tool to perform different phases of the investigation if and when the source of evidence is network data.

Although countermeasures can be taken against anti-forensic techniques used by the attackers, computer forensic tools and techniques are not always successful in getting hold of the cyber criminals. Anti-forensic tools/techniques can give a technically competent criminal an advantage over a criminal that do not use similar techniques. Since technique such as sanitization can wipe off data from hard drives, investigation can become a challenging task and can favour the attacker. Furthermore, nowadays, techniques such as encryption and password protection are legitimately contained within the operating systems in order to protect the privacy of data. This would permit an attacker to use such tools/techniques just as much as any normal person without any criminal intentions. Aided by the anti-forensic tools, attackers can acquire the capability of either hiding/removing the evidences or even the entire data from the hard-disk drive or initiating an attack on the computer forensic tools thereby sabotaging the investigation. Similarly, incorporation of SSDs as storage devices has increased the complexity of the investigation. Features such as garbage collection and TRIM ensure a permanent erasure of redundant data and deleted files that may have contained vital information. Since garbage collection is purely controlled by the SSD controller, it is difficult for the investigator to know the time-window before which the deleted data is gone forever. This adds more uncertainty to the situation and can compromise the integrity of the evidence.

As a result, despite spending ample time on conducting the investigation, lack of significant data on the storage device can leave the investigator with limited or no knowledge about the user activities or no useful forensic artefacts recovered. Under such circumstances, the extent to which the organizations and law enforcement can rely solely upon computer forensics tools/techniques that function only based on the data that is left behind on the suspected machines becomes questionable. Therefore, it is worth looking into alternative sources of evidence that can help in the investigation. For instance, in organisations, it is a common practice to capture and analyse network data for the purposes of network monitoring as well as intrusion detection. The captured network traffic will contain a copy of network activities – communications carried out between the computers and the online activities initiated by the employees – that took place within the organisation. An analysis of the network traffic can reveal interesting information about the employees' activities that may benefit the investigators. Due to the potential of the network traffic to be a valuable alternative source of evidence, the research will now turn its focus towards network forensics.

3 Network Forensics

3.1 Introduction

As briefed in Section 2.2.3, organisational networks are susceptible to various attacks. Using different tools and techniques, the cybercriminals may engage in activities such as penetration of network via the discovered vulnerabilities, DoS and DDoS attacks, deployment of malwares such as viruses, worms, ransomwares and Trojans, data breach, and insider threat. Financial gain, ruination of organisations' time/money, or sabotage of the organisations' reputation can be a few of the intentions behind the crimes. Network security measures and incident response procedure that are put in place in order to detect, give alerts, prevent and to restore the network to its normal behaviour may or may not succeed. However, in order to track down the criminal(s) that worked behind the attack and to find answers to the associated questions such as what, when, where and how, a forensic investigation will be required.

Palmer identified network forensics as a domain that uses scientifically proven techniques to collect, fuse, identify, examine, correlate, analyse and document digital evidence from multiple, actively processing and transmitting digital sources. The purpose of the investigation is to unveil facts about the intention or the success of unauthorised activities that are meant to disrupt, corrupt or compromise system components. The information gained through the investigation is expected to benefit in either responding to or recovering from the unauthorised activities (Palmer, 2001).

According to Yasinsac and Manzano (2001), network forensics involves monitoring the network traffic, determining if there is an anomaly and ascertaining whether it indicates an attack. If there is a confirmed attack, then its nature is determined and the network forensic techniques enable the investigators to track back the attacker. The ultimate aim is to provide enough evidence in order to present the culprit in the court of justice. Network forensics is defined as the capturing, recording and analysis of network events with the intention of discovering the source of security attacks or any other incidents that cause problems (Pilli *et al.*, 2010). Section 3.2 of the chapter discusses different forensic methodologies put forth through research that can be utilised for conducting investigation in networked environments. This section also discusses the challenges faced by the investigation. Features of a handful of the popular network forensic analysis tools are reviewed in Section 3.3. Section 3.4 of the Chapter discusses about the existing approaches for analysing the network traffic (i.e. the packet and flow based analysis), and establishes why the aforementioned approaches are not suitable for the proposed user oriented network forensic analysis tool. Section 3.5 discusses the Security Information and Event Management system, a component used by the incident response teams for discovering network trends and detecting security issues, followed by the discussion and conclusion in Sections 3.6 and Section 3.7 respectively.

3.2 Network Forensic Methodologies

During the first ever Digital Forensic Research Workshop (DFRWS) in 2001, an objective was set to extend the application of digital forensic science to include the networked environments and a framework was put forward. The framework proposed in the workshop (DFRWS) consisted seven steps namely identification, preservation, collection, examination, analysis, presentation and decision. Following this, different investigation process models were proposed over the years. However, the models recommended until 2005 were not with the sole purpose of conducting network forensic investigation as those models included network forensics only in a generalized form (Joshi et al, 2010).

In 2005, Ren and Jin proposed a generalised network forensic investigation process model that was designed with networked environments in mind, and was the first of its kind. The model contains capture, copy, transfer, analysis, investigation and presentation phases, and utilises different tools that help to automate the capture phase and the analysis phase. The model is designed to capture data from two types of sources – end-side sources (i.e. a victim or an attacker machine) and intermediate-side sources (i.e. the network-related sources). The model uses traditional computer forensic tools to capture data from various files residing in the victim or the attacker machine while on the end-side, and captures data from the network traffic, firewalls or IDSs while on the intermediate-side. Different phases and the process status transfer are shown in Figure 3.1 (Ren and Jin, 2005).

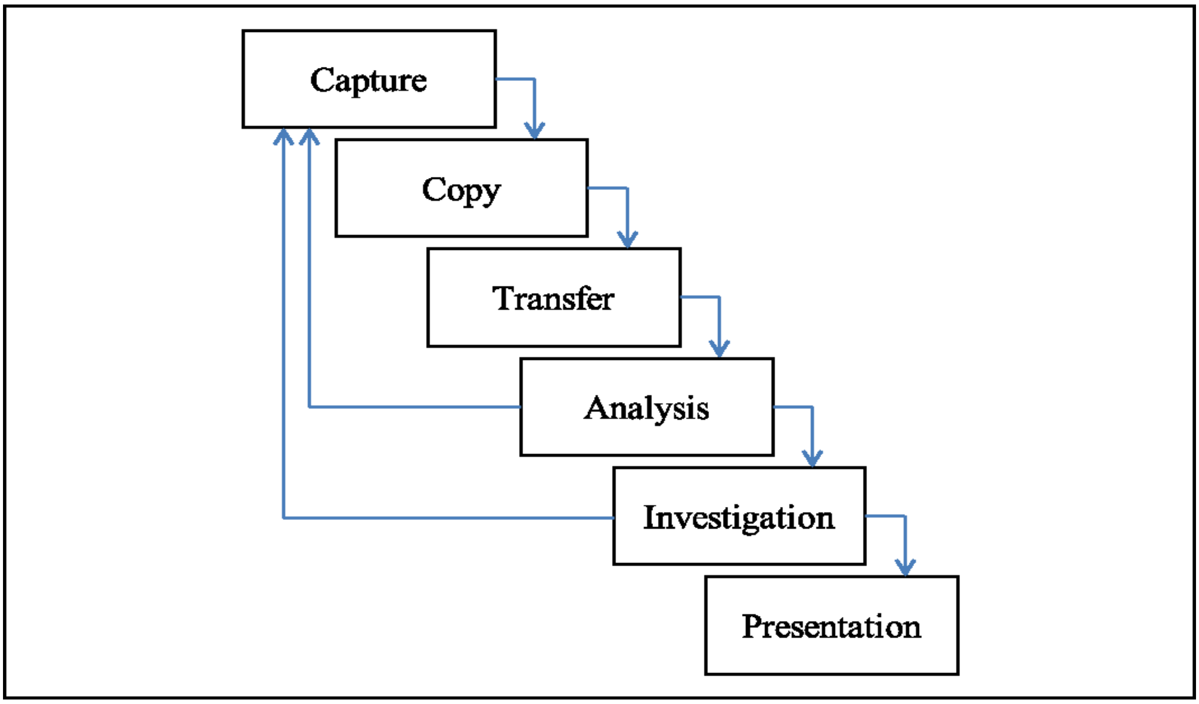


Figure 3.1: Phases of General NF Process Model (Ren and Jin, 2005)

Capture phase obtains the data from different sources such as network traffic packets, firewall log, IDS log, and router log without breaking the data privacy, followed by the second phase (i.e. ‘copy’) in which the original data is copied to read only memory, transfer network or analysis machine and this phase makes sure that raw data is preserved. The mission of the ‘transfer’ phase is to transport the copied data to the analysis machine without affecting the speed and security.

High significance is given to the ‘analysis’ as this is the most sophisticated phase in this process model, which includes data filter for data extraction, meta-analysis that includes IP packet statistics analysis, protocol analysis and session analysis, and integrated analysis performed after data fusion including association analysis, misuse replay and data mining analysis. Figure 3.2 is the diagrammatic representation of the analysis phase (Ren and Jin, 2005).

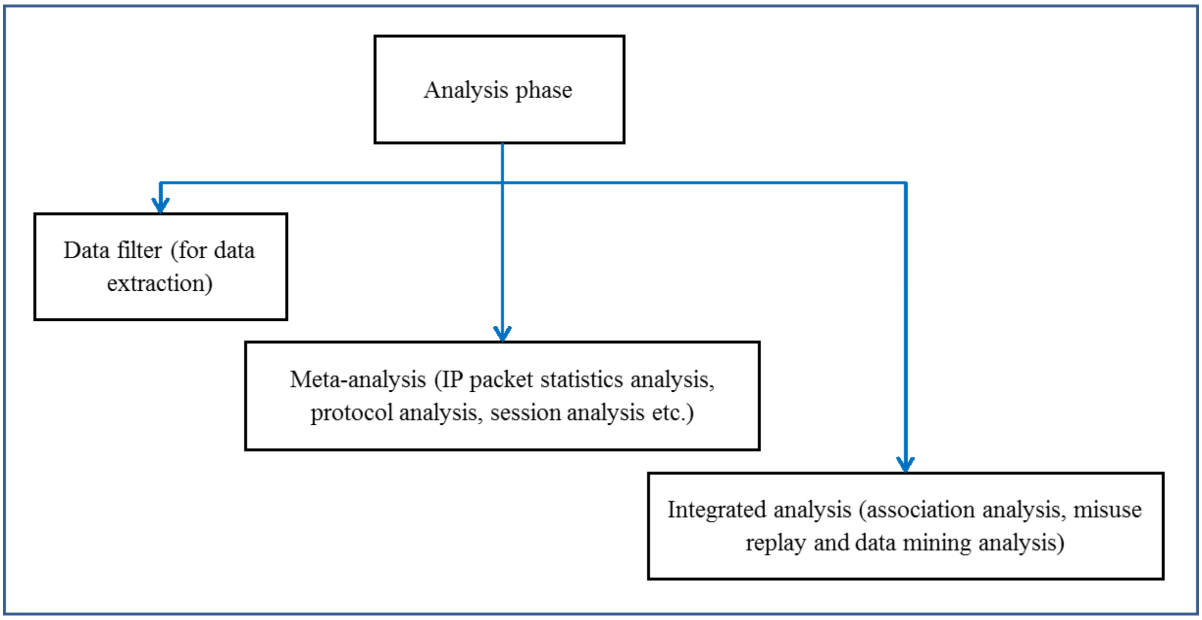

Analysis phase
Data filter (for data extraction)
Meta-analysis (IP packet statistics analysis, protocol analysis, session analysis etc.)
Integrated analysis (association analysis, misuse replay and data mining analysis)

Figure 3.2: Analysis phase of the process model

The 'Investigation' phase is where the information about the source or the perpetrator is acquired. Tools such as traceroute, whois and wget are used for this purpose in this phase for the determining the geographical location of the source. Following is the 'presentation' phase, which results in presenting the conclusion accompanied by the procedure for conclusion making. Even though different phases of this process model follow a sequential order, the analysis and investigation phase roll back to the capture phase (Ren and Jin, 2005).

The designed network forensic system had main parts called 'network forensics machine' and 'analysis machine' and the system had the potential to be deployed in both intranet and Internet environments. Network forensics machine doesn't have any IP address which eliminates the possibility of being detected by any external parties and only the analysis machine has access to the network forensics machine. According to the proposed design, the system can monitor and discover the inside attacks which can be analysed further. However, practical implementation of this model in an Internet environment encounters the problems of cost, scalability and efficiency. Within an Internet environment, the large volume of data that has to be monitored and recorded creates practical difficulties.

In 2010, Pilli *et al.* proposed a generic process model for performing network forensic analysis which contains a methodology designed specifically for network based investigation. Phases of this model are shown in Figure 3.3 followed by their respective functions.

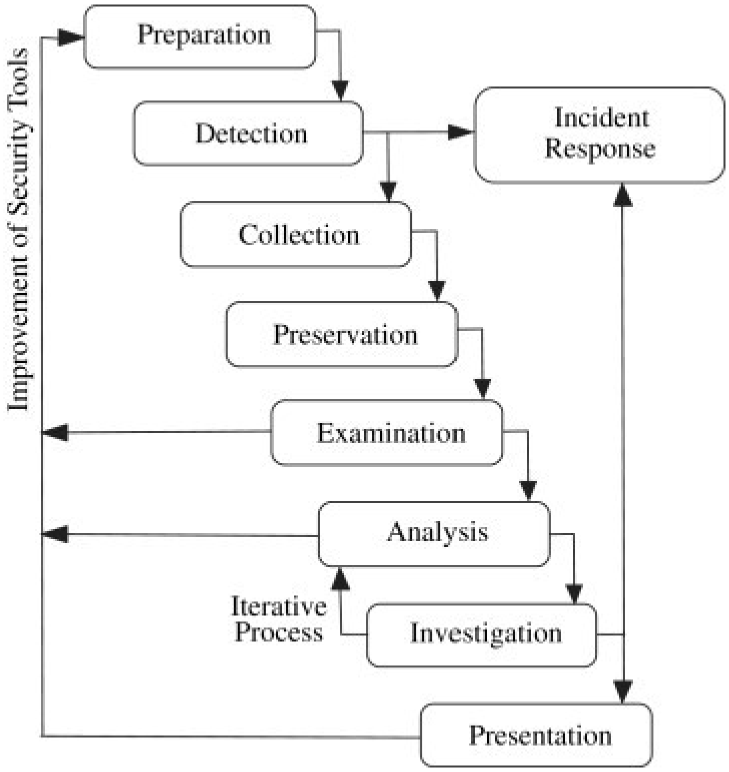


Figure 3.3: Generic process model for network forensics (Pilli *et al.*, 2010)

Preparation: Different network security and monitoring tools and network forensic analysis tools (NFATs) are carefully and strategically deployed at different points within the network with the intention of collecting the evidences. Obtaining legal warrants and authorizations are also part of this phase in order to avoid any privacy violations.

Detection: The network security and monitoring tools deployed in the preparation phase produces alerts if there are any policy violations or security breaches and these alerts are perceived in 'detection' phase and a quick validation is performed determining if it was a false alarm or a real attack without altering any evidence. Based on the outcome of this phase, the next phase can either be 'incident response' or 'collection'.

Incident response: In this phase appropriate response to the crime or intrusion incident is made based on the information gathered in the detection phase followed by initiating an action plan on how to defend future attacks and recover from existing damage. Simultaneously, decision is made whether to continue with the investigation or not. Incident response is guided by organization policy, legal and business constraints. This phase is connected to the 'investigation' phase as well where a different action plan may be necessitated based on the information obtained from the investigation phase.

Collection: This is where the network traffic data collected using hardware and software tools based on the alerts generated by the sensors. This phase can be tricky as the live traffic data can change at quicker speeds and will require the right tools like TCPDump, Wireshark, TCPFlow, NfDump, Sebek etc. in order to collect the data.

The challenge here is storing the huge quantity of data collected requiring massive storage space and the tools used should be capable of saving the data in appropriate log formats and the currently existing tools will need human assistance to perform this task.

Preservation, Examination and Analysis: After the 'preservation' phase in which the captured network traces and logs are copied to be analysed while the original data is stored in a read-only format along with their hash values, the copies of the data obtained from different security tools are combined together to one large data set and is examined methodically to extract specific crime indicators in 'examination' phase and also a feedback is given to improve the security tools.

The crime indicators generated in the examination phase are passed on to the analysis phase which are then classified and correlated with the existing attack patterns using statistical, soft computing and data mining methodologies to match attack patterns. Then these attack patterns are combined together to reconstruct and replay to understand the intention and methodology of the attacker. Like examination phase, analysis phase gives feedback to improve the security tools (Pilli *et al.*, 2010).

Investigation and presentation: In this model, the investigation and analysis phases are executed iteratively as different features from the analysis phase may be required for the functioning of the investigation phase in order to produce the 'attribution of attack' (i.e. to establish identity of the attacker). As mentioned above, the investigation phase feeds the resulting data to the 'incident response' phase and further on for the act of prosecution.

The presentation phase of this model consists of presenting the observations and conclusions with proper explanation and visualizations in front of the court in an understandable manner and documenting all these findings in order.

Designed with a purpose of eliminating any threat and to restore the operations back to its normal state, incident response plays an important role within the forensic investigation process. Incident response can be considered to be executed in two stages – an 'initial setup and preparation' stage followed by an 'incident response sequence' stage (Lamis, 2010). The first step of the 'initial setup and preparation' stage is to set up the incident response team and to assign the roles and responsibilities for the team members. The second step of this stage is to prepare the team in advance to face the security incidents. The 'incident response sequence' stage includes 'identifying' whether a particular event is a threat (and if so, then followed by determining the severity, documentation that continues throughout the process and evidence collection), 'containment' in which the incident will be stopped and the control of the compromised system will be regained, 'eradication' in which the vulnerability will be eliminated from the system, 'recovery' in which the system will be restored to its normal capabilities and operations, and 'follow up' in which the strengths and weaknesses of the preparation and response stages will be identified and appropriate steps will be taken to eliminate the weaknesses. Decision on whether or not to continue with the investigation to trace the attacker will be made at this point. If the incident has caused severe damage to the organisation, then further investigation will be carried out. Continuing with the forensic investigation, traces will need to be collected and preserved in a secure location for the analysis stage during which the indicators of crime will be extracted, which will be presented to the senior management for further actions to be taken (Lamis, 2010).

Similar to other phases, incident response is a significant phase of a network forensic investigation. In a network, any incident that causes any level of threat can be termed as a security incident. Incident response is the phase that performs the task of responding to any such security incident, and it involves much more than technical problems. Incident response has to deal with management, legal and social issues involved within any particular security problem (Khurana *et al.*, 2009). Incident response starts with the detection of a security breach and proceeds from there. Once the infected system is identified, an action plan will be initiated by the incident response team in order to recover it from the damage. This will be achieved in stages, beginning with the isolation of the infected system from the rest of the network followed by the elimination of the incident. Subsequently, the compromised system will be restored and put back into work. In order to keep track of all actions taken, documentation will take place in each stage of the incident response (Khurana *et al.*, 2009). A challenge to overcome within the incident response phase is ensuring that the response is quick and in real-time. If there is any delay in initiating the response, then it is only going to result in the loss of essential data that belongs to an individual or an organization or both. The incident response process should begin as soon as a security incident occurs and the alert is initiated. At the same time, the attacker should not be made aware of the response that is taking place on the opposite side (Khurana *et al.*, 2009). In their research, Khurana *et al.* pointed out that it is not sufficient to trust the incident response teams within the organisations alone, and proposed a framework for collaborative incident response and investigation. The research has put forth an idea of teaming up the incident response and investigation processes across multiple organisations and legal entities to identify, track and eliminate the threat and to prosecute the culprits behind the attack. The framework is called 'Palantir' which is discussed in the paper 'Palantir: A framework for collaborative incident response and investigation', published in 2009. The research also suggested that the aforementioned collaboration should be managed centrally by a trusted entity called the Independent Centre for Incident Management (ICIM). The proposed model has two components – a 'roles and responsibilities model' that defines the roles and responsibilities of different entities involved in the incident response and investigation and a 'process model' that defines different phases and the execution of responsibilities. The paper also talks about a security architecture that is designed based on the proposed idea, followed by the implementation of the collaborative incident response and investigation and the evaluation of the model (Khurana *et al.*, 2009). The work identified three challenges for the model, the trust issue being the first. Establishing trust between multiple organisations and personnel that are involved in the process becomes difficult due to the private nature and importance of the organisational data. Also, facilitation of the process raises financial concerns. The second challenge is associated with the management of all tasks and processes involved in the incident response and investigation. The third challenge is related to the analysis of data gathered from various digital devices within the organisational network and the duration for which such a huge volume of data need to be managed (Khurana *et al.*, 2009).

Alongside proposing different process models for conducting the NF investigation in a structured and forensic manner, the research has also focused on identifying the challenges that the investigation may run into and on proposing solutions that can minimise the effects of such challenges, which are discussed in the remainder of this section.

Since being the source of evidence, network traffic is a central element of the NF investigation. When network security events/incidents are detected, the corresponding network traces will have to be collected and analysed as a part of the investigation. Network traces can be collected from sources such as IDS and firewall logs, log files generated by the network services and applications, and packet captures by packet sniffers and different network forensic analysis tools (Nikkel, 2005). The enormous volume of data that traverse an organisational network not only demands bigger data storages to hoard the collected network traces but also creates a major challenge for the investigation. Furthermore, the investigators are faced by another challenge – the difficulty in understanding the captured network traffic due to the raw state of the data. A research proposal, put forward by Laurence D Merkle in 2008, termed these challenges as the ‘quantity problem’ and the ‘complexity problem’.

In the aforementioned research, reference is being made towards a typical network forensic analysis procedure that takes place in a series of stages. In the initial stages of the analysis, the investigator forms a hypothesis regarding the system being compromised. In the following stages, data is acquired from network traffic logs and various system events, which will be used to reinforce the hypothesis prepared by the investigator. Towards the final stage, the contents of individual packets are examined, which will either endorse or oppose the hypothesis formed by the investigator. If there is a confirmation of the hypothesis, then the data can become an essential link in the chain of evidence. Even though software tools that can perform each stage of the analysis exist, they initially will have to collect large volumes of data from various sources followed by the correlation of the collected data. Any irrelevant data can only be discarded after completing the tasks of collection and correlation, resulting in the ‘quantity problem’. Also, the raw state of the collected data adds another obstacle, which is called the ‘complexity problem’. In order to tackle these challenges, the research proposed an idea of integrating tools that are used in different stages of the forensic investigation to form a single system by utilising computational intelligence. Similarly, the proposed system will combine existing search techniques that are used in network forensic analysis. According to the proposal, forming a single tool can increase the efficiency of the entire system and reduce human interaction (Merkle, 2008). The methodology of this proposal was to generate reasonably realistic variations of known network attacks by implementing a hybrid evolutionary computational algorithm on the attacker system. The resulting data set would then utilised as the source for performing the forensic analysis. Tool for network analysis is selected, which is followed by the formal characterisation of the information produced and required by each stage of the analysis. After identifying the processes (the most time-consuming and error-prone) that needs to be automated, a design and prototype of the integrated system is developed. This system will apply the appropriate computational intelligence, thereby grouping those processes based on their characteristics. The evaluation of the system is performed using the component testing test, which is generated by randomly partitioning the network forensic data set (Merkle, 2008).

Another research, conducted by Mukkamala, was an attempt to tackle the quantity problem and contribute towards fulfilling the need for automation. The research proposed the automation of offline intrusion analysis by utilising artificial neural networks (ANN) and support vector machines (SVM), and suggested that eliminating or ranking the attributes of network traffic based on their significance can reduce the volume of network data that needs to be analysed upon the detection of network security events (Mukkamala, 2003). The network traffic can be stripped in order to extract numerous features such as the source/destination IP address, the timestamp, the source/destination port number, the protocol, the packet size, and many more. Eliminating the features that are useless and ranking the useful features according to their significance can reduce the volume of network traffic that needs to be analysed. Furthermore, the research also proposed that adoption of this approach can increase the accuracy and the overall performance of the detection process. The dataset required for the experiment was taken from the MIT's Lincoln Labs, which was developed for offline intrusion detection system evaluations by DARPA in 1999. The dataset was comprised of network traffic data that falls into different categories of network attacks such as normal, probe, denial of service, user to super-user, and remote to local. The experiment found out that the elimination or ranking of features did contribute towards reducing the volume of traffic, increasing the accuracy and the overall performance of detection. The experiment also concluded that the SVM technique performed better than ANN technique when it comes to scalability, training time and prediction accuracy (Mukkamala, 2003).

Another work, done by Stoffel *et al.* in 2010, used the application of fuzzy sets into a forensic investigation system with the goal of reducing the effort needed during the forensic analysis. Application of fuzzy sets will derive a set of if-then rules that are supposed to overcome two important constraints existing in forensic data analysis – the low accuracy of results and the difficulty in being easily understood by humans. The methodology used in the research involved division of the raw forensic data into groups, called clusters. Subsequently, membership functions were extracted from the cluster data followed by the creation of the fuzzy inference systems that will contain the set of if-then rules. These rules can be understood by the investigators without ambiguity, and can be used more efficiently in a court of justice as they are not too complex or technical. The data set that was used for testing the system came from numerous incidents of robberies and residential burglaries within a certain area. The set of if-then rules showed increased accuracy of forensic evidences retrieved and the rules derived by the fuzzy sets were easy to understand (Stoffel *et al.*, 2010). This research can be considered as an indication of a possibility of similar approach (i.e. fuzzy logic) being applicable to forensic data (i.e. network traffic) generated as a result of network intrusions as well, which may increase the accuracy of the results and the easiness to understand.

As mentioned in the beginning of this section, IDS logs are utilised as one of the sources from which network traces are collected for forensic analysis. Therefore, any approach (within the intrusion detection domain) that can reduce the volume of IDS logs may contribute towards reducing the volume of traffic that needs to be analysed. Research conducted in the area of intrusion detection by utilising machine learning techniques such as ANNs, K-nearest neighbour, decision trees, SVMs, and genetic algorithms has been on a rise since last decade. Using any one machine learning technique to approach the intrusion detection problem is termed as ‘single classifiers’. If several (typically two) machine learning techniques are combined, then the method is termed as ‘hybrid classifiers’. In hybrid classifier method, the first classifier will accept the raw data producing intermediate results, which will be accepted as inputs by the second classifier to generate the final results. A third method called ‘ensemble classifiers’ was proposed, which combined the weak learning algorithms that gave an effectively improved overall performance. Tsai *et al.* proposed an idea of using machine learning techniques for intrusion detection that will collect only the most useful network events that will contain the highest probable evidence, thereby helping the forensic analysis to become more efficient. The research proposed that if hybrid and ensemble classifiers are combined, then it can produce better results within intrusion detection (Tsai *et al.*, 2009). In this research, 55 different studies within the field of intrusion detection were reviewed, which took place during the time period of 2000 to 2007. Factors such as the total number of articles that were published during those years, the techniques (i.e. single, hybrid or ensemble classifiers) utilised in those studies, the specific machine learning algorithm that was used in each of the studies, and the dataset that was used in each of them were taken into account. While most of the studies used public datasets like DARPA 1998, DARPA 1999, some of them used private datasets. The research paper concluded by stating the future research options such as the architecture of multiple classifiers (by combining ensemble and hybrid classifiers), and feature selection to determine which feature selection method will serve best (Tsai *et al.*, 2009). Even though this research is specific to the domain of intrusion detection, as mentioned above, its ability to collect only the most useful network events may help with the quantity problem, thereby making an indirect contribution towards network forensic investigation.

Increasing usage of encryption by the Internet based services is another factor that can limit the investigation. In order to transfer private/confidential information in a secure manner, different online services such as websites that provides news, banking and shopping services, and email, file transfer and chat services started using application layer encryption. When encryption is used at the application layer, a secure channel that connects two hosts (e.g., a client and a server) is created. This standard technology used for establishing this secure channel was the Secure Socket Layer (SSL) protocol or is its successor, the Transport Layer Security (TLS) protocol (Symantec, 2019). Once a TLS handshake is established between the client and the server, the client can initiate the first Hyper Text Transfer Protocol (HTTP) request ensuring secure transmission of data (IETF RFC 2818, 2000). In order to facilitate secure data transmission between the hosts, the TLS protocol utilises encryption algorithms such as Elliptic Curve Cryptography (ECC), RSA algorithm or Digital Signature Algorithm (DSA), which encrypts the data and prevents a third party from reading the information (Symantec, 2019). The encrypted nature of data can limit the results of the investigation, especially if the analysis is focused on the payload of the network packets. If the decryption key is available, then the investigators might be able to decrypt the data. But, this will only add more time and complexity to the investigation process. Therefore, it is important to think about alternative options for analysing the network traffic.

3.3 Network Forensic Analysis Tools

Network monitoring and analysis tools are utilised in monitoring the organisational network traffic, in collecting network data if any anomalous activity detected, in assisting in the network forensic investigation and in facilitating incident response. The aforementioned actions belong to different phases of network forensic investigation procedure, thereby enabling Network Forensic Analysis Tools (NFATs) to play a crucial role in the investigation process.

NFATs capture the network traffic so that the network administrator/ investigator can analyse the data and extract the significant features. They can preserve the network traffic records for long periods of time for analysis at a later point and they can replay the attack traffic in order to understand the attacker's moves (Corey et al., 2002). Also, they can organize the network packets that they captured as individual transport layer connections between the machines helping us to see the packet content (Joshi et al., 2010).

However, it has to be mentioned at this point that there are many different network security and monitoring tools that are utilised in different phases of network forensic analysis. Those tools are not designed to have an edge in collecting evidences and analysing the captured data to reach at conclusions. Still, they are positioned at appropriate place and their services are borrowed in most of the phases during the process of network forensic investigation (Casey, 2004). This gives the clear indication that the network forensic analysis tools require further research into them. A paper entitled 'Network traffic as a source of evidence: tool strengths, weaknesses and future needs' proposes some basic requirements for tools to process network traffic that could be used as evidence:

- *"Support Tcpdump format (import and export).*
- *Reliable protocol identification and reconstruction (e.g., detect protocol violations, review/test for Tool Implementation Errors).*
- *Data reduction (e.g., various methods of locating desired items and excluding extraneous data).*
- *Known files (e.g., use MD5 values to exclude known files, flag known bad files such as hacker tools and exploits, and flag suspicious files such intellectual property).*
- *Data recovery (e.g., automated file extraction and display, image gallery, reconstitute KaZaA fragments from multiple sources).*
- *Recover hidden data in network traffic (e.g., detect protocol anomalies and steganography).*
- *Keyword search capabilities (across fragmented messages, regular expressions, Unicode).*
- *Documentation (e.g., audit trail of all digital evidence examiner actions, system performance, packet losses).*

- *Integrity (e.g., calculate MD5 value of captured digital evidence and record for future reference, save evidence onto write once media).*
- *Read-only during collection (e.g., no response on network including ARP replied, offline DNS resolutions).*
- *Read-only during examination (e.g., do not access graphics on Web servers when reconstructing and displaying a page).*
- *Complete collection (e.g., capture full packet, minimize losses).*
- *Security (e.g., secure remote access and administration, tools should not run as Administrator or root, review tool source code for vulnerabilities)"* (Casey, 2004).

Based on the proposed requirements, the paper compared and rated commercial network forensic analysis tools, NetIntercept and NetDectector, that were developed to analyse network traffic and process the evidence and open source network protocol analysers, TCPFlow and Ethreal, which is tabularised in Table 3.1.

Table 3.1: Commercial NFATs and open source protocol analysers rating (Casey, 2004)

Capability	Tcpflow	Ethreal	NetIntercept	NetDetector
Tcpdump import and export	Implemented	Implemented	Implemented	Implemented
Flow/stream reconstruction	Implemented	Implemented	Implemented	Implemented
Protocol decoding	Not implemented	Implemented	Well implemented	Well implemented
Data reduction	Not implemented	Implemented	Well implemented	Well implemented
Known file detection/ exclusion	Not implemented	Not implemented	Not implemented	Not implemented
Data recovery	Not implemented	Implemented	Well implemented	Well implemented
Hidden data detection	Not implemented	Not implemented	Implemented	Not implemented
Keyword searching	Not implemented	Not implemented	Well implemented	Well implemented
Audit log	Not implemented	Not implemented	Not implemented	Implemented
Integrity checking Mechanism	Not implemented	Not implemented	Not implemented	Not implemented
Loss documentation	Not implemented	Not implemented	Not implemented	Implemented
Read-only collection	System dependent	System dependent	Implemented	Implemented
Read-only examination	Not implemented	Not implemented	Implemented	Implemented
Security	System dependent	System dependent	Implemented	Implemented

This comparison shows that there is still scope for improvement in many of the features that network forensic tools exhibit. Also, some of the well-established network protocol analysers have the potential to grow many features into them and become more helpful in conducting a network forensic investigation. Further research can contribute towards improving the existing capabilities of NFATs such as protocol stream reconstruction, creating audit log, ability to re-establish web pages without contacting the web servers, and ability to be secure against vulnerability attacks. Also, sufficient amount of research is needed in order to implement features such as known file detection/exclusion, hidden data detection including steganography, and maintaining the integrity by calculating MD5 hash values for evidences.

Table 3.2 lists some of the most commonly used network forensic analysis tools based on whether they are commercial or open source products, type of appliance and description followed by a discussion of some of those important NFATs.

Table 3.2: Network Forensic Analysis Tools

Product	Owner	Appliance	Description
NetDetector	NIKSUN Inc.	Hardware/ software	• Along with a dynamic application recognition capability, this NFAT incorporates anomaly and signature-based IDSs. • Capable of reconstructing applications such as Voice, Video, Web, Email, Images, FTP file transfer and Chats and supports protocols such as TCP, UDP, IPv4, IPv6, IEEE 802.3, DNS, and SIP. • The tool has its own management console, includes NetOmni Suite for data aggregation, visualisation and reporting. • NetDetector Suite also contains NetDetectorLive, an appliance that can detect security violations via content-based rules and give alerts.
NetWitness	RSA (security division of EMC)	Hardware/ software	• A system designed to capture live network traffic, to recreate application layer network sessions and to analyse the data. • Consists of three main parts – a configurable 'decoder' for capturing, filtering and analysing the network data, a 'concentrator' providing a Linux based platform for the network appliance combining metadata from 'decoder' in real time for analysis and a 'broker' providing a real time, single point of access to all metadata from the entire enterprise network.
SilentRunner	AccessData Corp.	Software	• Captures and analyses wireless Ethernet 802.11a, 802.11n, webmail, chat and social media along with monitoring about 2500 protocols and services. • Identifies malicious insiders, malwares, and break-in attempts. • Allows to playback network events in the right order. • Uses interactive graphical representations to help analyse users, hosts, applications, protocols etc.
OmniPeek	LiveAction	Hardware/ software	• Supports both wired and wireless networks (802.11a/b/g/n/ac), view web traffic and monitors application performance. The tool also supports analysis of VoIP and Video applications. • Offers integrated flow and packet-level analysis. • The visualization part provides a top-down view of activities taking place within the network through packet visualizers.
PyFlag	Open source	Software	• This open source NFAT can examine captured traffic recursively at multiple levels. • Dissects packets at low level protocols (IP, TCP, UDP), forms protocol streams by combining associated packets followed by high level protocol dissection. • Main components are IO source, File System Loader, Virtual File System, Scanners, Database and Web GUI which are discussed later in this chapter.
NetworkMiner	NETRESEC (open source	Software (professional	• Capable of performing live network traffic capture and off-line analysis with a PCAP file parsing speed of 0.581 MB/s (in free edition) and 0.457 MB/s (in professional GUI

Product	Owner	Appliance	Description
	and professional version)	version comes in a USB flash drive)	version). • Information is grouped per host in the user interface. • Can reassemble transmitted files, certificates, user credentials and identify rogue hosts. • Protocols supported for file extraction are FTP, TFTP, HTTP and SMB.
Xplico	Open source	Software	• Captures network traffic, divides data at the protocol level and reassembles for analysis. • Tool has four macro-components – Dema (a decoder manager), Xplico (IP/network decoder that decodes the data), Manipulators (a set of applications to compare and combine data for analysis) and Visualisation System (to view the extracted data). • Supports protocols such as HTTP, SIP, IMAP, POP, SMTP, TCP, UDP etc.

NetDetector, a commercial network forensic analysis tool from NIKSUN enterprises, captures and analyses network traffic and by utilising the integrated IDS the tool is capable of providing more of a proactive response to the attacks. Main features are listed down below.

- A hardware appliance that captures, analyses and reports data. And its appliance has a large storage supply.
- It has alert mechanism (GUI pop-ups, and emails).
- The tool integrates both anomaly-based and signature-based Intrusion Detection System and is able to run a complete forensic investigation for the security administrator.
- Capable of reconstructing applications and packet level decodes.
- It can import and export data in a variety of formats (HTTP, FTP, SCP)
- It supports common network interfaces (10/100/1000 Ethernet, T1, FDDI) and protocols such as TCP, UDP, SCTP, IPv4, IPv6, SIP, DNS etc. (Sira, 2003).
- Not capable of full reconstruction of DNS protocol exchanges.

Being integrated with anomaly and signature-based intrusion detection systems, NetDetector is capable of monitoring the network traffic to find if there are variations from the normal network activities or to determine if there are any security breaches by comparing the captured packets with the pre-determined attack patterns (NIKSUN, 2013).

According to the tool datasheet, the tool makes use of hundreds of types of metadata in order to deliver the feature called application and session reconstruction and this feature comes with ‘quick mode’ allowing the investigator/ administrator to quickly analyse through terabytes of data and reconstructing the applications and sessions faster and ‘full mode’ allowing to go for thorough analysis (NIKSUN, 2013).

One important feature of NetDetector is that along with supporting both half and full duplex network interfaces with speed ranging from 10/100/1000 Mbps, 10Gbps and the tool was also able to deliver full packet capture, recording and analysis even while the system throughput came close to 20Gbps. NetDetector claims to reconstruct many different applications such as voice, video, web, FTP file transfers, chats, E-mail, images, NetBIOS, IRC, Microsoft and Adobe applications.

SilentRunner is a commercial network forensic analysis tool owned by AccessData Corporation that is capable of capturing, analysing and visualising the activities within a network that can have data rates of up to gigabytes.

- The tool does not have any hardware component as part of it.
- It can generate an interactive graphical representation of series of events and correlates actual network traffic.
- It can play back and reconstruct the security incidents in their exact sequence (Sira, 2003).
- Applications that the tool is capable of reconstructing include webmail (such as Gmail, and Microsoft Outlook), data and VoIP (Skype) (AccessData, 2013).

For collecting the data, the tool uses Red Hat Linux platform helping to make sure that no packets are missed while capturing and Oracle 11g is utilised for creating the database. The tool has four main components – collector, knowledge base, analysis engine and context-management.

Collector – captures the raw data while positioned at the appropriate place within the network and can also be used for real time network monitoring to find any anomalies or suspicious applications.

Data manager – the investigator uses it to write queries to the central database that stores all data captured by the collector and to save the analysis outcomes for future reference.

NetworkMiner is another tool, owned by NETRESEC, which provides both open source and professional versions of the tool into the market. Following are the main features of NetworkMiner

- Can store the OS details (OS fingerprinting)
- Can arrange the captured traffic based on sent/received packets or sent/received frames or based on OS
- Can store files that have been transmitting through the network locally into your system.
- Can extract and display images separately
- Can search for keywords
- Can store and display the login credentials (username, password) for some websites
- Can display DNS lookups, IP addresses, ports etc.

- NetworkMiner Professional (i.e. the commercial version of NetworkMiner) has the option of showing the geographical location of a certain server with limited accuracy. The feature is accomplished by utilising the GeoLite database created by MaxMind, a third party provider of IP geolocation databases.
- NetworkMiner Professional can perform ‘host highlighting’. That means, we can highlight a particular host and the tool automatically will highlight any frame or image sent/received from/to the same host in the same colour we used to highlight.
- It can classify and assemble the captured frames based on different hosts making it easier for the user (NETRESEC, 2013).

PyFlag (Python Forensic and Log Analysis GUI), initially developed by the Australian Department of Defence for the purpose of performing digital forensic analysis and later declared as a free software under General Public License (GPL) by the Free Software Foundation, was further developed making it an advanced network forensic analyser which is capable of supporting multiple sources of data such as disk images, memory images and network captures (Cohen, 2008). Figure 3.4 is the overview of the tool’s architecture:

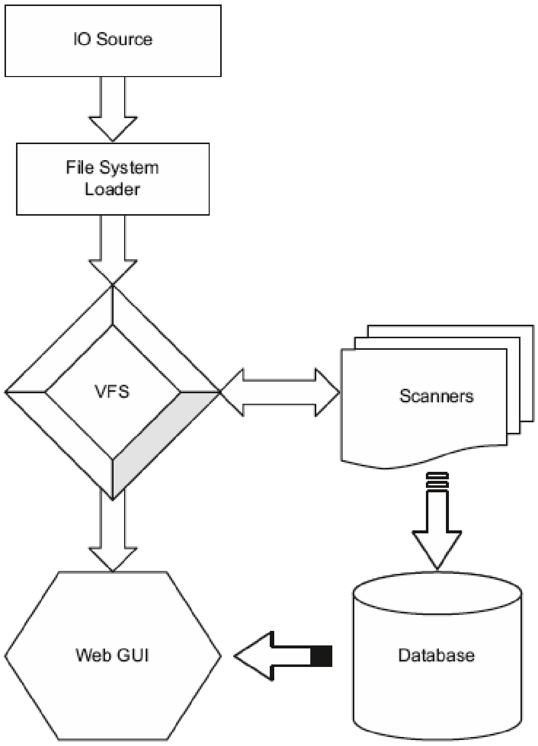


Figure 3.4: Overview of PyFlag architecture (Cohen, 2008)

With the support from interface provided by the IO subsystem and IO source drivers, PyFlag stores large PCAP files in EWF (eye witness evidence file format). A primary concept in the architecture of PyFlag is virtual file system (VFS) in which all objects are known as inodes and are represented using an ‘inode id’ (the actual path taken to arrive at that data).

Another part of PyFlag architecture is file system loader which is responsible for inhabiting VFS with inodes. Different file system loaders are used depending on the IO source (e.g. PCAP file system loader uses inodes to represent the TCP streams whereas the memory file system loader uses inodes to represent process IDs). PyFlag uses a ‘mount point’ in its VFS into which all inodes generated by a particular file system loader are loaded helping the tool to combine data from disk image, memory image and network traffic capture files. PyFlag uses scanners which are modules that operate on VFS inodes collecting and storing information about those inodes into the database. It also uses a typescanner to determine the type of file by scanning the file header and in certain cases the file scanners run recursively on the inodes (e.g. if there is a zip file, then the zip file scanner will run on the zip archive creating new VFS inodes for each member of that particular zip file followed by running file scanners on each of those archive member files) (Cohen, 2008).

In order to perform network forensic analysis, different components used in PyFlag include stream re-assembler, packet handlers (for obtaining information from non-TCP packets such as DNS lookup) and stream dissectors. The re-assembler used in PyFlag to reconstruct the TCP streams is based on the supposition that the hosts communicating may not be aware of network traffic being captured and analysed and therefore will not make a deliberate the re-assembler’s attempt to combine the forward and reverse streams in chronological order and result in creating the inodes. Stream dissectors are specialised scanners (e.g. HTTP dissectors) used to parse and if necessary decode different higher level protocol contents in order to obtain useful information. Main features of PyFlag are listed below:

- An ideal tool for the network protocols that layered in nature and supports a number of network protocols.
- Has the ability to recursively examine data at multiple levels.
- This tool can parse PCAP files and can dissect the packets at low level protocols (TCP, IP, and UDP).
- It can reassemble the related packets into streams and can dissect these data streams using high level protocol dissectors (HTTP, IRC, SMTP, and POP) (Cohen and Collet, 2007; Sourceforge, 2005).
- Capable of rendering static HTML pages. However, not fully capable to re-establish the dynamic web pages (depends on the role JavaScript plays in the page construction).

Even though NFATs exhibit important features the performance is not to a sufficient level. Areas such as reconstruction of high level information, data correlation, and usage of advanced data analytics, case management, checking the integrity of extracted evidences, hidden data detection, and efficient automation require further research. Also, the number of open source network forensic analysis tools are less and there is still room for more open source network forensic analysis tools that can perform efficiently and more research should be welcomed to address the existing issues in network forensics and develop NFATs to meet the requirements.

In network forensics, a particular tool that can perform all the required tasks does not exist yet. Even though research is going on in the field, an efficient network forensic analysis technique has not come in practise yet. This gives the hacker community motivation to enhance their skills on a regular basis knowing that undertaking a forensic investigation is just one of jobs that a network administrator will have to do. Putting the large amount of money invested in business by the organizations at stake give extra pleasure to the attackers. The cost involved in the production of new network forensic analysis tools and the training costs are other limiting factors. Another challenge existing with the current tools is that they are reactive, not proactive. What the existing tools do is that they monitor the network and if there is an attack detected they will notify the administrator. Even though they will assist in further actions taken by the administrator, the tools cannot interfere with the traffic on its own or they cannot perform any counter-attack on its own. They require human intervention and direction. This is why more research needs to be done in this field in order to develop more tools and techniques which can perform the tasks without human interference, in an efficient manner.

3.4 Current Approaches to Network Traffic Analysis

Network traffic analysis turned into an area of interest in the 1990s (Claffy et al, 1995; Debar et al, 1999). As a result, since then, network administrators started applying the network traffic analysis techniques to multiple domains such as management, prioritisation, performance, accounting, application behaviour analysis and security. Based on the granularity of the analysis, the network traffic analysis approaches can be classified into two categories – packet based (fine grained) analysis and flow based (coarser grained) analysis. While the packet based network traffic analysis method inspects the packet content (i.e. the payload), the flow based analysis technique analyses a summary of multiple IP packets that share a set of similar features for a period of time. Sections 3.4.1 and 3.4.2 discusses these two methods in detail including the working principles behind, followed by the existing works and the advantages and disadvantages.

3.4.1 Packet Based Network Analysis Method

The packet based network analysis method, also known as Deep Packet Inspection (DPI), involves the examination of content of all IP packets that traverse the network. Collection of these IP packets can be facilitated by making use of either hardware appliances or software tools such as tcpdump, Wireshark, Xplico and Microsoft Network Monitor. This type of network analysis benefits in the detection and prevention of different types of network attacks such as malwares, network intrusions and data exfiltration. In these attacks, the attackers conceal information within the header and payload sections of IP packets are made use of by the attackers.

The collected packet content (i.e. the IP address, port number and the payload information) will be compared against existing rulesets in order to identify the presence of an attack. Different pattern matching techniques such as string matching and self-organising maps are utilised for this purpose. Any match would be an indication of an attack. A number of counter measures, such as blocking the traffic or raising an alarm, could be taken against such attacks based on the application in which the packet based analysis method was used. Based on this principle, a number of studies were carried out that focused on counteracting the threats (Clarke *et al.*, 2017). Table 3.3 lists examples of such studies along with their applications, matching methods and performance.

Table 3.3: Examples of work conducted in packet based network analysis (Alotibi, 2017)

Studies	Applications	Methods	Performance
Mahoney and Chan (2001)	Anomaly based network IDS	Learns the normal ranges of values for each packet header field at the data link, network, and transport/control layers	65% Detection Rate
Wang and Stolfo (2004)	Anomaly based network IDS	Models the payload of normal applications in two stages: profiling byte frequency distribution and standard deviation during the training phase and the Mahalanobis distance for the detection phase.	100% detection rate with 0.1% false positive for port 80 traffic
Zanero (2005)	Anomaly based network IDS	Uses Self-Organising Map to analyse the payload of TCP packets	66.7% detection rate with 0.03% False positive
Wang et al 2005	Zero-day worm detection	Correlates ingress/egress payload alerts to identify the worm's initial propagation and also automatically generates signatures.	Over 95% of detection rate with less than 0.5% of false positive
Bolzoni et al (2006)	Anomaly based network IDS	Exams the packet payload via the combination of Self-Organising Map and a modified PAYL system	73.2% (Detection Rate) with less than 1% of False Positive
Wang et al (2006)	Buffer overflow attack blocker	Extracts instruction sequences from HTTP request and determines if the instruction contains malicious code	100% Detection rate on HTTP traffic
Ahmed and Lhee (2011)	Malware detection	Classifies the packet payload based upon three categories: multimedia (e.g. jpg), text (e.g. asp) or executables.	False negative (4.69%) and False Positive (2.53%)
Al-Bataineh and White (2012)	Detection of data exfiltration	Uses entropy and byte frequency distribution of HTTP POST request contents to detect the presence of data stealing botnets	99.97% detection rate on HTTP traffic
He et al (2014)	Detection of encrypted data exfiltration	Employs the entropy technology and user's network behaviour to detect data exfiltration in the cloud environment	Close to 90% detection rate with less than 1% of false positives

The main issues identified with the practical implementation of the packet based analysis method were the time consuming nature of the pattern matching technique and increasing amount of data (with a size of Giga bits per second) that needed to be processed by the network devices. With the intention of enhancing the performance and efficiency of the packet based network analysis based applications, a wide range of research was carried out. Research of this nature focused on packet sampling techniques (Jurga and Hulboj, 2007), hardware based solutions (Cho et al, 2002; Dharmapurikar et al, 2004; Sourdis et al, 2005; Smith et al, 2009) and novel algorithms (Liu et al, 2004; Dharmapurikar and Lockwood, 2006; Lu et al, 2006; Yu et al, 2006; Smallwood and Vance, 2011; Sun et al, 2011).

To add credit to the packet based network analysis method, it can be effective against different network related attacks such as malware distribution, data exfiltration, DoS attack and network intrusions. However, despite having a broad range of methods devised with the intention of improving the performance, the deep packet inspection method is still a time consuming process. This is mainly due to its bit-by-bit comparison nature making the analysis of Gigabits of organisational network traffic within a fraction of seconds a challenging task. Along with this issue, the increasing usage of SSL/TLS protocols creates more encrypted traffic posing another significant barrier to the packet based analysis method. Any attempt to obtain the payload in a plain text format by decrypting the data will not only introduce further delays but will also be a compromise on the confidentiality of the data.

3.4.2 Flow Based Network Analysis Method

A flow is the summary of a group of IP packets that share a set of common properties such as source and destination IP addresses, port numbers and time and date stamps traversing a certain network observation point during a given frame of time (Claise, 2008). The flow based analysis approach is the one that depends on the concept of analysing the IP flows to detect various network attacks. A typical flow record will normally contain the following attributes: the time and date stamps that will indicate the start and end of the flow, the IP addresses of the source and destination, the port numbers that were used, the total size of the packet payload, the total number of IP packets and the type of protocols (e.g. TCP or UDP). In order to obtain a network flow record, a flow generation application is used instead of directly gathering from the raw network traffic. Some of the established flow generation applications existing in the field are Cisco's NetFlow, sFlow, Juniper's J-Flow and IETF's IPFIX (Cisco 2015; Sflow, 2015; Juniper 2015; Claise, 2008). The criteria for completing the collection of flow record would be different for each individual flow generation application depending on their implementation and configuration. For example, one application would be completing the collection when the flow is idle for a certain period of time while for another application it would be when the FIN or RST flags of the TCP protocol are set. In a flow based network monitoring system, information about the current traffic flow is quickly compared against the historical flow data. The system does this by making use of various pattern classification techniques such as neural networks or statistical models. Any deviation from the historical flow data, such as the amount of data being sent or the type of traffic on a specific port number, can be considered as an incident (Clarke *et al.*, 2017). Many methods and tools have been proposed and devised within the flow based network analysis domain. Table 3.4 presents a number of selected examples of such studies along with their applications, pattern classification techniques and performance. Also, a number of sample techniques for the flow based analysis approach have been proposed including Estan and

Varghese, 2003; Duffield et al 2004; Duffield et al 2005; Androulidakis et al, 2007 and Canini et al, 2009.

Table 3.4: Examples of work conducted in flow based network analysis (Alotibi, 2017)

Studies	Applications	Methods	Performance
Kim et al, 2004	Anomaly network detection	Uses the flow header information and the traffic pattern to create the detection function and threshold	The method could detect scanning and flooding attacks
Pao and Wang, 2004	Signature based network IDS	Creates an access control list by using IP addresses and port numbers	The method was able to detect the Ping sweep, DoS and port scan attacks.
Crotti et al, 2006	Traffic classification	Uses statistical method to analysis network traffic at the IP flow level; traffic from both directions between the client and the server were considered.	99.4% and 97.5% detection rate for server's and client's HTTP traffic respectively
Song et al 2006	Anomaly based network IDS	Employs the back-propagation neural network to exam the statistically aggregated flow data	94% detection rate with 0.2% false positive
Muraleedharan et al, 2010	Anomaly based network IDS	Utilises a chi-square detection mechanism to analyse IP flow characteristics	100% detection rate on 15 attacks but with long delays
Braga et al, 2010	Anomaly based network IDS	Verifies IP flows by using Self-Organising Map to detect distributed DoS	99.11% detection rate and 0.46% false alarm rate
Winter et al, 2011	Anomaly based network IDS	Uses One-Class Support Vector Machines for the analysis of IP flow data	98% detection rate
Tegeler et al, 2012	Malware detection system	Utilises a statistical method to create training and identifying models for detecting botnet traffics	90% detection rate with 0.1% false positive rate
Jadidi et al, 2013	Anomaly based network IDS	Utilises the Multi-Layer Perceptron neural network with a Gravitational Search Algorithm to analyse the flow data	99.43% accuracy rate
Hofstede et al, 2013	Anomaly based network IDS	Mainly employs Exponentially Weighted Moving Average (EWMA) for mean calculation algorithm	95% detection rate with 1% false positive rate

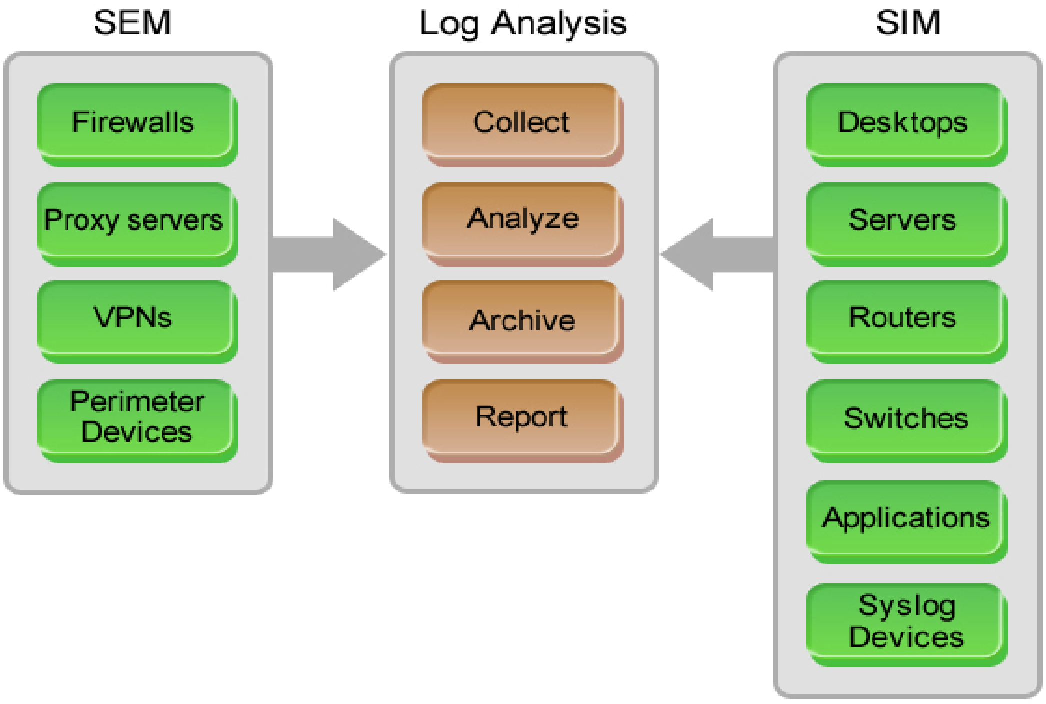
The ability to handle large volume of network traffic, both encrypted and unencrypted, in a timely manner was the crucial advantage of the flow based analysis method over the packet based analysis approach. This made the flow based network analysis method the primary option for security analysts when it came to investigating different network related issues such as malware distributions, network intrusions, Distributed Denial of Service (DDoS) attacks and traffic classifications. However, the high-level abstraction nature and further usage of sampling techniques raised certain issues. This had caused the flow based network analysis approach to lose the granularity of the traffic such as the number of the file being downloaded by an attacker during a given period of time or the number of emails being sent by the user. Furthermore, there were factors affecting the performance of the flow based network analysis approach such as the deployment of the DHCP protocol, the usage of wireless technologies (e.g. Wi-Fi and 4G), the use of network proxies by users (e.g. The Onion Router network) and the utilisation of IP spoofing by the attackers.

As demonstrated in Section 3.4.1 and Section 3.4.2, the analysis of network traffic is a well-established domain it has been widely used to detect various network related attacks. Nevertheless, the multi facets of the network environment generate an increasing set of challenges for the existing approaches to experience. It is envisioned that the identification of network activities/interactions would offer additional information to the network forensic investigators when an incident has occurred.

3.5 Security Information and Event Management (SIEM)

While network forensics is an essential part of the organisation taking the post-crime investigation into account, there is another component of the organisational network called Security Information and Event Management (SIEM), which acts as a core tool in detecting real-time threats and performing log management for forensic investigation, which has objectives similar to that of network forensics. While most of the SIEM systems can only carry out tasks such as collection, correlation and aggregation, SIEM systems such as Splunk are more refined and delivers more than just collecting, aggregating and managing the data. The tool is able to provide real-time visualisation of network events and Internet services in use, which will allow the analysts to find information that is of interest in a reduced-effort fashion. Despite coming from a network security domain, SIEM tools of this nature can become helpful to a forensic investigator in acquiring a human-level understanding of the network behaviour. Therefore, it is worth looking into different SIEM techniques, their features, limitations and challenges, which are discussed in this section.

SIEM systems were formed by combining Security Information Management systems (SIMs) that focus on collecting log file data from different sources such as desktops, applications, servers, routers, switches etc. to analyse network trends in order to detect any threat and Security Event Management systems (SEMs) that focus on activities of perimeter devices within a network such as firewalls, proxy servers, virtual private networks (VPNs) and IDSs in order to improve the incident response capabilities of those devices. SIEM systems operate on log data from various network devices performing collection, analysis, archiving and reporting. Figure 3.5 exemplifies the concept of Security Information and Event Management (SIEM):



© AdventNet, Inc. All rights reserved

Figure 3.5: Security Information and Event Management (ZOHO Corporation white paper, 2007)

Main features that SIM products aim to achieve includes log collection, archiving, historical reporting and forensics whereas SEM products implement features such as real time reporting, log collection, normalization, correlation and aggregation. These essential features were pooled, forming SIEM products, giving a complete view of the network (Cisco, 2010). Following are the main functions of an SIEM product:

- Log collection – is performed by an SIEM product in order to collect details of events recorded by different sources in an organizational network. This log collection process feeds both forensic tools in finding evidences and compliance reporting.

- Normalization – allows SIEM to convert the log data, which are stored in dissimilar formats depending on their sources, into a common data model. This puts the analysis and connection of events at ease for the organization.

- Correlation – is the process to identify the links between log data and events collected from different sources within the organization which will fuel the procedures of both detecting the security threats and reacting to those threats.
- Aggregation – allows SIEM to merge event records that are duplicate (as more than one source may provide records about the same event), so that the size of event data can be reduced.
- Reporting – involves producing long-term summaries and allows monitoring of correlated and aggregated event data in real time (Cisco, 2010).

Gartner defines SIEM as "*SIEM technology supports threat detection and security incident response through the real-time collection and historical analysis of security events from a wide variety of event and contextual data sources. It also supports compliance reporting and incident investigation through analysis of historical data from these sources. The core capabilities of SIEM technology are a broad scope of event collection and the ability to correlate and analyse events across disparate sources*" (Gartner, 2012).

SIEM systems were introduced in the early 2000s; in a time when the existing technologies lacked in their capabilities to deal with the growing amount and accuracy of security data and failed to provide a centralised visibility. SIEM products were designed and developed to assess the vulnerabilities, to reduce false positives from IDSs by handling and investigating log data collected from different sources such as firewalls and IDSs, and they used existing database management tools and specialized analytics for this. Since mid-2000s and followed by the regulation explosion, SIEM systems became a core tool for handling the security data and were adopted by many of the industries. Increased compliance regulations lead to an escalation of security instrumentation which resulted in an on-going growth of security data and gradually started questioning the data handling and analytic capabilities of SIEM systems. Innovative attackers developed more sophisticated methods of security breach exploiting the limitations of SIEM systems to scale and analyse security data resulting in the need for SIEM to detect anomalies fast and attacks that appear genuine in nature and attain contextual information (McAfee, 2013).

Some of the well-established vendors such as IBM/Q1, Novell (now owned by NetIQ), HP/ArcSight, Quest Software, Symantec, Splunk, NetIQ and Tripwire have their SIEM products in the market. Table 3.5 lists the important features exhibited by the above mentioned commercial SIEM vendor products:

Table 3.5: SIEM Product Features and Explanation

SIEM Product Features	Explanation
Real-time analysis for alerts	Quick analysis of data during incidents, this is the most important feature that any SIEM system should exhibit and it depends on how quickly the event data are displayed and loaded for analysis with high rate of accuracy. Benchmarking set by SANS suggests 1 minute from the actual event to display the data, 2-3 minutes to load the data for analysis and the rate of accuracy as 90% for an efficient SIEM product. Correlation, heuristics, behaviour-based anomaly detection etc. can be used to increase the accuracy (Butler, 2009)
Automated log collection from multiple sources	Another important feature of a SIEM product is its ability to collect log and event data from various sources within the organisation in an automated manner.
Search capabilities	Product feature that helps in analysis of event and log data by searching with keywords or conditional statements that are simple and effective in nature.
Root cause analysis and investigation of archived logs	This feature focuses on the analysis (not in real-time) of log data collected from different sources for the purpose of finding the cause of attack.
Real-time management	This feature decides the notification of and reaction time to events and depends on the factors such as the speed and accuracy of the reaction time.
Event correlation	This feature depends on how quickly (near real-time) the SIEM product can correlate between the events and longer time in correlation and analysis can delay the reaction to an event.
Operational dashboard	Interactive dashboards producing result-oriented graphs, texts and data which allow the security analysts to navigate through to further explore the results.
Secure log management	This feature provides a central repository for log data and using this feature SIEM system can perform a forensic analysis of incidents.
Event normalization	As the log data from different sources will be in different formats, it is important to convert the data into a normalized format.
Support for up to thousands of events per second (EPS)	Another important feature in determining the performance rate of a SIEM product. Generally, SIEM collectors can handle 10,000 – 15,000 events per second. Also, it is essential to prioritize the events based on their significance and to know the ‘peak EPS’ for each of the chosen devices in the organisation.
Out-of-the-box compliance reports	Ability to provide out-of-the-box compliance reports can make more organisations to accommodate that particular SIEM product.
Compression for efficient log storage	Compressing the log data is an important feature, as the data after normalization can use more space than raw data, so that its storage becomes more efficient.

Understanding the prominence and the primary factors which drive the use of SIEM systems within an organization is important. Since early 2000s, the incapability of existing technologies to collect and analyse enormous amounts of event and security data made SIEM system an essential tool of organisations and the need for stronger security has been on demand ever since. Real-time threat detection and meeting compliance requirements seem to be most significant and prevailing driving factors for SIEM use among the organisations and SIEM products use the collected event and log data in order to fulfil the purposes of real-time threat detection and compliance reporting. A 2012 SIEM vendor evaluation survey conducted by InformationWeek observed that among the respondents, 44% indicated real-time threat detection and 26% indicated meeting compliance requirements as the factors driving the use of SIEM systems (Francis, 2012). Figure 3.6 shows the survey results of factors that drive SIEM use.

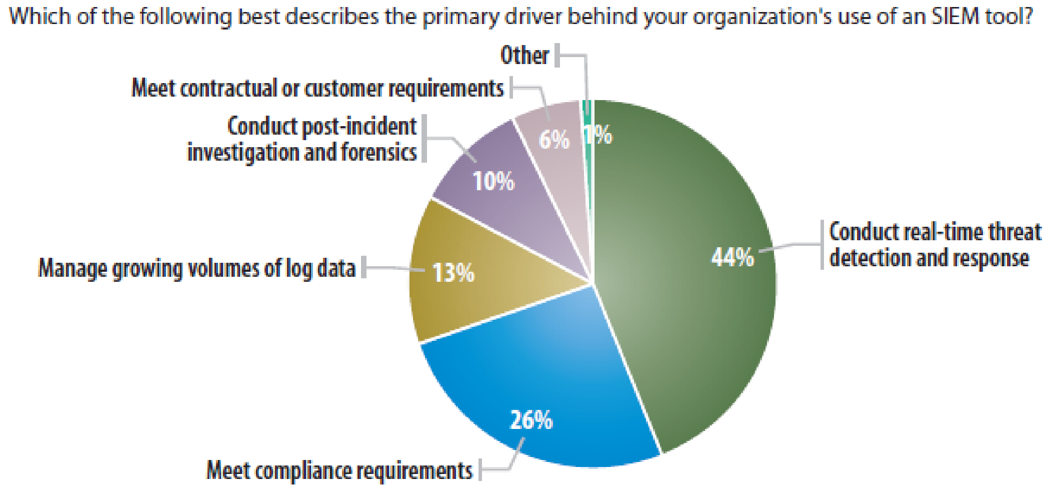


Figure 3.6: Factors that drive SIEM use (Francis, 2012)

Using the event data collected, SIEM systems detect potential attacks and policy violations and in case of any attack or violation, SIEM system assists the security operations teams to react quickly who aim to lessen the damage caused by the attack, to reduce the recovery time and, if possible, to defuse the threat at the beginning itself. In order to meet the compliance requirements, many organisations depend on SIEM systems which are capable of managing and reviewing log data up to a great extent. For example, companies that accept credit card payments or simply handle the card data need to meet the Payment Card Industry Data Security Standard (PCIDSS) or companies that deal with protected health information need to obey Healthy Insurance Portability and Accountability Act (HIPAA) which requires the respective organisation to review their log files on a daily basis (Francis, 2012).

Being a core security tool, SIEM systems extends its boundaries by integrating with other parts of the organizational infrastructure. Network and application configuration management, help desk and service products, performance management, identity and access management, network fault management are some of the tools that interoperate with SIEM based on the organisational requirements. With the objectives SIEM systems help to achieve and with the ability to integrate with other infrastructure tools, the prominence of security information and event management systems within an organisation is understandable.

SIEM systems were designed to act as a centralised repository of log and event data in order to modernise security incident management and compliance report within organisations. Presence of SIEM systems assists security analysts to focus more on resolving the security incidents instead of hunting for information from a large pool of data (RSA security brief, 2012). Table 3.6 lists the main capabilities and boundaries of these capabilities of traditional SIEM systems:

Table 3.6: SIEM system - Capabilities and Limitations

Capabilities	Limitations
Automates multiple functions – collection, analysis, archiving and reporting of data from log files and events from various sources such as network devices, firewalls, IDSs, servers, applications etc.	Over time, the diversity and volume of security data within an organization have increased and these factors limit the data handling capabilities of current SIEM products.
Provides a single point visibility of organization's data which is made possible by creating an integrated repository for security data allowing Security Operations Centre (SOC) analysts access these data in a centralized manner.	The visibility offered by SIEM systems depends on and limited to the log and event data collected from different sources within the organization and these sources of data make up only a small portion of the entire population of available resources which leaves the security analysts with only a small amount of potentially relevant activities.
In order to produce a complete security data warehouse, SIEM products combine log and event data from various sources.	Even though SIEM products collect a huge amount of log and event data, the quality of these data for being used in incident investigations is poor. SIEM tools may detect the presence of malwares, but they often fail to answer questions related to the scope, impact, business criticality and priority of the attack. SIEM systems also lack in providing sufficient amount of contextual data.
Generates different control reports which support up to a great extend in proving compliance with regulations set by the government and the industry.	Compliance reporting neither contributes towards regulating the security-related risks nor improves the security situation of the organization.
Offers an alerting mechanism for known threats using correlation rules and attack signatures.	Alert mechanism of SIEM systems are limited to signatures and attack methods that are known already. They do not possess signatures for advanced threats and are not capable of detecting attack methods that appears legitimate in nature.

SIEM is architected to detect and report the any bad behaviour in the network and it is the responsibility of the security analysts to conduct investigation and take responsive actions against the occurred security events. During the course of a security event at the SIEM level, it takes support from the evidences provided by the signature based detection systems and reports the event.

Two important external factors raising challenges for SIEM systems are certainly the increasing volume of security data and the number of attacks that are advanced and sophisticated in nature. Statistics show that “the amount of data analysed by enterprise information security organizations will double every year through 2016” (Gartner, 2012). Traditional SIEMs are incapable of detecting an attack that is well hidden under legitimate user credentials or under normal network traffic; it can raise a red flag only if there is a variation from the normal behaviour (Splunk, 2012).

SIEM systems have other internal challenges such as limitations with their databases, limited analytic capabilities and inability to perform historical analysis within itself. Traditional SIEMs uses relational databases which go under pressure by the increasing rates at which the log data being inserted and faces trouble in performing real-time correlation and historical reporting. For the same reason, many SIEM systems fail to execute real-time correlation and thereby failing to produce a true image of risk (McAfee, 2013). Analytic capabilities of SIEM systems are limited to static rules and known signatures and attack methods and the systems are incapable of analysing changing user behaviour and risk factors (RSA security brief, 2012). Inability to decide the reason for growing events per second (EPS) problem is another challenge persisting in many SIEM systems leaving the security analysts unable to attain situational awareness (McAfee, 2013). SIEM system architecture is not built to store historical event data and for this reason the system is unable to access any attack information of that kind for the purpose of assisting in current security data analysis (Splunk, 2012).

While having such limitations and challenges, the capability of SIEM systems for detecting and reporting security threats in an organization is up to a sufficient level and there is space for expansion. The proposed idea to meet the current challenges is by the application of advanced analytics of security data. SIEM system is preparing to equip itself with security analytics to handle the enormous volume of network traffic instead of not just handling the event and log data in order to reconstruct the attack session to understand the attack pattern. Also, acquiring threat intelligence from external sources is another technique SIEM systems is considering adopting and to redesign the user interface in order to incorporate security analytics. SIEM system is proposing to use the same security analytics to replace static correlation rules that the system is currently using with dynamic comparisons to analyse the change in behaviour.

3.6 Discussion

The potential that the network traffic has in being used as an alternative source of evidence in forensic investigation was established in Chapter 2. Whenever an organisational network is victimised by a cybercrime, the network traffic becomes useful from an investigation point of view. Understanding different investigation process models that are specifically designed for networked environments revealed that challenges exist within different investigation phases such as the detection, collection, incident response and the analysis phase. Two fundamental challenges that the NF investigation faces are the enormous volume and the raw state of data that traverse the networks. The quantity and complexity problems can limit the effectiveness of the detection, collection and analysis phases, and thus the overall effectiveness of the investigation. The investigation can be limited by the increasing usage of encryption by online services. The application layer encryption of data can limit the extraction of meaningful information. There may be scenarios in which certain tools are able to decrypt the data, provided the decryption key is available. For instance, using the RSA private key, Wireshark can decrypt TLS traffic when it is using the RSA algorithm (Wireshark, 2019). But this can increase the time and complexity of the investigation. These limitations extend into NFATs that are utilised for the investigation. Furthermore, they can increase the cognitive load on the investigators, both in terms of the time and effort needed from them.

Being a component utilised by the incident response team, SIEM systems have become a part of the organisational network. SIEM system, interconnected with other components of the organisational structure, attempts to perform real-time threat detection and acts as a proactive system trying to achieve similar goals as of network forensics. High cost, complexity, time taken for deployment, and noisy alerts are among the limitations of SIEM systems (Barraco, 2014). To overcome the limitations and challenges, research carried out within the SIEM domain is proposing to adapt security analytics and threat intelligence from external sources, thereby making the system more dynamic and preventive against network attacks. Even though refined SIEM system such as Splunk Enterprise has a growing demand, and enables the analysts to visualise network events, create interactive dashboards and generate reports, it still differs from the user-oriented NFAT that is being proposed in this research. While the former system is a proactive, the latter is a reactive in nature. The researcher is presenting a system that is capable of extracting user interactions, taking the investigation one step closer to the user. However, such SIEM systems can be beneficial for the investigation as they can be utilised as an additional source of useful data.

NFATs are not as sophisticated as the computer forensic tools such as FTK Imager and EnCase. There are functionalities exhibited by computer forensic tools that can be incorporated into NFATs while improving the existing features:

- Inclusion of 'case management' functionality in NFATs requires more attention as the same is available in computer forensic tools (e.g. ADLab from AccessData Corp.).
- More research is needed into developing open source NFATs that are capable of generating high-level information from raw network data to their full potential (e.g. PyFlag is capable of re-establishing the web pages, but not the dynamic ones).
- Implementation of calculating hash values for the extracted files and comparing them with the National Software Resource Library (NSRL) in order to reduce the volume of evidence to be analysed needs more focus (as this technique is already being used in computer forensic tools).
- Implementation of data correlation (both temporal and relational) techniques to find links between the generated high-level information, which have evidential value from an investigation point of view, requires more attention.
- Implementation of presentation techniques (documentation and visualisation of evidences) needs more focus, especially in open source NFATs.

Current NFATs have to deal with the enormous volume and the encrypted nature of raw network traffic. Also, investigators may have to depend on more than one tool to conduct the investigation. But this can be improved if there is an NFAT that is capable of performing the investigation based on the case management premise. For instance, a tool that can extract meaningful information such as online user activities and exhibit them in a visual manner such as a timeline can benefit the investigator. A tool that can assist in the investigation while maintaining the integrity and chain of custody, offer collaborative working, and that can generate case reports can reduce the cognitive load on the investigator.

3.7 Conclusion

Since identifying that network traffic data can be utilised as an alternative source of evidence, it was important to examine the network forensics - the domain that revolves around network traffic. The analysis of this domain was carried out by reviewing different investigation process models and some of the commercial and open source NFATs. Moreover, the existing challenges/limitations of the investigation were identified. Based on the study conducted into the field of network forensics, following observations are generated:

- The number of NFATs is limited compared to the available Computer Forensic tools, and the existing NFATs are not efficient enough to meet all the investigation requirements.
- Since a single tool cannot fulfil the NF investigation needs, investigators make use of more than one tool. This can include tools that are utilised for other purposes such as network security tools and SIEM systems.
- Enormous volume and raw state of network traffic can challenge different phases of the investigation (such as detection, collection and analysis), thereby making the investigation tasks more difficult.
- Lack of complete automation in NFATs and the human intervention it causes can limit the tools' efficiency.
- NFATs are not yet capable of performing the entire case management similar to some of the established computer forensic tools.

Analysis of the network forensic domain revealed that there is a need for generating meaningful information from raw network traffic, and, to achieve this goal, enormous volume of traffic has to be analysed in an efficient manner. It can be identified that the computer forensic domain is far more matured in comparison with network forensics and there is still room for more research in this field. More efficient NFATs can be developed by tackling the challenges and by mapping some of the features that are already implemented in computer forensic tools. At this juncture, it is important to analyse the prevailing methods that are used for analysing the network traffic, which the next Chapter will look into.

4 Identification of User Activities from Network Meta-data

4.1 Introduction

Based on the discussions made in Chapters 2 and 3, it is evident that the existing network analysis tools such as NetworkMiner, Xplico and PyFlag are not as sophisticated as the established computer forensic tools. The large volume and the encrypted nature of the network traffic pose more challenges for the investigation. These network analysis tools require an immense amount of cognitive input from the investigator. Naturally such factors would create hurdles for conducting the investigation in a timely fashion. Under these circumstances the investigation would end up demanding more time and money from the organisation.

In order to conduct the investigation effectively, it is essential for the investigator to have assistance from an efficient network forensic analysis tool. While assisting in the investigation, an ideal NFAT must be able to reduce the effort exerted by the investigator by reducing the cognitive load, reduce the time taken for the investigation process to complete or reduce the cost of the investigation. The research aims to achieve this through a novel network forensic analysis tool that is capable of generating high-level information from low-level network metadata, which will require a novel approach for network traffic analysis. Section 4.2 introduces the novel interaction based analysis approach that is capable of generating online user interactions from raw network metadata. Section 4.3 of the chapter discusses the methodology of the experiment that was designed to test the practicality of the hypothesised interaction based approach. The results obtained, which reassured the feasibility of the proposed approach are presented in Section 4.4, followed by the discussion and conclusion in Section 4.5 and Section 4.6 respectively.

4.2 A Novel Interaction-based Approach

Primarily, the packet based and the flow based analysis approaches are utilised for investigating network related issues and for the detection and prevention of different types of network attacks. Both approaches possess factors that make them desirable while having their own limitations. The research, however, is aimed at developing a system that has the capability to reduce the investigator’s effort by extracting information regarding the online user interactions through the analysis of low level network traffic metadata.

Identification of the online user interactions is crucial for reducing the cognitive load on the investigators. As mentioned above, this high-level information (i.e. the user interactions) will be generated by analysing the raw network metadata; generated within the organisational environment. Furthermore, there is a need for presenting the high-level information in a visual manner that is more cognitively manageable for the investigators. The existing approaches for analysing the network traffic (i.e. the packet based and flow based approaches) do not serve the purpose of identifying the user interactions and hence are not suitable candidates for achieving the research goal.

The reason that makes the flow based network analysis approach not suitable for the job is because of the possibility of a single flow containing more than one user interaction. And such a possibility would make the identification of user activities an impossible task. For instance, in TLS based connections once the client and the server establish the connection, the TLS session is left open for a period of time. A possibility of more than one user interaction occurring during this time duration and being part of a single flow does exist. To give an example, while accessing Facebook the user may have performed multiple activities such as posting a comment, chatting with a friend or sending an image via the chat window within a short duration of time. A flow record of this session will be containing metadata that comes from a combination of all of the aforementioned activities making the extraction of meaningful information (i.e. user interactions) using the flow based analysis approach impossible. The packet based analysis approach is also inappropriate as it is unable to offer the necessary abstraction that is needed to understand the nature of user interaction(s). The bit-by-bit comparison nature of header and payload data of each and every packet exerted by this approach, the difficulties in dealing with the enormous volume and the encrypted nature of network traffic will not the serve the notion of identifying the user activities which is central to the research hypothesis.

The lack of suitability exhibited by both packet based and flow based analysis approaches led the researcher to postulate a novel network traffic analysis method – the interaction based approach – that is capable of identifying the online user activities within an organisational network environment which would be beneficial for the digital forensic investigation.

The underlying principle behind the interaction based analysis approach is as follows. Users interact with many different Internet based applications performing various activities on a daily basis. These user interactions will result in the establishment of network connections and the traversing of network traffic between the hosts. These hosts can either be a client and a server or two clients on either end. Analysing the structure of such traffic metadata would reveal a unique pattern that may exist thereby leading to the identification of the corresponding user activity. For example, a user watching a video on YouTube via the web browser would establish a connection between the user's computer and the YouTube server. The activity will also generate a stream of packets travelling from the YouTube server to the user's computer. Detailed analysis of the generated network metadata would recognize the unique pattern and thus identifying the user interaction (i.e. watching the video). In the context of this proposed approach, the metadata will be a subset of the packet header data, and will consist of elements such as IP address, port numbers, network layer protocols, flags set for TCP/UDP protocols, timestamps, and packet size.

The postulated interaction based approach is central to the development of the UO-NFAT system, which is expected to assist in the investigation by identifying online user interactions. The primary application of the proposed system is going to be within an organisational environment where incidents such as insider threat/ misuse and policy violations could easily go unnoticed due to their un-detectable nature. In such scenarios the proposed UO-NFAT system, powered by the interaction based analysis approach, could provide the investigator with information about the online activities of the employees.

4.3 Experimental Methodology

Confirming whether the proposed interaction based approach is practically achievable was crucial for the continuation of the research. In order to test the hypothesis, an experiment that will investigate the relationship between the online user interactions (activities) and the corresponding network traffic (metadata) generated while using numerous Internet based services was structured. Different steps that were involved in the experiment are illustrated in Figure 4.1.

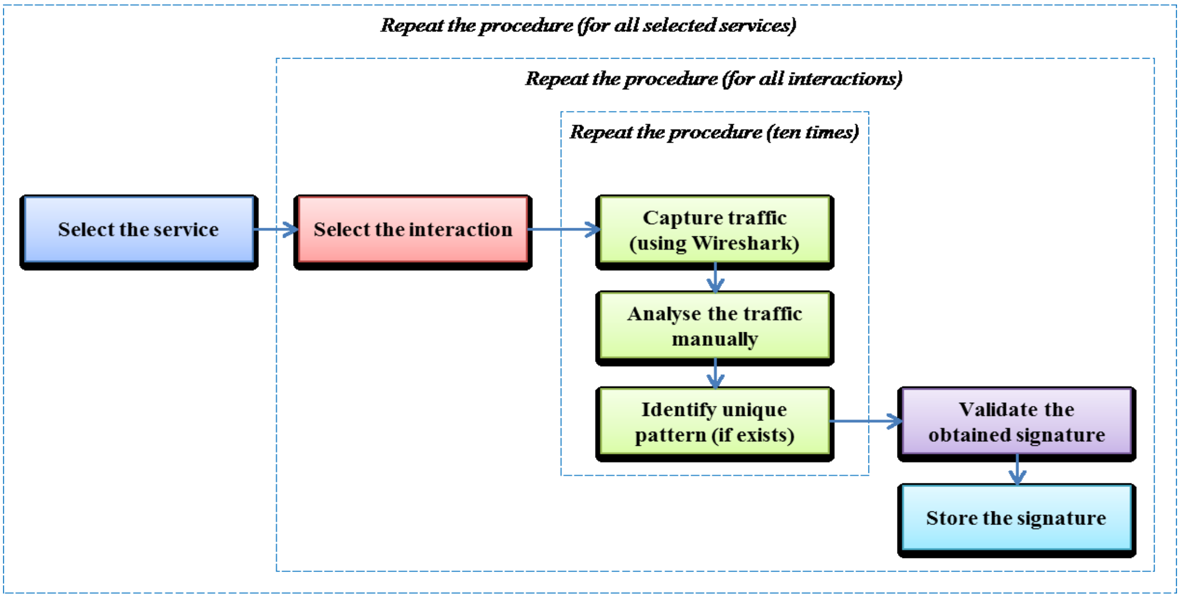

Repeat the procedure (for all selected services)
Repeat the procedure (for all interactions)
Repeat the procedure (ten times)
Select the service
Select the interaction
Capture traffic (using Wireshark)
Analyse the traffic manually
Identify unique pattern (if exists)
Validate the obtained signature
Store the signature

Figure 4.1: Experimental methodology for signature extraction

First step was to identify and select the Internet based services that were going to be included in the experiment. Nine different services were selected and all possible user interactions within these Internet services were identified. The primary reasons for selecting these nine services were their popularity and the fact that they belong to a group of Internet based services that is the biggest contributor of online traffic (Alexa, 2017). The best way to acquire information about the online activities of a user is by analysing the network traffic (metadata) generated by the top applications that represent a significant volume of the user's online traffic. Because this information is an outcome of an analysis that was performed on the major part of a user's online traffic, it could turn out to be useful for the investigation process. The chosen services and the corresponding possible user interactions are listed in Table 4.1.

Table 4.1: Internet based services and possible user interactions

Internet based services	Popular user interactions
BBC	Page navigation, watching video clips/television programs, listening to audio clips/radio, comment, news sharing
Dropbox	Download/upload files, share files/folders, folder navigations
eBay	Product browsing, payment transaction, product review writing
Facebook	Post, like, comment, share, find friends, attach files, chat, typing message
Google Docs	Create/edit/share/delete online documents

Google Search	Keyword search, page navigation
Hotmail	Compose/reply/forward/delete an email, insert recipients, read email, upload/download file attachments
Skype	Send text message, file transfer, click on contacts, audio/video call, change online status
Twitter	Tweeting, uploading image
Wikipedia	Search, read/modify an page content, upload media files
YouTube	Search, watch videos, listen to songs, upload audio/video files, like, dislike, comment, share, subscribe

The next part (step 2) of the experiment was intended to identify whether or not a unique signature within the network metadata was detected when each and every identified user activity was performed. The second step also included storing the parameters that determined the signature, if and when one was detected.

This fragment of the experiment was conducted manually in which Wireshark was used to capture the traffic. The experiment was designed with the intention to capture traffic only from a computer that was connected to the local area network and had access to the Internet via the Ethernet. A desktop computer with web browser(s) installed was used for the experiment. It was through the web browser that the selected online services were accessed and interacted with, except for the Skype. In order to interact with the Skype service, the installed Skype application was utilised. While Wireshark was running in the background, the researcher accessed a particular service and performed a certain interaction and the resultant traffic was captured by the tool. Following this, the captured traffic underwent manual analysis in order to identify the signature, if any. The fundamental nature of network communication between hosts is in such a way that the network packets travel in both directions (i.e. bidirectional traffic). When it comes to identifying the unique signature, the nature of traffic returning from the Internet based service is as equally important as the nature of traffic leaving from the user’s computer. Hence both directions of network traffic were included in the analysis.

While conducting the experiment, certain parameters of the network metadata were utilised for analysis. These parameters are the date and time stamp, the IP addresses of the source and destination, the port numbers used by the source and destination, the type of transport layer protocol (either TCP or UDP), the length of the datagram and TCP flags (e.g., SYN, ACK, PUSH and FIN).

The IP address of the server that delivers an Internet based services is one of the parameters used for finding the signature and play an important role in the identification of user activities. The web services included in this experiment are the ones that offer their services over version number 4 of the protocol (IPv4) and not the latest version (IPv6). Although historical trends show that, as of January 2018, 11.5% of the total number of websites and 28.3% of the websites that are ranked in the top 1000 use IPv6 (W3Techs, 2018), majority of the websites still offer services over IPv4, which led the experiment to opt for the IPv4. The port numbers used by the user's computer (client) and the server that provides the service that is being used while a certain user interaction occurs is another parameter that was considered. There are fixed port numbers utilised by the transport layer protocols (TCP/UDP) for specific purposes. For instance, a web server that offers its services over a secure communication channel makes use of port number 443 whereas a web server that uses unencrypted data uses port number 80. And such details are considered to be contributing towards the identification of user interactions. The type of transport layer protocol used for data transmission was another important parameter used for the metadata analysis. The experiment included services that use both connection oriented protocol (TCP) (e.g., Facebook, Google, YouTube etc.) and connectionless protocol (UDP) (e.g., Skype) as the transport layer protocols for transmitting data.

These parameters are among the prime features of a packet that traverse the network and are important in understanding about the behaviour of the network that these packets constitute. Therefore, the research decided to choose the aforementioned parameters under the assumption that these elements of the low level network metadata of this kind could be extremely valuable in extracting information about the online activities that the users are involved in.

Accessing the service and performing the interaction was repeated at least ten times capturing the resulting traffic on each occasion. This was done with the intention of strengthening the verification and making the result robust. Traffic for certain interactions such as chatting on Facebook and Skype were captured more than ten times as it was necessary to investigate the role of the length of the message on the network traffic produced. This whole step was performed repeatedly for all possible interactions associated with the selected Internet based services. Besides, this step of the experiment was carried out by two other fellow researchers in an independent manner (without disclosing any results of the observations to the other researchers) thereby strengthening the experiment. Once all three researchers completed the manual analysis task, the results were cross checked for validation.

4.4 Results

As mentioned in the previous section, analysis of the network metadata generated as a result of the user interacting with the selected Internet based services was conducted manually. The observations and results obtained are discussed in this section.

It was observed during each attempt of the user accessing a particular Internet based service that the traffic received by the client's computer is, in fact, a mixture of data that is coming from multiple IP sources. These multiple IP sources could be a part of the Content Delivery Network (CDN) that deliver contents, such as JavaScript, Cascaded Style Sheets (CSS), image files (such as JPEG, GIF or PNG), embedded video/audio clips, downloadable software and live streaming media, that the requested webpage requires for rendering properly. The requested website could incorporate links to IP sources that play a role in performing web analytics in the background and IP addresses that promote advertisements as well. For the identification of user interactions, it was necessary to exclude the host IP addresses that cater the aforementioned sources (i.e. JavaScript, CSS, image files, web analytics and advertisements) and focus on the packets that belong to the principal IP address that offers the online service because the activity signatures identified by the interaction based approach are based upon the core user interactions.

Furthermore, while being accessed on different occasions, most of the selected services were observed to be transmitting data from more than one dedicated IP address. This occurs as a result of a strategy used by the respective companies (i.e. storing copies of their data on multiple servers) for load balancing and operational ease. This finding led the researcher into realising that in order to separate the traffic that may contain the interaction signature, it is essential to have the range of principal IP addresses that these servers could operate from. As mentioned in methodology, the IP addresses of the source and destination hosts are among the parameters that contribute towards the identification of the interaction signatures. These observations resulted in extracting the range of principal IP addresses that each Internet based service was operating on, which was beneficial for the later analysis of a larger dataset as well. The detected IP address ranges for the servers that host each of the nine services are listed in Table 4.2.

Table 4.2: IP address range for selected Internet based applications

Services	IP Address Range	
BBC	212.58.244.xxx	212.58.246.xxx
Dropbox	108.160.172.xxx 108.160.165.xxx	108.160.165.xxx 162.125.64.xxx
Facebook	31.13.90.xxx	
Google Docs/ Google Search/ YouTube	216.58.208.xxx 216.58.210.xxx 173.194.78.xxx 173.194.20.xxx 173.194.135.xxx	74.125.133.xxx 74.125.230.xxx 74.125.195.xxx 74.125.214.xxx 74.125.174.xxx
Hotmail	204.79.197.xxx	157.56.122.xxx
Skype	157.56.193.xxx 157.56.192.xxx	157.56.126.xxx 13.79.153.xxx
Twitter	185.45.5.xxx	199.96.57.xxx
Wikipedia	91.198.174.xxx	

It can be noticed from the table that most of the services have more than one server; especially service such as Google, Skype and YouTube, which were transmitting data from several servers. As mentioned above, identifying such a range of server IP addresses for the selected Internet based services was an important result and it is safe to assume that similar Internet services that are not included in this research could display the same behaviour.

The manual analysis was performed on all popular user interactions, as shown in Table 4.1, that the selected services had to offer. However, unique patterns were not observed for some of the interactions as any such patterns were non-existent for those interactions. The services and the corresponding user interactions that successfully gave unique/identifiable signatures are listed in Table 4.3 followed by the explanation of the patterns obtained for the listed user activities.

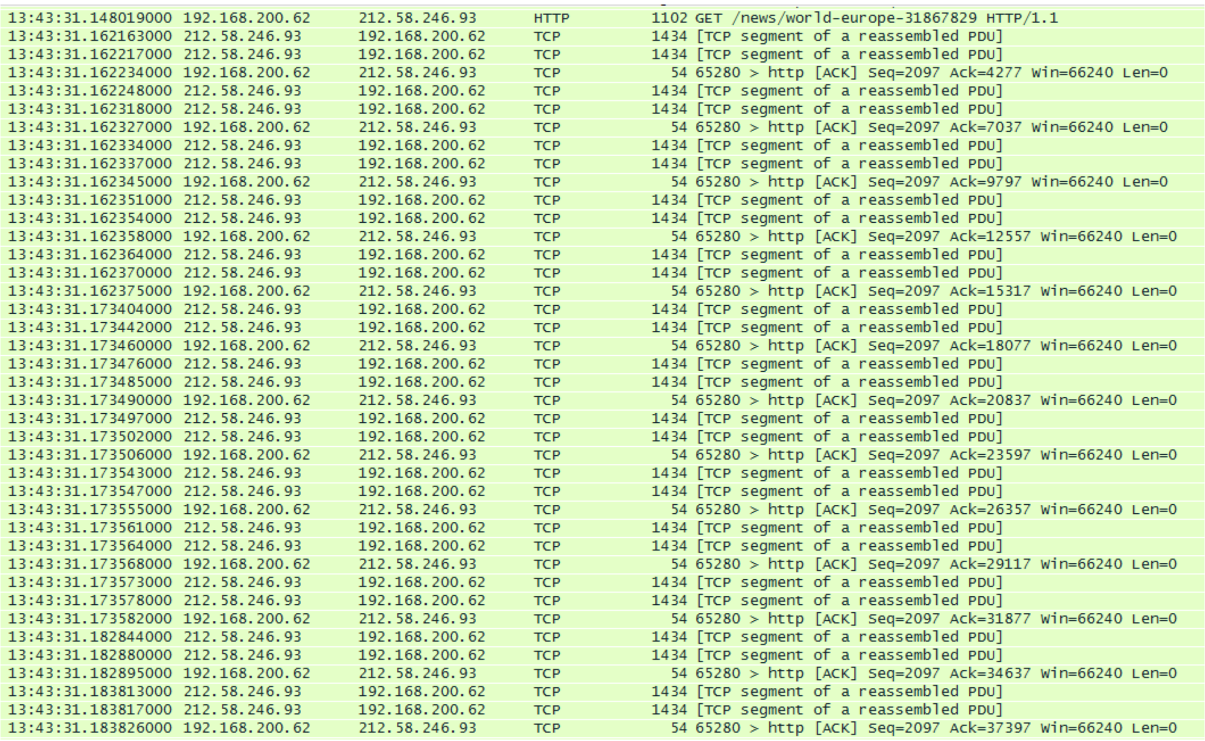
Table 4.3: Internet based services and identified user interactions

Services	User interactions			
BBC	Page navigation	Watching video	Listening to radio	
Dropbox	Downloading file	Uploading file		
Facebook	Page loading	Attaching file	Typing	Chat
Hotmail	File attachments	Composing email	Inserting recipient	
Google Docs	Editing document			
Google Search	Page navigation			
Skype	Text messages	Audio call	Video call	
	File transfer	Image transfer	Click on contacts	
		Idle		
Twitter	Page loading	Uploading photo		
Wikipedia	Page loading			
YouTube	Watching video	Uploading video		

4.4.1 BBC

BBC.co.uk is the news and media website owned by the British Broadcasting Corporation and offers its users information regarding the latest news, sports headlines, TV & radio and weather. As of the 1st of January 2018, 'bbc.co.uk' is ranked 7th among the top websites and 1st among the top news and media websites in the United Kingdom, based on the web traffic generated by the website. Web analytics shows that the BBC website has had 689 Million visits in January 2018 (SimilarWeb, 2018). Among the popular user activities that occurs on this website, the activities that were identified via analysing the network metadata are – page navigation, watching video and listening to radio.

When the user was navigating through the web pages, the resulting traffic was obtained as a stream of packets traversing from the server to the user's computer, with the size of each packet being close to the Maximum Transmission Unit (MTU) value. Screenshot illustrated in Figure 4.2 shows an example of the stream of packets that resulted from the page navigation activity.

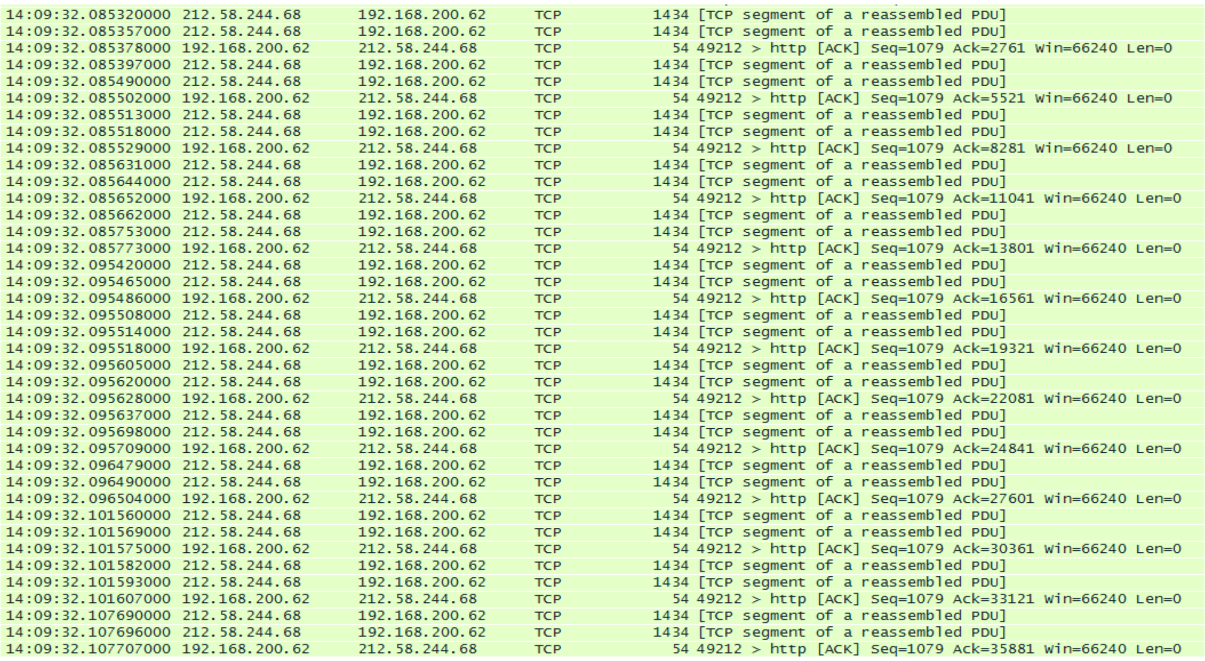


```
13:43:31.148019000 192.168.200.62   212.58.246.93    HTTP      1102 GET /news/world-europe-31867829 HTTP/1.1
13:43:31.162163000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162217000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162234000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=4277 Win=66240 Len=0
13:43:31.162248000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162318000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162327000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=7037 Win=66240 Len=0
13:43:31.162334000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162337000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162345000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=9797 Win=66240 Len=0
13:43:31.162351000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162354000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162358000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=12557 Win=66240 Len=0
13:43:31.162364000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162370000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.162375000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=15317 Win=66240 Len=0
13:43:31.173404000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173442000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173460000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=18077 Win=66240 Len=0
13:43:31.173476000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173485000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173490000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=20837 Win=66240 Len=0
13:43:31.173497000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173502000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173506000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=23597 Win=66240 Len=0
13:43:31.173543000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173547000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173555000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=26357 Win=66240 Len=0
13:43:31.173561000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173564000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173568000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=29117 Win=66240 Len=0
13:43:31.173573000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173578000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.173582000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=31877 Win=66240 Len=0
13:43:31.182844000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.182880000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.182895000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=34637 Win=66240 Len=0
13:43:31.183813000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.183817000 212.58.246.93    192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
13:43:31.183826000 192.168.200.62   212.58.246.93    TCP         54 65280 > http [ACK] Seq=2097 Ack=37397 Win=66240 Len=0
```

Figure 4.2: Page navigation on BBC website

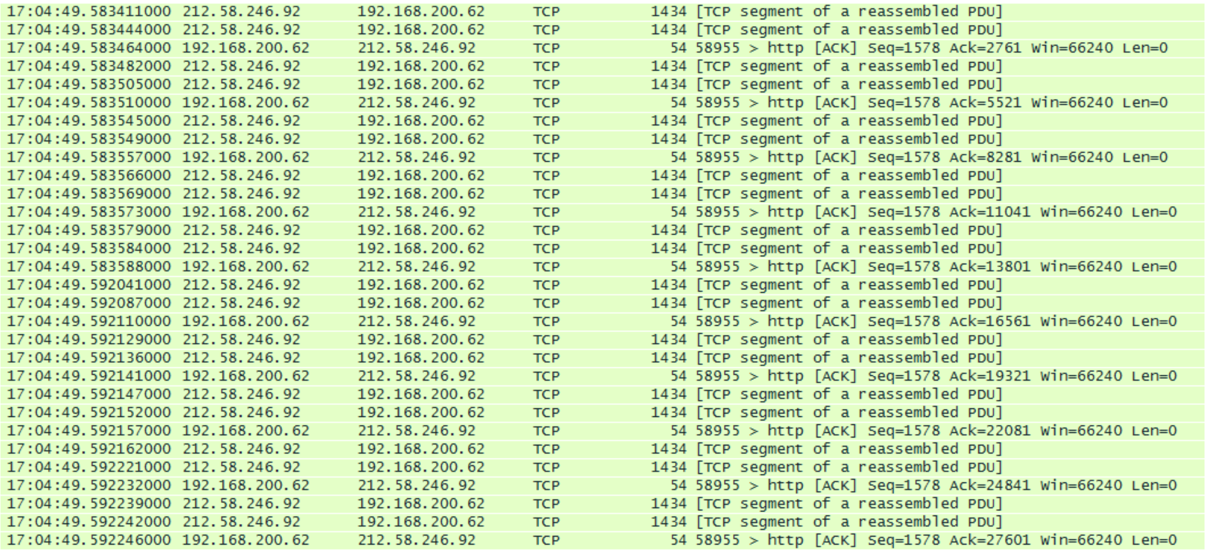
From the screenshot, it can be noticed that the MTU sized TCP packets (with ACK flag set) are transmitted from the HTTP port (80) of the server and in return, TCP acknowledgement packets (with ACK flag set) are sent back to the server from the user's computer.

In a similar fashion, the remaining activities (i.e. watching video and listening to the radio) resulted in a stream of packets that are sent from the server and TCP acknowledgment packets are retuned. Examples of the resulting packets captured for watching video and listening to radio activities are shown in Figure 4.3 and Figure 4.4 respectively.



```
14:09:32.085320000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085357000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085378000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=2761 Win=66240 Len=0
14:09:32.085397000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085490000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085502000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=5521 Win=66240 Len=0
14:09:32.085513000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085518000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085529000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=8281 Win=66240 Len=0
14:09:32.085631000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085644000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085652000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=11041 Win=66240 Len=0
14:09:32.085662000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085753000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.085773000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=13801 Win=66240 Len=0
14:09:32.095420000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095465000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095486000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=16561 Win=66240 Len=0
14:09:32.095508000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095514000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095518000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=19321 Win=66240 Len=0
14:09:32.095605000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095620000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095628000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=22081 Win=66240 Len=0
14:09:32.095637000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095698000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.095709000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=24841 Win=66240 Len=0
14:09:32.096479000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.096490000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.096504000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=27601 Win=66240 Len=0
14:09:32.101560000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.101569000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.101575000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=30361 Win=66240 Len=0
14:09:32.101582000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.101593000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.101607000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=33121 Win=66240 Len=0
14:09:32.107690000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.107696000 212.58.244.68   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
14:09:32.107707000 192.168.200.62  212.58.244.68    TCP     54 49212 > http [ACK] Seq=1079 Ack=35881 Win=66240 Len=0
```

Figure 4.3: Watching video on BBC website



```
17:04:49.583411000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583444000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583464000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=2761 Win=66240 Len=0
17:04:49.583482000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583505000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583510000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=5521 Win=66240 Len=0
17:04:49.583545000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583549000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583557000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=8281 Win=66240 Len=0
17:04:49.583566000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583569000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583573000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=11041 Win=66240 Len=0
17:04:49.583579000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583584000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.583588000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=13801 Win=66240 Len=0
17:04:49.592041000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592087000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592110000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=16561 Win=66240 Len=0
17:04:49.592129000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592136000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592141000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=19321 Win=66240 Len=0
17:04:49.592147000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592152000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592157000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=22081 Win=66240 Len=0
17:04:49.592162000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592221000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592232000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=24841 Win=66240 Len=0
17:04:49.592239000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592242000 212.58.246.92   192.168.200.62   TCP   1434 [TCP segment of a reassembled PDU]
17:04:49.592246000 192.168.200.62  212.58.246.92    TCP     54 58955 > http [ACK] Seq=1578 Ack=27601 Win=66240 Len=0
```

Figure 4.4: Listening to radio on BBC website

Key characteristics of the identified user interactions for the BBC website are listed in Table 4.4.

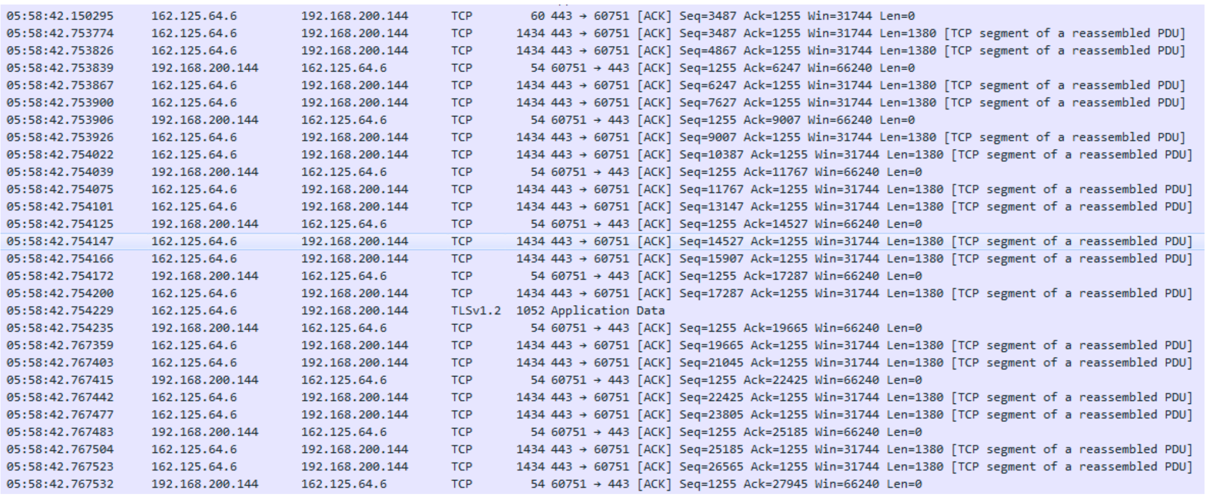
Table 4.4: BBC user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Page navigation	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Watching video	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Listening to radio	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)

4.4.2 Dropbox

Dropbox is a modern day online storage space that allows the users to access and share important files from multiple locations and devices. Being one among the top 100 websites in the United States and top 200 websites in the world as of January 2018, Dropbox has majority of its traffic generated from the United States and the United Kingdom; 27.95% and 5.05% respectively (SimilarWeb, 2018). It should be highlighted that it was the Dropbox website that was selected for the experiment, not the desktop application. Staying connected to the Internet was a criterion that was set in the experimental methodology, which is necessary for capturing the generated network traffic metadata. Since the Dropbox desktop application is designed in such a way that it can be used both online and offline (Dropbox, 2018), need for a constant Internet access is only optional thereby making the desktop application unsuitable for the experiment. Hence the decision to choose the Dropbox website was made.

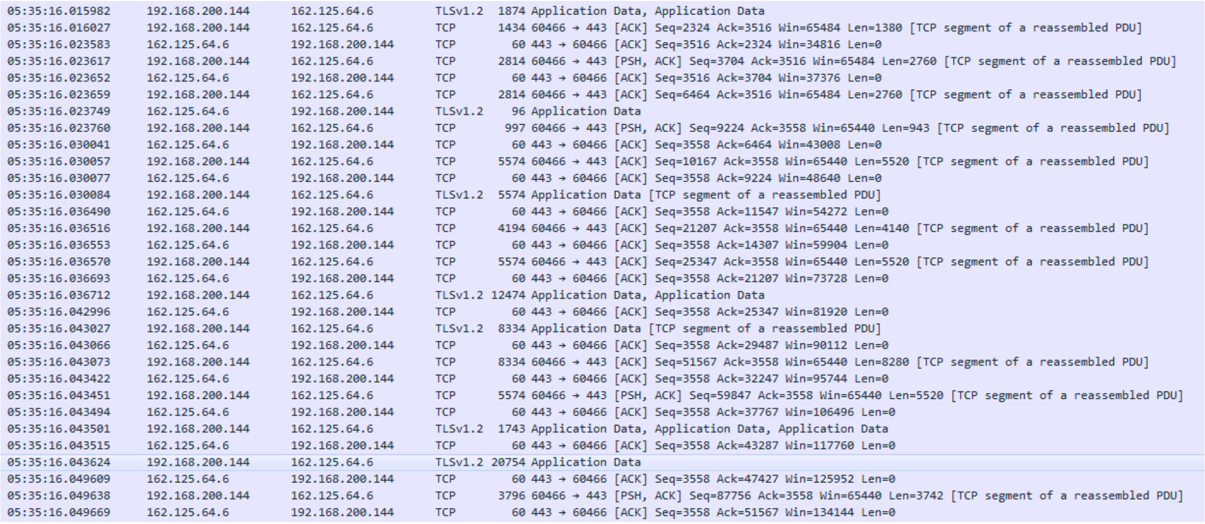
The main activities that the users perform are downloading from and uploading to the Dropbox. When a file was downloaded, a stream of packets was observed to be transmitting from the server through port number 443 and the size of the packets was approximately equal to the MTU value. An example of the packets that resulted from the downloading activity is shown in Figure 4.5. The transport layer protocol used in this activity was TCP and the ACK and PSH were the TCP flags that were set.



```
05:58:42.150295   162.125.64.6     192.168.200.144   TCP       60 443 → 60751 [ACK] Seq=3487 Ack=1255 Win=31744 Len=0
05:58:42.753774   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=3487 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.753826   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=4867 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.753839   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=6247 Win=66240 Len=0
05:58:42.753867   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=6247 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.753900   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=7627 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.753906   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=9007 Win=66240 Len=0
05:58:42.753926   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=9007 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754022   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=10387 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754039   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=11767 Win=66240 Len=0
05:58:42.754075   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=11767 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754101   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=13147 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754125   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=14527 Win=66240 Len=0
05:58:42.754147   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=14527 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754166   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=15907 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754172   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=17287 Win=66240 Len=0
05:58:42.754200   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=17287 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.754229   162.125.64.6     192.168.200.144   TLSv1.2 1052 Application Data
05:58:42.754235   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=19665 Win=66240 Len=0
05:58:42.767359   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=19665 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767403   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=21045 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767415   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=22425 Win=66240 Len=0
05:58:42.767442   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=22425 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767477   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=23805 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767483   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=25185 Win=66240 Len=0
05:58:42.767504   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=25185 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767523   162.125.64.6     192.168.200.144   TCP     1434 443 → 60751 [ACK] Seq=26565 Ack=1255 Win=31744 Len=1380 [TCP segment of a reassembled PDU]
05:58:42.767532   192.168.200.144  162.125.64.6      TCP       54 60751 → 443 [ACK] Seq=1255 Ack=27945 Win=66240 Len=0
```

Figure 4.5: Downloading file from Dropbox

Conversely, when the user was uploading a file, the main direction of packet flow was from the user's computer to the server. The communication was over a secure connection using port number 443 on the server side. The packet size was close to the MTU value and acknowledgement packets were returned from the server. Figure 4.6 shows an example of the traffic that was generated as a result of the uploading activity.



```
05:35:16.015982   192.168.200.144  162.125.64.6      TLSv1.2  1874 Application Data, Application Data
05:35:16.016027   192.168.200.144  162.125.64.6      TCP      1434 60466 → 443 [ACK] Seq=2324 Ack=3516 Win=65484 Len=1380 [TCP segment of a reassembled PDU]
05:35:16.023583   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3516 Ack=2324 Win=34816 Len=0
05:35:16.023617   192.168.200.144  162.125.64.6      TCP      2814 60466 → 443 [PSH, ACK] Seq=3704 Ack=3516 Win=65484 Len=2760 [TCP segment of a reassembled PDU]
05:35:16.023652   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3516 Ack=3704 Win=37376 Len=0
05:35:16.023659   192.168.200.144  162.125.64.6      TCP      2814 60466 → 443 [ACK] Seq=6464 Ack=3516 Win=65484 Len=2760 [TCP segment of a reassembled PDU]
05:35:16.023749   162.125.64.6     192.168.200.144   TLSv1.2    96 Application Data
05:35:16.023760   192.168.200.144  162.125.64.6      TCP       997 60466 → 443 [PSH, ACK] Seq=9224 Ack=3558 Win=65440 Len=943 [TCP segment of a reassembled PDU]
05:35:16.030041   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=6464 Win=43008 Len=0
05:35:16.030057   192.168.200.144  162.125.64.6      TCP      5574 60466 → 443 [ACK] Seq=10167 Ack=3558 Win=65440 Len=5520 [TCP segment of a reassembled PDU]
05:35:16.030077   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=9224 Win=48640 Len=0
05:35:16.030084   192.168.200.144  162.125.64.6      TLSv1.2  5574 Application Data [TCP segment of a reassembled PDU]
05:35:16.036490   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=11547 Win=54272 Len=0
05:35:16.036516   192.168.200.144  162.125.64.6      TCP      4194 60466 → 443 [ACK] Seq=21207 Ack=3558 Win=65440 Len=4140 [TCP segment of a reassembled PDU]
05:35:16.036553   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=14307 Win=59904 Len=0
05:35:16.036570   192.168.200.144  162.125.64.6      TCP      5574 60466 → 443 [ACK] Seq=25347 Ack=3558 Win=65440 Len=5520 [TCP segment of a reassembled PDU]
05:35:16.036693   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=21207 Win=73728 Len=0
05:35:16.036712   192.168.200.144  162.125.64.6      TLSv1.2 12474 Application Data, Application Data
05:35:16.042996   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=25347 Win=81920 Len=0
05:35:16.043027   192.168.200.144  162.125.64.6      TLSv1.2  8334 Application Data [TCP segment of a reassembled PDU]
05:35:16.043066   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=29487 Win=90112 Len=0
05:35:16.043073   192.168.200.144  162.125.64.6      TCP      8334 60466 → 443 [ACK] Seq=51567 Ack=3558 Win=65440 Len=8280 [TCP segment of a reassembled PDU]
05:35:16.043422   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=32247 Win=95744 Len=0
05:35:16.043451   192.168.200.144  162.125.64.6      TCP      5574 60466 → 443 [PSH, ACK] Seq=59847 Ack=3558 Win=65440 Len=5520 [TCP segment of a reassembled PDU]
05:35:16.043494   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=37767 Win=106496 Len=0
05:35:16.043501   192.168.200.144  162.125.64.6      TLSv1.2  1743 Application Data, Application Data, Application Data
05:35:16.043515   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=43287 Win=117760 Len=0
05:35:16.043624   192.168.200.144  162.125.64.6      TLSv1.2 20754 Application Data
05:35:16.049609   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=47427 Win=125952 Len=0
05:35:16.049638   192.168.200.144  162.125.64.6      TCP      3796 60466 → 443 [PSH, ACK] Seq=87756 Ack=3558 Win=65440 Len=3742 [TCP segment of a reassembled PDU]
05:35:16.049669   162.125.64.6     192.168.200.144   TCP        60 443 → 60466 [ACK] Seq=3558 Ack=51567 Win=134144 Len=0
```

Figure 4.6: Uploading file to Dropbox

The key characteristics of the identified user activities are recorded in Table 4.5.

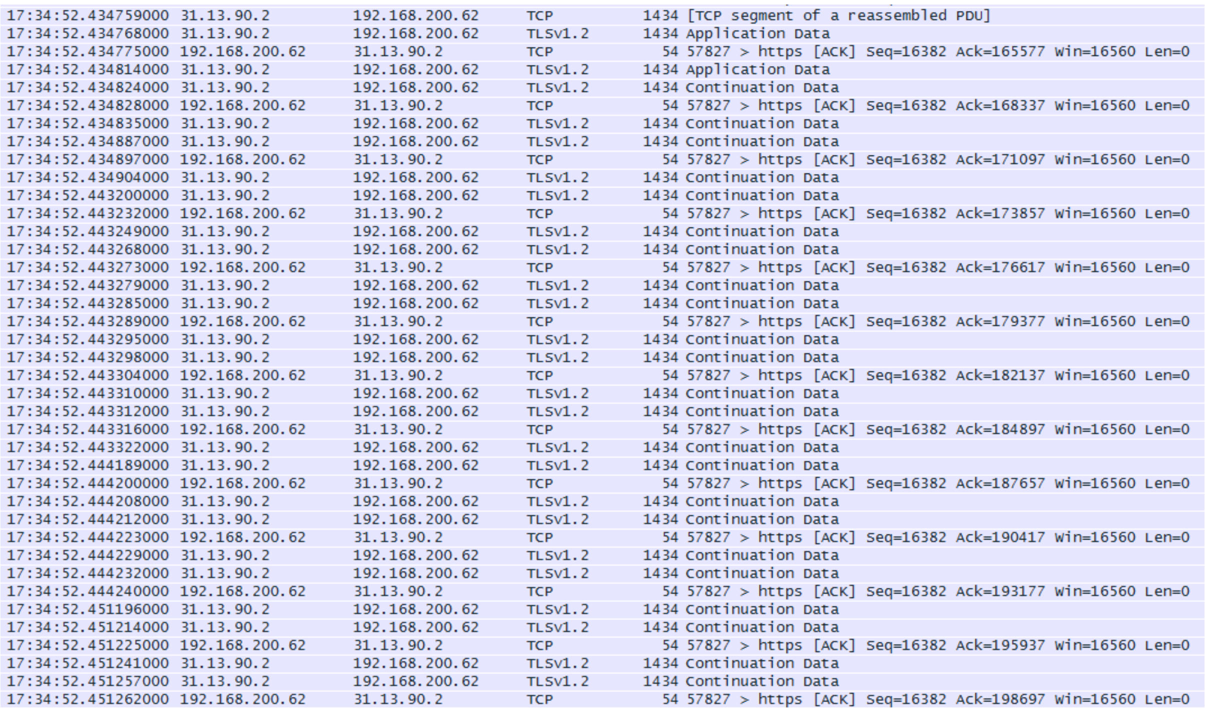
Table 4.5: Dropbox user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Downloading file	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Uploading file	Stream	MTU (approximate)	Client → Server	TCP (ACK, PSH/ACK)

4.4.3 Facebook

Facebook is the social media and social networking platform that allows its users to connect with friends and family, send/receive messages, post, like, share and comment on photos and videos and so much more. Based on the web analysis, as of January 2018, Facebook is 2nd among the top ranked websites in the world and in the United States. In the United Kingdom, Facebook is ranked number 3 based on the generated traffic. Facebook has had 31 Billion visits in January 2018 alone. The analysis shows that, on average, the users spends almost 14 minutes (13 minutes and 44 seconds) each time they visit Facebook (SimilarWeb, 2018). This is an ample amount of time for engaging in various activities, given the total number of visits and the average time spent on each visit.

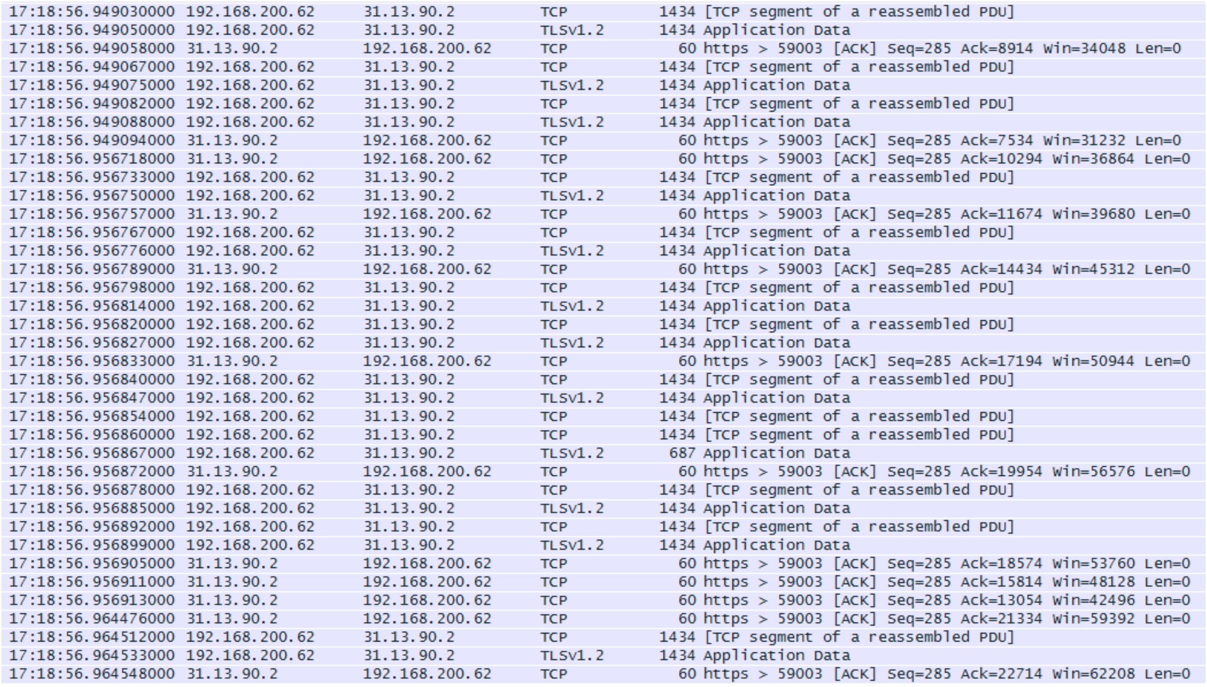
Each and every time when the user was involved in loading the main page, the server transmitted a stream of packets through port number 443 to the user's computer. TCP acknowledgement packets with the ACK flag set were sent back to the server. Data packets transferred from the server had either ACK or PSH and ACK flags set and had a size that is close to the MTU value. An example of the traffic that is responsible for the page loading activity is shown in Figure 4.7.



```
17:34:52.434759000 31.13.90.2       192.168.200.62   TCP       1434 [TCP segment of a reassembled PDU]
17:34:52.434768000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Application Data
17:34:52.434775000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=165577 Win=16560 Len=0
17:34:52.434814000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Application Data
17:34:52.434824000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.434828000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=168337 Win=16560 Len=0
17:34:52.434835000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.434887000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.434897000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=171097 Win=16560 Len=0
17:34:52.434904000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443200000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443232000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=173857 Win=16560 Len=0
17:34:52.443249000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443268000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443273000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=176617 Win=16560 Len=0
17:34:52.443279000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443285000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443289000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=179377 Win=16560 Len=0
17:34:52.443295000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443298000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443304000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=182137 Win=16560 Len=0
17:34:52.443310000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443312000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.443316000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=184897 Win=16560 Len=0
17:34:52.443322000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444189000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444200000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=187657 Win=16560 Len=0
17:34:52.444208000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444212000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444223000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=190417 Win=16560 Len=0
17:34:52.444229000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444232000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.444240000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=193177 Win=16560 Len=0
17:34:52.451196000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.451214000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.451225000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=195937 Win=16560 Len=0
17:34:52.451241000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.451257000 31.13.90.2       192.168.200.62   TLSv1.2   1434 Continuation Data
17:34:52.451262000 192.168.200.62   31.13.90.2       TCP         54 57827 > https [ACK] Seq=16382 Ack=198697 Win=16560 Len=0
```

Figure 4.7: Facebook page loading

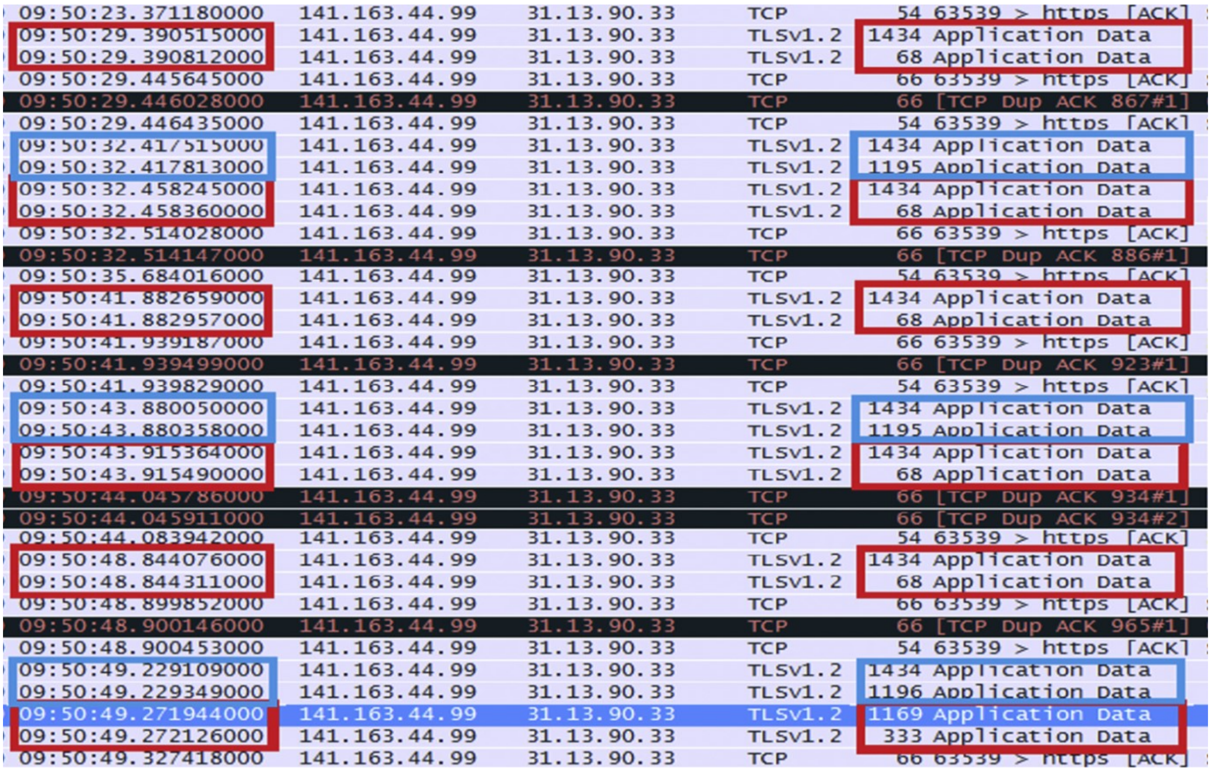
Similarly, another pattern was observed when the user was attaching a file through the chat window which turned out to be the signature for the activity. A stream of packets was transferred from the user's computer to port number 443 of the Facebook server. All packets had the size that is almost equal to the MTU value and had either ACK or PSH and ACK flags set for the TCP protocol used in the transport layer. In return, the server acknowledged with TCP packets that had the ACK flag set. Figure 4.8 shows the traffic pattern emerged when the user attached a file via the chat window.



```
17:18:56.949030000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.949050000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.949058000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=8914 Win=34048 Len=0
17:18:56.949067000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.949075000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.949082000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.949088000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.949094000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=7534 Win=31232 Len=0
17:18:56.956718000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=10294 Win=36864 Len=0
17:18:56.956733000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956750000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956757000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=11674 Win=39680 Len=0
17:18:56.956767000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956776000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956789000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=14434 Win=45312 Len=0
17:18:56.956798000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956814000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956820000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956827000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956833000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=17194 Win=50944 Len=0
17:18:56.956840000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956847000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956854000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956860000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956867000 192.168.200.62     31.13.90.2          TLSv1.2       687 Application Data
17:18:56.956872000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=19954 Win=56576 Len=0
17:18:56.956878000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956885000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956892000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.956899000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.956905000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=18574 Win=53760 Len=0
17:18:56.956911000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=15814 Win=48128 Len=0
17:18:56.956913000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=13054 Win=42496 Len=0
17:18:56.964476000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=21334 Win=59392 Len=0
17:18:56.964512000 192.168.200.62     31.13.90.2          TCP          1434 [TCP segment of a reassembled PDU]
17:18:56.964533000 192.168.200.62     31.13.90.2          TLSv1.2      1434 Application Data
17:18:56.964548000 31.13.90.2         192.168.200.62      TCP            60 https > 59003 [ACK] Seq=285 Ack=22714 Win=62208 Len=0
```

Figure 4.8: Attaching image in Facebook chat

A third activity pattern obtained was for when the user was typing a message in to the chat window. It was observed that two TCP packets with a total size of 1502 bytes were transmitted from the user's computer to the server within a time duration of less than a millisecond. In Figure 4.9, two packet with sizes 1434 bytes and 68 bytes can be noticed, which were the ones transmitted during the typing activity. But on repeating the activity, the packet sizes changed (1169 bytes and 333 bytes), yet the total size remained 1502 bytes. This meant that it is the combined value of 1502 bytes that needs to be focused upon while identifying the typing activity. Both packets had PSH and ACK flags set in the transport layer.



```
09:50:23.371180000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:29.390515000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:29.390812000  141.163.44.99  31.13.90.33  TLSv1.2    68 Application Data
09:50:29.445645000  141.163.44.99  31.13.90.33  TCP      66 63539 > https [ACK]
09:50:29.446028000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 867#1]
09:50:29.446435000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:32.417515000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:32.417813000  141.163.44.99  31.13.90.33  TLSv1.2  1195 Application Data
09:50:32.458245000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:32.458360000  141.163.44.99  31.13.90.33  TLSv1.2    68 Application Data
09:50:32.514028000  141.163.44.99  31.13.90.33  TCP      66 63539 > https [ACK]
09:50:32.514147000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 886#1]
09:50:35.684016000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:41.882659000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:41.882957000  141.163.44.99  31.13.90.33  TLSv1.2    68 Application Data
09:50:41.939187000  141.163.44.99  31.13.90.33  TCP      66 63539 > https [ACK]
09:50:41.939499000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 923#1]
09:50:41.939829000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:43.880050000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:43.880358000  141.163.44.99  31.13.90.33  TLSv1.2  1195 Application Data
09:50:43.915364000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:43.915490000  141.163.44.99  31.13.90.33  TLSv1.2    68 Application Data
09:50:44.045786000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 934#1]
09:50:44.045911000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 934#2]
09:50:44.083942000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:48.844076000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:48.844311000  141.163.44.99  31.13.90.33  TLSv1.2    68 Application Data
09:50:48.899852000  141.163.44.99  31.13.90.33  TCP      66 63539 > https [ACK]
09:50:48.900146000  141.163.44.99  31.13.90.33  TCP      66 [TCP Dup ACK 965#1]
09:50:48.900453000  141.163.44.99  31.13.90.33  TCP      54 63539 > https [ACK]
09:50:49.229109000  141.163.44.99  31.13.90.33  TLSv1.2  1434 Application Data
09:50:49.229349000  141.163.44.99  31.13.90.33  TLSv1.2  1196 Application Data
09:50:49.271944000  141.163.44.99  31.13.90.33  TLSv1.2  1169 Application Data
09:50:49.272126000  141.163.44.99  31.13.90.33  TLSv1.2   333 Application Data
09:50:49.327418000  141.163.44.99  31.13.90.33  TCP      66 63539 > https [ACK]
```

Figure 4.9: Typing and sending chat message on Facebook

Another signature obtained for a Facebook activity was for the user chatting. On each occasion when the chat message was sent (by pressing the 'Enter' key), two TCP packets were sent from the client (i.e. the user's computer) to port number 443 of the server. While repeating the experiment it turned out that the total size of these signature packets is a combination of a base value of 2625 bytes (i.e. 1434 + 1191 bytes) and the size of the characters in the sent message. It was also observed that these packets were getting transmitted within the duration of a millisecond. In figure 4.9, two TCP packets with a total size of 2629 bytes (i.e. 1434 + 1195 bytes) can be noticed, which were generated when 4 characters were sent via the chat. Again, two packets worth the total size of 2630 bytes were obtained when 5 characters were sent. In both typing and sending the chat message activities, the resulting packets seemed to be transmitting within less than a millisecond.

Finding unique patterns for Facebook interactions were quite challenging as majority of the activities provided fluctuating results or no results at all, which were discarded. Table 4.6 lists the user interactions that were identified positively by analysing the traffic that was produced.

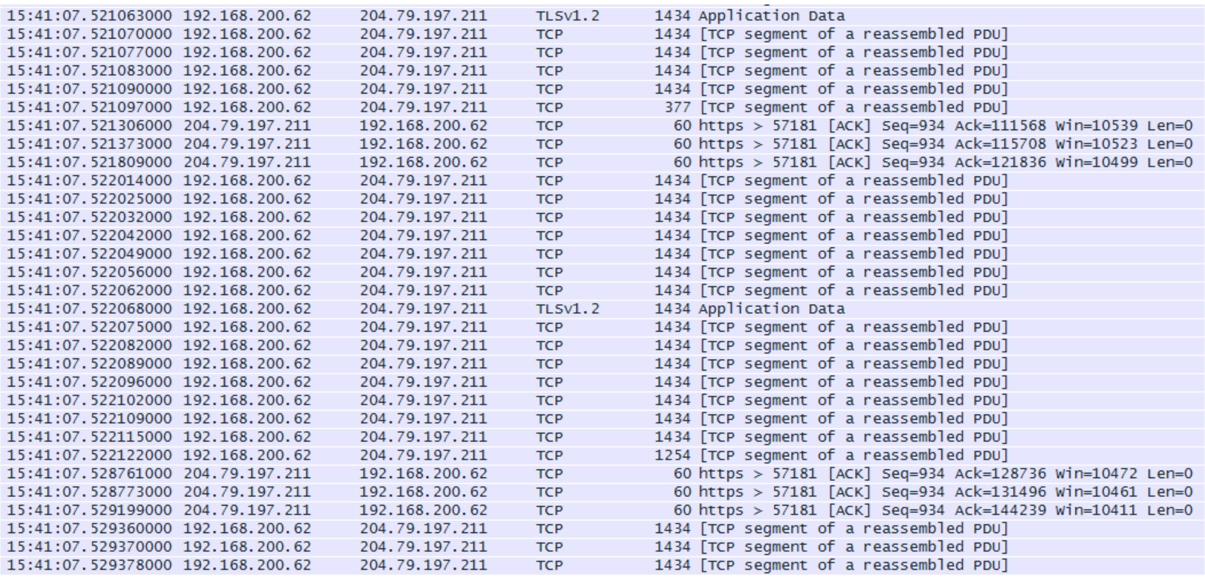
Table 4.6: Facebook user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Page loading	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Attaching file	Stream	MTU (approximate)	Client → Server	TCP (ACK, PSH/ACK)
Typing	Multiple	1502	Client → Server	TCP (PSH/ACK)
Chat	Multiple	2625+	Client → Server	TCP (PSH/ACK)

4.4.4 Hotmail

Hotmail, which is now upgraded to the new Outlook.com, was chosen as an example of a web based e-mail service. E-mail services such as Microsoft Outlook are an inevitable part of the organisational culture and employees use the service for communicating official matters, scheduling tasks and meetings and booking calendar events etc. Therefore, understanding the user activities related to the e-mail service would be beneficial speaking from an investigation perspective. The Hotmail user interactions that were identified during the experiment are attaching files, composing e-mail and inserting a recipient.

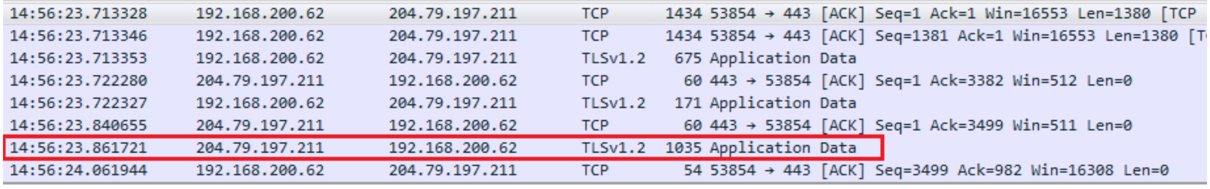
Each time when the user added a file as an attachment, a stream of packets seemed to be transmitting from the user's computer to the server. All TCP packets transferred from the client to server had a size that is approximately equal to the MTU value and had either the ACK or the PSH and ACK flags set in the transport layer. The server kept sending acknowledgement packets in return, which had the ACK flag set. The whole transmission of packets was initiated, continued and completed via port number 443 on the server side. Figure 4.10 shows an example of the pattern obtained while carrying out the activity.



```
15:41:07.521063000 192.168.200.62    204.79.197.211    TLSv1.2     1434 Application Data
15:41:07.521070000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.521077000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.521083000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.521090000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.521097000 192.168.200.62    204.79.197.211    TCP          377 [TCP segment of a reassembled PDU]
15:41:07.521306000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=111568 Win=10539 Len=0
15:41:07.521373000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=115708 Win=10523 Len=0
15:41:07.521809000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=121836 Win=10499 Len=0
15:41:07.522014000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522025000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522032000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522042000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522049000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522056000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522062000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522068000 192.168.200.62    204.79.197.211    TLSv1.2     1434 Application Data
15:41:07.522075000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522082000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522089000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522096000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522102000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522109000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522115000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.522122000 192.168.200.62    204.79.197.211    TCP         1254 [TCP segment of a reassembled PDU]
15:41:07.528761000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=128736 Win=10472 Len=0
15:41:07.528773000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=131496 Win=10461 Len=0
15:41:07.529199000 204.79.197.211    192.168.200.62    TCP           60 https > 57181 [ACK] Seq=934 Ack=144239 Win=10411 Len=0
15:41:07.529360000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.529370000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
15:41:07.529378000 192.168.200.62    204.79.197.211    TCP         1434 [TCP segment of a reassembled PDU]
```

Figure 4.10: File attachment in Hotmail

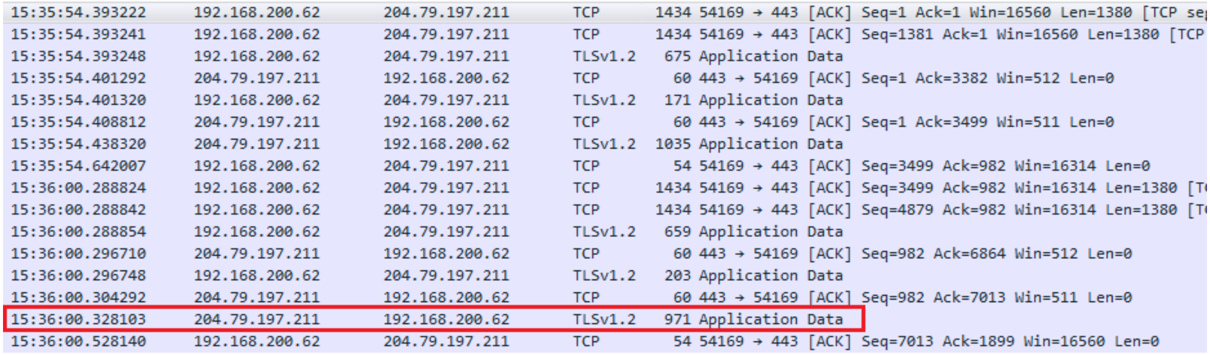
Every time, in order to compose an e-mail, once the user has clicked on the ‘Compose’ button, the server sent a packet that was 1035 bytes in size from port number 443 to the user’s computer. This packet had the PSH and ACK flags set in the transport layer. An example of the packet that resulted from the composing e-mail activity is shown in Figure 4.11.



```
14:56:23.713328     192.168.200.62     204.79.197.211     TCP       1434 53854 → 443 [ACK] Seq=1 Ack=1 Win=16553 Len=1380 [TCP
14:56:23.713346     192.168.200.62     204.79.197.211     TCP       1434 53854 → 443 [ACK] Seq=1381 Ack=1 Win=16553 Len=1380 [T
14:56:23.713353     192.168.200.62     204.79.197.211     TLSv1.2    675 Application Data
14:56:23.722280     204.79.197.211     192.168.200.62     TCP         60 443 → 53854 [ACK] Seq=1 Ack=3382 Win=512 Len=0
14:56:23.722327     192.168.200.62     204.79.197.211     TLSv1.2    171 Application Data
14:56:23.840655     204.79.197.211     192.168.200.62     TCP         60 443 → 53854 [ACK] Seq=1 Ack=3499 Win=511 Len=0
14:56:23.861721     204.79.197.211     192.168.200.62     TLSv1.2   1035 Application Data
14:56:24.061944     192.168.200.62     204.79.197.211     TCP         54 53854 → 443 [ACK] Seq=3499 Ack=982 Win=16308 Len=0
```

Figure 4.11: Composing e-mail in Hotmail

Similarly, each and every time when the user inserted a recipient’s e-mail ID into the recipients section, a packet with a size of 971 bytes was sent from port number 443 of the server to the user’s computer. PSH and ACK flags were set in the transport layer for this specific packet. The packet that resulted from this activity is displayed in Figure 4.12.



```
15:35:54.393222    192.168.200.62    204.79.197.211    TCP     1434 54169 → 443 [ACK] Seq=1 Ack=1 Win=16560 Len=1380 [TCP se
15:35:54.393241    192.168.200.62    204.79.197.211    TCP     1434 54169 → 443 [ACK] Seq=1381 Ack=1 Win=16560 Len=1380 [TCP
15:35:54.393248    192.168.200.62    204.79.197.211    TLSv1.2  675 Application Data
15:35:54.401292    204.79.197.211    192.168.200.62    TCP       60 443 → 54169 [ACK] Seq=1 Ack=3382 Win=512 Len=0
15:35:54.401320    192.168.200.62    204.79.197.211    TLSv1.2  171 Application Data
15:35:54.408812    204.79.197.211    192.168.200.62    TCP       60 443 → 54169 [ACK] Seq=1 Ack=3499 Win=511 Len=0
15:35:54.438320    204.79.197.211    192.168.200.62    TLSv1.2 1035 Application Data
15:35:54.642007    192.168.200.62    204.79.197.211    TCP       54 54169 → 443 [ACK] Seq=3499 Ack=982 Win=16314 Len=0
15:36:00.288824    192.168.200.62    204.79.197.211    TCP     1434 54169 → 443 [ACK] Seq=3499 Ack=982 Win=16314 Len=1380 [T
15:36:00.288842    192.168.200.62    204.79.197.211    TCP     1434 54169 → 443 [ACK] Seq=4879 Ack=982 Win=16314 Len=1380 [T
15:36:00.288854    192.168.200.62    204.79.197.211    TLSv1.2  659 Application Data
15:36:00.296710    204.79.197.211    192.168.200.62    TCP       60 443 → 54169 [ACK] Seq=982 Ack=6864 Win=512 Len=0
15:36:00.296748    192.168.200.62    204.79.197.211    TLSv1.2  203 Application Data
15:36:00.304292    204.79.197.211    192.168.200.62    TCP       60 443 → 54169 [ACK] Seq=982 Ack=7013 Win=511 Len=0
15:36:00.328103    204.79.197.211    192.168.200.62    TLSv1.2  971 Application Data
15:36:00.528140    192.168.200.62    204.79.197.211    TCP       54 54169 → 443 [ACK] Seq=7013 Ack=1899 Win=16560 Len=0
```

Figure 4.12: Inserting recipient in Hotmail

Key characteristics of all user activities that were identified during the analysis are listed in Table 4.7.

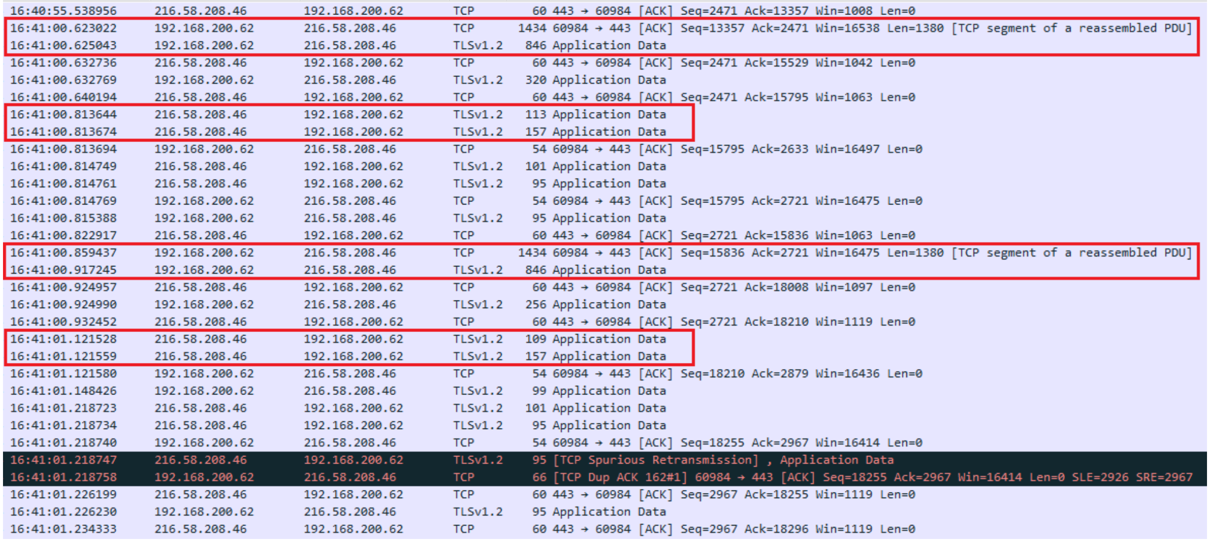
Table 4.7: Hotmail user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Attaching file	Stream	MTU (approximate)	Client → Server	TCP (ACK, PSH/ACK)
Composing e-mail	One	1035	Server → Client	TCP (PSH, ACK)
Inserting recipient	One	971	Server → Client	TCP (PSH, ACK)

4.4.5 Google Docs

Google Docs, part of a web-based software suite offered by Google, is a program that allows users to create and edit documents and perform several other actions on the document. Apart from Google Docs, the software suit contains Google Sheets, Google Slides and Google Forms. Google Docs was chosen as an example of popular online documents applications used by a diverse group of users such as employees and students. The user activity that was identified within Google Docs was editing the document.

When the user was editing the document, two TCP packets with lengths of 1434 bytes and 846 bytes were sent from the user's computer to port number 443 of the server. The transmitted TCP packets had ACK and PSH/ACK flags set in their transport layers. The server replied with two TCP packets with a combined length that varied between 266 bytes and 270 bytes, which were transferred from port number 443 to the client system. An example of the captured traffic is displayed in Figure 4.13.



```
16:40:55.538956   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2471 Ack=13357 Win=1008 Len=0
16:41:00.623022   192.168.200.62   216.58.208.46    TCP      1434 60984 → 443 [ACK] Seq=13357 Ack=2471 Win=16538 Len=1380 [TCP segment of a reassembled PDU]
16:41:00.625043   192.168.200.62   216.58.208.46    TLSv1.2   846 Application Data
16:41:00.632736   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2471 Ack=15529 Win=1042 Len=0
16:41:00.632769   192.168.200.62   216.58.208.46    TLSv1.2   320 Application Data
16:41:00.640194   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2471 Ack=15795 Win=1063 Len=0
16:41:00.813644   216.58.208.46    192.168.200.62   TLSv1.2   113 Application Data
16:41:00.813674   216.58.208.46    192.168.200.62   TLSv1.2   157 Application Data
16:41:00.813694   192.168.200.62   216.58.208.46    TCP        54 60984 → 443 [ACK] Seq=15795 Ack=2633 Win=16497 Len=0
16:41:00.814749   216.58.208.46    192.168.200.62   TLSv1.2   101 Application Data
16:41:00.814761   216.58.208.46    192.168.200.62   TLSv1.2    95 Application Data
16:41:00.814769   192.168.200.62   216.58.208.46    TCP        54 60984 → 443 [ACK] Seq=15795 Ack=2721 Win=16475 Len=0
16:41:00.815388   192.168.200.62   216.58.208.46    TLSv1.2    95 Application Data
16:41:00.822917   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2721 Ack=15836 Win=1063 Len=0
16:41:00.859437   192.168.200.62   216.58.208.46    TCP      1434 60984 → 443 [ACK] Seq=15836 Ack=2721 Win=16475 Len=1380 [TCP segment of a reassembled PDU]
16:41:00.917245   192.168.200.62   216.58.208.46    TLSv1.2   846 Application Data
16:41:00.924957   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2721 Ack=18008 Win=1097 Len=0
16:41:00.924990   192.168.200.62   216.58.208.46    TLSv1.2   256 Application Data
16:41:00.932452   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2721 Ack=18210 Win=1119 Len=0
16:41:01.121528   216.58.208.46    192.168.200.62   TLSv1.2   109 Application Data
16:41:01.121559   216.58.208.46    192.168.200.62   TLSv1.2   157 Application Data
16:41:01.121580   192.168.200.62   216.58.208.46    TCP        54 60984 → 443 [ACK] Seq=18210 Ack=2879 Win=16436 Len=0
16:41:01.148426   192.168.200.62   216.58.208.46    TLSv1.2    99 Application Data
16:41:01.218723   216.58.208.46    192.168.200.62   TLSv1.2   101 Application Data
16:41:01.218734   216.58.208.46    192.168.200.62   TLSv1.2    95 Application Data
16:41:01.218740   192.168.200.62   216.58.208.46    TCP        54 60984 → 443 [ACK] Seq=18255 Ack=2967 Win=16414 Len=0
16:41:01.218747   216.58.208.46    192.168.200.62   TLSv1.2    95 [TCP Spurious Retransmission] , Application Data
16:41:01.218758   192.168.200.62   216.58.208.46    TCP        66 [TCP Dup ACK 162#1] 60984 → 443 [ACK] Seq=18255 Ack=2967 Win=16414 Len=0 SLE=2926 SRE=2967
16:41:01.226199   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2967 Ack=18255 Win=1119 Len=0
16:41:01.226230   192.168.200.62   216.58.208.46    TLSv1.2    95 Application Data
16:41:01.234333   216.58.208.46    192.168.200.62   TCP        60 443 → 60984 [ACK] Seq=2967 Ack=18296 Win=1119 Len=0
```

Figure 4.13: Document editing in Google Docs

The identified user interaction for Google Docs and its key characteristics are listed in Table 4.8.

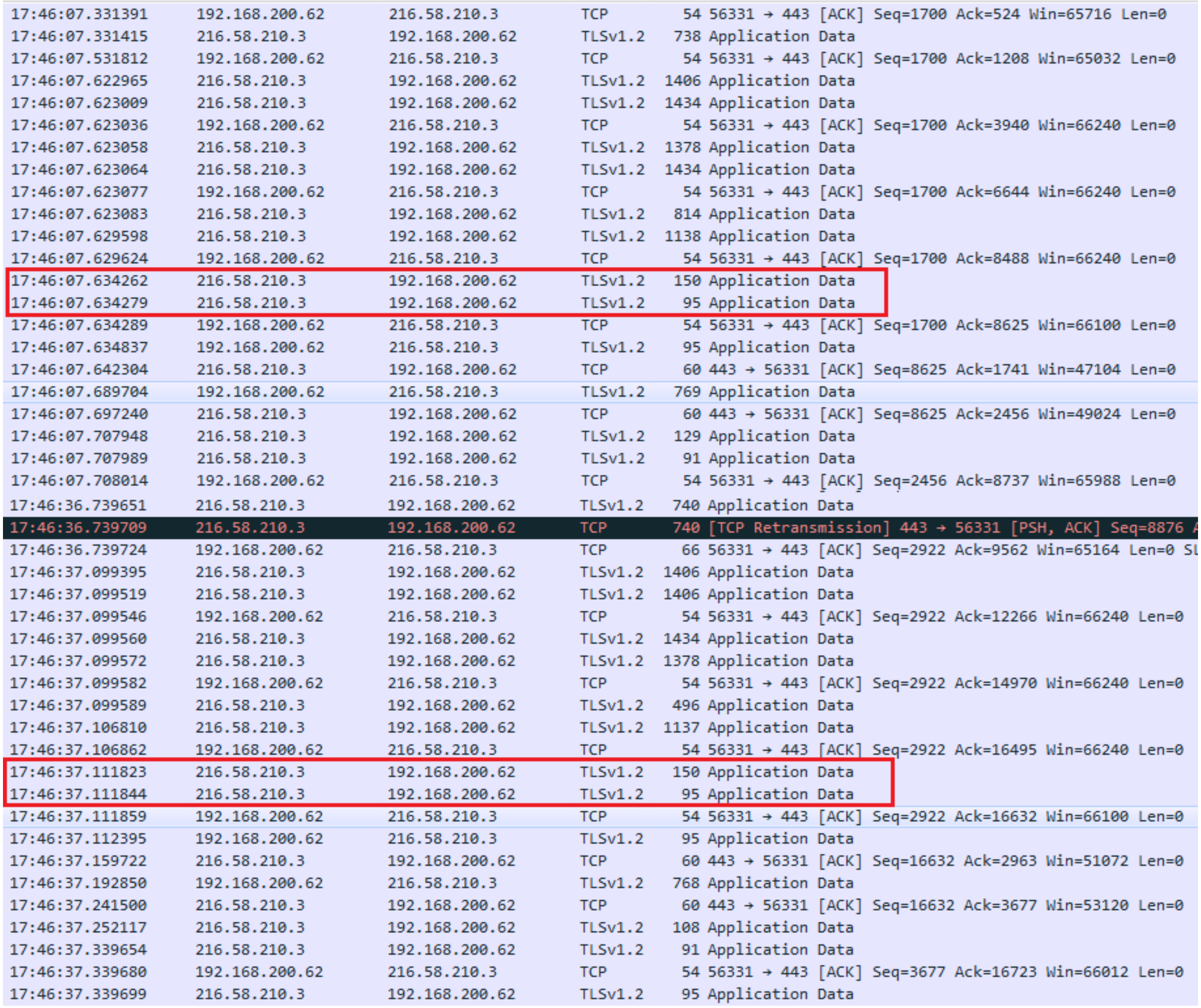
Table 4.8: Google Docs user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Editing document	Multiple	2280	Client → Server	TCP (ACK, PSH/ACK)

4.4.6 Google Search

Being the number one search engine in the United Kingdom, Google's search engine (google.co.uk) generated 3.57 Billion visits in January 2018 (SimilarWeb, 2018). A search engine such as the Google Search allows searching for information such as images, videos, news and helps the users to find the information that is being searched for.

A pattern for when the user navigates through pages containing the search results was identified during the manual analysis. After searching for information, when the user navigated from one page of results to another, two TCP packets were sent from port number 443 of the Google server to the user's computer. The transmitted TCP packets had a combined size of 245 bytes and had the PSH and ACK flags set in their transport layers. The obtained pattern while performing the activity is displayed in Figure 4.14.



```
17:46:07.331391   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=524 Win=65716 Len=0
17:46:07.331415   216.58.210.3     192.168.200.62   TLSv1.2  738 Application Data
17:46:07.531812   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=1208 Win=65032 Len=0
17:46:07.622965   216.58.210.3     192.168.200.62   TLSv1.2 1406 Application Data
17:46:07.623009   216.58.210.3     192.168.200.62   TLSv1.2 1434 Application Data
17:46:07.623036   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=3940 Win=66240 Len=0
17:46:07.623058   216.58.210.3     192.168.200.62   TLSv1.2 1378 Application Data
17:46:07.623064   216.58.210.3     192.168.200.62   TLSv1.2 1434 Application Data
17:46:07.623077   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=6644 Win=66240 Len=0
17:46:07.623083   216.58.210.3     192.168.200.62   TLSv1.2  814 Application Data
17:46:07.629598   216.58.210.3     192.168.200.62   TLSv1.2 1138 Application Data
17:46:07.629624   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=8488 Win=66240 Len=0
17:46:07.634262   216.58.210.3     192.168.200.62   TLSv1.2  150 Application Data
17:46:07.634279   216.58.210.3     192.168.200.62   TLSv1.2   95 Application Data
17:46:07.634289   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=1700 Ack=8625 Win=66100 Len=0
17:46:07.634837   192.168.200.62   216.58.210.3     TLSv1.2   95 Application Data
17:46:07.642304   216.58.210.3     192.168.200.62   TCP       60 443 → 56331 [ACK] Seq=8625 Ack=1741 Win=47104 Len=0
17:46:07.689704   192.168.200.62   216.58.210.3     TLSv1.2  769 Application Data
17:46:07.697240   216.58.210.3     192.168.200.62   TCP       60 443 → 56331 [ACK] Seq=8625 Ack=2456 Win=49024 Len=0
17:46:07.707948   216.58.210.3     192.168.200.62   TLSv1.2  129 Application Data
17:46:07.707989   216.58.210.3     192.168.200.62   TLSv1.2   91 Application Data
17:46:07.708014   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=2456 Ack=8737 Win=65988 Len=0
17:46:36.739651   216.58.210.3     192.168.200.62   TLSv1.2  740 Application Data
17:46:36.739709   216.58.210.3     192.168.200.62   TCP      740 [TCP Retransmission] 443 → 56331 [PSH, ACK] Seq=8876 A
17:46:36.739724   192.168.200.62   216.58.210.3     TCP       66 56331 → 443 [ACK] Seq=2922 Ack=9562 Win=65164 Len=0 SL
17:46:37.099395   216.58.210.3     192.168.200.62   TLSv1.2 1406 Application Data
17:46:37.099519   216.58.210.3     192.168.200.62   TLSv1.2 1406 Application Data
17:46:37.099546   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=2922 Ack=12266 Win=66240 Len=0
17:46:37.099560   216.58.210.3     192.168.200.62   TLSv1.2 1434 Application Data
17:46:37.099572   216.58.210.3     192.168.200.62   TLSv1.2 1378 Application Data
17:46:37.099582   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=2922 Ack=14970 Win=66240 Len=0
17:46:37.099589   216.58.210.3     192.168.200.62   TLSv1.2  496 Application Data
17:46:37.106810   216.58.210.3     192.168.200.62   TLSv1.2 1137 Application Data
17:46:37.106862   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=2922 Ack=16495 Win=66240 Len=0
17:46:37.111823   216.58.210.3     192.168.200.62   TLSv1.2  150 Application Data
17:46:37.111844   216.58.210.3     192.168.200.62   TLSv1.2   95 Application Data
17:46:37.111859   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=2922 Ack=16632 Win=66100 Len=0
17:46:37.112395   192.168.200.62   216.58.210.3     TLSv1.2   95 Application Data
17:46:37.159722   216.58.210.3     192.168.200.62   TCP       60 443 → 56331 [ACK] Seq=16632 Ack=2963 Win=51072 Len=0
17:46:37.192850   192.168.200.62   216.58.210.3     TLSv1.2  768 Application Data
17:46:37.241500   216.58.210.3     192.168.200.62   TCP       60 443 → 56331 [ACK] Seq=16632 Ack=3677 Win=53120 Len=0
17:46:37.252117   216.58.210.3     192.168.200.62   TLSv1.2  108 Application Data
17:46:37.339654   216.58.210.3     192.168.200.62   TLSv1.2   91 Application Data
17:46:37.339680   192.168.200.62   216.58.210.3     TCP       54 56331 → 443 [ACK] Seq=3677 Ack=16723 Win=66012 Len=0
17:46:37.339699   216.58.210.3     192.168.200.62   TLSv1.2   95 Application Data
```

Figure 4.14: Page navigation in Google Search

The identified user interaction in Google Search (i.e. page navigation) along with its key characteristics is listed in Table 4.9.

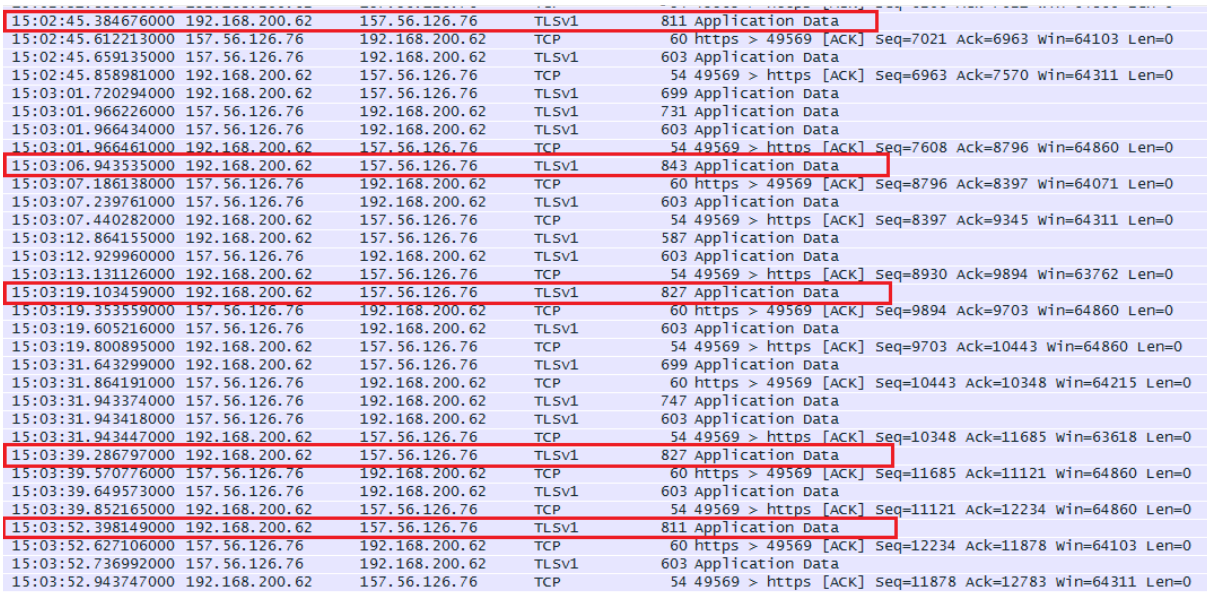
Table 4.9: Google Search user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Page navigation	Multiple	245	Server → Client	TCP (PSH/ACK)

4.4.7 Skype

Skype is a VoIP application, now owned by Microsoft, which allows users to communicate with other members via text chat, audio calls and video calls. The application also allows users to share images, files and to do video conferencing with multiple users. Skype is a software product utilised a lot within the organisational environment and therefore the ability to identify user interactions within Skype would be beneficial. The user interactions identified within Skype during the manual analysis are – text messaging, audio calling, video calling, transferring files, clicking on contacts and being idle.

Recognising the existence of a signature for the text messaging activity was more challenging compared to the other activities due to the variation in packet size. When the user was participating in text chat, a single TCP packet was observed to be transmitting from the user's computer to port number 443 of the Skype server. The size of the packet was observed as a combination of a baseline value (794 bytes) and the size of the characters. For example, when a 17 character long message was sent the resulting packet was 811 bytes long and when a 33 character long message was sent the packet was 827 bytes long, which are shown in Figure 4.15. The resulting packet had PSH and ACK flags set in the transport layer.

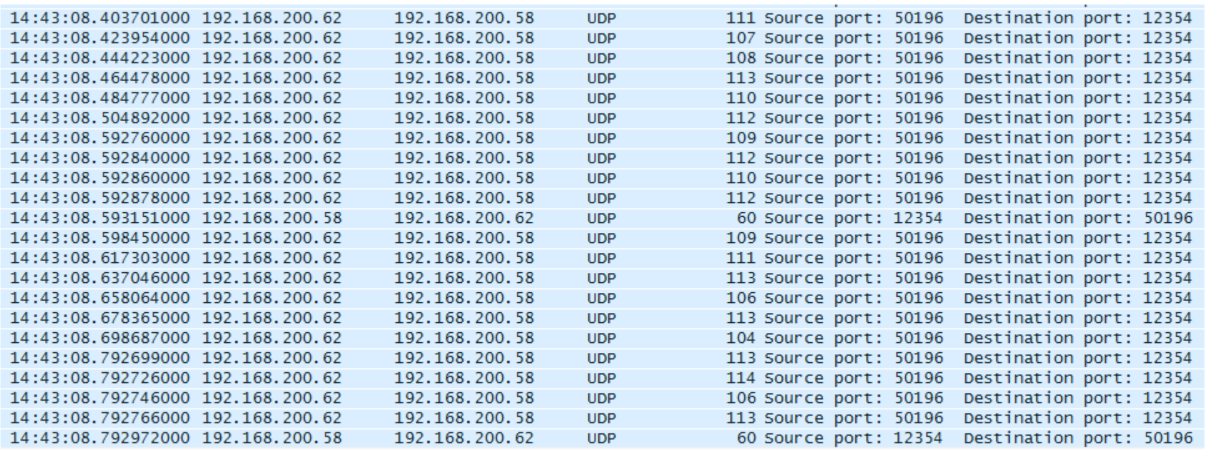


```
15:02:45.384676000 192.168.200.62   157.56.126.76    TLSv1      811 Application Data
15:02:45.612213000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=7021 Ack=6963 Win=64103 Len=0
15:02:45.659135000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:02:45.858981000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=6963 Ack=7570 Win=64311 Len=0
15:03:01.720294000 192.168.200.62   157.56.126.76    TLSv1      699 Application Data
15:03:01.966226000 157.56.126.76    192.168.200.62   TLSv1      731 Application Data
15:03:01.966434000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:01.966461000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=7608 Ack=8796 Win=64860 Len=0
15:03:06.943535000 192.168.200.62   157.56.126.76    TLSv1      843 Application Data
15:03:07.186138000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=8796 Ack=8397 Win=64071 Len=0
15:03:07.239761000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:07.440282000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=8397 Ack=9345 Win=64311 Len=0
15:03:12.864155000 192.168.200.62   157.56.126.76    TLSv1      587 Application Data
15:03:12.929960000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:13.131126000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=8930 Ack=9894 Win=63762 Len=0
15:03:19.103459000 192.168.200.62   157.56.126.76    TLSv1      827 Application Data
15:03:19.353559000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=9894 Ack=9703 Win=64860 Len=0
15:03:19.605216000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:19.800895000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=9703 Ack=10443 Win=64860 Len=0
15:03:31.643299000 192.168.200.62   157.56.126.76    TLSv1      699 Application Data
15:03:31.864191000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=10443 Ack=10348 Win=64215 Len=0
15:03:31.943374000 157.56.126.76    192.168.200.62   TLSv1      747 Application Data
15:03:31.943418000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:31.943447000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=10348 Ack=11685 Win=63618 Len=0
15:03:39.286797000 192.168.200.62   157.56.126.76    TLSv1      827 Application Data
15:03:39.570776000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=11685 Ack=11121 Win=64860 Len=0
15:03:39.649573000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:39.852165000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=11121 Ack=12234 Win=64860 Len=0
15:03:52.398149000 192.168.200.62   157.56.126.76    TLSv1      811 Application Data
15:03:52.627106000 157.56.126.76    192.168.200.62   TCP         60 https > 49569 [ACK] Seq=12234 Ack=11878 Win=64103 Len=0
15:03:52.736992000 157.56.126.76    192.168.200.62   TLSv1      603 Application Data
15:03:52.943747000 192.168.200.62   157.56.126.76    TCP         54 49569 > https [ACK] Seq=11878 Ack=12783 Win=64311 Len=0
```

Figure 4.15: Sending text message in Skype

Skype audio and video calls are examples of connectionless communication that use UDP in the transport layer. Skype operates in such a way that when a user logs into his/her Skype account through a computer, that particular device becomes a node within the Skype network. When a user begins an audio or video call with another Skype member, the two nodes involved (i.e. the caller's and the callee's computers) are connected directly, which is known as a peer to peer connection, creating the medium for communication without any intermediate system (i.e. the Skype server) controlling the communication.

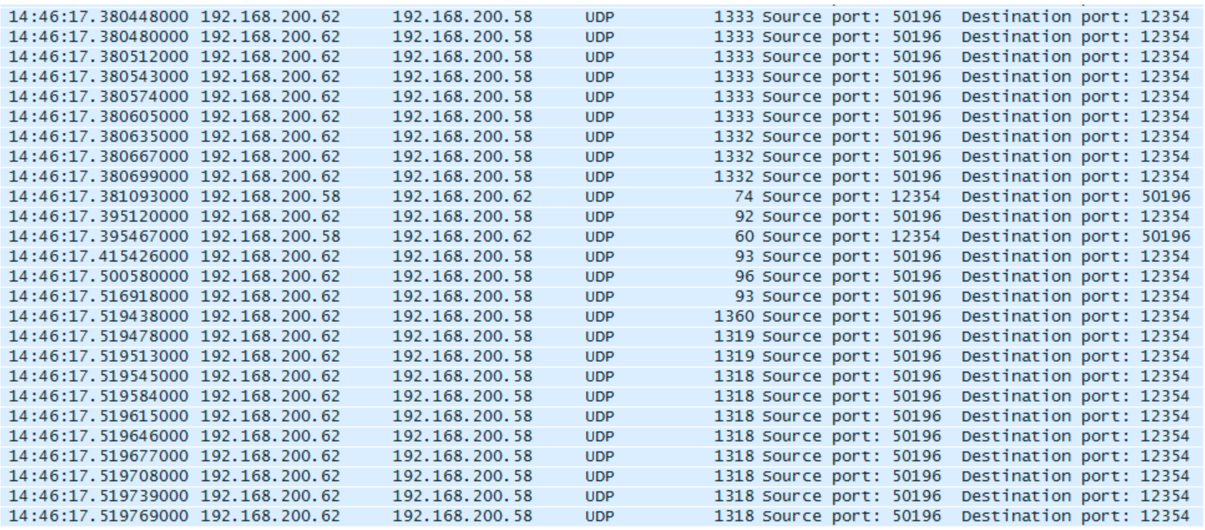
When the audio call was commenced, UDP packets were transmitted between UDP ports of the two clients that were involved in the call. In Figure 4.16, it can be noticed that the first client (i.e. user IP – 192.168.200.62) is sending a stream of UDP packets that are larger in size (between 104 and 147 bytes) compared to the 60 bytes long UDP packets returned by the second client (i.e. user IP – 192.168.200.58). This scenario was the outcome of an audio call in which the first client was using a microphone and the second client was not, meaning there was no audio returned.



```
14:43:08.403701000 192.168.200.62     192.168.200.58     UDP          111 Source port: 50196  Destination port: 12354
14:43:08.423954000 192.168.200.62     192.168.200.58     UDP          107 Source port: 50196  Destination port: 12354
14:43:08.444223000 192.168.200.62     192.168.200.58     UDP          108 Source port: 50196  Destination port: 12354
14:43:08.464478000 192.168.200.62     192.168.200.58     UDP          113 Source port: 50196  Destination port: 12354
14:43:08.484777000 192.168.200.62     192.168.200.58     UDP          110 Source port: 50196  Destination port: 12354
14:43:08.504892000 192.168.200.62     192.168.200.58     UDP          112 Source port: 50196  Destination port: 12354
14:43:08.592760000 192.168.200.62     192.168.200.58     UDP          109 Source port: 50196  Destination port: 12354
14:43:08.592840000 192.168.200.62     192.168.200.58     UDP          112 Source port: 50196  Destination port: 12354
14:43:08.592860000 192.168.200.62     192.168.200.58     UDP          110 Source port: 50196  Destination port: 12354
14:43:08.592878000 192.168.200.62     192.168.200.58     UDP          112 Source port: 50196  Destination port: 12354
14:43:08.593151000 192.168.200.58     192.168.200.62     UDP           60 Source port: 12354  Destination port: 50196
14:43:08.598450000 192.168.200.62     192.168.200.58     UDP          109 Source port: 50196  Destination port: 12354
14:43:08.617303000 192.168.200.62     192.168.200.58     UDP          111 Source port: 50196  Destination port: 12354
14:43:08.637046000 192.168.200.62     192.168.200.58     UDP          113 Source port: 50196  Destination port: 12354
14:43:08.658064000 192.168.200.62     192.168.200.58     UDP          106 Source port: 50196  Destination port: 12354
14:43:08.678365000 192.168.200.62     192.168.200.58     UDP          113 Source port: 50196  Destination port: 12354
14:43:08.698687000 192.168.200.62     192.168.200.58     UDP          104 Source port: 50196  Destination port: 12354
14:43:08.792699000 192.168.200.62     192.168.200.58     UDP          113 Source port: 50196  Destination port: 12354
14:43:08.792726000 192.168.200.62     192.168.200.58     UDP          114 Source port: 50196  Destination port: 12354
14:43:08.792746000 192.168.200.62     192.168.200.58     UDP          106 Source port: 50196  Destination port: 12354
14:43:08.792766000 192.168.200.62     192.168.200.58     UDP          113 Source port: 50196  Destination port: 12354
14:43:08.792972000 192.168.200.58     192.168.200.62     UDP           60 Source port: 12354  Destination port: 50196
```

Figure 4.16: Audio call in Skype

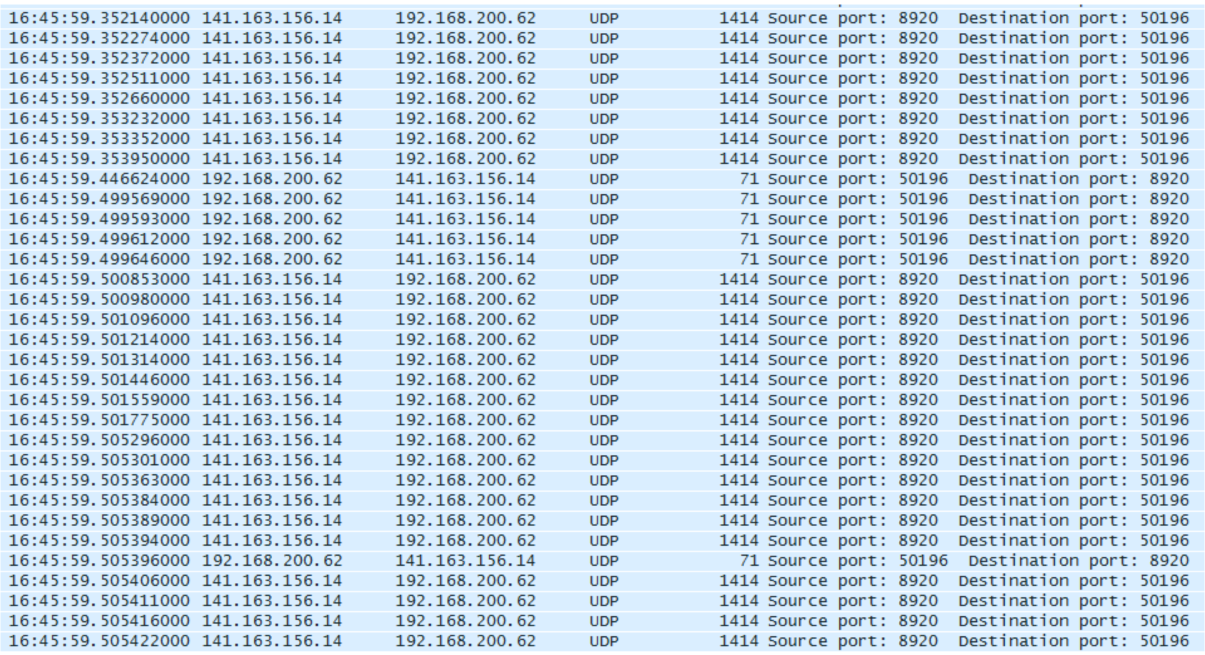
Similar to the audio call, when the user made a video call to another Skype user, UDP packets were transferred between the two clients. But in this case, the size of majority of the UDP packets sent from client one (i.e. user IP – 192.168.200.62) were ranging between 1165 bytes and 1382 bytes and the UDP packets returned by the second client (i.e. user IP – 192. 168.200.58) were much smaller in size, as shown in Figure 4.17. This is because only client one had the webcam turned on while the second client had not.



```
14:46:17.380448000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380480000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380512000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380543000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380574000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380605000 192.168.200.62     192.168.200.58     UDP         1333 Source port: 50196  Destination port: 12354
14:46:17.380635000 192.168.200.62     192.168.200.58     UDP         1332 Source port: 50196  Destination port: 12354
14:46:17.380667000 192.168.200.62     192.168.200.58     UDP         1332 Source port: 50196  Destination port: 12354
14:46:17.380699000 192.168.200.62     192.168.200.58     UDP         1332 Source port: 50196  Destination port: 12354
14:46:17.381093000 192.168.200.58     192.168.200.62     UDP           74 Source port: 12354  Destination port: 50196
14:46:17.395120000 192.168.200.62     192.168.200.58     UDP           92 Source port: 50196  Destination port: 12354
14:46:17.395467000 192.168.200.58     192.168.200.62     UDP           60 Source port: 12354  Destination port: 50196
14:46:17.415426000 192.168.200.62     192.168.200.58     UDP           93 Source port: 50196  Destination port: 12354
14:46:17.500580000 192.168.200.62     192.168.200.58     UDP           96 Source port: 50196  Destination port: 12354
14:46:17.516918000 192.168.200.62     192.168.200.58     UDP           93 Source port: 50196  Destination port: 12354
14:46:17.519438000 192.168.200.62     192.168.200.58     UDP         1360 Source port: 50196  Destination port: 12354
14:46:17.519478000 192.168.200.62     192.168.200.58     UDP         1319 Source port: 50196  Destination port: 12354
14:46:17.519513000 192.168.200.62     192.168.200.58     UDP         1319 Source port: 50196  Destination port: 12354
14:46:17.519545000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519584000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519615000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519646000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519677000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519708000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519739000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
14:46:17.519769000 192.168.200.62     192.168.200.58     UDP         1318 Source port: 50196  Destination port: 12354
```

Figure 4.17: Video call in Skype

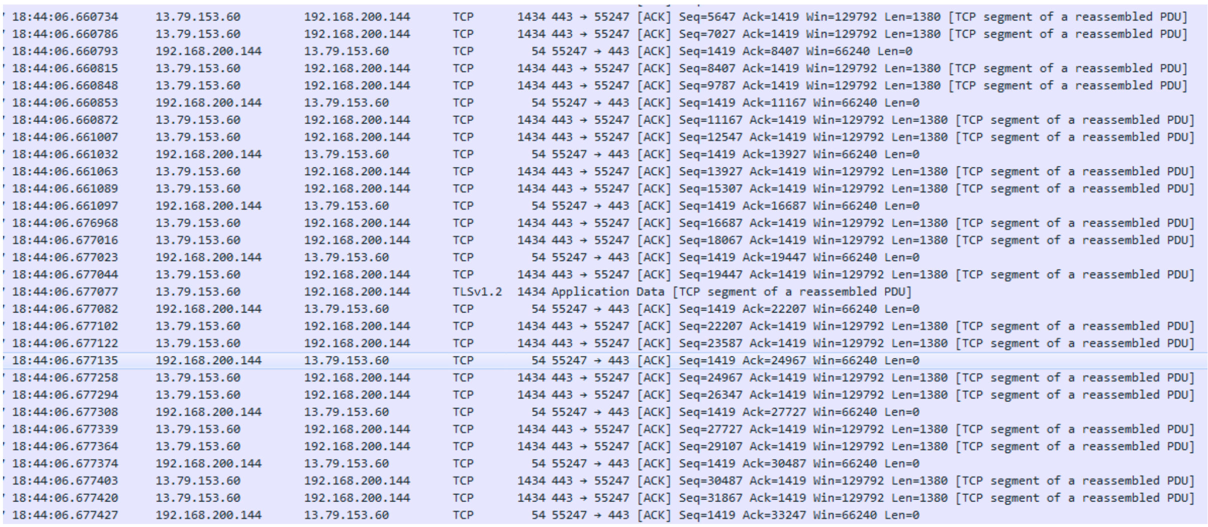
The signature obtained for transferring file between the Skype users was similar to the signatures obtained for audio and video calls in terms of the number of packets involved. The signature contained a stream of packets that were transferred from the sending client (i.e. user IP – 141.163.156.14) to the receiving client (i.e. 192.168.200.62), as shown in Figure 4.18. Size of the packets transmitted from the sending client was nearly the MTU. The files that were used for transmission were of type '.pdf' and '.docx'.



```
16:45:59.352140000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.352274000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.352372000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.352511000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.352660000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.353232000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.353352000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.353950000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.446624000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.499569000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.499593000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.499612000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.499646000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.500853000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.500980000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501096000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501214000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501314000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501446000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501559000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.501775000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505296000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505301000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505363000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505384000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505389000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505394000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505396000 192.168.200.62     141.163.156.14     UDP            71 Source port: 50196  Destination port: 8920
16:45:59.505406000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505411000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505416000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
16:45:59.505422000 141.163.156.14     192.168.200.62     UDP          1414 Source port: 8920  Destination port: 50196
```

Figure 4.18: File transfer in Skype

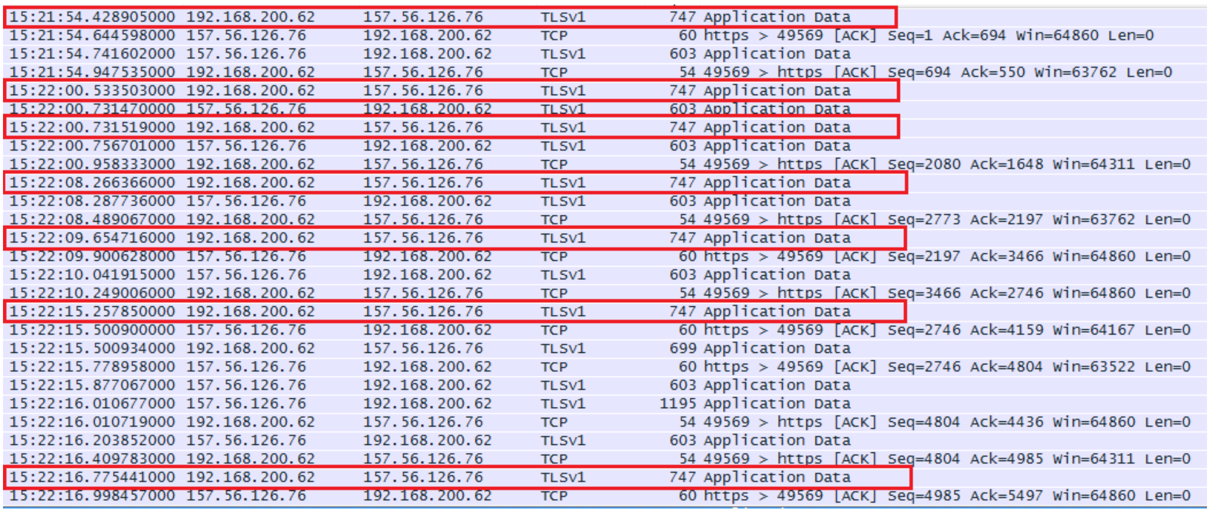
Another user activity that was tested and gave confirming result was transferring images while using Skype. Depending on whether the user is sending or receiving the image, the main direction of the flow of packets in the signature will change. It has to be highlighted that the Skype server is involved within the image transfer activity and it is not carried out directly between the users. Figure 4.19 shows the pattern obtained when the user was receiving an image. It can be noticed that the signature contained a stream of TCP packets that were transmitted from port number 443 of the server to the user's computer. The transmitted packets had either ACK or PSH, ACK flags set in the transport layer. In return, TCP acknowledgement packets were sent from the client system to port number 443 of the server.



```
18:44:06.660734   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=5647 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.660786   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=7027 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.660793   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=8407 Win=66240 Len=0
18:44:06.660815   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=8407 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.660848   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=9787 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.660853   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=11167 Win=66240 Len=0
18:44:06.660872   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=11167 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.661007   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=12547 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.661032   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=13927 Win=66240 Len=0
18:44:06.661063   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=13927 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.661089   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=15307 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.661097   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=16687 Win=66240 Len=0
18:44:06.676968   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=16687 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677016   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=18067 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677023   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=19447 Win=66240 Len=0
18:44:06.677044   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=19447 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677077   13.79.153.60      192.168.200.144   TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
18:44:06.677082   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=22207 Win=66240 Len=0
18:44:06.677102   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=22207 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677122   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=23587 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677135   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=24967 Win=66240 Len=0
18:44:06.677258   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=24967 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677294   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=26347 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677308   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=27727 Win=66240 Len=0
18:44:06.677339   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=27727 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677364   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=29107 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677374   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=30487 Win=66240 Len=0
18:44:06.677403   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=30487 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677420   13.79.153.60      192.168.200.144   TCP      1434 443 → 55247 [ACK] Seq=31867 Ack=1419 Win=129792 Len=1380 [TCP segment of a reassembled PDU]
18:44:06.677427   192.168.200.144   13.79.153.60      TCP        54 55247 → 443 [ACK] Seq=1419 Ack=33247 Win=66240 Len=0
```

Figure 4.19: Image transfer in Skype

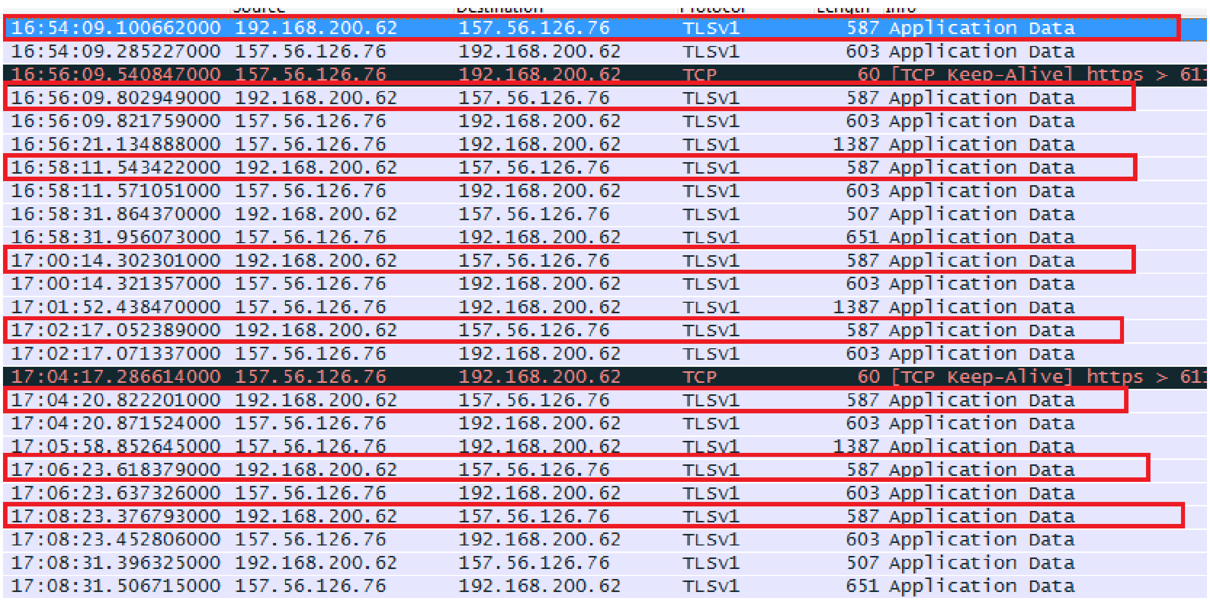
While using Skype, the activity of clicking on contacts generated a certain pattern that was consistent throughout the experiment. An example of the obtained pattern is shown in Figure 4.20. Every time the user clicked on a Skype contact, a single TCP packet with a size of 747 bytes was transmitted from the user's computer to port number 443 of the server. The TCP packet had PSH and ACK flags set in the transport layer.



```
15:21:54.428905000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:21:54.644598000 157.56.126.76     192.168.200.62    TCP          60 https > 49569 [ACK] Seq=1 Ack=694 Win=64860 Len=0
15:21:54.741602000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:21:54.947535000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=694 Ack=550 Win=63762 Len=0
15:22:00.533503000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:00.731470000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:00.731519000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:00.756701000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:00.958333000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=2080 Ack=1648 Win=64311 Len=0
15:22:08.266366000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:08.287736000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:08.489067000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=2773 Ack=2197 Win=63762 Len=0
15:22:09.654716000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:09.900628000 157.56.126.76     192.168.200.62    TCP          60 https > 49569 [ACK] Seq=2197 Ack=3466 Win=64860 Len=0
15:22:10.041915000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:10.249006000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=3466 Ack=2746 Win=64860 Len=0
15:22:15.257850000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:15.500900000 157.56.126.76     192.168.200.62    TCP          60 https > 49569 [ACK] Seq=2746 Ack=4159 Win=64167 Len=0
15:22:15.500934000 192.168.200.62    157.56.126.76     TLSv1       699 Application Data
15:22:15.778958000 157.56.126.76     192.168.200.62    TCP          60 https > 49569 [ACK] Seq=2746 Ack=4804 Win=63522 Len=0
15:22:15.877067000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:16.010677000 157.56.126.76     192.168.200.62    TLSv1      1195 Application Data
15:22:16.010719000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=4804 Ack=4436 Win=64860 Len=0
15:22:16.203852000 157.56.126.76     192.168.200.62    TLSv1       603 Application Data
15:22:16.409783000 192.168.200.62    157.56.126.76     TCP          54 49569 > https [ACK] Seq=4804 Ack=4985 Win=64311 Len=0
15:22:16.775441000 192.168.200.62    157.56.126.76     TLSv1       747 Application Data
15:22:16.998457000 157.56.126.76     192.168.200.62    TCP          60 https > 49569 [ACK] Seq=4985 Ack=5497 Win=64860 Len=0
```

Figure 4.20: Clicking on contacts in Skype

A signature obtained for another activity was when the Skype was left idle by the user. During the activity, a TCP packet of length 587 bytes was transmitted from the user's computer to port number 443 of the server. PSH and ACK flags were set in the transport layer of the transmitted TCP packet. An example of the obtained traffic is shown in Figure 4.21.



```
16:54:09.100662000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
16:54:09.285227000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
16:56:09.540847000 157.56.126.76   192.168.200.62  TCP       60 [TCP Keep-Alive] https > 61
16:56:09.802949000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
16:56:09.821759000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
16:56:21.134888000 157.56.126.76   192.168.200.62  TLSv1   1387 Application Data
16:58:11.543422000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
16:58:11.571051000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
16:58:31.864370000 192.168.200.62  157.56.126.76   TLSv1    507 Application Data
16:58:31.956073000 157.56.126.76   192.168.200.62  TLSv1    651 Application Data
17:00:14.302301000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
17:00:14.321357000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
17:01:52.438470000 157.56.126.76   192.168.200.62  TLSv1   1387 Application Data
17:02:17.052389000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
17:02:17.071337000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
17:04:17.286614000 157.56.126.76   192.168.200.62  TCP       60 [TCP Keep-Alive] https > 61
17:04:20.822201000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
17:04:20.871524000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
17:05:58.852645000 157.56.126.76   192.168.200.62  TLSv1   1387 Application Data
17:06:23.618379000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
17:06:23.637326000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
17:08:23.376793000 192.168.200.62  157.56.126.76   TLSv1    587 Application Data
17:08:23.452806000 157.56.126.76   192.168.200.62  TLSv1    603 Application Data
17:08:31.396325000 192.168.200.62  157.56.126.76   TLSv1    507 Application Data
17:08:31.506715000 157.56.126.76   192.168.200.62  TLSv1    651 Application Data
```

Figure 4.21: User idle in Skype

Skype is the Internet based application which had the most online user interactions identified for. The user interactions that were identified within Skype and the corresponding key characteristics are listed in Table 4.10.

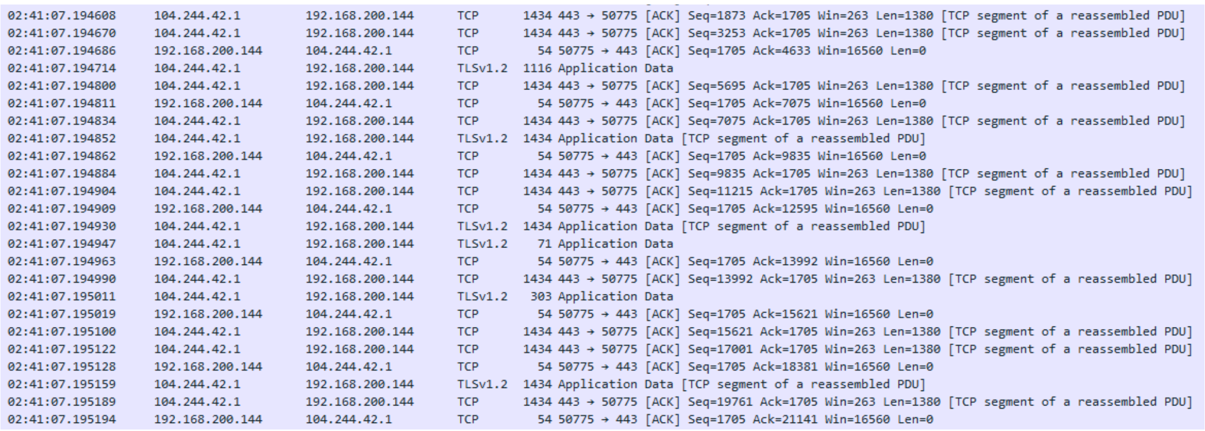
Table 4.10: Skype user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Text message	One	794+	Client → Server	TCP (PSH/ACK)
Audio call	Stream	104 – 147	Both clients	UDP
Video call	Stream	1165 – 1382	Both clients	UDP
File transfer	Stream	MTU (approximate)	Sending client → Receiving client	UDP
Image transfer	Stream	MTU (approximate)	Client → Server (sending) Server → Client (receiving)	TCP (ACK, PSH/ACK)
Click on contact	One	747	Client → Server	TCP (PSH/ACK)
Idle	One	587	Client → Server	TCP (PSH/ACK)

4.4.8 Twitter

Twitter is an example of microblogging that allows its users to post messages, known as tweets, that are 280 characters long and to interact with other users' messages. Social networking and news are the primary types of services offered by Twitter to its users. Twitter is ranked 13th among the top websites in the world as of February 2018 (Alexa, 2018) and has 4.49 Billion visits in January 2018 (SimilarWeb, 2018). Main user activities that were identified while interacting with Twitter are page loading and uploading photo.

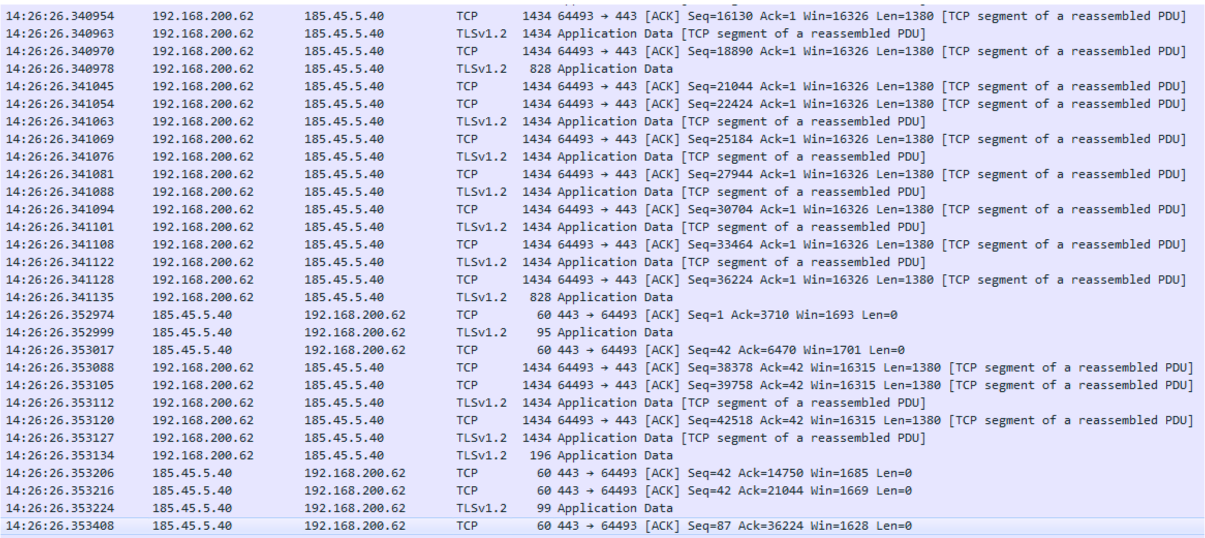
When the user carried out the page loading activity on Twitter, a resulting pattern that contained stream of TCP packets that were transmitted from port number 443 of the server to the user's computer was obtained. The TCP packets had a size that is approximately equal to the MTU and had either ACK or PSH/ACK flags set in the transport layer. Acknowledgement TCP packets, with ACK flag set in the transport layer, were returned from the client system to the server. Figure 4.22 shows an example of the traffic resulted from the page loading activity.



```
02:41:07.194608   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=1873 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194670   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=3253 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194686   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=4633 Win=16560 Len=0
02:41:07.194714   104.244.42.1      192.168.200.144   TLSv1.2  1116 Application Data
02:41:07.194800   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=5695 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194811   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=7075 Win=16560 Len=0
02:41:07.194834   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=7075 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194852   104.244.42.1      192.168.200.144   TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
02:41:07.194862   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=9835 Win=16560 Len=0
02:41:07.194884   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=9835 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194904   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=11215 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.194909   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=12595 Win=16560 Len=0
02:41:07.194930   104.244.42.1      192.168.200.144   TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
02:41:07.194947   104.244.42.1      192.168.200.144   TLSv1.2    71 Application Data
02:41:07.194963   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=13992 Win=16560 Len=0
02:41:07.194990   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=13992 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.195011   104.244.42.1      192.168.200.144   TLSv1.2   303 Application Data
02:41:07.195019   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=15621 Win=16560 Len=0
02:41:07.195100   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=15621 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.195122   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=17001 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.195128   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=18381 Win=16560 Len=0
02:41:07.195159   104.244.42.1      192.168.200.144   TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
02:41:07.195189   104.244.42.1      192.168.200.144   TCP      1434 443 → 50775 [ACK] Seq=19761 Ack=1705 Win=263 Len=1380 [TCP segment of a reassembled PDU]
02:41:07.195194   192.168.200.144   104.244.42.1      TCP        54 50775 → 443 [ACK] Seq=1705 Ack=21141 Win=16560 Len=0
```

Figure 4.22: Page loading in Twitter

Uploading photo to the Twitter feed is another user activity that was observed to have a signature. Stream of TCP packets with size approximately equal to the MTU value were transferred from the user's computer to port number 443 of the server. Either ACK or PSH/ACK flags were set in the transport layer of the transmitted TCP packets. In return, TCP acknowledgement packets, with the ACK flag set, were returned from port number 443 of the server. An example of the obtained traffic is shown in Figure 4.23.



```
14:26:26.340954   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=16130 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.340963   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.340970   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=18890 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.340978   192.168.200.62   185.45.5.40      TLSv1.2  828 Application Data
14:26:26.341045   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=21044 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341054   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=22424 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341063   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.341069   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=25184 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341076   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.341081   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=27944 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341088   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.341094   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=30704 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341101   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.341108   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=33464 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341122   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.341128   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=36224 Ack=1 Win=16326 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.341135   192.168.200.62   185.45.5.40      TLSv1.2  828 Application Data
14:26:26.352974   185.45.5.40      192.168.200.62   TCP       60 443 → 64493 [ACK] Seq=1 Ack=3710 Win=1693 Len=0
14:26:26.352999   185.45.5.40      192.168.200.62   TLSv1.2   95 Application Data
14:26:26.353017   185.45.5.40      192.168.200.62   TCP       60 443 → 64493 [ACK] Seq=42 Ack=6470 Win=1701 Len=0
14:26:26.353088   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=38378 Ack=42 Win=16315 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.353105   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=39758 Ack=42 Win=16315 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.353112   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.353120   192.168.200.62   185.45.5.40      TCP     1434 64493 → 443 [ACK] Seq=42518 Ack=42 Win=16315 Len=1380 [TCP segment of a reassembled PDU]
14:26:26.353127   192.168.200.62   185.45.5.40      TLSv1.2 1434 Application Data [TCP segment of a reassembled PDU]
14:26:26.353134   192.168.200.62   185.45.5.40      TLSv1.2  196 Application Data
14:26:26.353206   185.45.5.40      192.168.200.62   TCP       60 443 → 64493 [ACK] Seq=42 Ack=14750 Win=1685 Len=0
14:26:26.353216   185.45.5.40      192.168.200.62   TCP       60 443 → 64493 [ACK] Seq=42 Ack=21044 Win=1669 Len=0
14:26:26.353224   185.45.5.40      192.168.200.62   TLSv1.2   99 Application Data
14:26:26.353408   185.45.5.40      192.168.200.62   TCP       60 443 → 64493 [ACK] Seq=87 Ack=36224 Win=1628 Len=0
```

Figure 4.23: Uploading photo in Twitter

User interactions identified within Twitter and the corresponding key characteristics are listed in Table 4.11 below.

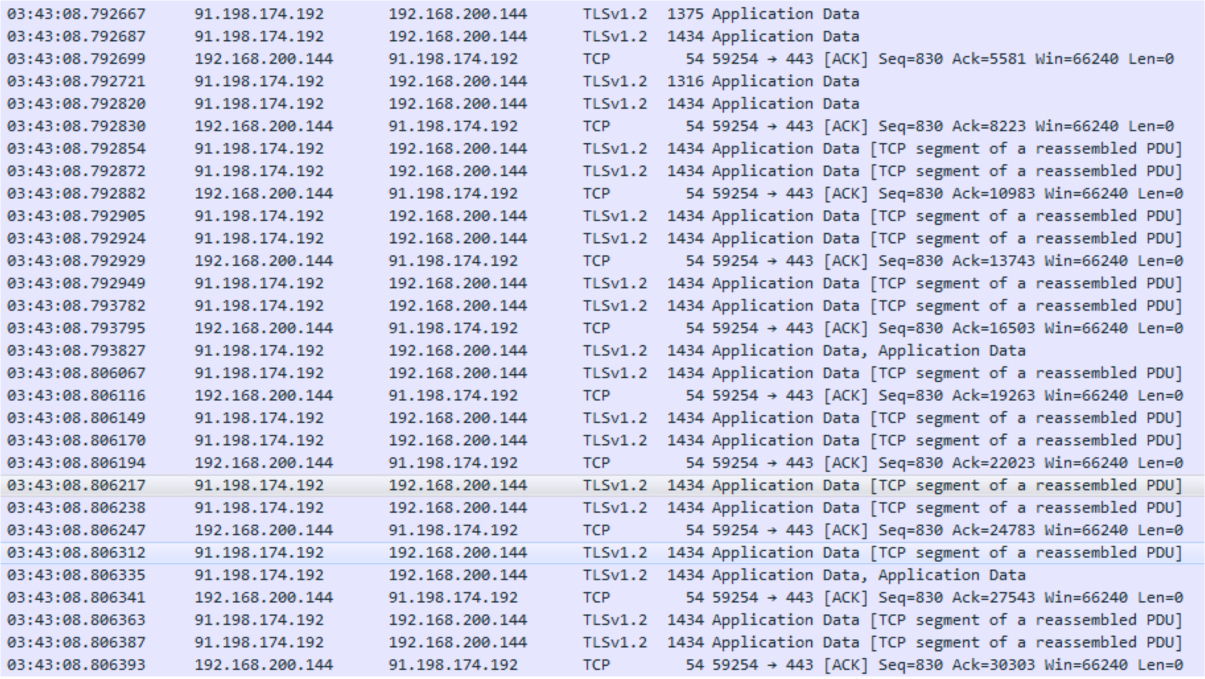
Table 4.11: Twitter user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Page loading	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Uploading photo	Stream	MTU (approximate)	Client → Server	TCP (ACK, PSH/ACK)

4.4.9 Wikipedia

Labelled as the free online encyclopaedia, Wikipedia is ranked as 5th among the top websites in the world and 8th among the top websites in the United Kingdom (Alexa, 2018). Being a website that possesses 6.77 Billion visits in January 2018 (SimilarWeb, 2018); it is an Internet based application worth considering for the analysis. The user interaction in Wikipedia that displayed a signature was page loading.

When the user searches for information, the resulting data is transmitted from port number 443 of the server to the user's computer as stream of TCP packets. The TCP packets had either ACK or PSH/ACK flags set in the transport layer. The client system acknowledged the reception of data with TCP packets that had the ACK flag set in the transport layer. The traffic obtained while performing the activity is shown in Figure 4.24.



```
03:43:08.792667    91.198.174.192     192.168.200.144    TLSv1.2  1375 Application Data
03:43:08.792687    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data
03:43:08.792699    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=5581 Win=66240 Len=0
03:43:08.792721    91.198.174.192     192.168.200.144    TLSv1.2  1316 Application Data
03:43:08.792820    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data
03:43:08.792830    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=8223 Win=66240 Len=0
03:43:08.792854    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.792872    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.792882    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=10983 Win=66240 Len=0
03:43:08.792905    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.792924    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.792929    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=13743 Win=66240 Len=0
03:43:08.792949    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.793782    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.793795    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=16503 Win=66240 Len=0
03:43:08.793827    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data, Application Data
03:43:08.806067    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806116    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=19263 Win=66240 Len=0
03:43:08.806149    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806170    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806194    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=22023 Win=66240 Len=0
03:43:08.806217    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806238    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806247    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=24783 Win=66240 Len=0
03:43:08.806312    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806335    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data, Application Data
03:43:08.806341    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=27543 Win=66240 Len=0
03:43:08.806363    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806387    91.198.174.192     192.168.200.144    TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
03:43:08.806393    192.168.200.144    91.198.174.192     TCP        54 59254 → 443 [ACK] Seq=830 Ack=30303 Win=66240 Len=0
```

Figure 4.24: Page loading in Wikipedia

There was only one user activity identified within the Wikipedia application. The activity and the corresponding key characteristics of the signature are listed in Table 4.12 below.

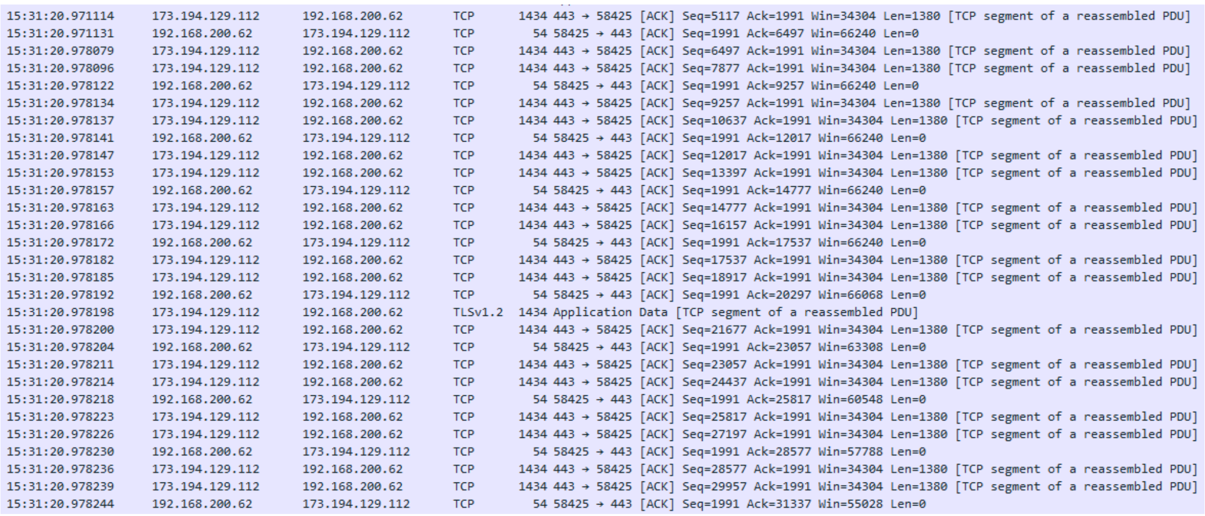
Table 4.12: Wikipedia user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Page loading	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)

4.4.10 YouTube

YouTube is the largest video sharing platform available on the Internet that allows its users to view, share and upload videos. The users can also rate and comment on the videos uploaded by other users and also subscribe to other users' channels. As of February 2018, YouTube is ranked 2nd among the top websites in the world and the United Kingdom (Alexa, 2018). With 26.09 Billion visits in January 2018 alone (SimilarWeb, 2018), YouTube can be resourceful when it comes to user activities. The main user interactions identified within YouTube were watching video and uploading video.

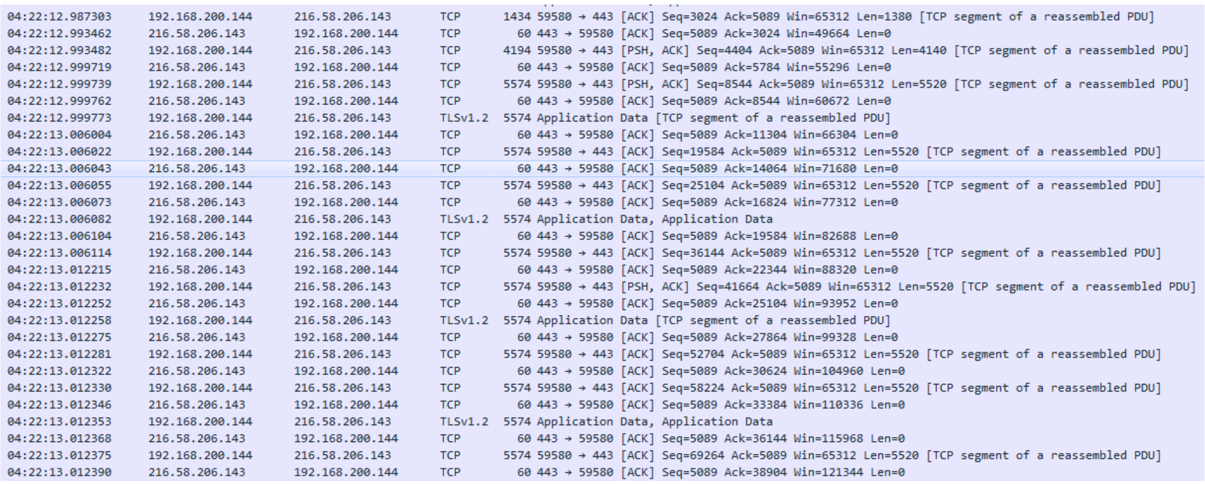
A pattern was identified when the user was watching video on YouTube, which is shown in Figure 4.25. During the activity, stream of TCP packets were sent from port number 443 of the server to the user's computer and the packets had a size that is almost equal to the MTU value. The TCP packets had either ACK or PSH/ACK flags set in the transport layer. The client system sent TCP acknowledgement packets, which had the ACK flag set in the transport layer, back to port number 443 of the server.



```
15:31:20.971114    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=5117 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.971131    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=6497 Win=66240 Len=0
15:31:20.978079    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=6497 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978096    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=7877 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978122    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=9257 Win=66240 Len=0
15:31:20.978134    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=9257 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978137    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=10637 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978141    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=12017 Win=66240 Len=0
15:31:20.978147    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=12017 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978153    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=13397 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978157    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=14777 Win=66240 Len=0
15:31:20.978163    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=14777 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978166    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=16157 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978172    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=17537 Win=66240 Len=0
15:31:20.978182    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=17537 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978185    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=18917 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978192    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=20297 Win=66068 Len=0
15:31:20.978198    173.194.129.112    192.168.200.62     TLSv1.2  1434 Application Data [TCP segment of a reassembled PDU]
15:31:20.978200    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=21677 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978204    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=23057 Win=63308 Len=0
15:31:20.978211    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=23057 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978214    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=24437 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978218    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=25817 Win=60548 Len=0
15:31:20.978223    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=25817 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978226    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=27197 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978230    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=28577 Win=57788 Len=0
15:31:20.978236    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=28577 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978239    173.194.129.112    192.168.200.62     TCP      1434 443 → 58425 [ACK] Seq=29957 Ack=1991 Win=34304 Len=1380 [TCP segment of a reassembled PDU]
15:31:20.978244    192.168.200.62     173.194.129.112    TCP        54 58425 → 443 [ACK] Seq=1991 Ack=31337 Win=55028 Len=0
```

Figure 4.25: Watching video on YouTube

Another activity for which a pattern was identified was when the user uploaded a video to YouTube. During the activity, the user's computer transmitted stream of TCP packets with the length equal to MTU to port number 443 of the server. The TCP packets had ACK or PSH/ACK flags set in the transport layer. The server returned the TCP acknowledgement packets, which had the ACK flag set in the transport layer. An example of the obtained traffic is displayed in Figure 4.26.



```
04:22:12.987303    192.168.200.144    216.58.206.143     TCP      1434 59580 → 443 [ACK] Seq=3024 Ack=5089 Win=65312 Len=1380 [TCP segment of a reassembled PDU]
04:22:12.993462    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=3024 Win=49664 Len=0
04:22:12.993482    192.168.200.144    216.58.206.143     TCP      4194 59580 → 443 [PSH, ACK] Seq=4404 Ack=5089 Win=65312 Len=4140 [TCP segment of a reassembled PDU]
04:22:12.999719    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=5784 Win=55296 Len=0
04:22:12.999739    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [PSH, ACK] Seq=8544 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:12.999762    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=8544 Win=60672 Len=0
04:22:12.999773    192.168.200.144    216.58.206.143     TLSv1.2  5574 Application Data [TCP segment of a reassembled PDU]
04:22:13.006004    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=11304 Win=66304 Len=0
04:22:13.006022    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=19584 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.006043    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=14064 Win=71680 Len=0
04:22:13.006055    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=25104 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.006073    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=16824 Win=77312 Len=0
04:22:13.006082    192.168.200.144    216.58.206.143     TLSv1.2  5574 Application Data, Application Data
04:22:13.006104    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=19584 Win=82688 Len=0
04:22:13.006114    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=36144 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.012215    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=22344 Win=88320 Len=0
04:22:13.012232    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [PSH, ACK] Seq=41664 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.012252    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=25104 Win=93952 Len=0
04:22:13.012258    192.168.200.144    216.58.206.143     TLSv1.2  5574 Application Data [TCP segment of a reassembled PDU]
04:22:13.012275    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=27864 Win=99328 Len=0
04:22:13.012281    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=52704 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.012322    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=30624 Win=104960 Len=0
04:22:13.012330    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=58224 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.012346    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=33384 Win=110336 Len=0
04:22:13.012353    192.168.200.144    216.58.206.143     TLSv1.2  5574 Application Data, Application Data
04:22:13.012368    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=36144 Win=115968 Len=0
04:22:13.012375    192.168.200.144    216.58.206.143     TCP      5574 59580 → 443 [ACK] Seq=69264 Ack=5089 Win=65312 Len=5520 [TCP segment of a reassembled PDU]
04:22:13.012390    216.58.206.143     192.168.200.144    TCP        60 443 → 59580 [ACK] Seq=5089 Ack=38904 Win=121344 Len=0
```

Figure 4.26: Uploading video to YouTube

The two user activities that were identified within YouTube and the corresponding key characteristics of the signature are listed in Table 4.13 below.

Table 4.13: YouTube user interactions - key characteristics

Interactions	No. of packets	Packet size (bytes)	Main direction	Protocol (flags)
Watching video	Stream	MTU (approximate)	Server → Client	TCP (ACK, PSH/ACK)
Uploading video	Stream	MTU (approximate)	Client → Server	TCP (ACK, PSH/ACK)

It is evident that when users interact with different Internet based services, the resulting network traffic can either be of an encrypted or of an unencrypted nature. Services that use encrypted traffic create a secure connection between the hosts and the transmitted data is encrypted using encryption algorithms (e.g., TLS). Although more and more services are opting for a secure connection, there are online services that still transmit plain-text data generating unencrypted traffic.

Based on the observations made during the analysis of network metadata resulted from user interactions that took place over a secure connection (HTTPS) between the hosts, the obtained patterns can be categorised into three types – a single packet category, multiple packets category and stream of packets category. This categorisation was based on the number of packets contained in the observed patterns. In contrast, when the hosts were communicating and the online user interactions were resulting in unencrypted traffic (HTTP) between the hosts, the corresponding patterns appeared as a stream of packets. These stream of packets were observed to be occurring when the SYN flag of the TCP protocol is set (i.e. when the 3-way handshake connection is made between the user's computer and the server) and when FIN flag of the TCP protocol is set (i.e. when the connection ends).

Thus signatures of all online user interactions, for the selected Internet based services (both encrypted and unencrypted), identified in the manual experiment conducted were observed to be constituted of either a single packet or multiple packets or a stream of packets. Based on the number of packets involved in the identification of each user interaction, the signatures obtained can be categorised into three groups – single-packet signature, multiple-packet signature and stream-of-packets signature.

For certain interactions during the manual analysis, when a user activity was carried out, a single packet with a unique payload length and TCP flag status appeared to be occurring. And this packet appeared repeatedly in all attempts made for that particular activity. In a multiple packet category, when a user activity took place the observed pattern usually contained between two and four packets. And these packets were transmitted within a timeframe of less than a millisecond. In the third category, the patterns obtained when certain user activities were performed can be represented as a stream of packets. In this type, the responsible packets were observed to be transmitted in a continuous fashion within a range of microseconds. Also, the payload size for these packets was the same, which was approximately equal to the MTU value. While the sending host was transmitting a stream of data packets the receiving host was returning a stream of acknowledgement packets to the sender. All online user interactions identified for the selected Internet based services and the key characteristics of obtained signatures are listed in Table 4.14.

Table 4.14: Online user interactions - key characteristics

Services	User Interactions	No. of Packets	Total Length (bytes)	Direction of packet(s)
BBC	Page navigation	Stream	Various	Server → Client
	Watching video	Stream	MTU (approximate)	Server → Client
	Listening to radio	Stream	MTU (approximate)	Server → Client
Dropbox	Downloading file	Stream	MTU (approximate)	Server → Client
	Uploading file	Stream	MTU (approximate)	Client → Server
Facebook	Page loading	Stream	MTU (approximate)	Server → Client
	Attaching file	Stream	MTU (approximate)	Client → Server
	Chat	Multiple	2625+	Client → Server
	Typing	Multiple	1502	Client → Server
Hotmail	File attachment	Stream	MTU (approximate)	Client → Server
	Composing email	One	1035	Server → Client
	Inserting recipient	One	971	Server → Client
Google Docs	Editing document	Multiple	2280	Client → Server
Google Search	Page navigation	Multiple	245	Server → Client
Skype	Text message	One	794+	Client → Server
	Audio call	Stream	104 to 147	Both clients
	Video call	Stream	1165 to 1382	Both clients
	File transfer	Stream	MTU (approximate)	Sending client → Receiving client
	Image transfer	Stream	MTU (approximate)	Client → Server (sending) Server → Client (receiving)
	Click on contact	One	747	Client → Server
	Idle	One	587	Client → Server
Twitter	Page loading	Stream	MTU (approximate)	Server → Client
	Uploading photo	Stream	MTU (approximate)	Client → Server
Wikipedia	Page loading	Stream	MTU (approximate)	Server → Client
YouTube	Watching video	Stream	MTU (approximate)	Server → Client
	Uploading video	Stream	MTU (approximate)	Client → Server

From the experiment conducted, a general structure for the signature can be drawn. Each signature is a unique pattern that will have a feature set that contains – start time of interaction, end time of interaction, server IP address, server port number, client port number, number of packets sent from client to server, total size of packets sent from client to server, number of packets sent from server to client, and total size of packets sent from server to client. While the elements such as the server IP, and port numbers are important in recognising the online services that are being used, the later elements of the feature set such as the number and total size of packets transferred between the servers and client, and the direction of packet transmission aid in narrowing down and identifying the online user activities.

In this research, all signatures for the selected services were synthesised by manually observing and analysing the network traffic during the initial stage of the experiment. Confirmation and validation of the signatures were obtained by repeating the manual procedure thirty times and independently between three researchers. The obtained signatures were stored and compared against while experimenting with a larger dataset, which was achieved through programing. During the automated analysis of the experiment, the dataset was be explored by applying filters such as server IP address, port numbers (HTTPS/HTTP), network layer protocols of interest (TCP/UDP), and flags that are set (PSH/ACK). The remaining features of the interactions such as the time duration of communication between the server and client, and total number and size of packets sent between the server and client will be calculated. This will start to form the information about the numerous interactions that took place between different servers and the client. Once the whole list of available interactions are generated, the list will be searched through to find any patterns that match with the stored signatures that were obtained during the manual analysis stage. Matches found during this search confirms the identification of the online user activities and will be stored in separate datasets for the proposed tool to utlise and perform multiple operations on.

As mentioned in the methodology section, the manual analysis experiment was conducted separately by three individual researchers and provided convincing results. However, a larger dataset was required to be tested on as the research desired to ensure the same outcome for more number of users and aimed to create a reliable basis for developing a UO-NFAT prototype for the later part of the research. The larger dataset used was collected by a postdoctoral researcher and was utilised as a common resource for two research projects including this one. In order to ensure reliability and to be scientifically sound, certain criteria were established before collecting the dataset and fulfilled during the collection process. Following are those criteria:

- The dataset must contain a sufficient number of participants to provide a basis for identifying the interactions for them.
- All users and the IP address of their systems must remain the same throughout the duration of data collection to ensure the relationship between the user and the IP address.
- The dataset must contain a sufficient number and type of samples for an adequate period of time to ensure identification of varieties of interactions.
- All network traffic metadata from all participants must be collected.

As the study involved the collection of traffic containing users' network activities ethical approval was obtained prior to the data collection. The dataset used was collected from 27 users during the period of 12 November 2014 to 20 January 2015. 18 of the users were full time PhD students situated in the same research laboratory. All network traffic metadata of these 18 users were monitored and captured via a network based approach. In order to meet the criterion of having fixed users and IP addresses, the DHCP on the host computers was disabled. The remaining 9 users were members of the university student population and their data was collected using a standalone application installed on their machines. Other than practicality, the reason for selecting the members from a graduate and post graduate population was the expected similarities in their Internet usage due to a shared background of being students and the idea that the behaviour of the group could be a close representation of staff behaviour within an organisational environment. All users were instructed not to allow anyone else to use their systems until the data collection was complete.

A dataset containing 63 Gigabytes of IP header information (metadata) was collected and the data for each individual user were stored in distinct SQLite database files. Each record of the database table contained the following fields – date and time stamp, sender's IP address, sender's port number, receiver's IP address, receiver's port number, packet length and the type of protocol (i.e. TCP/UDP) and the flags (i.e. SYN/ACK/PSH/FIN).

As mentioned in the data collection criteria, the initial dataset contains metadata of the whole network traffic. Therefore, pre-preocessing the metadata (prior to applying the user interaction signatures) was a crucial step so that the volume of the metadata that will undergo further analysis can be reduced. This objective was achieved by utilising different filters such as server IP addresses for the nine selected services and protocols and port numbers of interest. This allowed to eradicate any non-relevant traffic such as traffic associated with machine-to-machine network protocols and Content Delivery Networks (CDNs). Logically, this will result in a dataset that, in comparison, is lower in volume and higher in proportion of metadata associated with user related information. Data reduction achieved via pre-processing will reduce the time and computational power required for processing every single packet in the network.

Table 4.15 displays the data reduction that was achieved during the pre-processing stage. The total number of network metadata packets for each service, both prior to and after pre-processing are listed in the second and third column respectively. The total number of metadata packets were 49,675,318 at the beginning of preprocessing, which was reduced to 2,385,525 afterwards. It can be noticed that it aids in the reduction of metadata volume, which can then be analysed for the identification of online user activities. For instance, 68.9% of the total metadata was eradicated for the Skype service, leaving only the remaining 31.07% of metadata to be analysed in order to identify any matching signature(s).

Table 4.15: Data Reduction Results

Application	No. metadata packets	No. after pre-processing	Data Reduction %
YouTube	21,131,316	1,322,848	93.8
Facebook	5,727,953	386,741	93.3
Google	1,857,420	194,404	89.6
Twitter	747,584	71,403	90.5
Wikipedia	1,250,302	5,719	99.5
Hotmail	703,711	122,989	82.6
Dropbox	17,480,739	98,555	99.4
BBC	201,263	4,180	97.9
Skype	575,030	178,686	68.9

The results showed that the data volume reduction that expected as a part of the proposed approach was achieved. It was an indication of the results that the approach can potentially deliver with even bigger datasets in future tests. By performing this step, the researcher was able to validate the capability of the proposed approach to reduce the volume of metadata, which was a crucial element of the research objectives.

4.5 Discussion

Although thousands of Internet based services exist and numerous activities can be performed on those services, the experiment selected only nine services among the popular ones that exist. Nevertheless, the results obtained from the analysis of the nine selected services delivered sufficient evidence that the user activities can be extracted from the analysis of the low level network metadata. Obtained results were enough to justify the reason for not selecting a larger set of services for the experiment.

Furthermore, as explained in the principle, the interaction based approach performs the analysis only on the network metadata and the actual data content of the transmitted packets (payload) is untouched. Such a novel way of analysis was expected to enable the interaction based approach to overcome some of the challenges faced by the packet based network analysis approach. Results showed that the ability to focus only on the metadata allowed an enormous reduction in the volume of data to be analysed thereby reducing the computational power required. Focusing on metadata also created a way around the difficulty raised by the encrypted nature of data. Besides, by not examining the payload section of the network traffic the data privacy of the users was ensured. Another expected capability of the interaction based approach was to analyse the network metadata efficiently by drilling down to extract the specific packet(s) that resulted from a particular user interaction overcoming the challenge raised by the flow based approach (in which a flow record could contain metadata that originated as a result of multiple number of user interactions).

It has to be highlighted that the interaction based analysis approach is capable of identifying signatures that are targeted on the core user interactions instead of a wider range of interactions such as advertisements and supplementary data (such as JavaScript, CSS and images). It is important to exclude traffic that is generated from the transmission of such data as it does not contribute to the identification of user activities. By eliminating traffic of such kind, the volume of data that needs analysis is further reduced.

Given the current state of web applications, there can be services that are interconnected. This may lead to scenarios that could affect the level of detail obtained by the interaction based approach. For example, a user may watch a YouTube video that was shared on Facebook. The interaction based analysis will identify this as two separate interactions – a YouTube interaction of watching video and a Facebook interaction of page loading – instead of the two activities being identified as part of the same sequence. Even though this can be a limitation, the interaction based approach manages to apprehend the nature of both interactions thereby minimising the impact of the limitation.

Automating the system to generate signatures for more user activities and services can be a huge benefit to optimise the process and improve the efficiency of the system. It is worth pointing out that while the signatures that were observed while conducting the experiments served well in proving the research proposal, it is possible that signatures of other nature may exist for the same user activities. Only further research can explore the feasibility of existence of such signatures. The researcher believes that, if exists, then a comparison of different varieties of signatures could reveal more about the efficiency of results achieved through this research. Likewise, experimenting with network metadata in different geographical locations may disclose strengths or weaknesses of the interaction based approach. These can contribte towards the optimisation of the system. However, these factors are beyond the scope of this research at this point in time.

As mentioned in methodology, the experiment was designed to analyse the network metadata collected from a computer that had access to the Internet via the Ethernet cable. The experiment did not comprise the scenario in which the online user interactions take place in a wireless environment. The network traffic generated by a device that is connected to the Internet using the Wi-Fi technology (such as smartphones and tablets) could be different from the traffic produced in a wired Internet environment. The size difference of packets (such as TCP acknowledgement packets) and the possible delay and jitter that occur in Wi-Fi could be potential factors that make a difference. And therefore, the metadata analysis of Wi-Fi traffic may not produce the same results. Also, smartphones and tablets have integrated applications for all popular Internet based services, which is another factor that could result in different results. Therefore, traffic produced by such smartphone applications and under such situations needs to be analysed separately, which is beyond the scope of this research.

4.6 Conclusion

In order to reduce the effort exerted by the investigator along with reducing the cost and time of the investigation process, it was necessary to come up with a novel solution. Extraction of meaningful information (that information being the online user interactions) from low level network metadata was the proposed solution. In order to achieve this solution, an interaction based approach was hypothesised for the analysis of network metadata. To test the feasibility of the interaction based approach an experiment was designed that will manually analyse the network metadata that resulted from various user interactions performed within different Internet based services. The results obtained from the three individual researchers who performed the experiment independently showed sufficient evidence that the proposed approach is simultaneously capable of identifying the online user interactions and tackling the challenges caused due to the enormous volume and the encrypted nature of the network traffic.

The interaction based approach, compared to the flow based or packet based analysis approaches, provides the investigator with a much refined view that focuses on information linked to what the users were up to. Due to such a rich and sophisticated nature, the information provided could be a valuable resource that could reduce the investigator’s effort during the investigation.

5 User Oriented Network Forensic Analysis Tool: The Architecture

5.1 Introduction

A clear picture of the limitations of the current state of the art and the need for further research in the field of network forensics was established in the literature review chapters. Based on the review, need for a network forensic analysis tool (NFAT) that fulfils the requirements was envisioned. In order to structure such a tool, a novel approach for providing meaningful information about the user activities by analysing the low level network metadata was necessary. With this in mind, a novel interaction based analysis approach was hypothesised followed by designing an experiment to test the feasibility of the proposed approach, as discussed in chapter 4, which provided the researcher with results that assured the novel analysis approach's viability. As the successful testing of the interaction based approach provided a strong and scientific foundation for the envisioned tool, the next stage in the research was to present a novel architecture for the proposed User Oriented Network Forensic Analysis Tool (UO-NFAT). The Chapter begins with an appraisal of a set of requirements that must be met by the proposed system in order to achieve the research goal in Section 5.2. Section 5.3 of the chapter discusses the UO-NFAT architecture in detail along with the different entities (i.e. the processing engines and databases) that constitute the system architecture, followed by the discussion and conclusion in Section 5.4 and Section 5.5 respectively.

5.2 System Requirements

Before going further and discussing the system architecture, it is important to state the requirements that were identified for the proposed user oriented network forensic analysis tool (UO-NFAT) that could assist in the forensic investigation. The requirements were produced based on the nature of the proposed system as UO-NFAT is expected to assist the investigator by generating and presenting high level information from low level network metadata. The generated information should be a meaningful reflection of the employees' online activities within the organisational environment, which can help in the investigation of insider threats. Production of system requirements was also based on the limitations and challenges faced by the existing network forensic analysis tools. The requirements identified for the proposed UO-NFAT are listed below:

- The proposed system should be able to deal effectively with the issue caused by the enormous network traffic volume.
- Should be skilful in overcoming the difficulty of dealing with the encrypted traffic.
- Should have the ability to create high level information from raw network metadata.
- Should exhibit functionalities similar to that of the computer forensic tools such as case management, hashing, visualisation and reporting.
- Should effectively reduce the investigator's cognitive load.
- Should contribute towards reducing the overall time and cost of the investigation.
- Should provide an environment for different parties involved in the investigation to work in a collaborative manner.

Justification for the aforementioned set of system requirements took birth from the literature review phase of the research. Understanding the employee activities is important in resolving insider threat/misuse crimes. Computer Forensic tools play a huge role in the investigation of cybercrimes including insider threats within organisational environments. Sophisticated CF tools, such as the FTK and EnCase, with their advanced capabilities offer tremendous help in the investigation process. However, situations can arise in which the source of evidence (i.e., the hard disk drive) that the CF tools depend on have been tampered with. For instance, a technically competent employee can utilise one or more of the AF tools to hide the evidence, to eradicate them from the source or to place an attack on the CF tool itself. A challenge of such nature can either delay the investigation process or make it impossible to solve the case at all. In such situations, the organisational network traffic can be used as an alternative source to understand what the employees were up to. Since the network traffic contains a copy of the employees' online activities, an analysis of the traffic has the potential to provide valuable hints or evidences for resolving the case. Even in the absence of the aforementioned challenging situation, the network traffic can still be a useful source in insider threat investigation. For instance, the information gained from the analysis of the network traffic can help the investigator in narrowing down the number of individual computers that need to be analysed, which can help reduce the overall investigation time.

The literature review phase has identified limitations and challenges that current NF tools come with. Network activities on a normal day in an organisation can produce packets that are Terabytes in total size. Organisations, on a daily basis, could generate Gigabytes or Terabytes of Internet traffic, which might be data associated with work or personal interest.

While, as of June 2018, 74% of the total webpages loaded using the Google Chrome browser on Windows OS was encrypted (Google Transparency Report, 2018), Mozilla measured 70% of the total number of websites accessed through Firefox was encrypted as of April 2018 (Mozilla Internet Health Report, 2018). Analysing such an enormous volume of traffic to extract meaningful information will be a huge challenge; even more so with majority of the traffic encrypted.

An ideal NFAT should be able to handle the issues of large volume and encryption effectively. Extracting meaningful information about the employee activities is going to be crucial in determining their role in the event of an insider threat. The proposed system should be able to analyse the network metadata efficiently by incorporating an appropriate analysis technique (i.e. the interaction based analysis) in order to produce the online activities. In comparison with the established CF tools, current NFATs are limited in their ability to exhibit functionalities such as case management, hashing, visualisation, and reporting. Therefore, the proposed system should exhibit the aforementioned functionalities as it can effectively reduce the cognitive load on the investigator. It is possible that there might be multiple investigators working on the same case and the individuals are situated in different geographical locations. Therefore, capability to provide an environment that will allow different parties involved in the investigation to work in a collaborative manner should be a characteristic of the system (Drehel, 2019; AccessData, 2017; Miller, 2019). As briefed above, the limitations and challenges faced by the existing NFATs as well as the network traffic analysis techniques, and functionalities that can be adopted from established computer forensic tools are what have led to the generation of the system requirements. Listing the system requirements in the beginning of the Chapter will, later on, allow the researcher to verify whether or not the UO-NFAT architecture met those requirements.

5.3 The User Oriented Network Forensic Analysis Tool Architecture

Once the requirements that should be met by the proposed UO-NFAT were produced, the next step in the research was to design a novel architecture.

The system is designed to function in a web-based fashion in which the system will reside in a web server and the investigator will be interacting with the UO-NFAT from a local machine within the organisation intranet. The system is designed to function in a web-based fashion in which the system will reside in a web server that is set up inside the organisation. The interaction between the investigator and the tool will be made possible by means of the user interface that the tool provides via a web browser on the client system.

The standard N-tier web architecture was used as a reference point while creating the layered model of the UO-NFAT system, which is illustrated in Figure 5.1. The N-tier architecture is a standard model used in the development of client-server based web applications. The standard model consists of three layers – a Presentation layer, a Logic layer and a Data layer. Depending on whether the aforementioned layers are executed in a single machine or multiple machines, the standard model can be called a 1, 2 or 3-tier architecture; hence the name N-tier architecture (Lofberg and Molin, 2005; Papagelis, 2010; IBM, 2019).

The Presentation layer provides the user interface of the application and is responsible for handling the user interactions. The layer will pass the information received from user on to the second layer (i.e. the Logic layer). Likewise, information received in return will be presented to the user by rendering. Logic layer contains rules needed for processing the information received from the first and third layers of the architecture. Co-ordinating the other two layers, logical processing of the information, updating the other layers with latest/resulting information are among the tasks that will be performed by the Logic layer. This layer is capable of serving multiple users, which will play a crucial role while manifesting the collaborative working feature for the tool. The third layer, known as the Data layer, is in charge of the data storage used by the application. This layer will contain code for storing the information received from and/or retrieving/returning the stored information to the Logic layer. Depending on the nature of the developed application, the sources of data can be databases or file systems (Gartner, 2019; JReport, 2019).

As mentioned above, each of the three layers will be executed on different machines for a 3-tier web architecture. While the Presentation layer will run on the client computers, the Logic layer and the Data layer will run on the web server and the database server respectively.

The 3-tier web model can be considered unidirectional in its communication, in which each layer will only be communicating with the adjacent layer. For instance, the Presentation layer and the Data layer will not be talking to each other directly. The Logic layer, being the middle layer, will be able to communicate with the other two layers (Gartner, 2019).

The web application developed based on the 3-tier model will pack certain benefits such as maintainability, scalability, re-usability and security. Since the clients will not have direct access to the data stored in the backend database servers, the model provides increased security to the data. In this model, the three layers exist separated from each other, which offers ease when it comes to maintaining the layers independently (IBM, 2019). Figure 5.1 illustrates an overview of the different layers (i.e. the presentation, logic and data layers) of the UO-NFAT system and where they reside in an organisational network infrastructure.

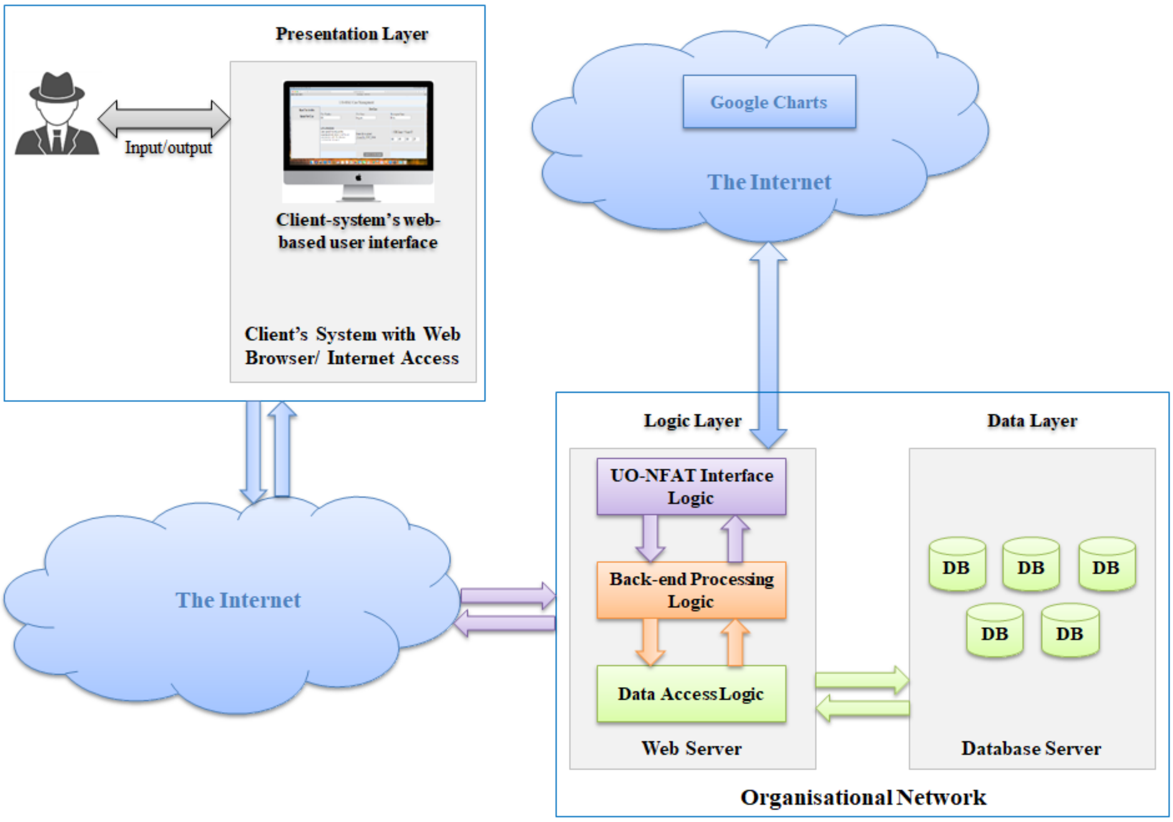


Figure 5.1: Layers of UO-NFAT Architecture

In the UO-NFAT structure, the 'Logic layer' can be considered to be constituted by the 'UO-NFAT Interface Logic', the 'Back-end Processing Logic' and the 'Data Access Logic' layer, as depicted in Figure 5.1. These three parts create and control the overall logic of the system. The engines that work in the background to facilitate the user interface constitute the 'UO-NFAT Interface Logic'. The 'case management engine', the 'visualisation-configuration engine', the 'visualisation engine' and the 'reporting engine' belong to the 'UO-NFAT Interface Logic'. The 'Back-end Processing Logic' part of the 'Logic layer' consists of the engines that carry out the back-end processes – the 'pre-processing engine' and the 'user-activity analyser'. The 'Data Access Logic' part of the 'Logic layer' will be responsible for the data querying operations. In a nutshell, all six engines of the UO-NFAT system will work together to form the Logic layer. The databases that store all kinds of relevant data form the third layer of the system (i.e. the 'Data layer').

The Presentation layer will pass the information received as a result of the interactions from the investigator to the Logic layer. Based upon the input received, the Logic layer will generate the logic required to process the information. The processing may require accessing data stored in the database server, which could include creating/modifying/deleting the data. The Data layer will select the appropriate data based on the queries received and return the results back to the Logic layer. The Logic layer will pass the resulting information back to the Presentation layer.

Although not labelled as a separate layer, there is an external link that connects the system to the Internet when the 'visualisation engine' and the 'reporting engine' utilise the Google Charts in order to plot multiple, interactive charts in real time. Nevertheless, this external link is equally important and is a part of the UO-NFAT system.

In comparison with a standalone design, the web-based client-server design will equip the tool with certain advantages, which will justify the decision for selecting the aforementioned design. The web-based design will deliver the developed tool software compatibility, allowing the investigators to access and interact with the system irrespective of the operating system running on the client machines. Investigators would be needing only a compatible web browser and an Internet connection that is fast enough in order to utilise the tool. Another advantage of this design is that it will aid the tool in facilitating collaborative working. Sometimes, it is possible that the work-load of an investigation is split between multiple investigators. The web-based design of the tool will allow the members of the investigation team to work collaboratively on the same case allowing them to refine analysis results according to interest, insert comments and bookmark interesting results. The web-based design, unlike the standalone design, will give the tool access to more powerful and efficient hardware resources such as the processor, memory and storage. Considering the enormous volume of network metadata that needs processing, the tool will certainly require dedicated backend servers. The tool can utilise resources of the server to process and analyse large volumes of network metadata and to store the results, meaning that the tool's performance will not be limited by the resources of the investigator's machine. Finally, ease of upgradation of the tool is another advantage that the web-based design offers. Because of the client-server nature, any upgradation of the tool will be carried out on the server-side. Therefore, unlike in a standalone version of the tool, the investigators would not have to worry about upgrading when the latest version of the tool is available.

Since being modelled on the 3-tier client-server architecture, the tool will be benefitted from more advantages such as flexibility, maintainability, re-usability, scalability and reliability. Database security will be ensured as the Logic layer separates the Presentation and Data layers and hence the clients will not have direct access to the databases preventing any illegitimate data modification. It is worth pointing out that the three layers of the aforementioned model can reside in one or more physical devices such as a web server, application server and a database server. Having a separate web server and database server can increase data security in the event of any attack on the web server. Also, introducing more than one web servers can aid the model to serve numerous clients while being able to deal with any downtime effectively. Similarly, including more than one database servers can ensure that necessary data is available at all time by one of the data servers being able to take over when the other server is down.

The layered overview of the UO-NFAT system, shown in Figure 5.1 is further expanded in order to form the fully developed architecture consisting of all vital processing modules of the system, which is exemplified in Figure 5.2. The fully developed architecture consists of all elements of the UO-NFAT system and shows the interconnections between those elements and their arrangement. A detailed explanation that follows the architecture helps the reader to understand how each element functions and how they work together to implement the system (i.e. the UO-NFAT). The UO-NFAT architecture is designed based on a case management premise, which will help in maintaining chain of custody and integrity of the evidence. Along with case management capability, the proposed tool is also expected to incorporate functionalities such as hashing, visualisation and reporting, that are exhibited by established computer forensic tools. Ultimately, a tool implemented based on this architecture is expected to reduce the effort exerted by the investigator in the overall investigation process.

The architecture constitutes of the case management engine, the pre-processing engine, the user-activity analyser, the visualisation-configuration engine, the visualisation engine and the reporting engine that enables the UO-NFAT interface, the back-end processing and the database operations. Also, there are multiple databases that contribute towards fulfilling the architecture. Starting from Subsection 5.3.1 through Subsection 5.3.6, a detailed discussion of each component of the UO-NFAT architecture is presented. The discussion is divided into six subsections, each covering each of the six engines and the databases that are associated with each engine. Discussion of the system's engines may contain usage of a limited number of screenshots of the prototype as examples. Those screenshots are used merely to aid understanding of the core functionality and do not represent a definitive implementation of the system.

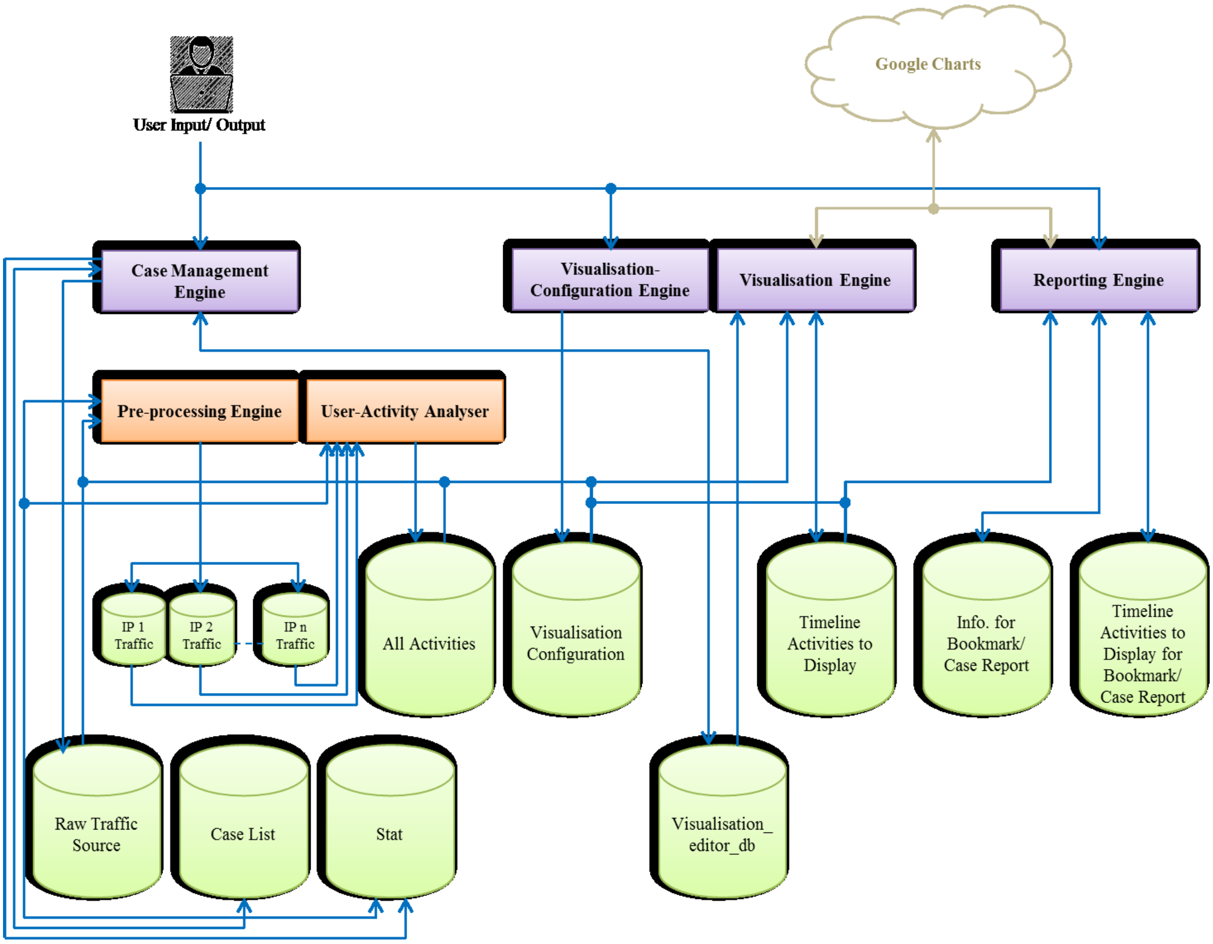


Figure 5.2: The UO-NFAT Architecture

5.3.1 The Case Management Engine

Being a part of the 'UO-NFAT interface logic layer', the case management engine is an entity of the system architecture that involves interaction with the investigator. The engine consists of four core parts – an 'AAA manager', a 'case archive manager', a 'visualisation editor', and a 'new case creator' – through which the system maintains the integrity of the evidence and chain of custody, permits collaborative working, and aids in the overall ease of analysis and reporting to the investigator.

5.3.1.1 AAA Manager

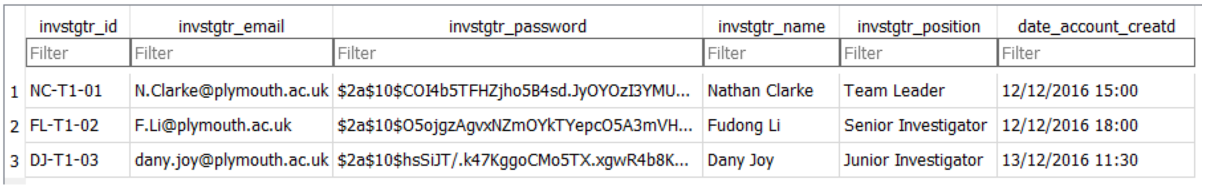
The Authentication, Authorisation and Accounting (AAA) functionality will allow the tool to ensure that only the authenticated individuals are given access to the tool, each investigator is provided with the appropriate set of operational rights depending on their designation, and a log of activities that are carried out by the investigators is maintained. The 'case management engine' will be responsible for enabling and controlling this functionality. Integration of AAA will benefit the UO-NFAT system not only to secure itself but also to contribute towards maintaining the chain of custody.

In order to facilitate the AAA functionality, the UO-NFAT design will be prompting the investigators to create their own user accounts and login to the account prior to start using the tool. The engine will supply the appropriate options via the tool's user interface. Also, in association with this, the engine will create an 'AAA_primary' table to store the details of the investigators.

The 'case management engine' will create an 'AAA_primary' table to store the necessary details. The database will contain six fields – 'invstgtr_id', 'invstgtr_email', 'invstgtr_password', 'invstgtr_name', 'invstgtr_position', and 'date_account_creatd'. The fields – 'invstgtr_email', 'invstgtr_password', 'invstgtr_name' and 'invstgtr_position' – will store the email address, password, name and their role within the investigation team respectively for each investigator. The engine will populate these fields using the mandatory data that were gathered from the investigator during the creation of the account. The 'invstgtr_id' and the 'date_account_creatd' fields will contain a unique ID for the investigator and the date/time of the account creation respectively. Unlike the other fields of the 'AAA_primary' table, values of these two fields will be generated automatically by the engine. Upon completion of the account creation process, the engine will notify the investigator by returning the unique investigator-ID.

The table will maintain a record for each investigator, and will be updated according to the changes (i.e. addition or removal of investigator(s)) that will occur within the investigation team. The 'AAA_primary' table will be utilised by the engine while carrying out each part of the AAA process. The 'invstgts_id', 'invstgtr_email' and the 'invstgtr_password' fields will be used while conducting authentication. While the ‘invstrgtr_position’ field will be utilised for assigning each individual with appropriate privileges (i.e. authorisation), the ‘invstgtr_name’ field will be used for logging the activities (i.e. accounting).

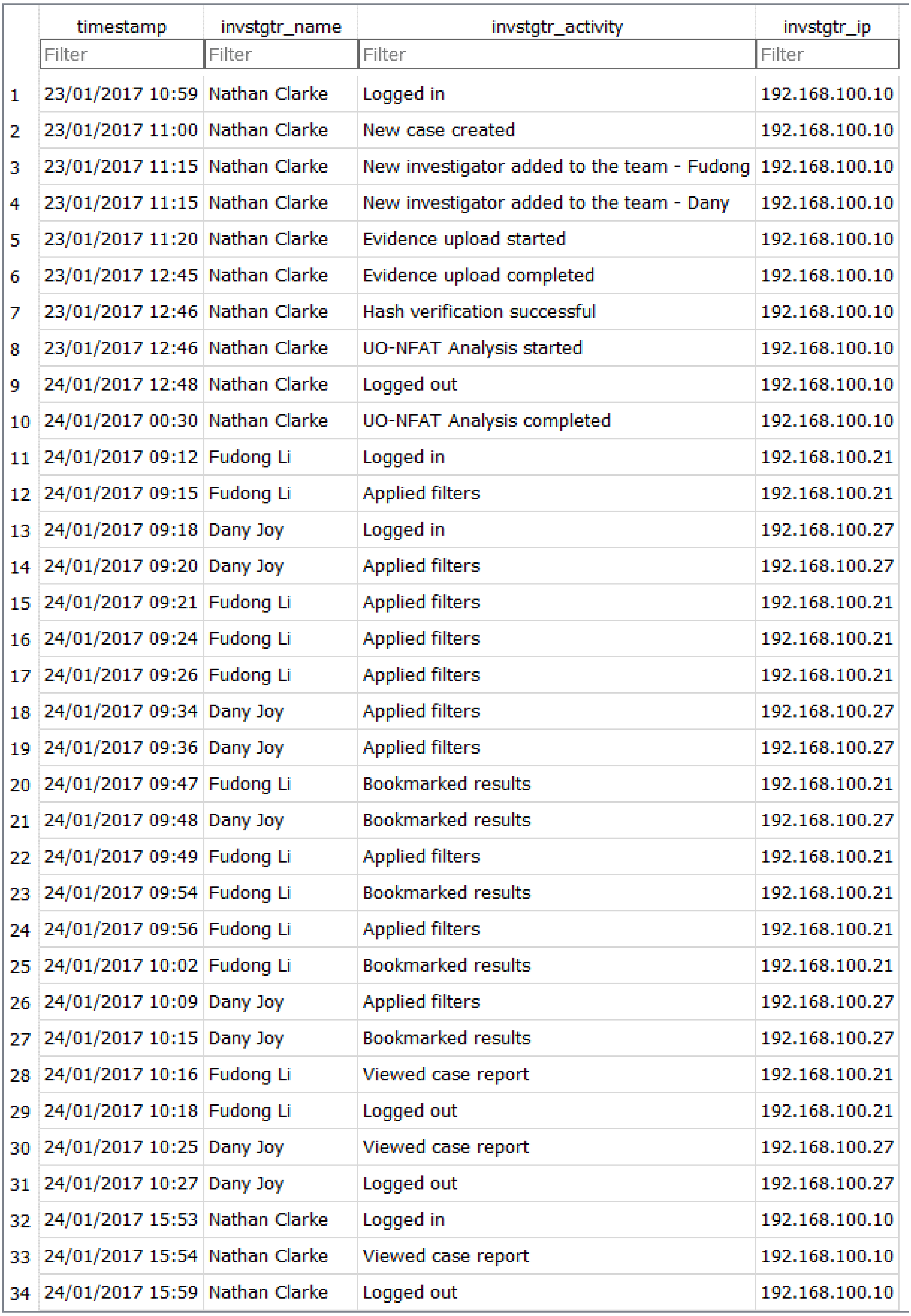
The investigator passwords will not be stored in a plain-text format as it will maximise the possibility of password revelation in the event of a tool exploitation (Convery, 2007). Therefore, during account creation, the engine will use a password hashing function such as BCrypt or SCrypt that has a high immunity against attacks (Laravel Docs, 2018; PHP Docs, 2018). The generated hash value will be stored in the ‘invstgtr_password’ field of the ‘AAA_primary’ database. During the login process, the engine will accept either the email address or the investigator-ID along with the password. The engine will authenticate the investigator that provides matching values for the ‘invstgtr_name’ or ‘invstgtr_id’ field and the corresponding ‘invstgtr_password’ field. For password verification, the engine will use the password supplied and the hash value stored in the database. An example of the ‘AAA_primary’ table is shown in Figure 5.3.



	invstgtr_id	invstgtr_email	invstgtr_password	invstgtr_name	invstgtr_position	date_account_creatd
	Filter	Filter	Filter	Filter	Filter	Filter
1	NC-T1-01	N.Clarke@plymouth.ac.uk	$2a$10$COI4b5TFHZjho5B4sd.JyOYOzI3YMU...	Nathan Clarke	Team Leader	12/12/2016 15:00
2	FL-T1-02	F.Li@plymouth.ac.uk	$2a$10$O5ojgzAgvxNZmOYkTYepcO5A3mVH...	Fudong Li	Senior Investigator	12/12/2016 18:00
3	DJ-T1-03	dany.joy@plymouth.ac.uk	$2a$10$hsSiJT/.k47KggoCMo5TX.xgwR4b8K...	Dany Joy	Junior Investigator	13/12/2016 11:30

Figure 5.3: AAA_primary (AAA Manager)

It can be noticed from the table that the ‘invstgr_id’ field has unique values for each row (i.e. investigator). For instance, this value can be a string that is a combination of the first and last letters of the investigator’s name, the team number that the investigator belongs to and the row-ID of the corresponding record within the table.



	timestamp	invstgtr_name	invstgtr_activity	invstgtr_ip
	Filter	Filter	Filter	Filter
1	23/01/2017 10:59	Nathan Clarke	Logged in	192.168.100.10
2	23/01/2017 11:00	Nathan Clarke	New case created	192.168.100.10
3	23/01/2017 11:15	Nathan Clarke	New investigator added to the team - Fudong	192.168.100.10
4	23/01/2017 11:15	Nathan Clarke	New investigator added to the team - Dany	192.168.100.10
5	23/01/2017 11:20	Nathan Clarke	Evidence upload started	192.168.100.10
6	23/01/2017 12:45	Nathan Clarke	Evidence upload completed	192.168.100.10
7	23/01/2017 12:46	Nathan Clarke	Hash verification successful	192.168.100.10
8	23/01/2017 12:46	Nathan Clarke	UO-NFAT Analysis started	192.168.100.10
9	24/01/2017 12:48	Nathan Clarke	Logged out	192.168.100.10
10	24/01/2017 00:30	Nathan Clarke	UO-NFAT Analysis completed	192.168.100.10
11	24/01/2017 09:12	Fudong Li	Logged in	192.168.100.21
12	24/01/2017 09:15	Fudong Li	Applied filters	192.168.100.21
13	24/01/2017 09:18	Dany Joy	Logged in	192.168.100.27
14	24/01/2017 09:20	Dany Joy	Applied filters	192.168.100.27
15	24/01/2017 09:21	Fudong Li	Applied filters	192.168.100.21
16	24/01/2017 09:24	Fudong Li	Applied filters	192.168.100.21
17	24/01/2017 09:26	Fudong Li	Applied filters	192.168.100.21
18	24/01/2017 09:34	Dany Joy	Applied filters	192.168.100.27
19	24/01/2017 09:36	Dany Joy	Applied filters	192.168.100.27
20	24/01/2017 09:47	Fudong Li	Bookmarked results	192.168.100.21
21	24/01/2017 09:48	Dany Joy	Bookmarked results	192.168.100.27
22	24/01/2017 09:49	Fudong Li	Applied filters	192.168.100.21
23	24/01/2017 09:54	Fudong Li	Bookmarked results	192.168.100.21
24	24/01/2017 09:56	Fudong Li	Applied filters	192.168.100.21
25	24/01/2017 10:02	Fudong Li	Bookmarked results	192.168.100.21
26	24/01/2017 10:09	Dany Joy	Applied filters	192.168.100.27
27	24/01/2017 10:15	Dany Joy	Bookmarked results	192.168.100.27
28	24/01/2017 10:16	Fudong Li	Viewed case report	192.168.100.21
29	24/01/2017 10:18	Fudong Li	Logged out	192.168.100.21
30	24/01/2017 10:25	Dany Joy	Viewed case report	192.168.100.27
31	24/01/2017 10:27	Dany Joy	Logged out	192.168.100.27
32	24/01/2017 15:53	Nathan Clarke	Logged in	192.168.100.10
33	24/01/2017 15:54	Nathan Clarke	Viewed case report	192.168.100.10
34	24/01/2017 15:59	Nathan Clarke	Logged out	192.168.100.10

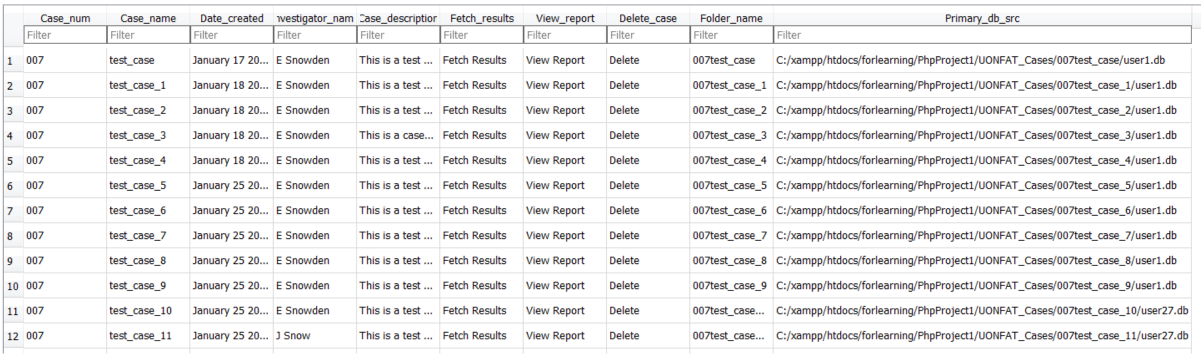
Figure 5.4: Accounting (AAA Manager)

The engine will create an ‘accounting’ table in association with each case in order to log information about the activities. The table will contain four fields – ‘timestamp’, ‘invstgtr_name’, ‘invstgtr_activity’, and ‘invtgtr_ip’. While the ‘timestamp’ field will store the date/time of occurrence of each activity, the ‘invstgtr_name’ field will contain the name of the investigator that is accountable for the corresponding activity. The ‘invstgtr_activity’ field will store the activity that was carried out and the ‘invstgtr_ip’ will store the IP address the investigator logged in from during the session. The AAA Manager will be able to keep a log of activities that were carried out by the investigators. An example of the ‘accounting’ table is shown in Figure 5.4.

5.3.1.2 Case Archive Manager

The key objective of the ‘case archive manager’ is to provide the investigator with complete information about the previously analysed cases. The information provided contains details such as case number, case name, date and time of creation, name of the investigator and case description. Alongside the case details, the case archive manager allows the investigator to take certain actions on the cases. The investigator can either fetch the previous analysis results or view the report that was generated previously or delete the entire case from the archive.

The case archive manager retrieves this information from a database named ‘case list’ that is created by the system and is part of the ‘database layer’. The contents of the case list database are presented in Figure 5.5.



	Case_num	Case_name	Date_created	ıvestigator_nam	Case_descriptior	Fetch_results	View_report	Delete_case	Folder_name	Primary_db_src
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	007	test_case	January 17 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case/user1.db
2	007	test_case_1	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_1	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_1/user1.db
3	007	test_case_2	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_2	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_2/user1.db
4	007	test_case_3	January 18 20...	E Snowden	This is a case...	Fetch Results	View Report	Delete	007test_case_3	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_3/user1.db
5	007	test_case_4	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_4	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_4/user1.db
6	007	test_case_5	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_5	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_5/user1.db
7	007	test_case_6	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_6	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_6/user1.db
8	007	test_case_7	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_7	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_7/user1.db
9	007	test_case_8	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_8	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_8/user1.db
10	007	test_case_9	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_9	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_9/user1.db
11	007	test_case_10	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case...	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_10/user27.db
12	007	test_case_11	January 25 20...	J Snow	This is a test ...	Fetch Results	View Report	Delete	007test_case...	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_11/user27.db

Figure 5.5: 'Case list' database (Case Archive Manager)

Each record in this database holds information about the previously created cases. It can be noticed that apart from the information presented to the investigator, the case list database consists of certain data fields, such as folder_name and primary_db_src that are private to the tool. These two data fields are used internally by the system for locating the source file containing the raw network metadata that was uploaded to the web server for analysis. The action of keeping these two fields undisclosed is taken by the UO-NFAT system in order to maintain the integrity of the source of evidence.

5.3.1.3 Visualisation Editor

The second component of the 'case management engine' is the 'visualisation editor', which provides the investigator with the option to edit the 'visualisation window' of the tool. In other words, the 'visualisation editor' allows the investigator to choose the type of visualisation chart that will be displayed inside the visualisation window of the UO-NFAT's dashboard.

Fundamentally, the 'visualisation editor' element of the 'case management engine' carries out three operations. The first operation (i.e. the 'add' operation) is allowing the addition of charts that the investigator thinks are appropriate for the current case. The second operation (i.e. the 'remove' operation) is allowing the removal of charts that the investigator finds inadequate. The final operation carried out by the 'visualisation editor' is allowing the addition of fresh charts into the existing pool by connecting remotely to servers that offer visualisation charts service using their APIs.

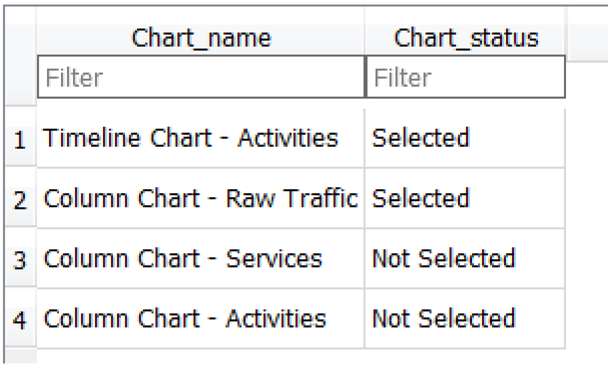
Through its user interface window, the 'visualisation editor' will display the list of visualisation charts that have been utilised in all cases investigated so far. The investigator can enable the 'visualisation editor' to perform the first or second of the aforementioned operations by utilising the 'add' or 'remove' buttons made available on the user interface window.

By default, the 'visualisation editor' will display the type/name of the charts that were used in the most recently investigated case as selected (i.e. ticked in the checklist) and the ones that were used in the cases prior to the most recent one as un-selected (i.e. un-ticked in the checklist). The investigator is allowed to add more charts to or remove the already selected charts from the available list. Or the current options can be left as they are.

In order to enable the 'visualisation editor' to perform the third operation (i.e. the 'add new' operation), the investigator can make use of the 'add new' button displayed on the user interface window. Upon selecting this option, the 'visualisation editor' will open a new user interface window.

By providing the URL for the visualisation API through this interface window, the investigator can enable the ‘case management engine’ to connect to the server and receive the list of available charts and their descriptions. Following this, by selecting the name, the investigator can make the engine include a fresh chart into the pool of existing visualisation charts.

In order to facilitate all three of the aforementioned operations, a database is necessary. A database to store the names of the visualisation charts and their corresponding status values (i.e. selected or not-selected) would have been created by the ‘case management engine’ when the tool was used for the first time. The then created database is common to all UO-NFAT cases and an example of this database, called the ‘visualisation_editor_db’, is shown in Figure 5.6.



	Chart_name	Chart_status
	Filter	Filter
1	Timeline Chart - Activities	Selected
2	Column Chart - Raw Traffic	Selected
3	Column Chart - Services	Not Selected
4	Column Chart - Activities	Not Selected

Figure 5.6: 'Visualisation_editor_db' database (Visualisation Editor)

The ‘visualisation_editor_db’ database has two fields – a ‘Chart_name’ field to hold the name of the visualisation chart and a ‘Chart_status’ field to hold the corresponding status value. Each time when the investigator is interested in editing the visualisation window of the tool’s dashbaord, the ‘visualisation editor’ segment of the ‘case management engine’ will read all records from the ‘visualisation_editor_db’ database and display them to the investigator via the corresponding user interface. Depending on the modifications made by the investigator during the chart selection process (i.e. during the ‘add’ or ‘remove’ operations), the database will be updated by writing the updated status values (i.e. ‘selected’ or ‘not selected’) into the corresponding ‘Chart_status’ field. During the third operation (i.e. the ‘add new’ operation), the database will be updated by adding a new record containing the name of the new visualisation chart that was selected by the investigator via the visualisation API.

It is important to point out that the status value of the timeline chart for displaying the chronology of user interactions will be set as ‘selected’, by default. This is due to the significance that the timeline of user activities has in the investigation. Also, it has to be mentioned that the ‘visualisation_editor_db’ will be accessed by the ‘visualisation engine’ at a later point prior to the engine plotting the visualisation charts into the ‘visualisation window’ of the UO-NFAT dashboard.

5.3.1.4 New Case Creator

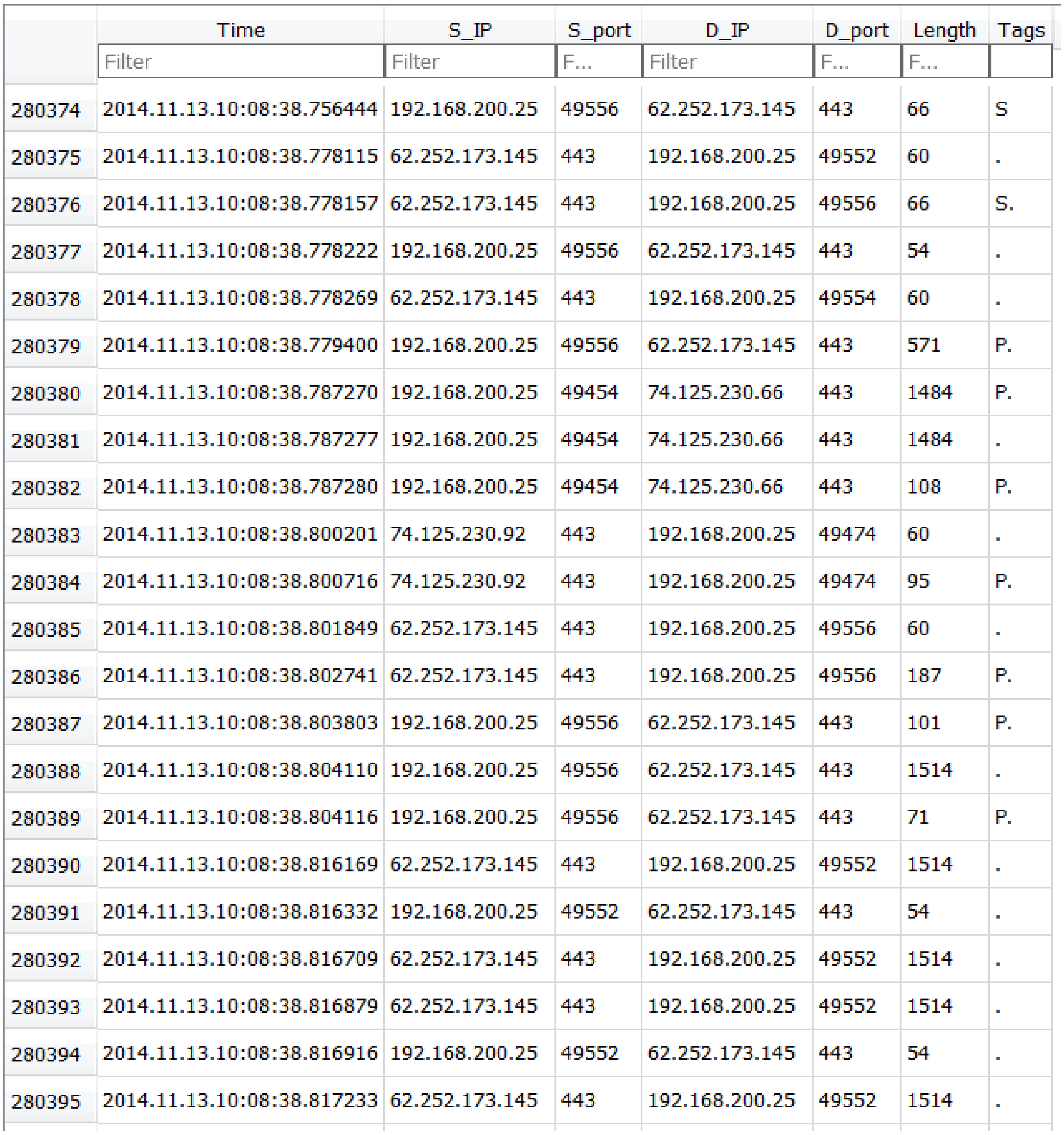
The ‘new case creator’, as the name implies, focuses on creating a new case file prior to the pre-processing and the analysis. This segment of the ‘case management engine’ is aimed at collecting the input information that the investigator has to provide.

All mandatory information including the case number, the case name, the name of the investigator and the case description needs to be fed into the system at this point. Following this step, the source database file containing the raw network meta-data needs to be selected for upload.

Furthermore, the investigator should provide the system with either a particular IP address or the Classless Inter-Domain Routing (CIDR) range that is required for the analysis. A specific IP address is given as the input when the investigator is certain about the IP address of the host that needs to be analysed for identifying the user activities. The CIDR option could be utilised in a scenario that will allow the investigator to view the available user activities generated by the active IPs among the 254 host IP addresses in that particular class C network (assuming that the first 24 bits of the IP address given in the CIDR notation represent the network). For instance, in Figure 5.10 it can be noticed that the investigator wants to analyse the raw traffic file (TEST_DB.db, in this case) for identifying the online activities of a specific user whose system IP address is 192.168.200.25. However, if the investigator wishes to analyse the traffic for another IP address then the UO-NFAT would facilitate this by providing an option to analyse a new IP at a later stage from the UO-NFAT dashboard. This option will be explained in section 5.3.5 as this option is initiated by the ‘visualisation engine’.

An assumption made while designing the UO-NFAT architecture was that the investigator must provide an IP address of interest from an analysis perspective, which needs to be fed into the system. An alternative to this option is to modify the architecture that will allow the system to determine the user IP addresses that are present in the uploaded network traffic. This can be achieved by carefully locating where the 3-way handshake connections are initiated and ended and thereby identifying the active user IP addresses within the subnet. But one drawback to this approach is that if the collected network traffic does not contain packet information about the connection establishment or ending, then the system will not be able to identify the presence of such IP addresses. This can lead the system not to analyse the corresponding traffic thereby not generating any user activities. This justifies the requirement for either a specific IP address or the CIDR range to be supplied by the investigator.

Based on the aforementioned input information supplied by the investigator, the ‘new case creator’ segment of the case management engine will create a system folder for the case that is currently being created for investigation. Name for this newly created system folder will be a combination of the case number and the case name provided by the investigator. The selected source file (i.e. the network metadata) will be uploaded into the web server and stored as a database called the ‘raw traffic source’ inside the case folder that was just created in preparation for the pre-processing and analysis. An example for the content of the ‘raw traffic source’ is shown in Figure 5.7.

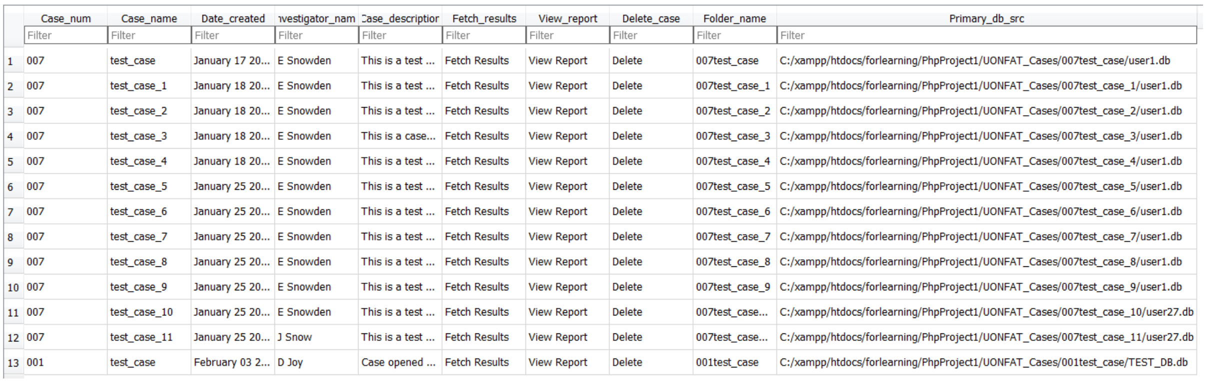


	Time	S_IP	S_port	D_IP	D_port	Length	Tags
	Filter	Filter	F...	Filter	F...	F...	
280374	2014.11.13.10:08:38.756444	192.168.200.25	49556	62.252.173.145	443	66	S
280375	2014.11.13.10:08:38.778115	62.252.173.145	443	192.168.200.25	49552	60	.
280376	2014.11.13.10:08:38.778157	62.252.173.145	443	192.168.200.25	49556	66	S.
280377	2014.11.13.10:08:38.778222	192.168.200.25	49556	62.252.173.145	443	54	.
280378	2014.11.13.10:08:38.778269	62.252.173.145	443	192.168.200.25	49554	60	.
280379	2014.11.13.10:08:38.779400	192.168.200.25	49556	62.252.173.145	443	571	P.
280380	2014.11.13.10:08:38.787270	192.168.200.25	49454	74.125.230.66	443	1484	P.
280381	2014.11.13.10:08:38.787277	192.168.200.25	49454	74.125.230.66	443	1484	.
280382	2014.11.13.10:08:38.787280	192.168.200.25	49454	74.125.230.66	443	108	P.
280383	2014.11.13.10:08:38.800201	74.125.230.92	443	192.168.200.25	49474	60	.
280384	2014.11.13.10:08:38.800716	74.125.230.92	443	192.168.200.25	49474	95	P.
280385	2014.11.13.10:08:38.801849	62.252.173.145	443	192.168.200.25	49556	60	.
280386	2014.11.13.10:08:38.802741	62.252.173.145	443	192.168.200.25	49556	187	P.
280387	2014.11.13.10:08:38.803803	192.168.200.25	49556	62.252.173.145	443	101	P.
280388	2014.11.13.10:08:38.804110	192.168.200.25	49556	62.252.173.145	443	1514	.
280389	2014.11.13.10:08:38.804116	192.168.200.25	49556	62.252.173.145	443	71	P.
280390	2014.11.13.10:08:38.816169	62.252.173.145	443	192.168.200.25	49552	1514	.
280391	2014.11.13.10:08:38.816332	192.168.200.25	49552	62.252.173.145	443	54	.
280392	2014.11.13.10:08:38.816709	62.252.173.145	443	192.168.200.25	49552	1514	.
280393	2014.11.13.10:08:38.816879	62.252.173.145	443	192.168.200.25	49552	1514	.
280394	2014.11.13.10:08:38.816916	192.168.200.25	49552	62.252.173.145	443	54	.
280395	2014.11.13.10:08:38.817233	62.252.173.145	443	192.168.200.25	49552	1514	.

Figure 5.7: 'Raw traffic source' database (New Case Creator)

The ‘raw traffic source’ database (the content being the raw network metadata that was uploaded for analysis) will have seven fields such as the date and time stamp, the IP address of the source, the port number on the source, the IP address of the destination, the port number on the destination, the length of the packet and the tag that reveals the type of prototype (such as TCP or UDP) along with the set TCP flags (such as SYN, ACK, PSH or FIN). These seven fields are chosen as they are the basis for the unique signatures that were observed for the online user interactions.

It has to be pointed out that all six inputs – case number, case name, and name of the investigator, case description, source file to upload and the IP address of interest – are mandatory for the pre-processing and analysis to commence. Therefore, if any of the input information field is empty in the user interface then the system will notify the investigator to complete the information in order to proceed. All pieces of information, along with the date and time of creation, will be stored in the 'case list' database. As mentioned above, the folder name and the destination path to the source file selected for uploading are generated by the engine internally. The date and time of creation is also assigned by the case management engine based on the local date/time. The newly created case will be added as a new record into the 'case list' database and the database structure is shown in Figure 5.8.

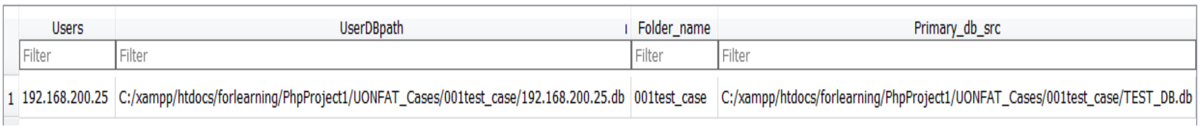


	Case_num	Case_name	Date_created	nvestigator_nam	Case_descriptior	Fetch_results	View_report	Delete_case	Folder_name	Primary_db_src
	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	007	test_case	January 17 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case/user1.db
2	007	test_case_1	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_1	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_1/user1.db
3	007	test_case_2	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_2	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_2/user1.db
4	007	test_case_3	January 18 20...	E Snowden	This is a case...	Fetch Results	View Report	Delete	007test_case_3	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_3/user1.db
5	007	test_case_4	January 18 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_4	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_4/user1.db
6	007	test_case_5	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_5	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_5/user1.db
7	007	test_case_6	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_6	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_6/user1.db
8	007	test_case_7	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_7	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_7/user1.db
9	007	test_case_8	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_8	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_8/user1.db
10	007	test_case_9	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case_9	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_9/user1.db
11	007	test_case_10	January 25 20...	E Snowden	This is a test ...	Fetch Results	View Report	Delete	007test_case...	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_10/user27.db
12	007	test_case_11	January 25 20...	J Snow	This is a test ...	Fetch Results	View Report	Delete	007test_case...	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/007test_case_11/user27.db
13	001	test_case	February 03 2...	D Joy	Case opened ...	Fetch Results	View Report	Delete	001test_case	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/001test_case/TEST_DB.db

Figure 5.8: 'Case list' database (New Case Creator)

In the database, it can be noticed that a new record (entry number 13) has been added containing all information provided by the investigator via the user interface. For example, the name of the new system folder (i.e. 001test_case) that was created by combining the case number and case name can be seen under the field 'folder_name'. Similarly the file-path to the source file that was uploaded (i.e. TEST_DB.db) can be seen under the field 'primary_db_src'. As mentioned earlier, the 'case list' database will be available for access to the 'case archive manager' if and when it needs the data for presentation.

At this point, another database called 'stat' will be created by the case management engine to store information about either the single IP address or the complete CIDR range of IP addresses that the investigator has chosen for analysis. The 'stat' database will be used again later by the 'pre-processing engine', which will be discussed in Section 5.3.2.



	Users	UserDBpath	Folder_name	Primary_db_src
	Filter	Filter	Filter	Filter
1	192.168.200.25	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/001test_case/192.168.200.25.db	001test_case	C:/xampp/htdocs/forlearning/PhpProject1/UONFAT_Cases/001test_case/TEST_DB.db

Figure 5.9: 'Stat' database (New Case Creator)

The structure of the ‘stat’ database is displayed in Figure 5.9. The database has four data fields – a ‘user’ field containing the IP address of interest, a ‘userdbpath’ field that holds the path to the database file that will contain the traffic for the IP address of interest extracted from the raw traffic source, a ‘folder_name’ field that contains the name of the current case folder and a ‘primary_db_src’ field that has the path to the raw traffic (metadata) file that was uploaded. The example given in Figure 5.9 has only one record because the investigator was interested in one particular IP address (i.e. 192.168.200.25). It has to be pointed out that the ‘userdbpath’ field will be filled only after the ‘pre-processing engine’ creates a new database extracting the traffic for the corresponding IP address from the raw source file. As mentioned above, this step will be explained in Section 5.3.2. The database would be having more records if either the investigator had opted for either a CIDR range such as 192.168.200.0/24 or the investigator had asked the system to analyse the traffic for more than one IP address of interest.

5.3.2 The Pre-processing Engine

The ‘pre-processing engine’ is one of the two components that belong to the ‘back-end processing logic layer’ of the UO-NFAT architecture. The main tasks carried out by this engine are hashing and data structuring.

5.3.2.1 Hashing

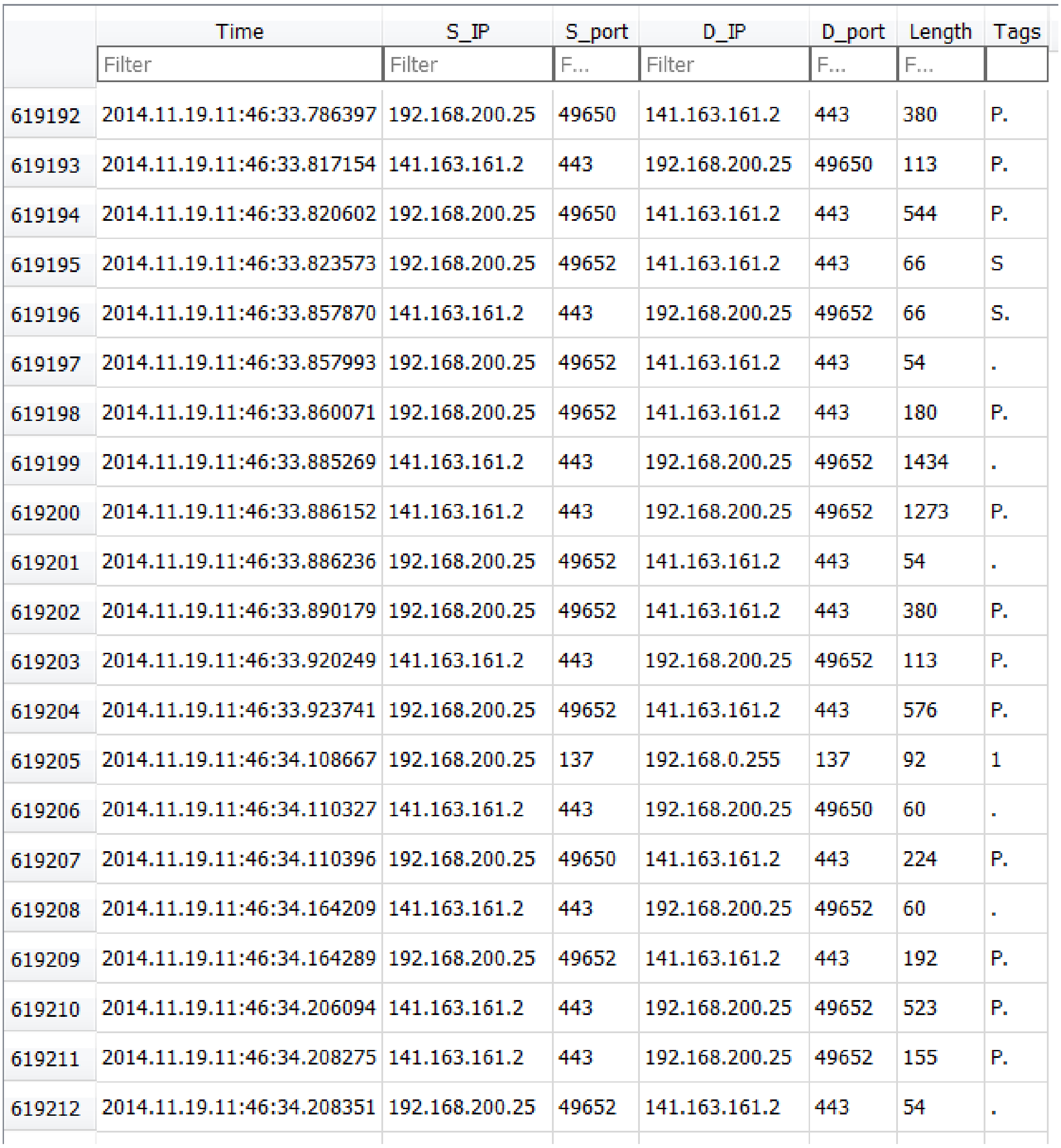
Hashing, in digital forensic investigation, is a method for confirming the integrity of the evidence. During the investigation, a copy of the original source of evidence is taken first and it is the copy that is analysed. By taking this approach, the investigator would be able to preserve the original source of evidence. But it is also essential to ensure that the copy has not been altered in any manner prior to, during or after the analysis. Hashing helps the investigator to achieve this goal. Once the raw traffic file is uploaded, the initial task undertaken by the ‘pre-processing engine’ is ‘hashing’. The engine will perform the task by generating the hash key for the uploaded copy of the source file containing the raw metadata using a hashing algorithm such as the Secure Hash Algorithm (SHA). Once generated, the engine will compare the hash key of the uploaded copy (which is located in the web server) against the hash key of the original source file (that could be stored in another device). If the hash key match is positive then the engine will confirm the authenticity of the uploaded source. The generated hash key can be stored in the ‘case list’ database in accordance with the appropriate source file. A negative match, would reveal that the uploaded data source has been altered.

Even though it is not likely, there is one more way the ‘pre-processing engine’ could use the hash key usefully and possibly save the investigator some time. The engine can successfully compare the generated hash key of the uploaded source file against the hash values of all raw traffic files that were previously uploaded into the ‘case list’ database. If the comparison does not result in a match, then the possibility of duplication of the uploaded source file can be discarded. At this point, the SHA hash value of the current file will be stored in the ‘case list’ database appropriately for future comparison purposes. In comparison, a matching hash key confirms the presence of an identical traffic meta-data file in the system. This could mean that the ‘case list’ database has got a raw traffic file that is exactly the same as the currently uploaded source file and that has already been analysed in the past. The tool will point out this information along with the matching case details and will take the investigator back to the ‘case archive’ allowing the investigator to choose between the given options (i.e. fetch results of, view report of or delete the old case). If the investigator wishes to, further analysis can be done on the previously investigated case. Even though the probability of occurrence of such a scenario is less, the hashing will help to scrutinise for any duplication of the source file which, if found positive, can save considerable amount of time on the investigation process.

5.3.2.2 Data Structuring

The second task executed by the ‘pre-processing engine’ is called ‘data structuring’. The source database file containing the raw network meta-data will be targeted in order to create separate databases for individual IP addresses. To fulfil this task, the engine will utilise the entries from the ‘stat’ database which could be either one or more IP address (i.e. a single IP or a CIDR value), depending on the input given by the investigator earlier. In the given example (as shown in Figure 5.9), there is only one IP address of interest which is 192.168.200.25. One by one, the engine will look for any of these IP addresses that are present in the raw data source file (stored in the ‘raw traffic source’ database) and will extract the traffic corresponding to that specific IP in order to create the new database. Database queries will be used by the pre-processing engine to accomplish the task and the resulting databases will be stored in the same case folder that contains the uploaded raw source file. The resulting databases will be named after the IP address that is being investigated. An example of a smaller dataset extracted from the raw traffic source is presented in Figure 5.10.

In this example, the extracted dataset is the traffic generated by the IP address – 192.168.200.25 – which was the user IP that the investigator was interested in. It is noticeable that the newly extracted dataset and the raw traffic source have the same data fields.



	Time	S_IP	S_port	D_IP	D_port	Length	Tags
	Filter	Filter	F...	Filter	F...	F...	
619192	2014.11.19.11:46:33.786397	192.168.200.25	49650	141.163.161.2	443	380	P.
619193	2014.11.19.11:46:33.817154	141.163.161.2	443	192.168.200.25	49650	113	P.
619194	2014.11.19.11:46:33.820602	192.168.200.25	49650	141.163.161.2	443	544	P.
619195	2014.11.19.11:46:33.823573	192.168.200.25	49652	141.163.161.2	443	66	S
619196	2014.11.19.11:46:33.857870	141.163.161.2	443	192.168.200.25	49652	66	S.
619197	2014.11.19.11:46:33.857993	192.168.200.25	49652	141.163.161.2	443	54	.
619198	2014.11.19.11:46:33.860071	192.168.200.25	49652	141.163.161.2	443	180	P.
619199	2014.11.19.11:46:33.885269	141.163.161.2	443	192.168.200.25	49652	1434	.
619200	2014.11.19.11:46:33.886152	141.163.161.2	443	192.168.200.25	49652	1273	P.
619201	2014.11.19.11:46:33.886236	192.168.200.25	49652	141.163.161.2	443	54	.
619202	2014.11.19.11:46:33.890179	192.168.200.25	49652	141.163.161.2	443	380	P.
619203	2014.11.19.11:46:33.920249	141.163.161.2	443	192.168.200.25	49652	113	P.
619204	2014.11.19.11:46:33.923741	192.168.200.25	49652	141.163.161.2	443	576	P.
619205	2014.11.19.11:46:34.108667	192.168.200.25	137	192.168.0.255	137	92	1
619206	2014.11.19.11:46:34.110327	141.163.161.2	443	192.168.200.25	49650	60	.
619207	2014.11.19.11:46:34.110396	192.168.200.25	49650	141.163.161.2	443	224	P.
619208	2014.11.19.11:46:34.164209	141.163.161.2	443	192.168.200.25	49652	60	.
619209	2014.11.19.11:46:34.164289	192.168.200.25	49652	141.163.161.2	443	192	P.
619210	2014.11.19.11:46:34.206094	141.163.161.2	443	192.168.200.25	49652	523	P.
619211	2014.11.19.11:46:34.208275	141.163.161.2	443	192.168.200.25	49652	155	P.
619212	2014.11.19.11:46:34.208351	192.168.200.25	49652	141.163.161.2	443	54	.

Figure 5.10: 'IPn traffic' database (Pre-processing Engine)

The main purpose of the 'data structuring' task is to reduce the volume of raw data that needs to be analysed thereby reducing the time taken for the investigation. For instance, consider a scenario in which the size of the raw file is 60GB and the investigator is interested in only one particular IP address that is responsible for 5GB of the entire traffic. Analysing the entire proportion of the raw file is going waste both time and computational power. By utilising ‘data structuring’ the system can extract and analyse just the 5GB dataset which could save considerable amount of time. The only difference between the databases for the individual IPs and the raw traffic file is the size; all will have the same data fields.

5.3.3 The User-Activity Analyser

As discussed in Chapter 4, the manual analysis of network meta-data resulted in signatures for the selected services and the corresponding user activities. In the UO-NFAT architecture, the ‘user activity analyser’ can be designated as the core element of the ‘back-end processing layer’. The ‘user activity analyser’ undertakes the task of analysing the meta-data looking to identify specific user activities for online services using the unique interaction signatures. The ‘user activity analyser’ will access information – the IP address selected for the analysis and the path to the file containing traffic that belongs to the selected IP – from the ‘stat’ database. Using this information, the ‘user activity analyser’ will locate the appropriate database containing the traffic, which will be followed by the analysis. The interaction based analysis approach that was hypothesised and tested positive in Chapter 4 is the backbone of the ‘user activity analyser’. The interaction based approach utilises a number network metadata parameters such as the date/time stamp, the IP address of the source and destination, the port numbers used by the source and destination, the type of protocol (TCP/UDP), the length of the packet and certain TCP flags (SYN, ACK, PSH and FIN) to identify the user interaction signatures. Using these parameters, unique signatures for different online interactions are identified.

Figure 5.11 illustrates the ‘user activity analyser’ which consists of three segments – ‘the service IP traffic extractor’, ‘the protocol traffic extractor’ and ‘the user activity extractor’. These three segments enable the ‘user activity analyser’ in parsing the raw network meta-data to generate meaningful user activities.

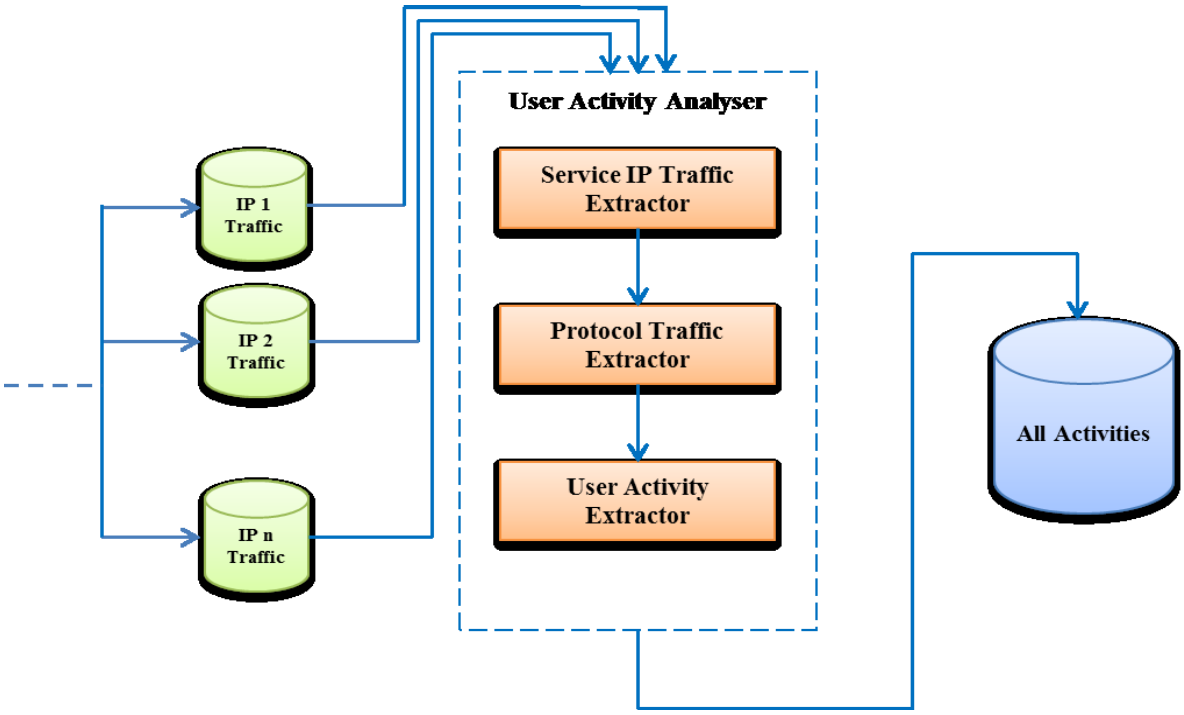
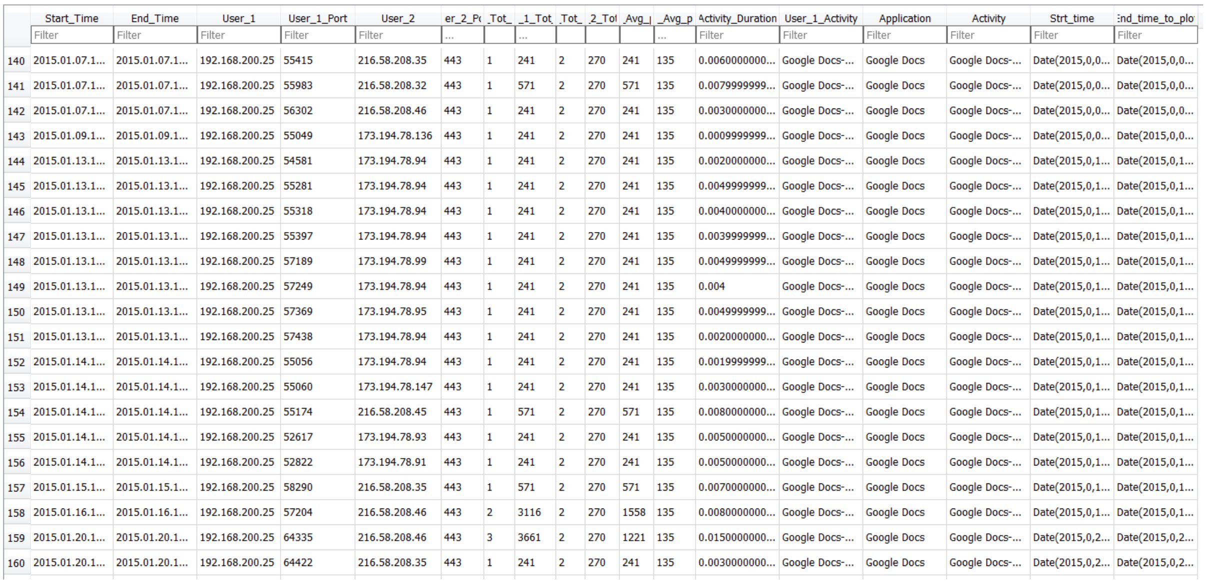


Figure 5.11: The User Activity Analyser

The 'service IP extractor' will extract packets from the raw metadata file of the selected IP address using the server IP addresses of the Internet based services. This will further reduce the volume of packets that need to be analysed. Following this, the 'protocol traffic extractor' will begin to separate the packets based on whether the transport layer protocols within the IP packets are TCP or UDP. As all of the chosen online user interactions occur either over a secure communication channel (HTTPS) or in an unencrypted manner (HTTP) and as both use TCP as their transport layer protocol, the 'protocol traffic extractor' will extract those TCP packets based on their server side port numbers (443 for HTTPS and 80 for HTTP). The 'protocol traffic extractor' will start combining the packets that match these standards. The resulting TCP/UDP related packets will be further extracted based on the signature parameters observed while experimenting with the interaction based approach.

The resulting data structure will be dealt with by the 'user activity extractor' that will apply the unique signatures. It has been observed that interactions that take place over a secure communication channel will have their signatures in the form of a single packet, multiple packets (between 2 and 4 packets) sent within a millisecond timeframe or a stream of packets (with maximum transmission unit size) sent within a 50 microseconds timeframe. And the unencrypted communications take place in a stream of packets fashion between the SYN and FIN flags of the TCP protocol. Any match found by the 'user activity extractor' will be identified as the specific online user interaction and will be stored in a database called ‘all activities’, created by the ‘user activity analyser’. An example of the structure of the ‘all activities’ database is illustrated in Figure 5.12.



	Start_Time	End_Time	User_1	User_1_Port	User_2	er_2_P	_Tot_	_1_Tot	Tot_	_2_Tot	_Avg_	_Avg_p	Activity_Duration	User_1_Activity	Application	Activity	Strt_time	End_time_to_plo
	Filter	Filter	Filter	Filter	Filter	...		...				...	Filter	Filter	Filter	Filter	Filter	Filter
140	2015.01.07.1...	2015.01.07.1...	192.168.200.25	55415	216.58.208.35	443	1	241	2	270	241	135	0.0060000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,0...	Date(2015,0,0...
141	2015.01.07.1...	2015.01.07.1...	192.168.200.25	55983	216.58.208.32	443	1	571	2	270	571	135	0.0079999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,0...	Date(2015,0,0...
142	2015.01.07.1...	2015.01.07.1...	192.168.200.25	56302	216.58.208.46	443	1	241	2	270	241	135	0.0030000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,0...	Date(2015,0,0...
143	2015.01.09.1...	2015.01.09.1...	192.168.200.25	55049	173.194.78.136	443	1	241	2	270	241	135	0.0009999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,0...	Date(2015,0,0...
144	2015.01.13.1...	2015.01.13.1...	192.168.200.25	54581	173.194.78.94	443	1	241	2	270	241	135	0.0020000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
145	2015.01.13.1...	2015.01.13.1...	192.168.200.25	55281	173.194.78.94	443	1	241	2	270	241	135	0.0049999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
146	2015.01.13.1...	2015.01.13.1...	192.168.200.25	55318	173.194.78.94	443	1	241	2	270	241	135	0.0040000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
147	2015.01.13.1...	2015.01.13.1...	192.168.200.25	55397	173.194.78.94	443	1	241	2	270	241	135	0.0039999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
148	2015.01.13.1...	2015.01.13.1...	192.168.200.25	57189	173.194.78.99	443	1	241	2	270	241	135	0.0049999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
149	2015.01.13.1...	2015.01.13.1...	192.168.200.25	57249	173.194.78.94	443	1	241	2	270	241	135	0.004	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
150	2015.01.13.1...	2015.01.13.1...	192.168.200.25	57369	173.194.78.95	443	1	241	2	270	241	135	0.0049999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
151	2015.01.13.1...	2015.01.13.1...	192.168.200.25	57438	173.194.78.94	443	1	241	2	270	241	135	0.0020000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
152	2015.01.14.1...	2015.01.14.1...	192.168.200.25	55056	173.194.78.94	443	1	241	2	270	241	135	0.0019999999...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
153	2015.01.14.1...	2015.01.14.1...	192.168.200.25	55060	173.194.78.147	443	1	241	2	270	241	135	0.0030000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
154	2015.01.14.1...	2015.01.14.1...	192.168.200.25	55174	216.58.208.45	443	1	571	2	270	571	135	0.0080000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
155	2015.01.14.1...	2015.01.14.1...	192.168.200.25	52617	173.194.78.93	443	1	241	2	270	241	135	0.0050000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
156	2015.01.14.1...	2015.01.14.1...	192.168.200.25	52822	173.194.78.91	443	1	241	2	270	241	135	0.0050000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
157	2015.01.15.1...	2015.01.15.1...	192.168.200.25	58290	216.58.208.35	443	1	571	2	270	571	135	0.0070000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
158	2015.01.16.1...	2015.01.16.1...	192.168.200.25	57204	216.58.208.46	443	2	3116	2	270	1558	135	0.0080000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,1...	Date(2015,0,1...
159	2015.01.20.1...	2015.01.20.1...	192.168.200.25	64335	216.58.208.46	443	3	3661	2	270	1221	135	0.0150000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,2...	Date(2015,0,2...
160	2015.01.20.1...	2015.01.20.1...	192.168.200.25	64422	216.58.208.35	443	1	241	2	270	241	135	0.0030000000...	Google Docs-...	Google Docs	Google Docs-...	Date(2015,0,2...	Date(2015,0,2...

Figure 5.12: 'All activities' database (User Activity Analyser)

Different fields of the database represent: the start time of the interaction, the end time of the interaction, user 1’s IP address, user’s port number, web service’s IP address, web service’s port number, total number of packets sent by the user, total size of packets sent by the user, total number of packets sent by the web service, total size of packets sent by the web service, average size of packets sent by the user, average size of packets sent by the web service, activity duration and the user activity. There are four more fields that exist for the purpose of plotting the visualisation (timeline) charts. These four fields contain information such as the Internet service that was used, the identified user interaction (activity), the start and end time needed for plotting that timeline.

5.3.4 The Visualisation-Configuration Engine

The UO-NFAT architecture is conceptualised and designed in such a way that the tool can present the information resulting from the analysis in a visual manner and allow the investigator to interact with the graphically displayed results, which would help the investigator in the process of investigation. In order to facilitate this notion, the UO-NFAT has incorporated a dashboard that would graphically portray the results and allow interactions with those results.

It is important for a tool that assists in the investigation to allow the investigator to interact with the system in order to choose from a set of analysis results based on certain filtering options. The need for such a type of interaction could be due to the investigation needs or the investigator's intuitions or a mere interest in the results.

In order to accomplish this goal, the 'visualisation-configuration engine' was designed that accepts different types of inputs (i.e. filter options) from the investigator in order to modify the visualisation charts that are plotted in the visualisation window of the dashboard. The UO-NFAT was designed with four types of filter options in mind – user activities, time-line from, time-line till and select the IP address. 'User activities' provides the investigator a list of all online activities performed by user(s) while they were interacting with the corresponding Internet based services. The 'time-line from' and 'time-line till' filter options allows the investigator to select the start and end points for the visualisation window. By utilising the 'Select the IP address', the investigator can select a specific user IP address or all user IP addresses that are responsible for the detected user activities.

The filtering options provided by the 'visualisation configuration window' allow the investigator to interact with the 'visualisation window' dynamically. It provides the investigator with the option to modify and filter through the visualisation charts till desirable results are obtained. Another characteristic feature of the filter options provided by the tool is that they can be targeted at different charts. For instance, while all filter-options are applicable to the timeline chart, only the start and end time filters have their effect on the column charts. Table 5.1 lists different types of filter options that are incorporated into the UO-NFAT prototype, along with the supporting information.

Table 5.1: Filter Options (Visualisation Configuration Engine)

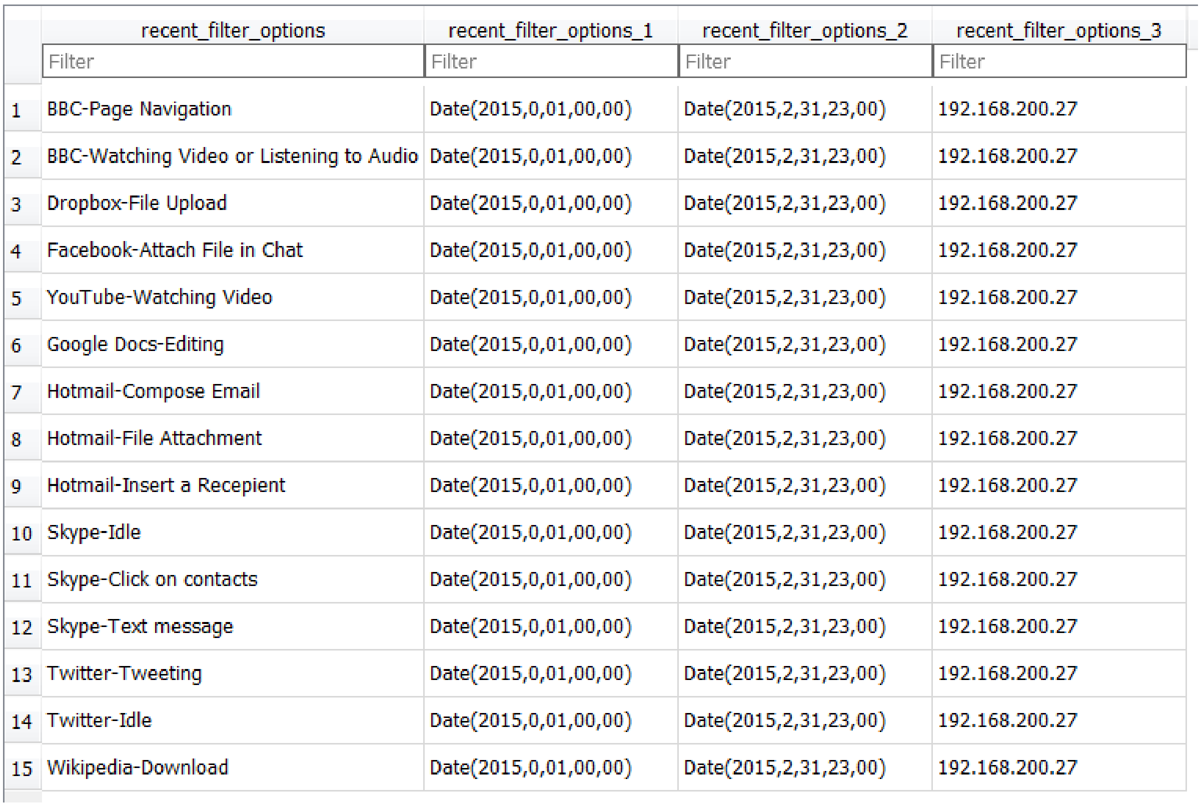
Filter Options	Details	
User Activities	BBC	Page Navigation
		Watching Video or Listening to Audio
	Dropbox	File Download
		File Upload
		Folder Navigation
	Facebook	Typing
		Chat
		Attach a File in Chat
		Page Loading
	YouTube	Watching Video
		Video Uploading
	Google Search	Page Navigation
	Google Docs	Editing
	Hotmail	Compose E-mail

		File Attachment
		Insert a Recipient
	Skype	Idle
		Click on Contacts
		Text Message
		Audio Call
		Video Call
		File Transfer (sending)
		File Transfer (receiving)
	Twitter	Tweeting
		Upload
		Click on Contacts
		Idle
	Wikipedia	Page loading
Time-line From	Starting date and time for the timeline	
Time-line Till	Ending date and time for the timeline	
Select the IP Address	User IP address of interest	

It has to be highlighted that the nature of the investigator's interaction with the 'visualisation-configuration engine' is different from that of with the 'case management engine', whether it is the type of information or the purpose of interaction. While the 'case management engine' allows the investigator to interact with the tool to deal with both new and old cases, the 'visualisation-configuration engine' offers the interaction so that the investigator can interact with the results of the analysis.

Being part of the 'UO-NFAT interface logic layer', the 'visualisation-configuration engine' interacts with the investigator via its user interface. The UO-NFAT will be facilitating this by means of a visualisation configuration window (containing the filter options) as part of the tool's dashboard.

The 'visualisation-configuration engine' will store the input information of this nature (i.e. the filter options) received through the tool's dashboard in a database called 'visualisation configuration'. The structure of this database is shown in Figure 5.13.



	recent_filter_options	recent_filter_options_1	recent_filter_options_2	recent_filter_options_3
	Filter	Filter	Filter	Filter
1	BBC-Page Navigation	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
2	BBC-Watching Video or Listening to Audio	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
3	Dropbox-File Upload	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
4	Facebook-Attach File in Chat	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
5	YouTube-Watching Video	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
6	Google Docs-Editing	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
7	Hotmail-Compose Email	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
8	Hotmail-File Attachment	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
9	Hotmail-Insert a Recepient	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
10	Skype-Idle	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
11	Skype-Click on contacts	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
12	Skype-Text message	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
13	Twitter-Tweeting	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
14	Twitter-Idle	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27
15	Wikipedia-Download	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27

Figure 5.13: 'Visualisation configuration' database (Visualisation Configuration Engine)

It can be noticed that the database has four fields that store the user activities, the start date/time for the timeline, the end date/time for the timeline and the user IP address.

Once the database is populated by the ‘visualisation-configuration engine’ with the filter options provided by the investigator, the UO-NFAT will make use of this data in preparation for plotting the visualisation charts. This task will be carried out by the ‘visualisation engine’, which will be discussed below in sub-section 5.3.5.

The ‘visualisation configuration’ database can be considered as a temporary database as the ‘visualisation-configuration engine’ will replace the contents of this database every time the investigator supplies a new set of filter options. The contents will be copied to and stored in a permanent database only if the investigator finds the resulting timeline chart interesting and chooses to bookmark that particular chart. This operation will be undertaken by the ‘reporting engine’, which will be discussed in sub-section 5.3.6.

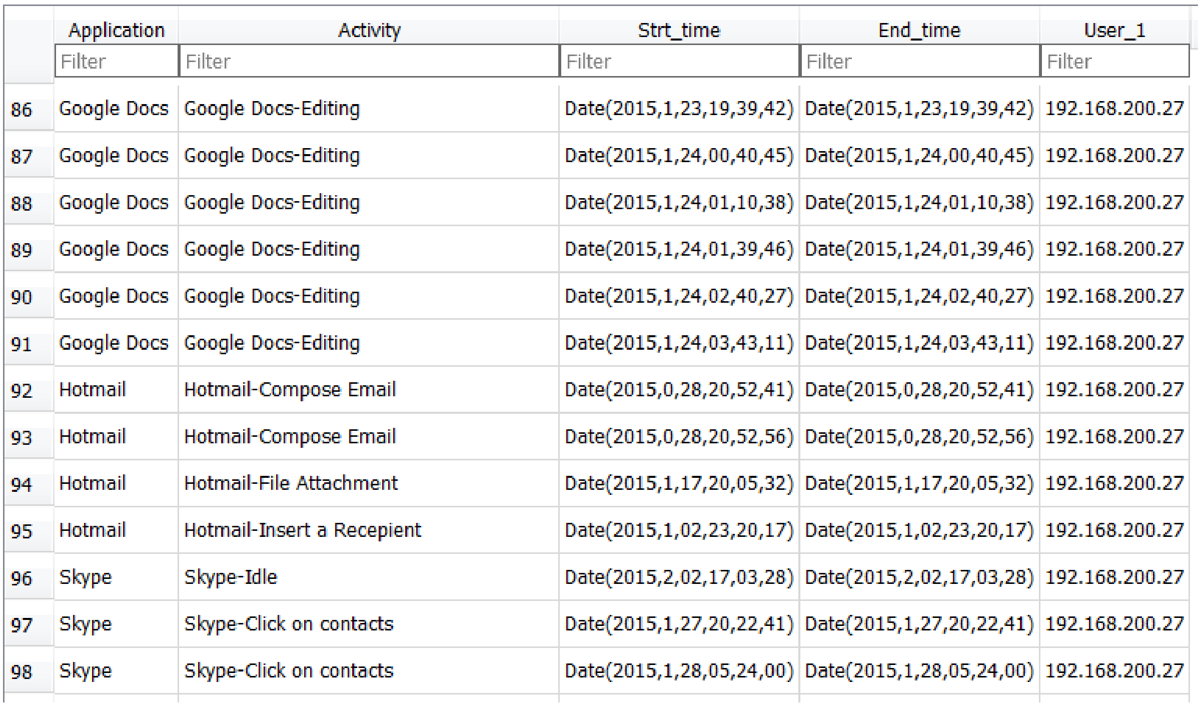
The ‘visualisation-configuration engine’ exists and operates to receive input information from the investigator and storing the data temporarily. This data plays an important role in generating the visualisation charts that the UO-NFAT uses to exhibit the identified online user activities. Moreover, the engine thus contributes in providing the UO-NFAT dashboard an interactive nature.

5.3.5 The Visualisation Engine

As discussed in the beginning of sub-section 5.3.4, the UO-NFAT that is based on the proposed architecture is expected to be capable of presenting the identified online user interactions with the support of visualisation charts and allowing the investigator to interact with the portrayed charts through filter options. In order to facilitate this functionality, a visualisation engine is required. It has to be mentioned that the 'visualisation-configuration engine' also plays a role in delivering the visualisation functionality. The receiving of the selected filter options via the user interface and the database associated with it are controlled by the 'visualisation-configuration engine'. The filter options stored in the database sets the stage for the 'visualisation engine's' actions.

Prior to performing any task, the 'visualisation engine' will access the 'visualisation_editor_db' database (as mentioned in subsection 5.3.1) to determine the number of tabs that the visualisation window should be having and the type of visualisation charts that each of the tab should be filled with. The values retrieved from the aforementioned database will be based on the selection of charts made by the investigator using the 'visualisation editor' in the beginning of the investigation. Once the data is acquired, the 'visualisation engine' will start with the task of creating the visualisation charts. For the purpose and convenience of explaining the operations carried out by the engine, an assumption that the investigator had opted for a column chart for displaying user IPs based on the number of packets for raw traffic, a column chart for displaying user IPs based on the number of packets for services and a column chart for displaying user IPs based on the number of packets for activities is made. As mentioned in subsection 5.3.1, the timeline chart for displaying the user activities is included in the visualisation window, by default. The remaining part of the subsection will explain the operations carried out by the 'visualisation engine' using aforementioned charts.

The 'visualisation engine' begins the task by accessing the 'visualisation configuration' database. The engine will read the database records (containing the filter options) one by one and each record will be compared against the records within the 'all activities' database, shown in Figure 5.12 that contains all identified online user activities. This logical comparison will extract the relevant data fields of the online user interactions that took place between the start date/time and the end date/time for the user IP address that the investigator provided via the filter options in the ‘visualisation configuration window’. The relevant data fields extracted in this manner will be stored in a database named ‘timeline activities to display’. The structure of this database is shown in Figure 5.14.



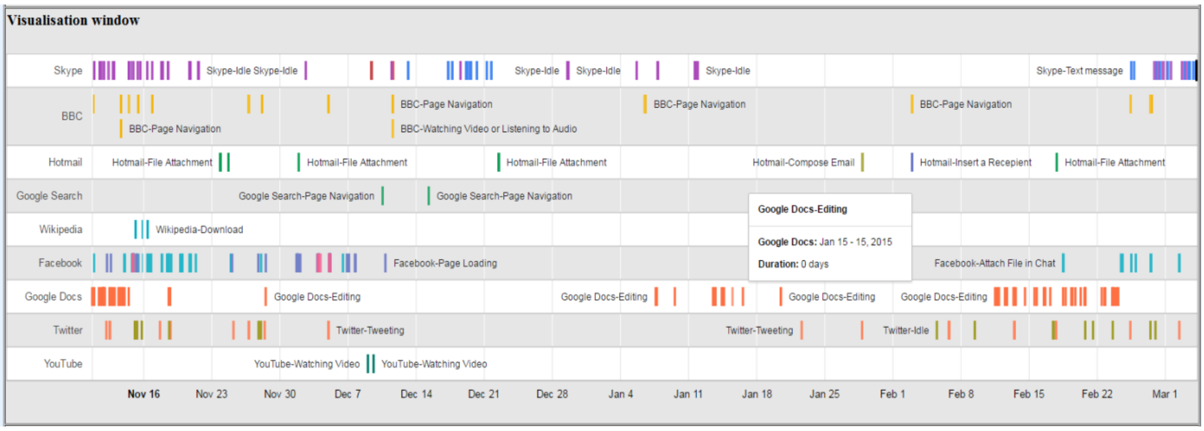
	Application	Activity	Strt_time	End_time	User_1
	Filter	Filter	Filter	Filter	Filter
86	Google Docs	Google Docs-Editing	Date(2015,1,23,19,39,42)	Date(2015,1,23,19,39,42)	192.168.200.27
87	Google Docs	Google Docs-Editing	Date(2015,1,24,00,40,45)	Date(2015,1,24,00,40,45)	192.168.200.27
88	Google Docs	Google Docs-Editing	Date(2015,1,24,01,10,38)	Date(2015,1,24,01,10,38)	192.168.200.27
89	Google Docs	Google Docs-Editing	Date(2015,1,24,01,39,46)	Date(2015,1,24,01,39,46)	192.168.200.27
90	Google Docs	Google Docs-Editing	Date(2015,1,24,02,40,27)	Date(2015,1,24,02,40,27)	192.168.200.27
91	Google Docs	Google Docs-Editing	Date(2015,1,24,03,43,11)	Date(2015,1,24,03,43,11)	192.168.200.27
92	Hotmail	Hotmail-Compose Email	Date(2015,0,28,20,52,41)	Date(2015,0,28,20,52,41)	192.168.200.27
93	Hotmail	Hotmail-Compose Email	Date(2015,0,28,20,52,56)	Date(2015,0,28,20,52,56)	192.168.200.27
94	Hotmail	Hotmail-File Attachment	Date(2015,1,17,20,05,32)	Date(2015,1,17,20,05,32)	192.168.200.27
95	Hotmail	Hotmail-Insert a Recepient	Date(2015,1,02,23,20,17)	Date(2015,1,02,23,20,17)	192.168.200.27
96	Skype	Skype-Idle	Date(2015,2,02,17,03,28)	Date(2015,2,02,17,03,28)	192.168.200.27
97	Skype	Skype-Click on contacts	Date(2015,1,27,20,22,41)	Date(2015,1,27,20,22,41)	192.168.200.27
98	Skype	Skype-Click on contacts	Date(2015,1,28,05,24,00)	Date(2015,1,28,05,24,00)	192.168.200.27

Figure 5.14: 'Timeline activities to display' database (Visualisation Engine)

The example shows that the resulting database has five different fields – an ‘application’ field containing the name of the online service, an ‘activity’ field containing the identified user activity, a ‘strt_time’ field holding the start date/time of the activity, an ‘end_time’ field holding the end date/time of the activity and a ‘user_1’ field holding the IP address of the user that was supplied through the filter options.

It has to be pointed out that similar to the ‘visualisation configuration’ database the ‘timeline activities to display’ database is also considered as a temporary database because its contents will be replaced when supplied with a new set of filter options. However, if the investigator opts for the bookmark option (discussed in subsection 5.3.6) then the database records will be stored in another permanent database.

Following this the ‘visualisation configuration’ engine will use the data within the database in order to plot a timeline chart of all online user interactions that are stored in the ‘timeline activities to display’ database. The engine makes use of Google Charts, a service that facilitates the formation of interactive visualisation charts, to plot the required timeline chart. The UO-NFAT user interface will have a dashboard including a visualisation window into which the ‘visualisation engine’ can plot the timeline chart. An example of the visualisation window is shown in Figure 5.15.



Google Docs-Editing

Google Docs: Jan 15 - 15, 2015

Duration: 0 days

Figure 5.15: Visualisation window (Timeline)

Figure shows an example of the timeline chart the ‘visualisation engine’ plotted using Google Charts. The timeline shows all user interactions that took place between 12th November 2014 and 15th March 2015. All nine of the selected services were accessed by the users during this period and the user interactions for each service are colour coded for convenience. The chart’s interactive nature allows the investigator to hover over any interaction that gives a pop up information window as seen in the example. The timeline provides the investigator information about the user activities in an easy to understand manner. Being able to know the events (i.e. user activities) in chronological order is crucial for the investigation. Time of occurrence can be an element that can correlate one or more activities to the crime. It is an element that will allow the investigator to paint a picture in its entirety, and to answer questions or even ask new questions that might help in the investigation process.

The ‘visualisation engine’ utilises the data stored in the ‘timeline activities to display’ database (shown in Figure 5.14) in order to plot column charts that will display the total number of packets that are responsible for the raw traffic, the selected services and the identified user activities. The engine will access the four distinct data fields (i.e. the ‘activity’ field, the ‘Strt_time’ field, the ‘End_time’ field and the ‘User_1’ field) using which the engine will operate on the user ‘IP traffic’ database (shown in Figure 5.10) that was created by the ‘pre-processing engine’ and on the ‘all activities’ database (show in Figure 5.12) that was created by the ‘user-activity analyser’ in order to calculate the aforementioned type of total number of packets. The obtained results will be stored in three different databases – ‘user/num of packets (traffic)’, ‘user/num of packets (services)’ and ‘user/num of packets (activities)’, which will be accessed by the ‘visualisation engine’ to plot the column charts with the help of Google Charts. Figures 5.16, 5.17 and 5.18 show examples of the column charts produced by the ‘visualisation engine’. It is important to note that the three column charts and the corresponding databases were created only based on the selection made by the investigator earlier via the ‘visualisation editor’ (as explained in subsection 5.3.1). Both the charts and the databases would have been different depending on the type(s) of chart(s) selected by the investigator, which is/are stored in the ‘visualisation_editor_db’ database.

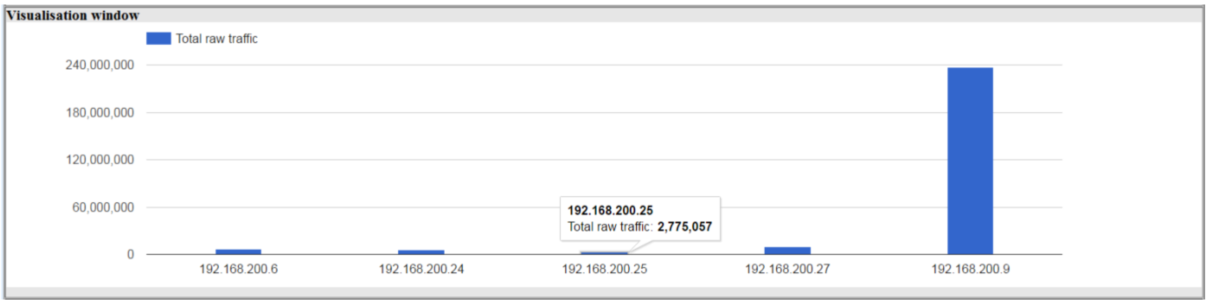


Figure 5.16: Visualisation window (Raw traffic comparison)

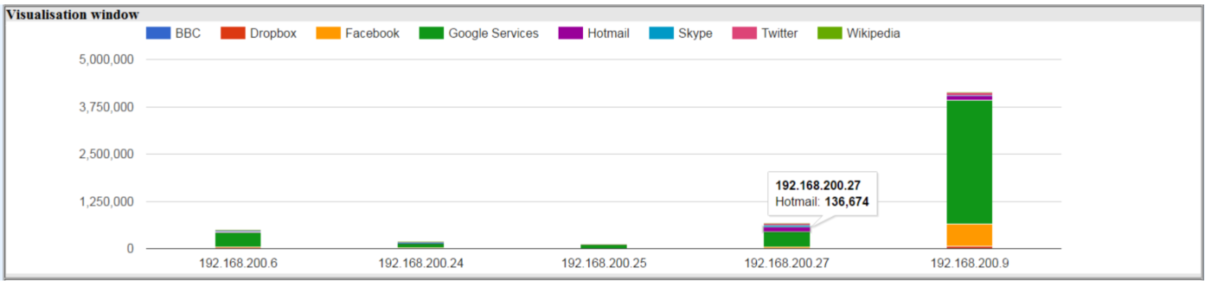


Figure 5.17: Visualisation window (Services comparison)

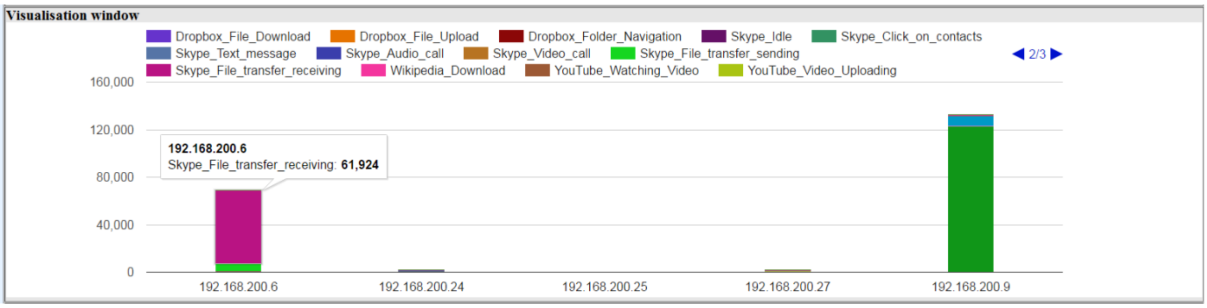
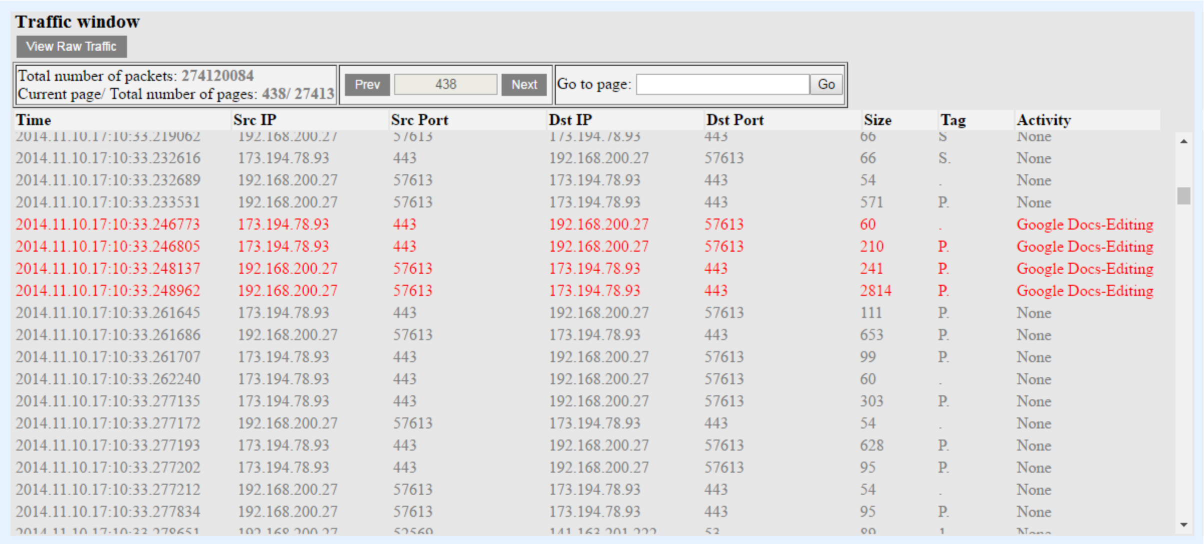


Figure 5.18: Visualisation window (Activities comparison)

Similar to the timeline chart, the engine will plot the column charts into the visualisation window of the UO-NFAT dashboard user interface. The column charts produced by the ‘visualisation engine’ will help the investigator to understand the activeness of each user in terms of their traffic creation, service usage and activity usage. The investigator can also compare between the users to find the most and least active one or to the mostly used service or activity for each user. These pieces of information could be useful for the investigation.

Another feature that the proposed UO-NFAT is expected to possess is the capability to provide an option that would allow the investigator to view the raw network traffic metadata, which will allow the system to maintain the integrity. In the UO-NFAT architecture, this task will be undertaken by the 'visualisation engine'. If and only if the investigator wishes to view the raw traffic, the engine will access the 'raw traffic source' database containing the source network traffic metadata and would start displaying the packets via user interface window of the UO-NFAT dashboard. In order to eliminate unnecessary delay, the UO-NFAT is designed in such a way that the dashboard does not display the raw network metadata by default. Loading the raw network traffic can consume more time as the visualisation engine would take longer to read through and process the source traffic file for the purpose of displaying the raw network traffic. An example of the corresponding user interface traffic window is displayed in Figure 5.19.



Traffic window

Total number of packets: 274120084
Current page/ Total number of pages: 438/ 27413

Time	Src IP	Src Port	Dst IP	Dst Port	Size	Tag	Activity
2014.11.10.17:10:33.219062	192.168.200.27	57613	173.194.78.93	443	66	S	None
2014.11.10.17:10:33.232616	173.194.78.93	443	192.168.200.27	57613	66	S.	None
2014.11.10.17:10:33.232689	192.168.200.27	57613	173.194.78.93	443	54	.	None
2014.11.10.17:10:33.233531	192.168.200.27	57613	173.194.78.93	443	571	P.	None
2014.11.10.17:10:33.246773	173.194.78.93	443	192.168.200.27	57613	60	.	Google Docs-Editing
2014.11.10.17:10:33.246805	173.194.78.93	443	192.168.200.27	57613	210	P.	Google Docs-Editing
2014.11.10.17:10:33.248137	192.168.200.27	57613	173.194.78.93	443	241	P.	Google Docs-Editing
2014.11.10.17:10:33.248962	192.168.200.27	57613	173.194.78.93	443	2814	P.	Google Docs-Editing
2014.11.10.17:10:33.261645	173.194.78.93	443	192.168.200.27	57613	111	P.	None
2014.11.10.17:10:33.261686	192.168.200.27	57613	173.194.78.93	443	653	P.	None
2014.11.10.17:10:33.261707	173.194.78.93	443	192.168.200.27	57613	99	P.	None
2014.11.10.17:10:33.262240	173.194.78.93	443	192.168.200.27	57613	60	.	None
2014.11.10.17:10:33.277135	173.194.78.93	443	192.168.200.27	57613	303	P.	None
2014.11.10.17:10:33.277172	192.168.200.27	57613	173.194.78.93	443	54	.	None
2014.11.10.17:10:33.277193	173.194.78.93	443	192.168.200.27	57613	628	P.	None
2014.11.10.17:10:33.277202	173.194.78.93	443	192.168.200.27	57613	95	P.	None
2014.11.10.17:10:33.277212	192.168.200.27	57613	173.194.78.93	443	54	.	None
2014.11.10.17:10:33.277834	192.168.200.27	57613	173.194.78.93	443	95	P.	None
2014.11.10.17:10:33.278651	192.168.200.27	52569	141.163.201.222	53	80	1	None

Figure 5.19: Traffic window (Visualisation Engine)

The raw metadata will be presented in a fashion that imitates the existing network analysis tools such as Wireshark, tcpdump, NetworkMiner etc. Also, the 'visualisation engine' will highlight the packets that are responsible for the user interactions that were already displayed by the engine through the visualisation window (shown in Figure 5.15). Apart from the same seven fields from the original traffic data (i.e. the date/time stamp, the source IP, the source port, the destination IP, the destination port, the packet size and the tag field that contains the set TCP flags) the displayed traffic will have an additional field named ‘activity’. The ‘activity’ field will display the name of the user interaction that the highlighted packets represent. The user interface window will also provide options for the investigator to navigate through the raw traffic. This feature, enabled by the ‘visualisation engine’, gives the investigator an option of cruise through the raw traffic while giving the UO-NFAT a certain level of credibility as it makes the tool capable of referring back to the specific packets (low-level metadata) that resulted the particular user interaction (high-level information).

The proposed UO-NFAT would be interacting with the investigator by providing options (via menu features on the tool’s dashboard) to either analyse a new IP address or stop the analysis. These options will be made available through the user interface of the UO-NFAT dashboard. Whenever the investigator interacts through either of the aforementioned options, the ‘visualisation engine’ will take control of the situation. If the ‘analyse new IP’ option is selected, the engine will redirect the investigator to the user interface that is dedicated for the task, which is controlled by the ‘case management engine’. This option can be utilised when the investigator is interested in analysing the source traffic file for identifying the activities of a different user IP address. Once submitted, the tool will run the whole analysis procedure, which will be a combined effort of all engines involved. The investigator can continue to work on the dashboard while the tool performs the analysis of the newly supplied user IP and the investigator will be notified once the analysis is complete. At this point the tool will reopen the dashboard with updated analysis results. The ‘analyse new IP’ option on the dashboard will enable the tool to analyse the raw traffic for the activities of new user IP addresses while letting the investigator to continue examining the results that are already provided. This feature can be an ideal option for a scenario in which the investigator had chosen to analyse the raw traffic for only one user IP address but wishes to view the activities for a different user IP address at a later point in the investigation. Similarly, if the investigator opted for the ‘stop analysis’ option on the tool’s dashboard, then the ‘visualisation engine’ will terminate the dashboard operations, will redirect the investigator from the dashboard to the starting point of the tool (i.e. the user interface that displays the tool’s case archive) and will hand over the control to the case management engine.

Being part of the UO-NFAT architecture, the ‘visualisation engine’ achieves tasks such as producing the timeline chart that portrays the chronology of the online user activities, producing the column charts that plots the total number of packets responsible for the raw traffic, all selected services and all identified user activities generated by each IP address that has been analysed and providing option to view the raw network traffic metadata with the packets that are responsible for the identified user activities highlighted. The engine makes use of the associated databases when needed. The outputs of the aforementioned tasks accomplished by the 'visualisation engine' are delivered to the investigator via the user interface windows (i.e. the visualisation window and the traffic window) of the UO-NFAT dashboard. The visualisation engine will have the capacity to provide a variety of visualisation charts including the timeline and column charts mentioned in the example.

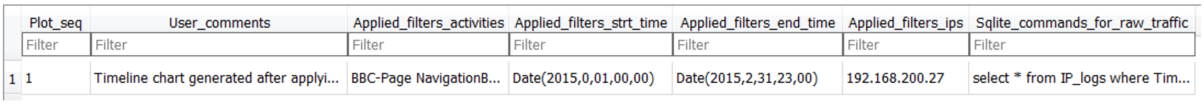
Through the combined effort put forth by the 'visualisation-configuration engine', the 'visualisation engine' and the corresponding databases that are involved, UO-NFAT accomplishes the visualisation functionality.

5.3.6 The Reporting Engine

In a digital forensic investigation, a case repot is the final product. It is this case report that the investigator will be handing over to a higher authority and that may contain information that has the potential of becoming the evidence in an investigation. Prior to designing the architecture a set of requirements were generated for the proposed system. As per the system requirements, one of the proficiencies that the proposed UO-NFAT is expected to have is the ability to produce a case report at the end of the investigation. With this intention, the sixth and final component that constitutes the UO-NFAT was added to the architecture, which is the 'reporting engine'. Two primary responsibilities that the 'reporting engine' is designed to accomplish are 'bookmarking and 'case report generation'.

The bookmarking feature, presented on the user interface of the UO-NFAT dashboard, will allow the investigator to save any of the timeline charts that were found interesting, for later. When the bookmarking option is initiated, the 'reporting engine' will enable the UO-NFAT's dashboard to present the investigator a user interface window to add any optional comments. This is also the starting point of the 'case report generation' as the procedure used for gathering information is same for both 'bookmarking' and 'case report generation'. The investigator will have the freedom to add any useful information that could be worth a reference at a later point or add no comment at all.

Once the optional comments are submitted, the engine will create an associated database named the ‘information for bookmark/case report’ that will store the relevant data. An example of the structure of this database is shown in Figure 5.20.

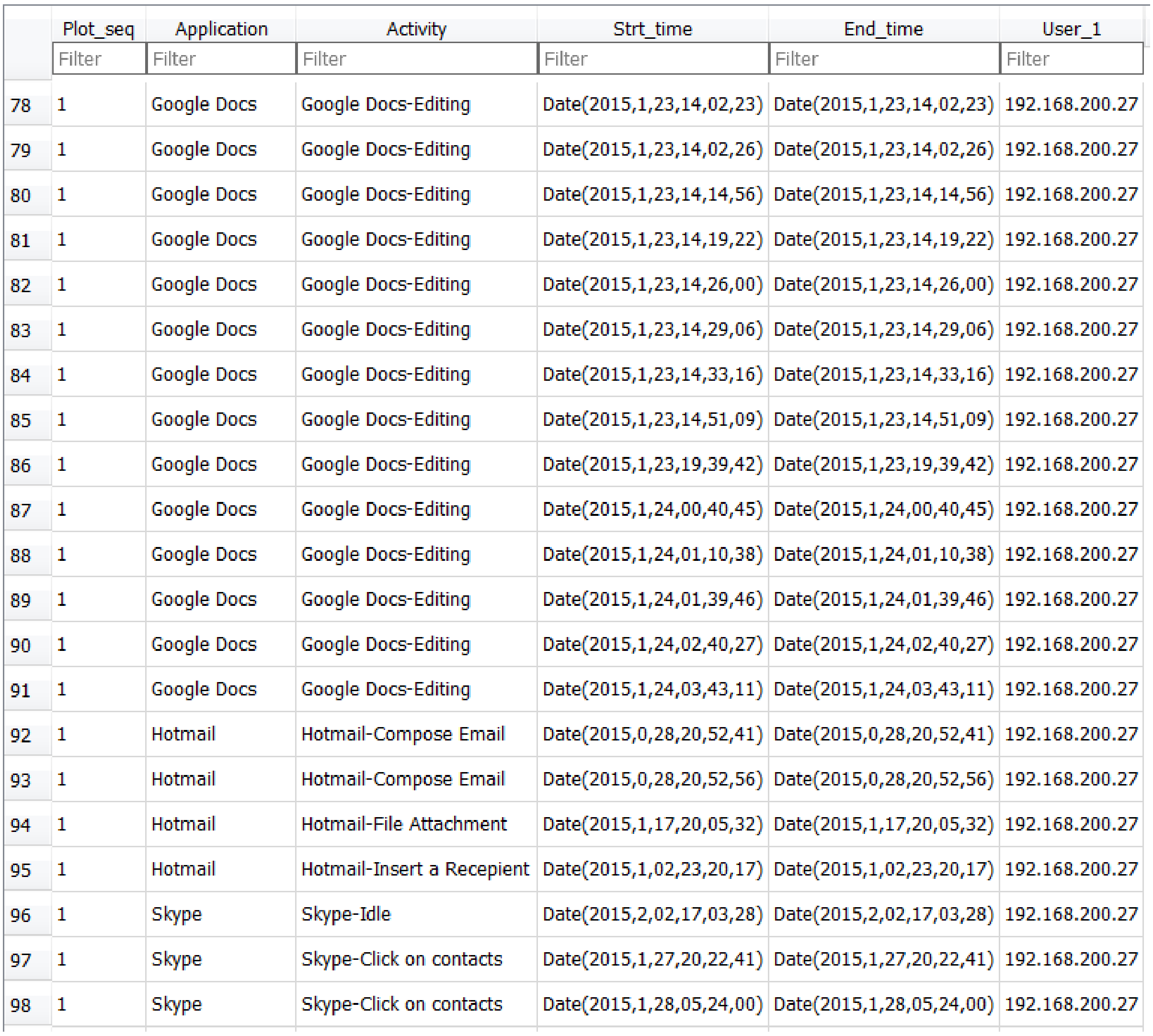


	Plot_seq	User_comments	Applied_filters_activities	Applied_filters_strt_time	Applied_filters_end_time	Applied_filters_ips	Sqlite_commands_for_raw_traffic
	Filter	Filter	Filter	Filter	Filter	Filter	Filter
1	1	Timeline chart generated after applyi...	BBC-Page NavigationB...	Date(2015,0,01,00,00)	Date(2015,2,31,23,00)	192.168.200.27	select * from IP_logs where Tim...

Figure 5.20: 'Information for bookmark/case report' database (Reporting Engine)

The database has seven fields – a ‘plot_seq’ field for storing a sequence number that will be utilised later for retrieving the records for the purpose of reproducing the bookmarked timeline charts and the case report, a ‘user_comments’ field that will contain the optional comments supplied by the investigator, followed by the next four fields (i.e. ‘applied_filters_activities’, ‘applied_filters_strt_time’, ‘applied_filters_end_time’ and ‘applied_filters_ips’) for storing the filter options that were supplied by the investigator in order to view the timeline chart that is being bookmarked and a ‘sqlite_commands_for_raw_traffic’ field that will contain the sqlite queries that were used for locating and highlighting the raw packets that are responsible for the user interactions shown in the timeline chart. Input data needed for field numbers three, four, five and six are copied from the ‘visualisation configuration’ database (shown in Figure 5.13 in subsection 5.3.4) by the engine.

Following this, the ‘reporting engine’ will create another database called the ‘timeline activities to display for bookmark/case report’, which will be having six fields – a ‘plot_seq’ field similar to that in the aforementioned database followed by the next five fields (i.e. ‘application’, ‘activity’, ‘strt_time’, ‘end_time’ and ‘user_1’). Figure 5.21 shows an example of the structure of the ‘timeline activities to display for bookmark/case report’ database.



	Plot_seq	Application	Activity	Strt_time	End_time	User_1
	Filter	Filter	Filter	Filter	Filter	Filter
78	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,02,23)	Date(2015,1,23,14,02,23)	192.168.200.27
79	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,02,26)	Date(2015,1,23,14,02,26)	192.168.200.27
80	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,14,56)	Date(2015,1,23,14,14,56)	192.168.200.27
81	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,19,22)	Date(2015,1,23,14,19,22)	192.168.200.27
82	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,26,00)	Date(2015,1,23,14,26,00)	192.168.200.27
83	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,29,06)	Date(2015,1,23,14,29,06)	192.168.200.27
84	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,33,16)	Date(2015,1,23,14,33,16)	192.168.200.27
85	1	Google Docs	Google Docs-Editing	Date(2015,1,23,14,51,09)	Date(2015,1,23,14,51,09)	192.168.200.27
86	1	Google Docs	Google Docs-Editing	Date(2015,1,23,19,39,42)	Date(2015,1,23,19,39,42)	192.168.200.27
87	1	Google Docs	Google Docs-Editing	Date(2015,1,24,00,40,45)	Date(2015,1,24,00,40,45)	192.168.200.27
88	1	Google Docs	Google Docs-Editing	Date(2015,1,24,01,10,38)	Date(2015,1,24,01,10,38)	192.168.200.27
89	1	Google Docs	Google Docs-Editing	Date(2015,1,24,01,39,46)	Date(2015,1,24,01,39,46)	192.168.200.27
90	1	Google Docs	Google Docs-Editing	Date(2015,1,24,02,40,27)	Date(2015,1,24,02,40,27)	192.168.200.27
91	1	Google Docs	Google Docs-Editing	Date(2015,1,24,03,43,11)	Date(2015,1,24,03,43,11)	192.168.200.27
92	1	Hotmail	Hotmail-Compose Email	Date(2015,0,28,20,52,41)	Date(2015,0,28,20,52,41)	192.168.200.27
93	1	Hotmail	Hotmail-Compose Email	Date(2015,0,28,20,52,56)	Date(2015,0,28,20,52,56)	192.168.200.27
94	1	Hotmail	Hotmail-File Attachment	Date(2015,1,17,20,05,32)	Date(2015,1,17,20,05,32)	192.168.200.27
95	1	Hotmail	Hotmail-Insert a Recepient	Date(2015,1,02,23,20,17)	Date(2015,1,02,23,20,17)	192.168.200.27
96	1	Skype	Skype-Idle	Date(2015,2,02,17,03,28)	Date(2015,2,02,17,03,28)	192.168.200.27
97	1	Skype	Skype-Click on contacts	Date(2015,1,27,20,22,41)	Date(2015,1,27,20,22,41)	192.168.200.27
98	1	Skype	Skype-Click on contacts	Date(2015,1,28,05,24,00)	Date(2015,1,28,05,24,00)	192.168.200.27

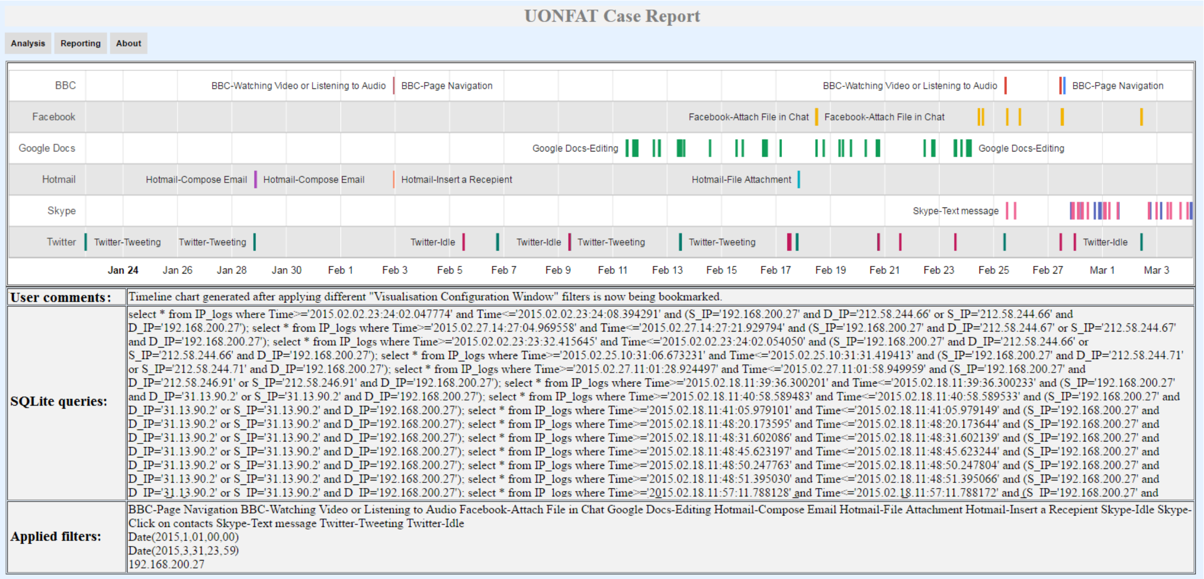
Figure 5.21: 'Timeline activities to display for bookmark/case report' database (Reporting Engine)

The last five fields of this database will be filled in by the engine with the data copied from field numbers one to five of the 'timeline activities to display' database (shown in Figure 5.14 in subsection 5.3.5). Each time the investigator bookmarks a timeline chart, relevant data will be kept on added into both of the associated databases by the 'reporting engine'. The 'plot_seq' data field in these two associated databases are used as an index for identifying the records that belong to each submission. On each occasion when a new bookmark is added, the 'plot_seq' field value will be incremented.

It is important to point out that both databases, created by the 'reporting engine', are utilised as a common source of data for both recovering the saved bookmarks and producing the case report. While all data fields of the 'info for bookmark/case report' database are used for producing the case report, only the 'plot_seq' and 'user_comments' fields are utilised for recreating the saved bookmarks. However, all data fields of the 'timeline activities to display for bookmark/case report' database will be utilised for both the recreation of saved bookmarks and the production of case report.

Earlier in subsections 5.3.4 and 5.3.5, it was pointed out that both 'visualisation configuration' and 'timeline activities to display' databases are temporary databases. Data stored in those databases are replaced whenever the investigator supplies a new set of filter options. Doing so allows the investigator to apply various filters and examine the resulting timeline charts, yet at the same time enables the system to save storage space by not saving all of those filters and the related data. The assumption is that the filters and the associated timeline chart data needs to be saved only if the investigator finds it interesting enough to be saved for the case report. Thus only if and when the bookmarking feature is selected, the 'reporting engine' will copy all records present in those two databases permanently into the 'info for bookmark/case report' and 'timeline activities to display for bookmark/case report' databases respectively.

At a later point, when the investigator opts to view the saved bookmarks, the engine will recreate the bookmarked timeline charts along with the comments added by the investigator earlier using data retrieved from the appropriate data fields of both databases. Similarly, the engine will produce the case report that will include the saved timeline charts, the comments added by the investigator, the filter options supplied for generating the saved timeline chart and the SQLite queries that were used for identifying the packets responsible for the user activities. An example of the case report is illustrated in Figure 5.22.



User comments:	Timeline chart generated after applying different "Visualisation Configuration Window" filters is now being bookmarked.
SQLite queries:	select * from IP_logs where Time>='2015.02.02.23:24:02.047774' and Time<='2015.02.02.23:24:08.394291' and (S_IP='192.168.200.27' and D_IP='212.58.244.66' or S_IP='212.58.244.66' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.27.14:27:04.969558' and Time<='2015.02.27.14:27:21.929794' and (S_IP='192.168.200.27' and D_IP='212.58.244.67' or S_IP='212.58.244.67' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.02.23:23:32.415645' and Time<='2015.02.02.23:24:02.054050' and (S_IP='192.168.200.27' and D_IP='212.58.244.66' or S_IP='212.58.244.66' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.25.10:31:06.673231' and Time<='2015.02.25.10:31:31.419413' and (S_IP='192.168.200.27' and D_IP='212.58.244.71' or S_IP='212.58.244.71' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.27.11:01:28.924497' and Time<='2015.02.27.11:01:58.949959' and (S_IP='192.168.200.27' and D_IP='212.58.246.91' or S_IP='212.58.246.91' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:39:36.300201' and Time<='2015.02.18.11:39:36.300233' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:40:58.589483' and Time<='2015.02.18.11:40:58.589533' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:41:05.979101' and Time<='2015.02.18.11:41:05.979149' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:20.173595' and Time<='2015.02.18.11:48:20.173644' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:31.602086' and Time<='2015.02.18.11:48:31.602139' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:45.623197' and Time<='2015.02.18.11:48:45.623244' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:50.247763' and Time<='2015.02.18.11:48:50.247804' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:51.395030' and Time<='2015.02.18.11:48:51.395066' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:57:11.788128' and Time<='2015.02.18.11:57:11.788172' and (S_IP='192.168.200.27' and

Applied filters:	BBC-Page Navigation BBC-Watching Video or Listening to Audio Facebook-Attach File in Chat Google Docs-Editing Hotmail-Compose Email Hotmail-File Attachment Hotmail-Insert a Recepient Skype-Idle Skype-Click on contacts Skype-Text message Twitter-Tweeting Twitter-Idle
	Date(2015,1,01,00,00)
	Date(2015,3,31,23,59)
	192.168.200.27

Figure 5.22: Case report

The user interface of the UO-NFAT dashboard will contain options (i.e. view saved bookmarks, view case report) for viewing the saved bookmarks and producing the case report, which when selected will initiate the 'reporting engine', the final part of the UO-NFAT interface layer, to perform the tasks of bookmarking and producing the case report.

5.4 Discussion

The research was set to propose a novel network forensic analysis tool that will assist in the investigation by generating user interactions from low level network traffic metadata and thus reduce the time and effort put forth by the investigator. The architecture was developed so as to incorporate a case management approach and other industry standard forensic processes into the proposed tool that are exhibited by the established forensic file system analysers such as FTK and Encase.

Prior to developing the architecture, a list of system requirements for the proposed UO-NFAT was produced. Those were the requirements that the proposed tool was expected to meet in order for it to fulfil the research goal. Table 5.1 provides a revisit to the listed system requirements along with an overview of the approach utilised by the tool in meeting each of those requirements.

Table 5.2: System requirements and UO-NFAT approach

System Requirements	UO-NFAT Approach
Effectively deal with the issue of enormous traffic volume	Eliminate noise and use metadata
Overcome the difficulty of dealing with the encrypted traffic	Focus on the network metadata
Create user interactions from raw network traffic metadata	Use the interaction based approach
Exhibit Computer Forensic tools' functionalities	Develop the selected functionalities into the tool (case management, hashing, visualisation & reporting)
Effectively reduce the investigator's cognitive load and the time/cost of the investigation	By achieving the above steps

After going through the architecture, it is evident that the UO-NFAT system was successful in implementing each of the approaches resulting in meeting the system requirements. It is through the combined effort exerted by all engines and the associated databases of the architecture that the different approaches utilised by the UO-NFAT for tackling each of the system requirements were implemented.

The developed UO-NFAT architecture has six different engines that operate in harmony with each other in order to fulfil the research goal. Table 5.2 contains a list constructed that holds the essence of the previous sections of this chapter. The table shows all processing engines of the UO-NFAT architecture, the tasks that they have accomplished and the gains that they deliver to the system they constitute, that system being the UO-NFAT.

Table 5.3: UO-NFAT architecture review table

Processing Engines	Tasks Accomplished	UO-NFAT Gains
The Case Management Engine	Authentication, Authorisation and Accounting	Incorporates case management functionality, helps to maintain integrity, chain of custody, and collaborative working
	New Case Creation	
	Case Archive Management	
	Visualisation Editing	
The Pre-processing Engine	Hashing Verification	Capability to authenticate the data source and to maintain the integrity of evidence
	Data Structuring	Reduces data volume prior to analysis, aids in reducing investigation time
The User-Activity Analyser	Analysis of network traffic metadata using the novel interaction based approach	Reduction of data volume
		Way around encryption
		Reduces the investigation time
		Identification of Online User Interactions
The Visualisation-Configuration Engine	Accepting filtering options	Ability to filter results based on user activities, start date/time, end date/time, user IP address. Interactive with the investigator
The Visualisation Engine	Visualisation Charts Generation	Incorporates visualisation functionality
	Raw Traffic Display	Incorporates raw traffic display to imitate existing network analysis tools
	Analysis of new user IP	Ability to analyse traffic for new user without interruption
The Reporting Engine	Bookmarking	Adds bookmarking feature
	Case Report Generation	Incorporates reporting functionality

Since the novel interaction based analysis approach utilises the network traffic metadata, it largely reduces the volume of data that needs to be analysed. Compared to the packet based and flow based analysis techniques, this approach addresses the issue of enormous data volume by eliminating noise (such as the machine-to-machine packets) and by drilling down through the network metadata to isolate the packet(s) that are responsible for the user interactions. Also, by using only the metadata, the interaction based approach finds a way of dealing with the issue of encryption and manages to identify the user activities. Since this novel interaction based approach has laid the foundation (in terms of research contribution) for the UO-NFAT architecture, the tool inherits the capability to deal with the issues of enormous data volume and encryption. This is also a huge advantage in reducing the time taken for the investigation. And the tool can exhibit these capabilities by means of the 'user-activity analyser' within the architecture.

The proposed tool is also expected to be built based upon a case management premise thus being able to function in fashion similar to that of the established computer forensic tools such as FTK and Encase. The 'case management engine' of the architecture allows the system to create new cases and manage old cases from the archive, which will allow the UO-NFAT to hold an industry standard approach. The hashing verification and data structuring powered by the 'pre-processing engine' will make the UO-NFAT capable of authenticating the source of data by checking for any alteration and reducing the volume of metadata to be analysed by separating data only for the selected user IP address. This, in certain scenarios, will reduce the time taken for analysis and, in effect, the time taken for the investigation.

The remaining three engines of the architecture (i.e. 'the visualisation-configuration engine', 'the visualisation engine' and the 'reporting engine') equip the UO-NFAT with an interactive dashboard that offers multiple functionalities. Along with providing a user interactive nature to the tool, these three engines enable the tool with visualisation and reporting functionalities, which are ideal for an industry standard forensic analysis tool. The engines also let the UO-NFAT to offer the investigator to analyse for new user IP address without breaking the continuity of the investigation and to bookmark interesting charts. The tool will be able to display the raw network traffic in a way that mimics the existing network analysis tools such as Wireshark and tcpdump.

It is evident that the different engines of the architecture enable the UO-NFAT to possess functionalities of a professional forensic analysis tool and to overcome limitations and challenges of the existing network forensic tools to certain extend. Also, the novel interaction based approach has powered the tool to analyse the network traffic metadata effectively and thereby reducing the investigator's effort and the time/cost of the investigation.

In comparison to the existing network analysis tools such as tcpdump, UO-NFAT offers a graphical user interface based interaction with the investigator. Perhaps, most importantly, the UO-NFAT reduces the time and effort required from the investigator in the analysis of the network traffic. Existing network analysis tools require the investigator to go through each and every data packets to extract any interesting and meaningful information. The volume and raw nature of the traffic makes the analysis a less convenient task for the investigator. The UO-NFAT, in comparison, generates high level information about the user interactions and provides the investigator with an interactive dashboard that reduces the input effort and creates a user friendly atmosphere to work in. Compared to the existing commercial NFATs such as NIKSUN's NetDetector Suite, RSA's Netwitness Suite and the open source NFATs such as PyFlag and Xplico, the prime factor that separates the UO-NFAT is the tool's ability to identify online user interactions for the popular Internet based services from the network metadata. The aforementioned tools perform the traffic analysis either based on a deep packet inspection approach or a flow based approach. The novel interaction based approach built into the UO-NFAT encompasses the tool to target the specific packets' metadata that are responsible for the user interactions. Compared to tools that analyse the packet content, UO-NFAT provides the users with data privacy by focusing only on the metadata.

UO-NFAT is a tool that can assist the investigator by identifying the user interactions through the analysis of network traffic metadata and thus can reduce the investigator's effort. Since being built on a case management concept and being able to incorporate functionalities such as hashing, visualisation and reporting, the time taken for investigation can be reduced. However, UO-NFAT is not a tool that can single-handedly conduct an investigation and find all evidences for a cybercrime. It is important to understand that UO-NFAT is a tool that can assist in the investigation by providing valuable information that would help the investigator to narrow down the suspects or to confirm or discard any hypothesis made during the investigation. UO-NFAT is designed to identify the online user interactions, which can be useful when it comes to dealing with incidents such as crimes involving the insiders (e.g. current employees, former employees, and business associates), misuse of organisational data, and policy violations within the organisational environment. Knowing what the employees were up to when a particular incident occurred is crucial in dealing with such incidents and that is when the UO-NFAT can be of use; being able to provide the investigator with information that may have the potential of becoming supporting evidences. In the changing landscape of user privacy, the extent to which organisations can access information about the online activities of their employees are becoming questionable. Even though the architecture proposed in this research is purely aimed at cybercrime investigation, in the wrong hands it could become a system for exploiting private information thereby raising thoughts of its legal and ethical implications.

The developed UO-NFAT architecture provides a platform for the investigator to not only analyse the network traffic metadata without having the burden of going through the raw content of network packets but also to carry out the investigation in a case management based fashion. Combined with the web based user interface of the tool, the UO-NFAT offers a convenient and effective way of conducting the investigation and reduces both the time taken by the investigation and the cognitive load on the investigator.

5.5 Conclusion

The current stage of the research developed the architecture for a novel web based network forensic analysis tool that utilises an innovative approach (i.e. the interaction based analysis approach) for generating meaningful information (i.e. online user interactions) from low level network traffic metadata. The User Oriented Network Forensic Analysis Tool (UO-NFAT) is also based upon the industry standard approach of case management that brings together all core forensic activities under a single application process thereby reducing the time taken to complete an investigation.

The UO-NFAT developed based on the proposed architecture would have a dashboard themed user interface that will portray different interactive visualisation charts to graphically represent the analysis results, provide different filter options that would allow the investigator to interact with the visualisation charts and give the provision to display the raw network metadata if the investigator prefers to examine. Combining all these features and functionalities, the proposed UO-NFAT is assumed to assist in the investigation and reduce the investigator’s cognitive load along with the time of the investigation.

6 Prototype Implementation

6.1 Introduction

The stage after designing the architecture for the proposed User Oriented Network Forensic Analysis Tool (UO-NFAT) was to develop a functional prototype. Prime goals to achieve in this stage was to build the prototype that is capable of proving the concept – identifying the online user activities within organisational environment in order to assist in the investigation, utilising the interaction based analysis of the raw network meta-data – that was discussed in Chapter 4 and eventually giving shape to the designed UO-NFAT architecture that was discussed in Chapter 5. As discussed in the architecture chapter, the proposed UO-NFAT is a web based tool that will reside in a web server within the organisational intranet. Therefore, the investigator will be interacting with the system via the tool's user interface. Due to the web based nature of the tool, the overall appearance of the user interface will be inspired by a web-page based appearance.

This chapter discusses different parts of the prototype's user interface through screenshots and explanations in Section 6.2 and Section 6.3 so that the reader gets a better understanding of how the prototype functions from an investigator's point of view. From a visual perspective, different user interface windows put forth by the UO-NFAT prototype can be classified into two segments – a case management segment and the dashboard segment that make Sections 6.2 and 6.3 of the Chapter. The importance is given to the prototype's user interface and how the investigator will be operating the tool during the course of an investigation. Section 6.4 and Section 6.5 includes the discussion and conclusion respectively.

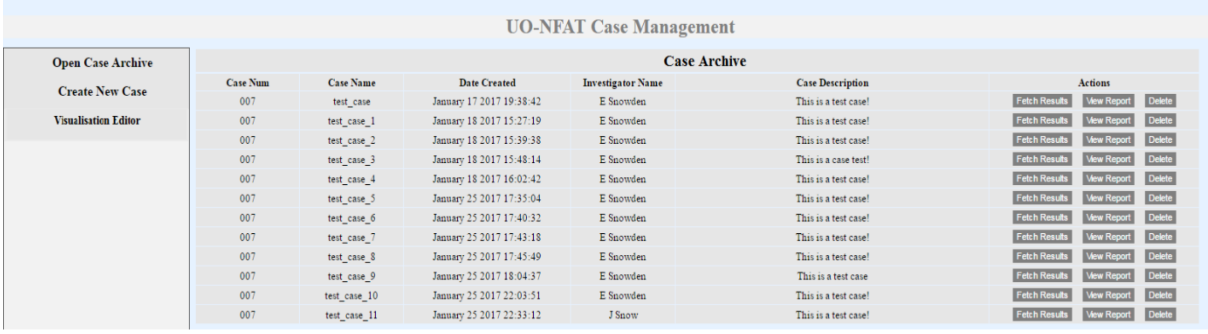
6.2 Case Management

In UO-NFAT prototype, 'case management' is used as a term that collectively represents different operations carried out by the investigator that shapes the case management functionality exhibited by the tool. The user interface windows that come under 'case management' are – case archive, visualisation editor and the new case creator, which are discussed in subsections 6.2.1, 6.2.2 and 6.2.3 below.

6.2.1 Case Archive

Figure 6.1 is for the 'UO-NFAT Case Management' segment and is where the UO-NFAT user interface begins. The page has two menu items – Open Case Archive and Create New

Case – that allows the tool to accomplish the ‘case management’ functionality. As displayed in Figure 6.1, the ‘Case Archive’ is the default starting point of the prototype’s user interface.



UO-NFAT Case Management

Case Archive

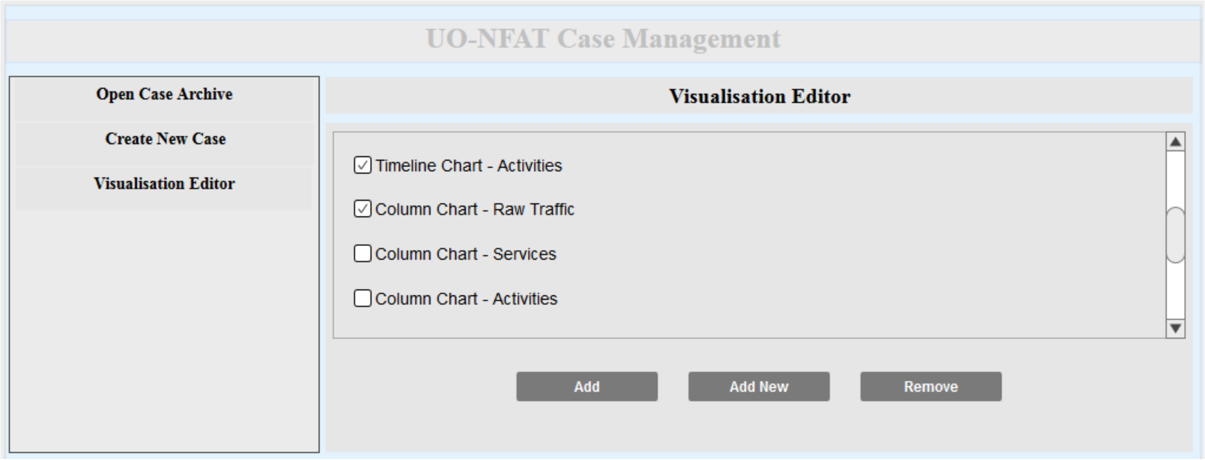
Case Num	Case Name	Date Created	Investigator Name	Case Description	Actions
007	test_case	January 17 2017 19:38:42	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_1	January 18 2017 15:27:19	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_2	January 18 2017 15:39:38	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_3	January 18 2017 15:48:14	E Snowden	This is a case test!	Fetch Results View Report Delete
007	test_case_4	January 18 2017 16:02:42	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_5	January 25 2017 17:35:04	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_6	January 25 2017 17:40:32	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_7	January 25 2017 17:43:18	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_8	January 25 2017 17:45:49	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_9	January 25 2017 18:04:37	E Snowden	This is a test case	Fetch Results View Report Delete
007	test_case_10	January 25 2017 22:03:51	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_11	January 25 2017 22:33:12	J Snow	This is a test case!	Fetch Results View Report Delete

Figure 6.1: UO-NFAT Case Management → Case Archive

The ‘case archive’ lists all cases which were created/analysed so far permitting the investigator to gather information such as case number, case name, date/time of creation, name of the investigator who performed the analysis and the case description. Also, the investigator is either permitted to ‘fetch the analysis results’ or to ‘view the case report’ or to delete the entire case. The ability of an investigator to view different elements of the case archive will depend on their role within the investigation team. The authorisation rights assigned to each investigator by the AAA property of the tool will be acting as a regulating factor in order to maintain the confidentiality of cases. For instance, details shown in Figure 6.1 can be considered as an archive that is available to an individual that fills the role of a team leader within the investigation team. A junior investigator, on the other hand, may not be presented with the entire case-list, may not be able to view the reports of certain cases or may not be able to delete any case files.

6.2.2 Visualisation Editor

Selecting the ‘Visualisation Editor’ option from the menu on the left-hand side of the case management segment will take the investigator to a new user interface window as displayed in Figure 6.2.

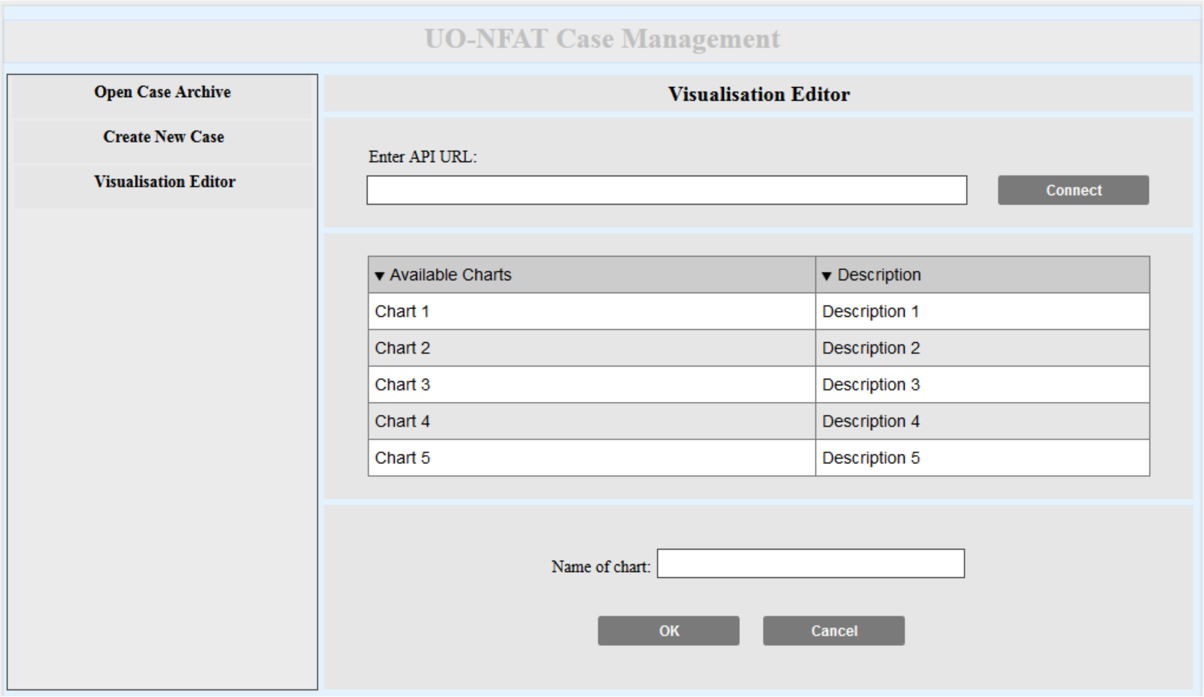


☑ Timeline Chart - Activities
☑ Column Chart - Raw Traffic
□ Column Chart - Services
□ Column Chart - Activities

Figure 6.2: Case management → Visualisation Editor (primary window)

Visualisation editor is a part of the prototype's user interface that allows the investigator to select the visualisation charts that will be displayed in the visualisation window of the dashboard. As displayed in Figure 6.2, the investigator will be provided with a list of visualisation charts that the tool is currently capable of plotting. Using the checklist, the current charts can be selected or unselected before the investigator chooses to 'add' or 'remove' those charts to or from the visualisation widow of the dashboard. Clicking the 'add' button after selecting the charts will result in adding new tabs to the visualisation window in which each tab will be allotted for each of the selected chart. Similarly, clicking the 'remove' button after selecting the charts will result in removing the corresponding visualisation tabs and charts from the visualisation window. Any selection or un-selection of visualisation charts made via the primary window of the visualisation editor will persist with all upcoming cases unless the options are modified by the investigator at some point in time.

The 'visualisation editor' primary window has a third button called 'Add New', which will allow the investigator to connect to third party visualisation APIs in order to add new visualisation chart(s). Clicking the 'add new' button will open a new interface window (i.e. the secondary window) that is illustrated in Figure 6.3.



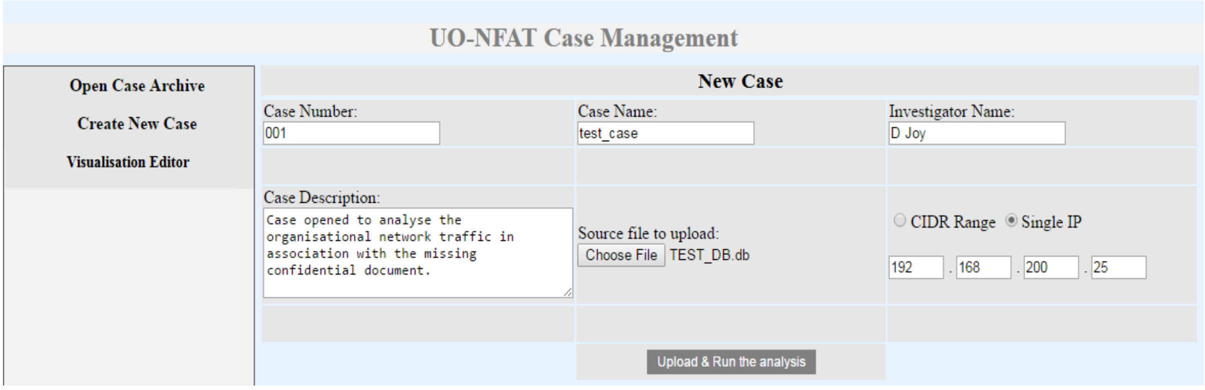
▾ Available Charts	▾ Description
Chart 1	Description 1
Chart 2	Description 2
Chart 3	Description 3
Chart 4	Description 4
Chart 5	Description 5

Figure 6.3: Case management → Visualisation Editor (secondary window)

In this interface window, the investigator can provide the URL for the third party visualisation API and 'connect' to the corresponding server. Following this, the connected server will provide the list of currently available charts along with their descriptions, as shown in the figure, so that the investigator can know what each chart is useful for. From this list, the investigator can decide which new visualisation chart can be added to the set of charts that the tool is already utilising. The name of the selected chart can be given in the text area provided and clicking 'OK' will successfully add the new chart and close the secondary window that was shown in Figure 6.3. The newly selected chart will be added to the list of existing charts (shown in Figure 6.2), which the investigator can add to the visualisation window of the dashboard by following the procedure that was explained via Figure 6.2.

6.2.3 New Case Creator

The 'Create New Case' menu item, as the name implies, directs the investigator towards creating a new case as illustrated in Figure 6.4. The investigator needs to provide mandatory information such as a case number, a case name, name of the investigator and a case description. Following this, the investigator must select the source database (.db) file containing the raw data (network traffic meta-data) to upload and also provide either a specific IP address or a range of IP addresses (the CIDR range) that the investigator is interested in.



UO-NFAT Case Management

Open Case Archive
Create New Case
Visualisation Editor

New Case

Case Number: 001

Case Name: test_case

Investigator Name: D Joy

Case Description: Case opened to analyse the organisational network traffic in association with the missing confidential document.

Source file to upload: Choose File TEST_DB.db

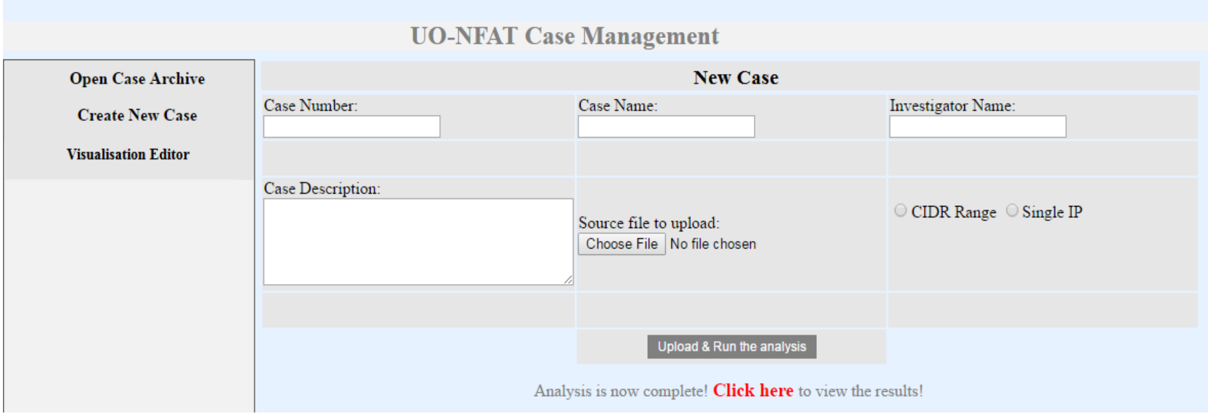
○ CIDR Range ● Single IP

192 . 168 . 200 . 25

Upload & Run the analysis

Figure 6.4: UO-NFAT Case Management → New Case

By clicking the ‘upload & run the analysis’ button the investigator can upload the source file into the web server and the tool will perform the ‘hashing verification’ to check for any alteration and duplication. If the uploaded file is already existing in the case archive, then the UO-NFAT will indicate it (by pointing out the corresponding case from the archive) so that the investigator can fetch the results and/or even perform a further analysis if needed. If the new hash value is not matching with any of the previous ones, then the tool will create a new case folder and continue with the analysis.



UO-NFAT Case Management

Open Case Archive
Create New Case
Visualisation Editor

New Case

Case Number:

Case Name:

Investigator Name:

Case Description:

Source file to upload: Choose File No file chosen

○ CIDR Range ○ Single IP

Upload & Run the analysis

Analysis is now complete! Click here to view the results!

Figure 6.5: UO-NFAT Case Management → New Case (analysis complete)

The analysis of traffic will be carried out in the back end which will be facilitated by Matlab and SQLite. Once the analysis is complete, the tool will display the message on the same interface page as shown in Figure 6.5 along with a link that says ‘Click Here’ to view the results. Clicking the link will take the investigator to the prototype’s dashboard.

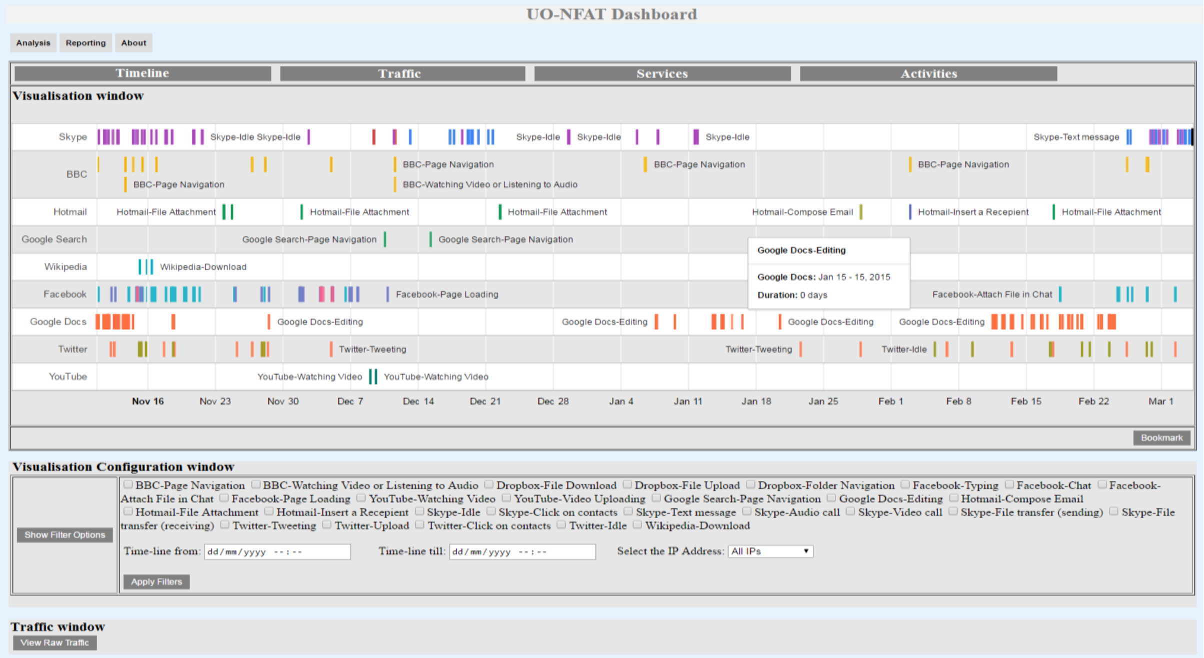
6.3 The Dashboard

The dashboard of the web based UO-NFAT prototype was designed and developed in such a way that it can present the results of the processed and analysed network traffic meta-data in a visual fashion. The prototype utilises timeline charts and column charts facilitated by

Google Charts in order to achieve the visual representation of the results. The dashboard of the prototype also provides various filter options that allows the investigator to interact with the displayed visualisation charts. Along with this, the dashboard provides a medium to view the raw network traffic meta-data if the investigator wants to take a look at the packets that are responsible for the identified user activities.

Structure wise, the dashboard consists of four segments – a navigation menu at the top of the dashboard followed by three display and operational windows – that allow the investigator to manage his/her way around the dashboard and perform multiple actions on the analysis results. More about the navigation menu of the dashboard will be discussed in subsection 6.3.4 because of its association with the details explained in that subsection.

Both the structure and functioning of the dashboard can be easily understood if explained with the help of screenshots of the UO-NFAT prototype's dashboard. Figure 6.6 is an illustration of the UO-NFAT Dashboard consisting of three main segments – a visualisation window, a visualisation configuration window and a traffic window.



Google Docs-Editing
Google Docs: Jan 15 - 15, 2015
Duration: 0 days

□ BBC-Page Navigation □ BBC-Watching Video or Listening to Audio □ Dropbox-File Download □ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat □ Facebook-Attach File in Chat □ Facebook-Page Loading □ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation □ Google Docs-Editing □ Hotmail-Compose Email □ Hotmail-File Attachment □ Hotmail-Insert a Recepient □ Skype-Idle □ Skype-Click on contacts □ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) □ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts □ Twitter-Idle □ Wikipedia-Download

Time-line from: dd/mm/yyyy --:-- Time-line till: dd/mm/yyyy --:-- Select the IP Address: All IPs ▾

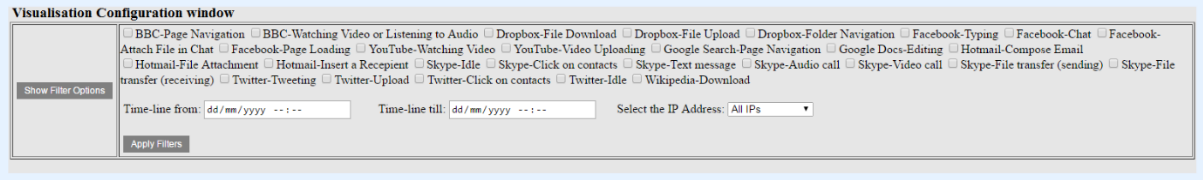
Figure 6.6: UO-NFAT Dashboard → Visualisation Window (Timeline)

6.3.1 The Visualisation Configuration Window

The visualisation configuration window is one of the crucial parts of the UO-NFAT dashboard as it serves as a user interface for receiving input from the investigator. The visualisation configuration window is designed in such a way that the investigator can make use of the different filter options in order to interact with different visualisation charts that will be plotted in the visualisation window. Depending on the filter options selected, different charts (e.g., timeline and/or column charts) within the visualisation window will be affected providing the investigator with desirable results.

The visualisation configuration window of the developed prototype offers four types of filter options – a 'user activity' filter options that contain all online user interactions included in this research, a 'time-line from' filter option that accepts the starting date/time for the timeline (in dd/mm/yyyy hh:mm format) of activities, a 'time-line till' filter option for accepting the ending date/time for the timeline (in dd/mm/yyyy hh:mm format) of activities and a 'select the IP address' filter option that can accept either one or all of the user IP addresses that was supplied previously (i.e. when the case was created) for the analysis.

There can be two scenarios when it comes to applying the filter options within the visualisation configuration window. First scenario is if and when the investigator does not select any filter option at all. The prototype is designed to automatically apply all filter options in such a scenario. Second scenario is when the investigator selects one or more of the filter options followed by clicking on the 'Apply Filters' button. In this case, the prototype will display user activities resulting from the selected filter options only. Illustrations for both scenarios are shown in Figure 6.7 and Figure 6.8 respectively.



Visualisation Configuration window

□ BBC-Page Navigation □ BBC-Watching Video or Listening to Audio □ Dropbox-File Download □ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat □ Facebook-Attach File in Chat □ Facebook-Page Loading □ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation □ Google Docs-Editing □ Hotmail-Compose Email □ Hotmail-File Attachment □ Hotmail-Insert a Recepient □ Skype-Idle □ Skype-Click on contacts □ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) □ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts □ Twitter-Idle □ Wikipedia-Download

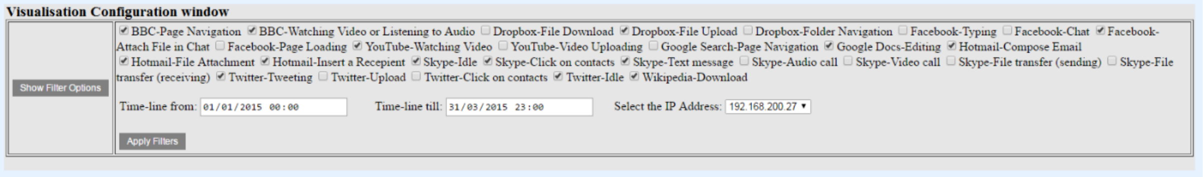
Time-line from: dd/mm/yyyy --:--

Time-line till: dd/mm/yyyy --:--

Select the IP Address: All IPs

Figure 6.7: Visualisation configuration window (no filter options selected)

In Figure 6.7, it can be noticed that no filter options are selected by the investigator. Therefore, the tool will automatically apply all filter options (i.e. all activities, from the beginning date/time till the end date/time of the identified user activities and all analysed user IP addresses). The engine uses this approach because it is only sensible to make the tool display all results when the investigator has not opted for any specific ones.


Visualisation Configuration window

☑ BBC-Page Navigation ☑ BBC-Watching Video or Listening to Audio □ Dropbox-File Download ☑ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat ☑ Facebook-Attach File in Chat □ Facebook-Page Loading ☑ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation ☑ Google Docs-Editing ☑ Hotmail-Compose Email
☑ Hotmail-File Attachment ☑ Hotmail-Insert a Recepient ☑ Skype-Idle ☑ Skype-Click on contacts ☑ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) ☑ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts ☑ Twitter-Idle ☑ Wikipedia-Download

Time-line from: 01/01/2015 00:00 Time-line till: 31/03/2015 23:00 Select the IP Address: 192.168.200.27 ▾

Figure 6.8: Visualisation configuration window (filter options selected)

Conversely, in Figure 6.8 the investigator has selected certain filter options. For example, the investigator would like to see the online user activities such as BBC page navigation, file upload on Dropbox and watching video on YouTube (to name a few) for the user IP address 192.168.200.27 during a time period starting from the midnight of 1st of January 2015 till 11:00 PM of the 31st of March 2015.

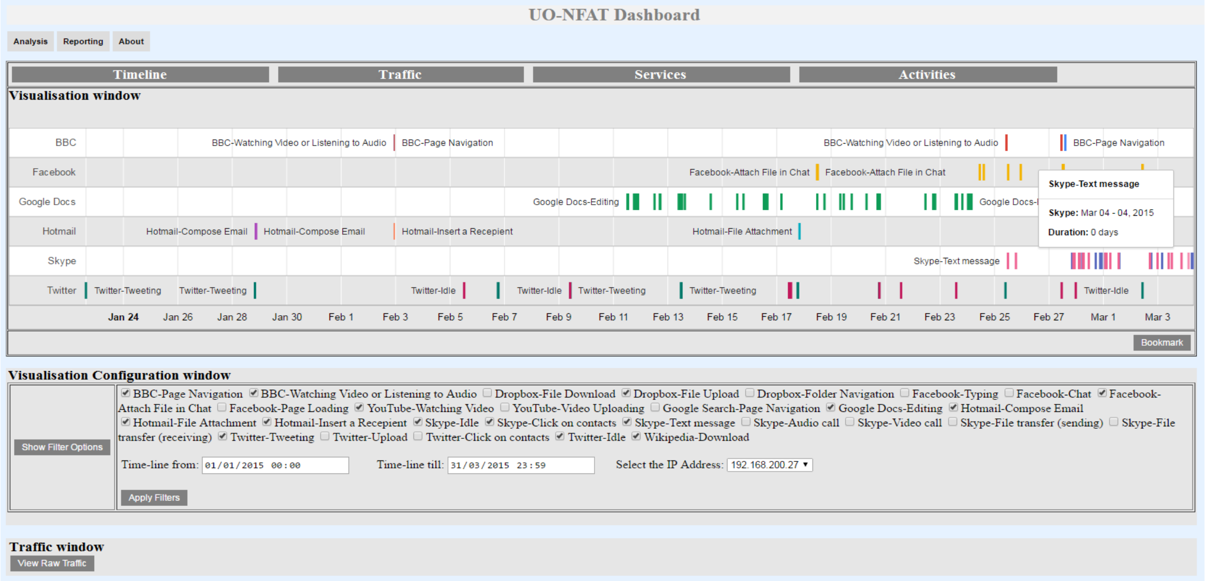
In both scenarios, the ‘visualisation configuration engine’ of the prototype will read all or the applied filter options from the user interface window and write the values into the appropriate backend database followed by passing the control to the ‘visualisation engine’ in order to plot the visualisation charts. The ‘visualisation configuration window’ permits the investigator to dynamically interact with the charts in the ‘visualisation window’ to obtain desirable results. Utilisation of filter options will aid the investigation process as it will allow the investigator to refine the results thereby eliminating the need to bookmark every single visualisation chart. Thus the investigator can be selective with the contents that can be bookmarked for the case report.

6.3.2 The Visualisation Window

Situated at the top of the dashboard, the ‘visualisation window’ plots the analysis results using Google Chart's time-line and column charts. The visualisation window also facilitates the investigator to steer through the plotted charts and to bookmark charts of particular interest.

The visualisation window consists of three parts – a top section containing the navigation buttons, a middle section containing a space to display the different visualisation charts and a bottom section containing the button to initiate the bookmarking option. The navigation panel at the top of the ‘visualisation window’ has four buttons – Timeline, Traffic, Services and Activities – that allow the investigator to steer through different visualisation charts displayed by the prototype. Utilising these buttons, the investigator can switch between the plotted timeline and column charts. It has to be pointed out that in each case, the tabs or buttons used for displaying different charts in the visualisation window could change according to the preferences made by the investigator during the case management segment. Because when the investigator uses the ‘visualisation editor’, it could result in adding or removing tabs and charts from the visualisation window in the dashboard. The middle section is where the ‘visualisation window’ will plot the visualisation charts on. While the time-line chart plots the extracted user activities against time, the three column charts plot the number of packets of total raw traffic, services and identified user activities generated by all analysed user IPs. The column charts give a cross comparison between the user IPs in terms of traffic associated with raw data, services and activities. By default, the visualisation window will display the timeline chart as it being the most important and the leading one among all displayed charts that are plotted off of the analysed results. The ‘Bookmark’ button in the bottom section will assist the investigator in operating the bookmarking functionality which will be discussed later in this section.

Figure 6.9 shows an example where the investigator has applied different filters (in the ‘visualisation configuration window’) in order to refine the results. The ‘visualisation window’ shows the resulting timeline chart. In the example illustrated in Figure 6.9, the generated timeline is for user IP – 192.168.200.27 and a time period between January 1st and March 31st of 2015 and for 15 selected user activities.



☑ BBC-Page Navigation ☑ BBC-Watching Video or Listening to Audio □ Dropbox-File Download ☑ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat ☑ Facebook-Attach File in Chat □ Facebook-Page Loading ☑ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation ☑ Google Docs-Editing ☑ Hotmail-Compose Email ☑ Hotmail-File Attachment ☑ Hotmail-Insert a Recepient ☑ Skype-Idle ☑ Skype-Click on contacts ☑ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) ☑ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts ☑ Twitter-Idle ☑ Wikipedia-Download

Time-line from: 01/01/2015 00:00 Time-line till: 31/03/2015 23:59 Select the IP Address: 192.168.200.27 ▾

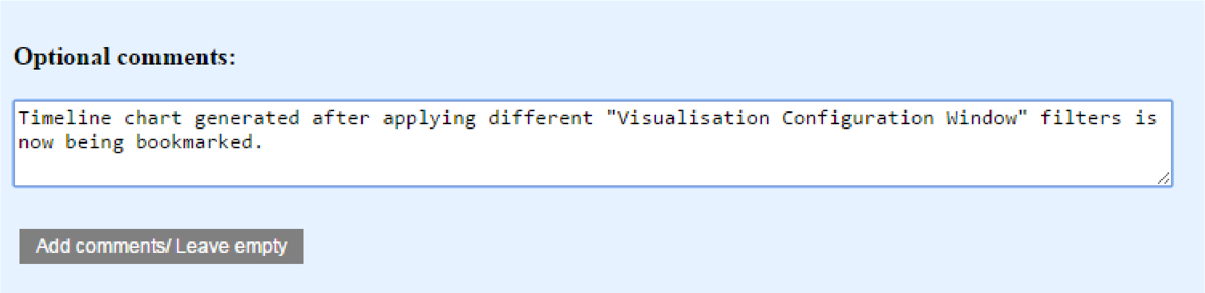
Figure 6.9: UO-NFAT Dashboard → Visualisation Window (Timeline)

From the displayed timeline chart, the investigator can understand that the user worked on the machine with IP address 192.168.200.27 accessed six online services such as BBC, Facebook, Google Docs, Hotmail, Skype and Twitter between January 1st and March 31st of 2015. The investigator can see that the user has performed online activities such as watching video or listening to audio and page navigation on BBC, attaching file while chatting on Facebook, editing Google documents, different activities on Hotmail such as inserting a recipient, composing e-mail and attaching file, clicking on contacts and text chatting on Skype and tweeting on Twitter. Likewise, the investigator had also selected user activities such as downloading (page loading) activity on Wikipedia, watching video on YouTube and file uploading activity on Dropbox along with the other filter options. However, it is interesting to notice that these three activities are not displayed in the timeline chart indicating their absence. From the resulting timeline, the investigator can understand that the user of interest (with IP address 192.168.200.27) did not access Wikipedia, watched YouTube or uploaded any file to Dropbox during the specified period of time (i.e. between January 1st and March 31st of 2015).

The timeline chart plays its part in reducing the investigator's effort by presenting all user activities that took place during a specific date/time window in a visual manner. This eliminates the need for the investigator having to manually examine large volumes of raw network traffic to find information about any interaction. In any crime investigation, whether it is conventional or cyber in nature, understanding the order of occurrence of the associated events is a crucial factor. The timeline chart fulfils this goal by plotting the user activities chronologically. Some user activities can be relevant to the case while others can be ignored. The investigator can determine the relevance of a specific user activity based on the type and the date/time of occurrence of that activity, which can be drawn from the timeline chart. Based on this information, the investigator can speculate a correlation between that activity and the crime that is being investigated.

The filter options that are specifically selected in the aforementioned example are only a representation of the options that are available to be utilised by the investigator and will alter according to the case requirements/investigator's interest. As mentioned above, with the use of filter options, the timeline chart provides a visual representation of the online user activities in a chronological order as well as in a refined fashion thereby enabling the investigator to have a better understanding of what the users were up to. Such a level of information could be interesting and useful while conducting cyber-crime investigation involving the employees in organisations (e.g., insider misuse). For instance, in a scenario where a classified organisational document was accessed and uploaded to Dropbox, the aforementioned information, provided by the tool, would help the investigator in finding out whether or not the employee using the IP address 192.168.200.27 was involved in such an activity. Such information could help the investigator in either ruling out suspects or strengthening the doubts, which could take him/her one step closer to solving the case.

It is important to discuss the role of bookmarking before proceeding to the remaining types of visualisation charts. The primary purposes of bookmarking are to allow the investigator to mark the current timeline chart as a saved bookmark and to save the timeline so that it can be reproduced later in the case report. The bookmarking functionality is facilitated by means of a 'Bookmark' button and a pop-up bookmark window for adding optional comments. When the investigator clicks on the 'Bookmark' button that is positioned in the bottom right hand corner of the 'visualisation window', the dashboard will trigger the bookmarking functionality. Following this action, the dashboard will open a new pop-up browser window that is illustrated in Figure 6.10.



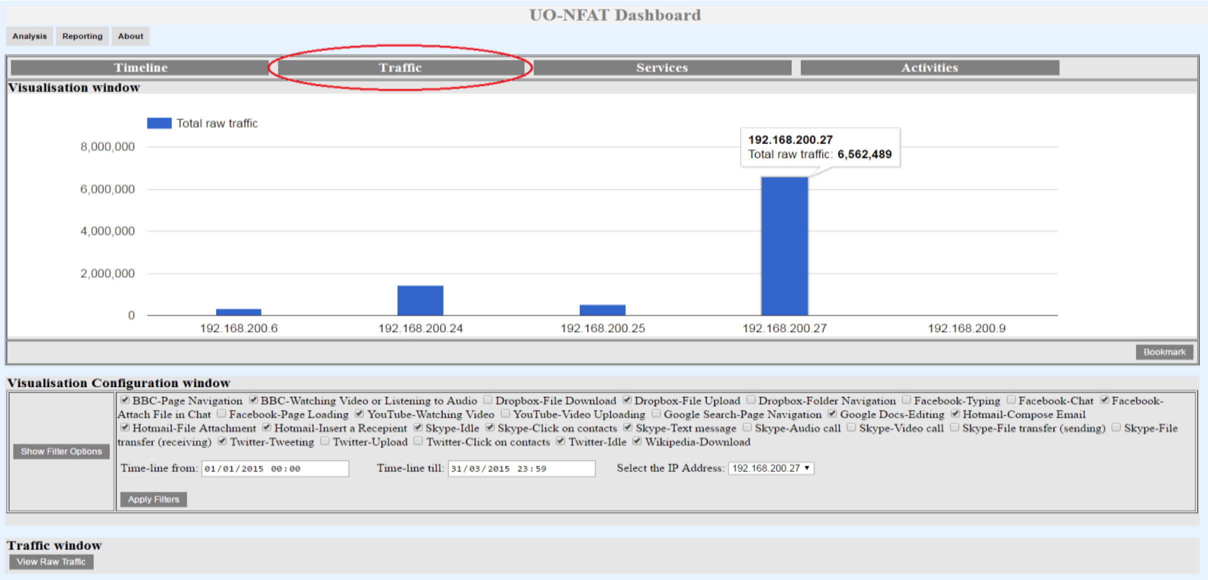
Optional comments:

Timeline chart generated after applying different "Visualisation Configuration Window" filters is now being bookmarked.

Figure 6.10: UO-NFAT Dashboard → Pre-Bookmark Window

The bookmark window provides the investigator with the option of adding appropriate comments within the text-area provided. The investigator has the freedom to decide whether or not to deliver any comments. Once the investigator is satisfied with the action taken, the 'Add comments or Leave empty' button can be utilised in order to complete the bookmarking process. This will store the relevant information such as the optional comments given by the investigator, the SQLite queries, and the filter options that were selected in the visualisation configuration window in the appropriate database. Pressing of this button will automatically close the bookmark window thereby returning the investigator to the dashboard. The investigator can repeat these steps involved in bookmarking for each and every timeline chart that is found interesting and is relevant to the case that is being investigated. As a result, the bookmaking functionality allows the investigator to select the preferable content that needs to be included in the case report. Later, the reporting engine will access and make use of this information prior to generating the case report. More on the case report and how the investigator can view the report is discussed in subsection 6.3.4.

As highlighted in Figure 6.11, if the investigator clicks on the ‘Traffic’ button from the top section of the visualisation window then the prototype will plot a column chart. This column chart will be displaying the IP address based distribution of raw packets within the limits of the start and end time filters that were applied in the visualisation configuration window. The investigator can draw some useful conclusions from the displayed column chart.



☑ BBC-Page Navigation ☑ BBC-Watching Video or Listening to Audio □ Dropbox-File Download ☑ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat ☑ Facebook-Attach File in Chat □ Facebook-Page Loading ☑ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation ☑ Google Docs-Editing ☑ Hotmail-Compose Email ☑ Hotmail-File Attachment ☑ Hotmail-Insert a Recepient ☑ Skype-Idle ☑ Skype-Click on contacts ☑ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) ☑ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts ☑ Twitter-Idle ☑ Wikipedia-Download

Time-line from: 01/01/2015 00:00 Time-line till: 31/03/2015 23:59 Select the IP Address: 192.168.200.27 ▾

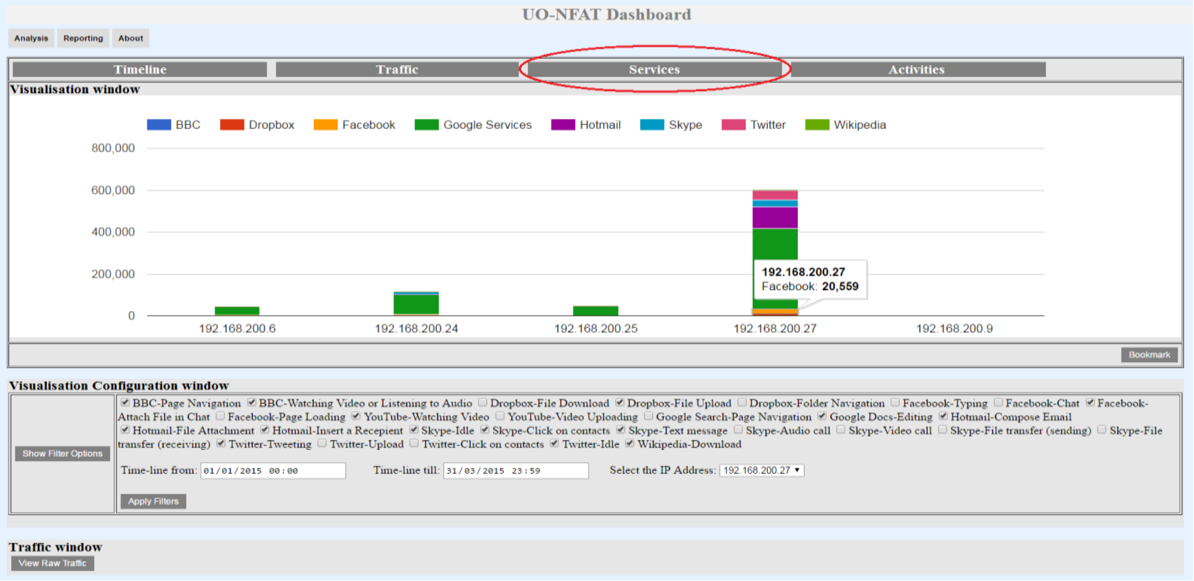
Figure 6.11: UO-NFAT Dashboard → Visualisation Window (Traffic)

For instance, the investigator can understand that between January 1st and March 31st of 2015 the IP address 192.168.200.6 was the least active one among the user IP addresses that were searched for. User with the IP address 192.168.200.27 was the most active one resulting in 6,562,489 packets in total during the specified period of time. The investigator can also notice that the IP address 192.168.200.9 was completely inactive during this time period (i.e. between Jan 1st and Mar 31st of 2015).

Information of such nature that is presented by the prototype will help the investigator in seeing what the user participation trends are and who the biggest and smallest contributors of traffic are. There could be a scenario where such information could feed some input into the investigation. For example, if the investigator, out of suspicion, decides to search for a user with a specific IP address (e.g. 192.168.200.9) but is clueless about the IP address being inactive, then the column chart in figure 6.11 can show that the aforementioned IP address was inactive during the time period that was searched for. Perhaps this could be a piece of detail that helps the investigator to think differently about the case before inspecting the timeline chart of user activities. Or, in another scenario where the same user IP address was a secondary suspect, the investigator can confirm that the IP address was not involved in any activity during the specified time duration. Perhaps, from an investigation standpoint, it only tells the investigator where not to look; but by taking that into account, the investigator can now modify the search criteria or refocus the user IP address that he/she is looking for.

Figure 6.12 exemplifies another visualisation chart, produced by the prototype's visualisation engine, that is plotted if and when the investigator clicks on the 'Services' button at the top of the visualisation window. The produced chart is a column chart that shows the IP address based distribution of packets for the nine Internet based services that are being considered. The purpose of the this visualisation chart is to assist the investigator to easily find out which the most and the least accessed online services are for each and every analysed user IP address.

In order to differentiate between the services, the chart assigns different colours to each and every Internet based services thereby making the service usage identification easy for the investigator. The visualisation chart can also generate callout boxes containing information such as the user IP address, the name of the service and the total number of packets that traversed the network due to the interaction with that specific service. Such a callout box can be viewed if the mouse is hovered over any given area of the individual column within the chart.



☑ BBC-Page Navigation ☑ BBC-Watching Video or Listening to Audio □ Dropbox-File Download ☑ Dropbox-File Upload □ Dropbox-Folder Navigation □ Facebook-Typing □ Facebook-Chat ☑ Facebook-Attach File in Chat □ Facebook-Page Loading ☑ YouTube-Watching Video □ YouTube-Video Uploading □ Google Search-Page Navigation ☑ Google Docs-Editing ☑ Hotmail-Compose Email ☑ Hotmail-File Attachment ☑ Hotmail-Insert a Recepient ☑ Skype-Idle ☑ Skype-Click on contacts ☑ Skype-Text message □ Skype-Audio call □ Skype-Video call □ Skype-File transfer (sending) □ Skype-File transfer (receiving) ☑ Twitter-Tweeting □ Twitter-Upload □ Twitter-Click on contacts ☑ Twitter-Idle ☑ Wikipedia-Download

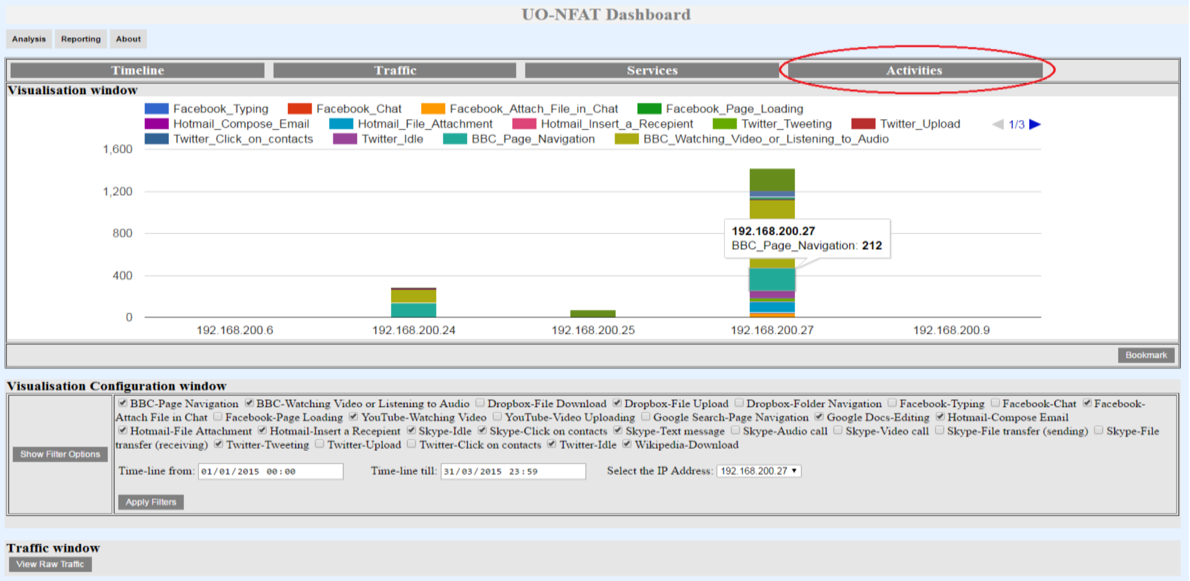
Time-line from: 01/01/2015 00:00 Time-line till: 31/03/2015 23:59 Select the IP Address: 192.168.200.27

Figure 6.12: UO-NFAT Dashboard → Visualisation Window (Services)

For instance, from the visualisation chart given in Figure 6.12, the investigator can notice that the user IP address 192.168.200.27 has accessed most of the Internet based services (i.e. BBC, Facebook, Google Services, Hotmail, Skype and Twitter) between January 1st and March 31st of 2015. It can also be noticed that the most accessed service for this IP address was Google Services. And user with the IP address 192.168.200.27 generated 20,559 packets while using the Facebook service.

With this type of visualisation chart, the investigator can visually compare between users based on their usage of the Internet based services. From an investigation point of view, viewing an individual user's Internet service usage may help the investigator to confirm his suspicion or, to understand what the employees were up to within the given time duration.

Figure 6.13 illustrates the user IP address based distribution of packets traversing the network that were generated as a result of the individual online user activity. The investigator can view the chart by clicking the 'Activities' button positioned at the top section of the visualisation window. The resulting visualisation chart will be plot with the user IP addresses along the horizontal axis and the number of network packets along the vertical axis. The chart will help the investigator to view the trends when it comes to the online activities of different users. Similar to the previous chart (shown in Figure 6.10), this visualisation chart also utilises colour coding scheme in order to differentiate between the plotted online user activities.



☑ BBC-Page Navigation ☑ BBC-Watching Video or Listening to Audio ☐ Dropbox-File Download ☑ Dropbox-File Upload ☐ Dropbox-Folder Navigation ☐ Facebook-Typing ☐ Facebook-Chat ☑ Facebook-Attach File in Chat ☐ Facebook-Page Loading ☑ YouTube-Watching Video ☐ YouTube-Video Uploading ☐ Google Search-Page Navigation ☑ Google Docs-Editing ☑ Hotmail-Compose Email ☑ Hotmail-File Attachment ☑ Hotmail-Insert a Recepient ☑ Skype-Idle ☑ Skype-Click on contacts ☑ Skype-Text message ☐ Skype-Audio call ☐ Skype-Video call ☐ Skype-File transfer (sending) ☐ Skype-File transfer (receiving) ☑ Twitter-Tweeting ☐ Twitter-Upload ☐ Twitter-Click on contacts ☑ Twitter-Idle ☑ Wikipedia-Download

Time-line from: 01/01/2015 00:00 Time-line till: 31/03/2015 23:59 Select the IP Address: 192.168.200.27

Figure 6.13: UO-NFAT Dashboard → Visualisation Window (Activities)

By viewing the chart, the investigator can understand that the user with the IP address 192.168.200.27 has performed most of the online activities between January 1st and March 31st of 2015. Making use of the hover-over callout boxes, the investigator can view more details such as the user IP address, the online user activity and the number of packets that were generated by that user activity. For example, in Figure 6.13 it can be observed that when the user with IP address 192.168.200.27 performed page navigation activity on BBC, 212 packets traversed the network between the aforementioned time duration.

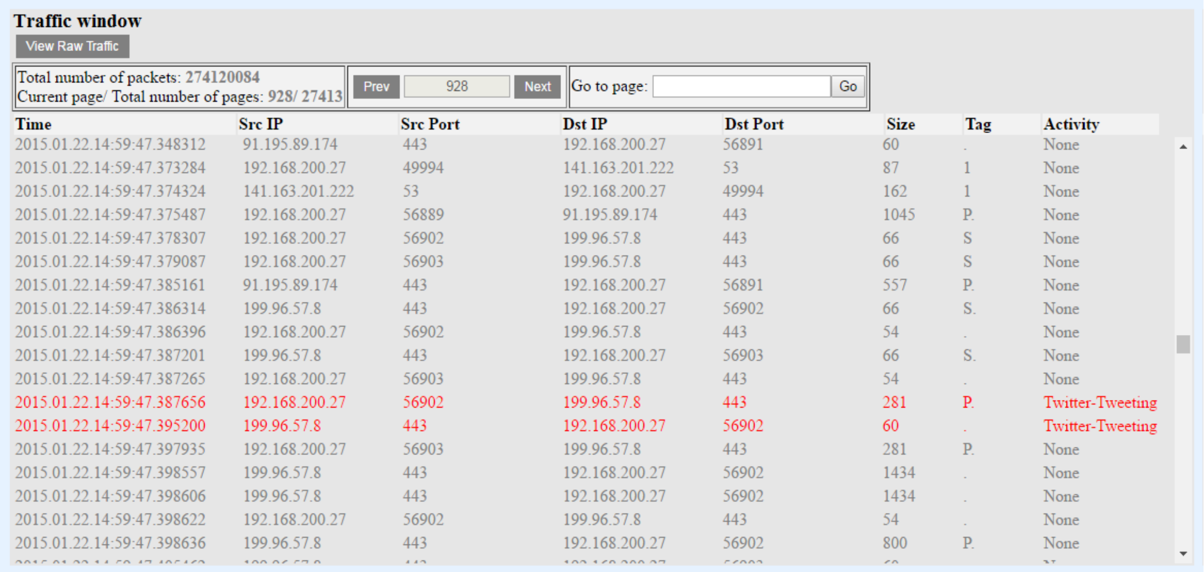
Examining the three column charts generated by the UO-NFAT prototype that were exemplified in Figure 6.11, Figure 6.12 and Figure 6.13 will lead to an important observation about the capability of the prototype. In Figure 6.11 in which the raw traffic was plotted against the user IP addresses; the IP address 192.168.200.9 did not produce any traffic during the specified time period. This implies that there should not be any packets detected for any of the Internet based services or the online user activities from this specific IP address. The confirmation of this implication can be performed by checking Figure 6.12 (that displays the online services for each analysed user) and Figure 6.13 (that displays the online user activities for each analysed user). And the confirmation turns out to be true, which gives strength to the ability of the tool to detect packets that are accountable for the overall traffic, Internet services and the online user activities.

The visualisation window is one of the key parts of the prototype's dashboard that contributes towards reducing the cognitive load on the investigator. Incorporating this functionality allows the investigator to spend a less amount of time manually browsing through the raw network traffic looking for packets of interest. It is evident that this can reduce both the pressure on the investigator and the time of investigation.

6.3.3 The Traffic Window

The 'Traffic Window' is the third and final display and operational window that the UO-NFAT prototype's dashboard is equipped with. As the name implies, the 'traffic window' allows the investigator to view the raw network meta-data from the source file that was uploaded at the beginning of the investigation. The window will display the traffic containing meta-data of the packets that are responsible for the identified online user activities in a highlighted fashion. And all of the highlighted traffic will be in relation with the online user activities that are displayed in the timeline chart within the 'visualisation window'.

Figure 6.14 illustrates what the ‘Traffic Window’ looks like on the UO-NFAT prototype’s dashboard. The ‘traffic window’ consists of three main parts – a ‘View Raw Traffic’ button at the top, a ‘navigation bar’ in the middle and the ‘traffic display section’ at the bottommost part of the window.



Traffic window

Total number of packets: 274120084
Current page/ Total number of pages: 928/ 27413

Go to page:

Time	Src IP	Src Port	Dst IP	Dst Port	Size	Tag	Activity
2015.01.22.14:59:47.348312	91.195.89.174	443	192.168.200.27	56891	60	.	None
2015.01.22.14:59:47.373284	192.168.200.27	49994	141.163.201.222	53	87	1	None
2015.01.22.14:59:47.374324	141.163.201.222	53	192.168.200.27	49994	162	1	None
2015.01.22.14:59:47.375487	192.168.200.27	56889	91.195.89.174	443	1045	P.	None
2015.01.22.14:59:47.378307	192.168.200.27	56902	199.96.57.8	443	66	S	None
2015.01.22.14:59:47.379087	192.168.200.27	56903	199.96.57.8	443	66	S	None
2015.01.22.14:59:47.385161	91.195.89.174	443	192.168.200.27	56891	557	P.	None
2015.01.22.14:59:47.386314	199.96.57.8	443	192.168.200.27	56902	66	S.	None
2015.01.22.14:59:47.386396	192.168.200.27	56902	199.96.57.8	443	54	.	None
2015.01.22.14:59:47.387201	199.96.57.8	443	192.168.200.27	56903	66	S.	None
2015.01.22.14:59:47.387265	192.168.200.27	56903	199.96.57.8	443	54	.	None
2015.01.22.14:59:47.387656	192.168.200.27	56902	199.96.57.8	443	281	P.	Twitter-Tweeting
2015.01.22.14:59:47.395200	199.96.57.8	443	192.168.200.27	56902	60	.	Twitter-Tweeting
2015.01.22.14:59:47.397935	192.168.200.27	56903	199.96.57.8	443	281	P.	None
2015.01.22.14:59:47.398557	199.96.57.8	443	192.168.200.27	56902	1434	.	None
2015.01.22.14:59:47.398606	199.96.57.8	443	192.168.200.27	56902	1434	.	None
2015.01.22.14:59:47.398622	192.168.200.27	56902	199.96.57.8	443	54	.	None
2015.01.22.14:59:47.398636	199.96.57.8	443	192.168.200.27	56902	800	P.	None

Figure 6.14: UO-NFAT Dashboard → Traffic Window (View raw traffic)

Clicking the ‘view raw traffic’ button triggers the tool to display the corresponding traffic into the ‘traffic display section’. It seemed more sensible to go with a design that would display the raw network traffic only on command and not by default. Hence the ‘View Raw Traffic’ button exists that will allow to view the raw network traffic only if the investigator wishes to do so. In fact, this approach lets the user interface of the dashboard to be loaded quickly as the viewing of raw network traffic is an optional activity.

The ‘navigation bar’ is situated right below the ‘view raw traffic’ button. The ‘navigation bar’ allows the investigator to browse through the traffic along with offering general information about the traffic window itself. Left-hand side panel of the ‘navigation bar’ displays the total number of packets in the source file, the current page number and the total number of pages. The mid-section of the navigation bar provides the investigator with the option to go to the previous or the next page from the current page that is being displayed. The right-hand side panel gives the option to go to any page number that the investigator wishes to in order to view the traffic on that page.

The traffic window of the prototype is designed to display 10,000 packets on each page. For instance, in the navigation bar shown in Figure 6.14, it can be noticed that there are more than

274 million packets in total (274120084 packets to be precise) within the uploaded source file and the values for current page and total number of pages are 928 and 27,413 respectively. This implies that it would take 27,413 pages to allocate the entire file containing the raw network meta-data as the traffic window displays only 10,000 packets per page. The dashboard has performed the backend calculations in order to display page number 928 where the investigator can find the packets (in highlights) that are responsible for the identified online activity. The tool will calculate the initial page number to display (928, in this case) based on the time-stamp of the earliest user activity that was identified and that appeared on the timeline chart in the 'visualisation window'.

The 'traffic display section' displays the network metadata belonging to the displayed activities highlighted along with the name of the activity. An idea that contributed towards the design of the 'Traffic Window' was to display the raw network traffic metadata in such a way that it imitates the existing network analysis tools such as Wireshark, tcpdump and NetworkMiner. From Figure 6.14 it can be noticed that each row of the displayed traffic contains information (i.e. metadata) about the packet such as the timestamp, source IP address, source port number, destination IP address, destination port number and the size of the packet (in bytes), that are similar to the type of information found in the aforementioned network analysis tools. Along with this, the dashboard also provides information such as the TCP flag that is set and the name of the online user activity that the packet is responsible for.

For instance, in this analysis it can be noticed from the timeline chart shown in Figure 6.9 that the first online activity that was identified for the user with the IP address 192.168.200.27 occurred on January 22nd of 2015 and it was a 'tweeting' activity while using the Twitter service. So when the investigator clicked on the 'view raw traffic' button, the UO-NFAT performed the calculations and displayed page number 928 (shown in the mid-section of the navigation bar of Figure 6.14) of the raw network traffic (shown in the 'traffic display section' of Figure 6.14) that contains the packets responsible for the aforementioned tweeting activity. The packets accountable for the online user activity are highlighted in red colour. And in the example that is being stated here, there are two packets in particular that traversed the network while the user did the online tweet. The first packet was sent from a port that is numbered 56902 of the user's system with the IP address 192.168.200.27 to the HTTPS port (port number 443) of the 'Twitter' server with the IP address 199.96.57.8 at 14:59:47.387656 hours on the 22nd of January 2015. This particular packet was 281 bytes long in size and had two of the TCP flags (PUSH and ACK flags) set. As a response to this, a packet was returned from the HTTPS port of the 'Twitter' server (with IP address 199.96.57.8) to the same port number (56902) of the user's system (with IP address 192.168.200.27). This packet was 60 bytes long and had the ACK flag of its TCP protocol set. This is an example in which the signature of the online user interaction was caused by a single packet.

The 'traffic window' becomes significant and useful when the investigator is interested in going through the raw network meta-data. As mentioned above, the traffic window exhibits the network traffic in a fashion similar to that of the established network analysis tools. By incorporating this functionality to the prototype, the UO-NFAT helps the investigation process in different ways. Firstly, it allows the investigator to track the identified online user activities back to the single, multiple or stream of raw network packet(s) that are responsible, if needed. Secondly, speaking from tool's credibility perspective, this functionality allows seeing that the identified user activities did originate from the raw traffic that is actually present in the source file; this demonstrates that the prototype offering the browsing of raw network meta-data certainly adds more value to the UO-NFAT as a capable network forensic analysis tool.

6.3.4 The Menu Tabs and Reporting

The main menu at the top of the dashboard has three tabs namely 'Analysis', 'Reporting' and 'About'. Utilising the 'Analysis' menu tab, the investigator can either analyse a new IP address that was not included in the current analysis or put a stop to the current analysis. Clicking on the menu items 'Analyse New IP' or 'Stop Analysis' under the 'Analysis' tab will allow the investigator to undertake either of the aforementioned tasks. Figure 6.15 shows the 'Analyse New IP' menu option, which is pointed out by the red arrow.

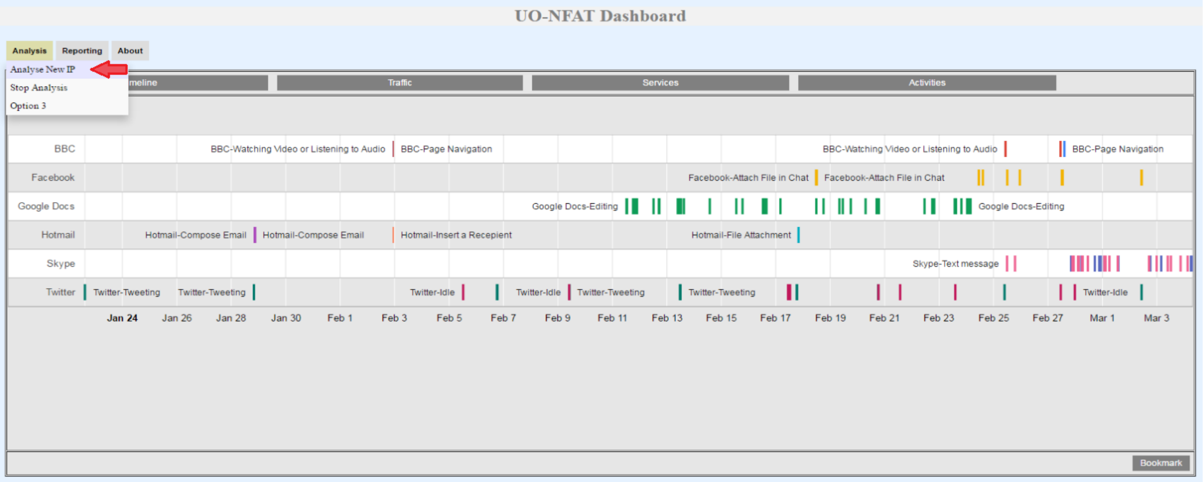
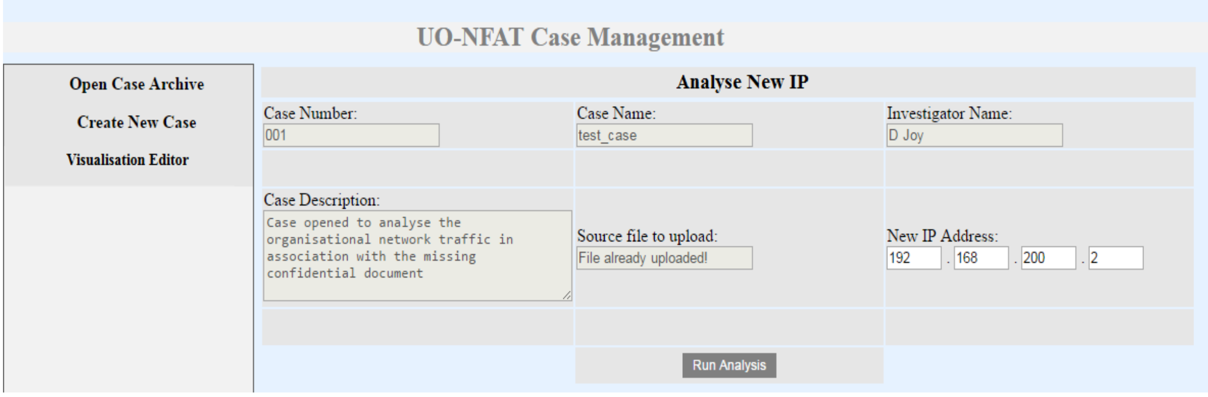


Figure 6.15: UO-NFAT Dashboard → Menu Option (Analyse New IP)

If the 'Analyse New IP' menu item is selected, the prototype will direct the investigator to the part of the 'Case Management' section of the prototype's user interface that is associated with analysing for a new IP address. The user interface appearance of this section is illustrated in Figure 6.16. It can be noticed that there are fields such as case number, case name, investigator name and case description in this figure that are already displaying information related to the current case. This is to offer confirmation that the investigator is dealing with the right case. At this point, the investigator should provide the new user IP address that needs to be analysed in the respective field. Once the IP address is entered and the 'Run Analysis' button is clicked, the prototype will start the analysis of the source file (containing the raw network traffic meta-data) and indicate the investigator when the process is complete.



Analyse New IP

Case Number: 001

Case Name: test_case

Investigator Name: D Joy

Case Description: Case opened to analyse the organisational network traffic in association with the missing confidential document

Source file to upload: File already uploaded!

New IP Address: 192 . 168 . 200 . 2

Figure 6.16: UO-NFAT Dashboard → Menu Option (Analyse New IP) → UO-NFAT Case Management → Analyse New IP

If the investigator chooses to select the 'Stop Analysis' menu item that is pointed out by the red arrow in Figure 6.17, then the prototype will direct the investigator back to 'Case Archive' part of the 'Case Management' section.

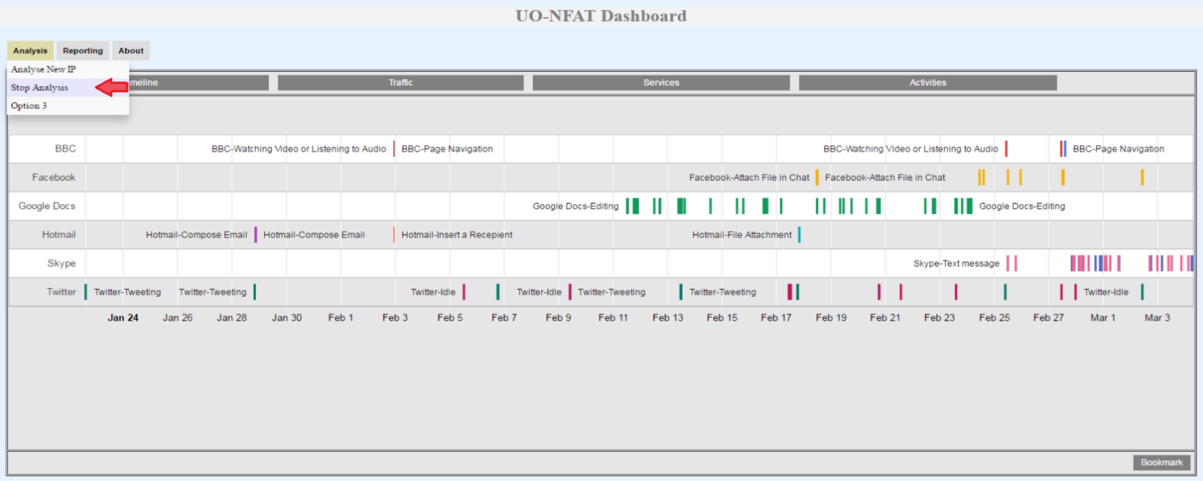
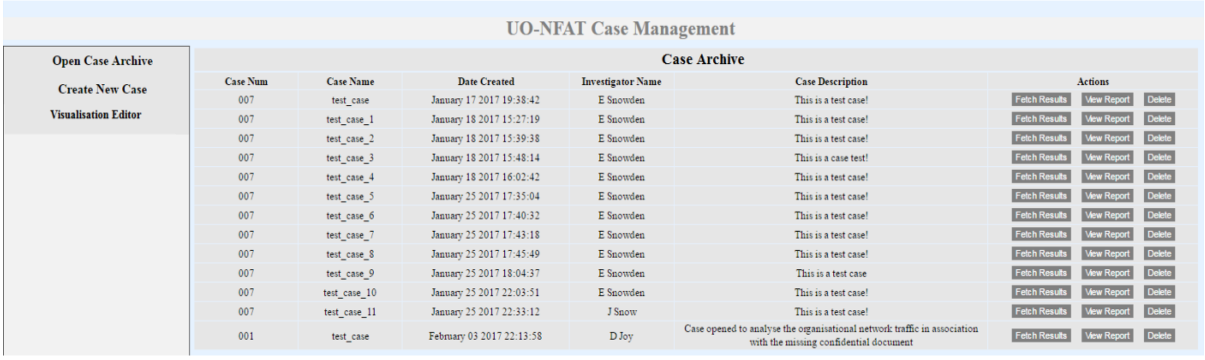


Figure 6.17: UO-NFAT Dashboard → Menu Option (Stop Analysis)

Figure 6.18 shows what the latest version of 'Case Archive' looks like. The prototype returns and displays the updated information from the corresponding case archive database with the last entry in the 'Case Archive' being the case that was analysed most recently.



Case Archive

Case Num	Case Name	Date Created	Investigator Name	Case Description	Actions
007	test_case	January 17 2017 19:38:42	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_1	January 18 2017 15:27:19	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_2	January 18 2017 15:39:38	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_3	January 18 2017 15:48:14	E Snowden	This is a case test!	Fetch Results View Report Delete
007	test_case_4	January 18 2017 16:02:42	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_5	January 25 2017 17:35:04	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_6	January 25 2017 17:40:32	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_7	January 25 2017 17:43:18	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_8	January 25 2017 17:45:49	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_9	January 25 2017 18:04:37	E Snowden	This is a test case	Fetch Results View Report Delete
007	test_case_10	January 25 2017 22:03:51	E Snowden	This is a test case!	Fetch Results View Report Delete
007	test_case_11	January 25 2017 22:33:12	J Snow	This is a test case!	Fetch Results View Report Delete
001	test_case	February 03 2017 22:13:58	D Joy	Case opened to analyse the organisational network traffic in association with the missing confidential document	Fetch Results View Report Delete

Figure 6.18: UO-NFAT Dashboard → Menu Option (Stop Analysis) → UO-NFAT Case Management → Case Archive

By exploiting the second tab of the menu – i.e. the 'Reporting' tab – the investigator can either open the case report for the current investigation or open the case archive. The 'View Report' and 'Open Archive' options under the 'Reporting' tab will allow the investigator to perform either of the actions mentioned above depending on the nature of the task.

The UO-NFAT prototype was designed and developed with the intention of incorporating different features into and 'Reporting' was one of them. The implemented prototype has the capability of producing a case report built into it. Because the prototype's design allows the investigator to activate the reporting feature through the menu on the dashboard, it seems more sensible to discuss the reporting feature in conjunction with the corresponding menu tab (i.e. View Report).

Selecting the ‘View Report’ option will open a new tab on the browser that will generate the case report for the investigation of the case that is currently being undertaken. The corresponding menu item is displayed in Figure 6.19 with the red arrow pointing towards the ‘View Report’ option.

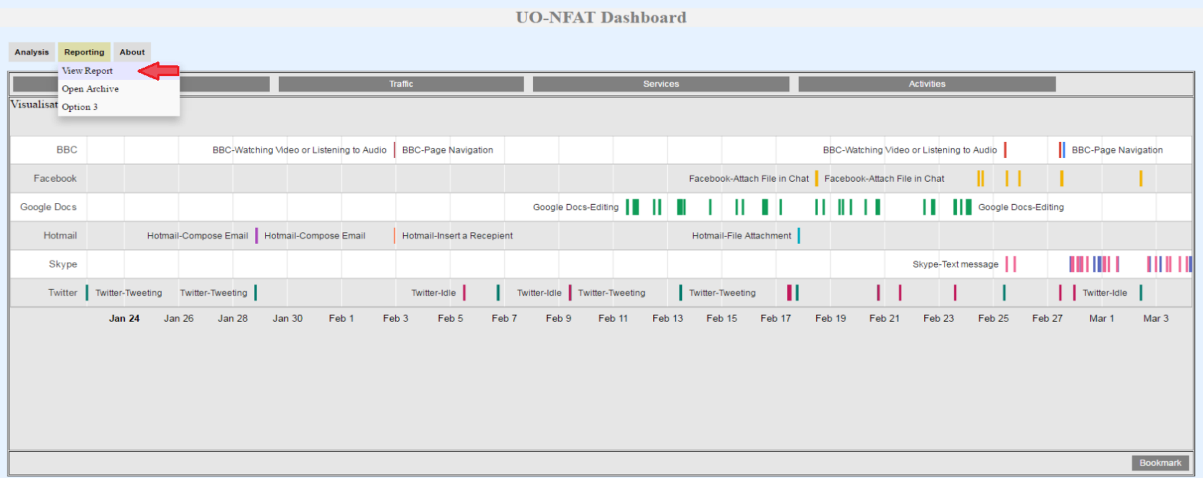
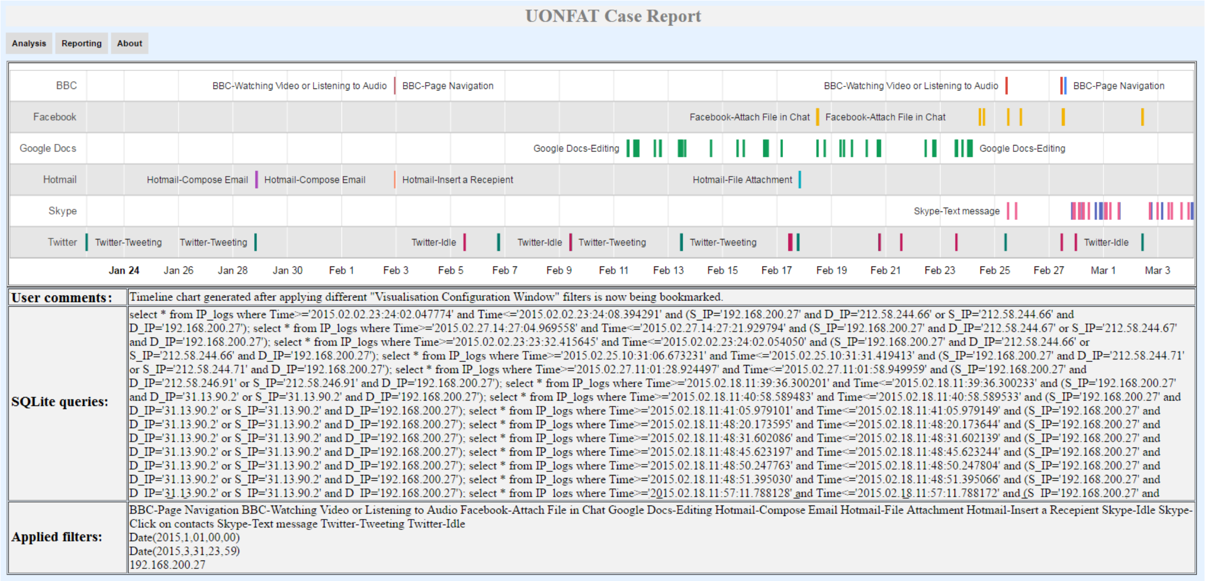


Figure 6.19: UO-NFAT Dashboard → Menu Option (View Report)

As discussed in the architecture in Chapter 5, it is the reporting engine that controls the process of generating the report. Figure 6.20 illustrates what the generated report would look like. It can be noticed from this figure that the case report window has four sections through which the relevant information is presented. The top section of the window displays the timeline of user activities. Following the timeline, it is the ‘User comments’ section that shows all comments that were added by the investigator while bookmarking the timeline chart earlier in the investigation. Next in line is the ‘SQLite queries’ section that lists all SQLite queries that would allow the investigator to extract the raw traffic that was responsible for each and every user activity that is presented in the corresponding timeline chart. These SQLite queries could become useful if the investigator is away from the tool but still wishes to explore the raw traffic by other means such as a SQLite browser. It could also be useful to an individual, perhaps a higher authority other than the investigator to whom the case report is being submitted, with an interest in the raw network traffic. The fourth and final section of the case report window is the ‘Applied filters’ that displays all filter options that were used when the investigator bookmarked the timeline chart earlier in the investigation. While the prototype generates the information required for the ‘User comments’ and ‘Applied filters’ sections based on the details provided by the investigator earlier, the ‘SQLite queries’ section is filled in by the tool itself without any direct input from the investigator. The main purpose of having sections such as user comments, SQLite queries and applied filters is to make the report easily understandable for the authority personnel.



User comments:	Timeline chart generated after applying different "Visualisation Configuration Window" filters is now being bookmarked.
SQLite queries:	select * from IP_logs where Time>='2015.02.02.23:24:02.047774' and Time<='2015.02.02.23:24:08.394291' and (S_IP='192.168.200.27' and D_IP='212.58.244.66' or S_IP='212.58.244.66' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.27.14:27:04.969558' and Time<='2015.02.27.14:27:21.929794' and (S_IP='192.168.200.27' and D_IP='212.58.244.67' or S_IP='212.58.244.67' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.02.23:23:32.415645' and Time<='2015.02.02.23:24:02.054050' and (S_IP='192.168.200.27' and D_IP='212.58.244.66' or S_IP='212.58.244.66' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.25.10:31:06.673231' and Time<='2015.02.25.10:31:31.419413' and (S_IP='192.168.200.27' and D_IP='212.58.244.71' or S_IP='212.58.244.71' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.27.11:01:28.924497' and Time<='2015.02.27.11:01:58.949959' and (S_IP='192.168.200.27' and D_IP='212.58.246.91' or S_IP='212.58.246.91' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:39:36.300201' and Time<='2015.02.18.11:39:36.300233' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:40:58.589483' and Time<='2015.02.18.11:40:58.589533' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:41:05.979101' and Time<='2015.02.18.11:41:05.979149' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:20.173595' and Time<='2015.02.18.11:48:20.173644' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:31.602086' and Time<='2015.02.18.11:48:31.602139' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:45.623197' and Time<='2015.02.18.11:48:45.623244' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:50.247763' and Time<='2015.02.18.11:48:50.247804' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:48:51.395030' and Time<='2015.02.18.11:48:51.395066' and (S_IP='192.168.200.27' and D_IP='31.13.90.2' or S_IP='31.13.90.2' and D_IP='192.168.200.27'); select * from IP_logs where Time>='2015.02.18.11:57:11.788128' and Time<='2015.02.18.11:57:11.788172' and (S_IP='192.168.200.27' and

Applied filters:	BBC-Page Navigation BBC-Watching Video or Listening to Audio Facebook-Attach File in Chat Google Docs-Editing Hotmail-Compose Email Hotmail-File Attachment Hotmail-Insert a Recepient Skype-Idle Skype-Click on contacts Skype-Text message Twitter-Tweeting Twitter-Idle
	Date(2015,1,01,00,00)
	Date(2015,3,31,23,59)
	192.168.200.27

Figure 6.20: UO-NFAT Dashboard → Menu Option (View Report) → UO-NFAT Case Report → Charts, User Comments, SQLite Queries, Applied Filters

The case report generated by the UO-NFAT prototype could be much larger in size and may contain multiple windows similar to what is shown in Figure 6.20 depending on how many timeline charts the investigator had bookmarked. For example, if the investigator bookmarked ‘n’ different timelines then the case report will have ‘n’ windows and each of those windows will have its own timeline chart, user comments, SQLite queries and applied filters sections.

The type information that the case report is comprised of are thought to be both important and helpful from an investigation point of view. The investigator could generate and submit such a report at the end of an investigation to the authority. The authorised individual would then be able to not only understand the user activities in a timely manner but also see what the investigator’s thoughts were from the comments section and what date/time, IP address and activity filters the investigator used to accumulate those user activities. Most part of the information presented in the report can be understood by the authorised individual that does not come from a technical background of understanding the network traffic. However, the report also supplies information, through the SQLite queries, that might be appealing to a technically competent person. In other words, the case report acts as a medium for conveying valuable information about the case that the investigator was able to gather during the investigation to the authority.

It is through the case report; the UO-NFAT prototype fulfils incorporating the functionality of reporting. For one thing, it allows the tool to imitate the reporting feature exhibited by the established computer forensic analysis tools. Also, it helps the investigator greatly in handing over the information that was discovered in a more meaningful and easy to understand manner thereby reducing some of the burden.

If the investigator opts for 'Open Archive' under the 'Reporting' menu tab as shown in Figure 6.21, the prototype will direct the investigator to the case archive.

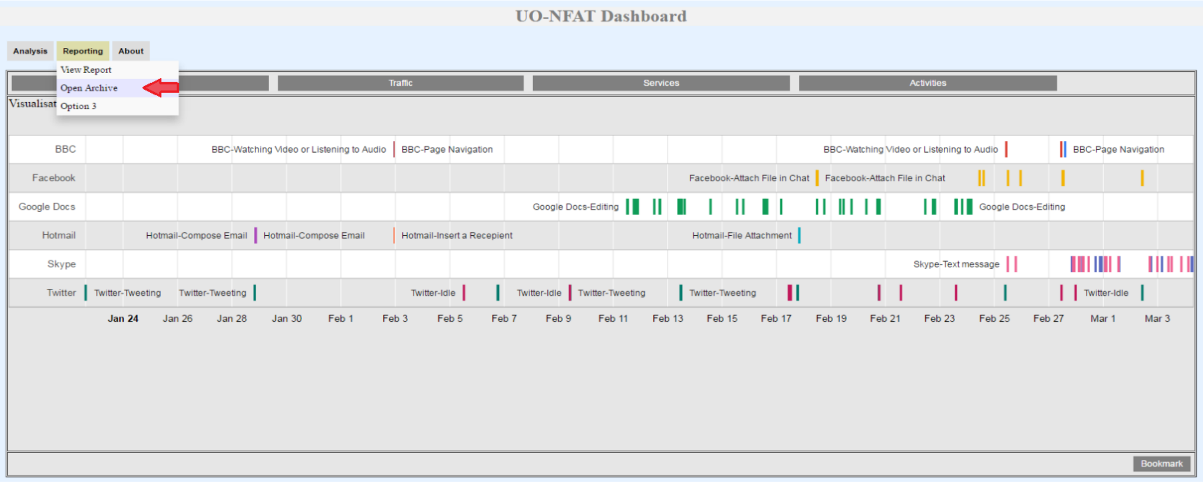


Figure 6.21: UO-NFAT Dashboard → Menu Option (Open Archive)

The main purpose of including the 'Open Archive' menu item is to provide the investigator with an option to view the existing report of any previously investigated case that the investigator thinks might be helpful for the on-going investigation. The previous case reports that are available to the investigator will be based upon the authorisation rights of that particular investigator within the investigation team, thereby maintaining confidentiality. This menu item will allow the investigator to view other reports without affecting the current investigation as the prototype will open the 'Case Archive' in a new browser tab enabling the investigator to switch between the dashboard tab and the case archive tab. A separate screenshot of the 'Case Archive' window is not included as it is already illustrated in Figure 6.18.

6.4 Discussion

There are multiple ways to implement the proposed UO-NFAT system and the prototype developed as a part of this research is only one of them. Nevertheless, certain challenges and limitations were identified during the prototype development phase, which are discussed in this section. The main challenge was the time taken for the development process. The researcher had to go through a steep learning curve in order to attain the necessary programming skills. Starting from the basics of programming languages such as HTML, CSS, and JavaScript for developing the front-end, PHP and Matlab for enabling the back-end, and SQLite for maintaining the databases, the process was time consuming. The developed prototype is able to perform the forensic analysis while showcasing functionalities such as case management, visualisation and reporting, and allows the investigator to view the corresponding raw meta-data. However, due to lack of time, incorporating features such as AAA and Visualisation Editor was not possible.

Not being able to consider and follow the Human Computer Interaction standards was another limitation of the prototype implementation stage. HCI is a branch that focuses on understanding how humans interact with computational devices and overlaps with a number of disciplines such as computer science, psychology, ergonomics, design and sociology. Personal computers, self-check-out tills at the supermarket, smart phones, tablets, GPS navigation devices used in vehicles and microwave ovens are a few examples of the computational devices that are widely used. The communication between the user and the device will be facilitated via the user interface that lies on top of the underlying software application(s) and the operating system. The domain intertwines researchers that are focused on the aspects of human mind that are involved while interacting with the computational devices and developers that are focused on the practical aspects of the interaction such as user interface, usability of the products and user experience (Dix, 2009; Interaction Design Foundation, 2018). The knowledge acquired by these two groups (i.e. HCI researchers and developers) can be picked up in the form of the International Organisation for Standardisation (ISO) standards and can be applied while developing a quality product (UO-NFAT, in this case) that will have an HCI element to it.

For instance, in order to maintain the usability aspects of the tool, it would be ideal to follow the ISO 9241-11:2018 (Ergonomics of human-system interaction – Usability) standard, which will provide guidelines for the usability aspects of the tool. Another International standard that may be beneficial for improving the quality aspects of the tool would be the ISO/IEC 25010:2011 (Systems and software engineering – Systems and software Quality Requirements and Evaluation (SQuaRE) – Systems and software quality models) standard. According to the definition, the standard is comprised of a ‘quality in use’ model and a ‘product quality’ model. While the former relates to the degree to which specific users (the investigators, in the context of this research) can use a software product or a computer system in order to meet their needs to achieve specific goals, the latter relates to the quality of the static properties of the software product and the dynamic properties of the computer systems (ISO, 2018). The ‘quality in use’ model aims to deliver effectiveness, efficiency, satisfaction, freedom from risk and context coverage. The ‘product quality’ model lays out the guidelines for achieving properties such as functional suitability, performance efficiency, compatibility, usability, reliability, security, maintainability and portability. Also, since the tool has a web based user interface, considering the World Wide Web Consortium (W3C) Accessibility, Usability, and Inclusion guidelines will be useful (W3C, 2019). The researcher believes that following the necessary ISO standards and W3C guidelines is crucial for elevating the overall usability and quality of UO-NFAT, and the future work will certainly attempt to achieve this goal.

The UO-NFAT prototype was designed/implemented to analyse the raw network meta-data in a reactive manner. Even though the prototype displayed promising results while analysing the previously collected dataset, the capability of the tool to undertake a task of analysing live network traffic is yet to be discovered. Finding out how well the tool can keep up with the enormous volume of live traffic may require carrying out further experiments and modifying the tool appropriately to improve the efficiency, which will be fulfilled as a part of the future research.

6.5 Conclusion

Implementing a functional prototype that is based on the novel UO-NFAT architecture was the primary goal of this stage of research. That task was completed by developing a web based UO-NFAT prototype that was capable of capturing the essence of the tool that was conceptualised in the architecture chapter. The chapter has successfully presented the developed prototype through screenshots giving the reader a complete picture of the journey that the investigator will be taking if the UO-NFAT was to be utilised in the process of an investigation. Finally, the ‘use cases’ section demonstrated how the UO-NFAT can assist in the investigation to reduce the burden on the investigator. The tool’s capability to extract online user interactions by using the novel interaction based approach to analyse raw network metadata has equipped UO-NFAT to deal with the enormous volume and the encrypted nature of network traffic. Also, the capability to assist in the investigation process by offering functionalities such as case management, visualisation and reporting have contributed towards reducing the investigation time.

7 The Evaluation

7.1 Introduction

The core area of the research focused on developing an 'interaction based network forensic analysis tool' – a novel approach that is capable of identifying the online user activities from the low-level network metadata – followed by designing the UO-NFAT system architecture that was built around the aforementioned approach and by integrating functionalities such as case management, visualisation and reporting. The research project went on to develop a functional prototype of the proposed system to prove the possibility of a real world formation of the tool that was hypothesised.

The novelty of the research and different facets of the developed system needed to be evaluated by an external group of experts. With this intention, the evaluation stage of the research was undertaken, which involved the assessment of the research done by the academic experts within the field of digital forensics. Section 7.2 of the chapter discusses the methodology that was adopted for the evaluation, followed by an introduction of the academic experts that participated in the evaluation in Section 7.3. The evaluation results that were obtained from the experts during the interview sessions are presented in Section 7.4 of the Chapter. Section 7.5 of the Chapter discusses three different use cases to test the prototype and to show how the prototype would be able to assist in the investigation, indicating how the tool would perform in real investigative scenarios. A detailed discussion of the experts' feedback and the conclusion are presented in Section 7.6 and Section 7.7 respectively.

7.2 Methodology

In order to conduct the evaluation stage of the research, it was necessary to establish a suitable methodology. During the development of evaluation methodology, certain factors that are capable of altering the structure of the evaluation process were taken into account. For instance, deciding whether engaging in an interview with each of the expert evaluators or gathering responses via a questionnaire is more suitable for obtaining the feedback was an important factor. Conducting an interview would allow the researcher to pose each and every question without losing its essence and resolve any confusion or query that may rise in the minds of the evaluators. Additionally, choosing interview over questionnaire would provide the researcher with an opportunity to book each evaluator for a session that is long enough to combine the tasks of presenting the research via the video podcast and obtaining the feedback; both tasks being completed within the time-limits of the same session. This approach will allow tapping full attention of the evaluators, which might not have been feasible if opted for a questionnaire based approach. Hence, the interview based approach was chosen to be incorporated into the evaluation methodology. Subsequently, choosing whether to structure the interview questions in an open-ended or a closed-ended format was another factor that required attention in developing the evaluation methodology. Open-ended questions will allow the evaluators to open up their minds and express their thoughts in a convenient manner thereby allowing them to assess the quality of the research effectively. Closed-ended questions, on the other hand, would have limited the evaluators' freedom to respond. Hence, seemingly more appropriate and logical, an open-ended-questions format was adopted in order to create the interview questions for the expert evaluators. Another factor that had its impact on the evaluation process was the decision to target academic experts as the designated group of evaluators; not practitioners. Although the project has produced a functional prototype, the core contribution of the research comes from the interaction-based approach for analysing the network metadata and from the accompanying UO-NFAT architecture. Because of the research nature of the project, the ideal group of individuals that can evaluate the venture would be a batch of experts with proficiency in the respective research domain (i.e. the academic experts). Practitioners, in comparison with the academic experts, are less likely to have a deeper grasp of the research aspects of the project, which was the reason behind concentrating on academic experts for conducting the evaluation. Addressing and assessing the aforementioned factors allowed the researcher to design a methodology that can fulfil the purpose of the evaluation stage in a logical, qualitative and an efficient manner. A diagrammatic representation of different phases of the evaluation process is illustrated in Figure 7.1.

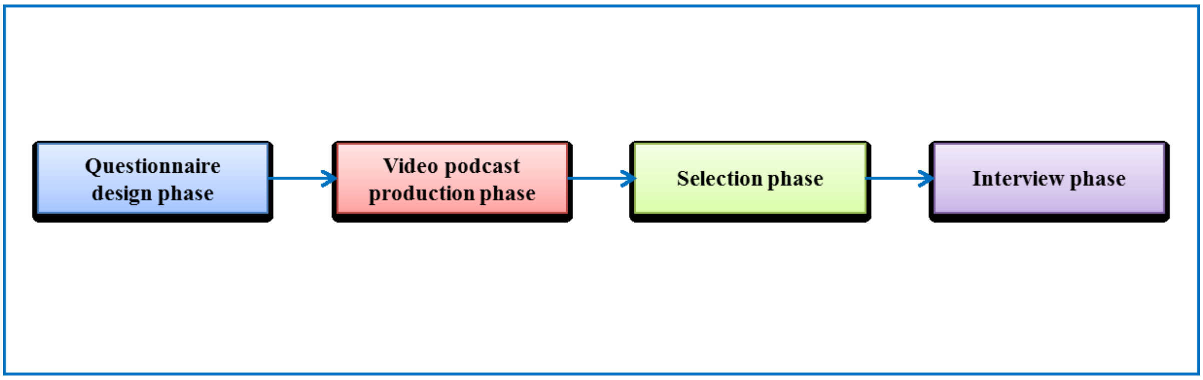


Figure 7.1: Phases of Evaluation

The researcher identified that the evaluation process can be divided into four different phases – the questionnaire design phase, the video podcast preparation phase, the selection phase and the interview phase. The questionnaire design and the video podcast production phases were set to be executed in a parallel fashion producing the necessary outputs (i.e. the interview questions and the video podcast). The goal was to produce a set of open-ended interview questions and a 15 to 20 minutes long video podcast that will portray the entire work done by the researcher. These outputs are sufficient to carry on with the evaluation process and will serve as the resources required later in the interview phase of the process. The reason behind limiting the video podcast to 20 minutes was to provide the evaluators with enough time to answer the questions and discuss their thoughts during the interview sessions. The third phase (i.e. the selection phase) was designed to select the expert evaluators based on their expertise in the domain, who will then assess the research in question. The first two phases when combined with the third phase will lead up to the fourth and final phase (i.e. the interview phase) of the evaluation process providing the ultimate feedback. It was important to ensure that the time allotted for each interview session falls into a sweet spot so that it can keep the evaluators' interest and attention alive. In order to achieve this goal, the interview phase was designed to include 45 minutes to 1 hour long sessions. As mentioned above, the interview questions and the video podcast that emerged from the first two phases will fulfil their respective roles in the interview phase. Identifying and understanding each of the phases made the whole process easy and served as a guideline to follow through. Each phase of the evaluation process is discussed in detail and justified in the respective subsection of the methodology section.

7.2.1 The Questionnaire Design Phase

The preliminary phase of the evaluation process involved designing and producing an interview questionnaire containing carefully crafted open ended questions for the expert evaluators. It was necessary to set the right questions and to frame them in a sensible manner so that the expert academics can deliver a proper assessment of different aspects of the research that was done. As mentioned in the evaluation methodology, the researcher had chosen an open-ended format for designing the interview questions allowing the evaluators to assess the research and the prototype in a non-restrictive manner. Starting with the first set of questions, the researcher went through three rounds of alterations in order to produce a sophisticated questionnaire. The finalised questionnaire for the expert evaluation contained fourteen different questions, which are listed in Table 7.1.

Table 7.1: Interview Questions for Expert Evaluators

Num.	Questions
1	The creation of user interactions derived from an analysis of network metadata forms a core contribution towards the new NFAT – what are your thoughts on this approach?
2	What do you think about the system utilising low level meta-data enabling it to find its way around the issues of encryption and the processing of large volumes of payload data?
3	What are your thoughts about the tool's capability to extract user activities and its effectiveness in reducing the investigator's cognitive load?
4	The tool is premised on the case management concept – providing full analysis and reporting in a single tool. What are your thoughts on this?
5	What are your comments on the 'reporting' feature? Considering the information it provides (i.e. timeline plots, user comments, filter options selected and database queries), how useful do you think it is in presenting the information of possible evidential value?
6	What are your thoughts on the effectiveness of the following functionalities offered by the dashboard? a) Chronological display of activities b) Filter options (i.e. User activities, Date & Time and IP addresses) c) Highlighted raw traffic display
7	Speaking from an insider threat/misuse investigation perspective, how beneficial/effective do you think the timeline-based display of user activities is?
8	Do you think any other filter options would be useful on top of the ones (user activities, date & time and IP addresses) that are already in place? If so, which one(s) and why?
9	What do you think of the dashboard feature that compares user IPs against the number of packets belonging to raw traffic, services and activities?
10	How useful do you think the tool would be in dealing with each of the following activities of insider crimes in organisations? Please justify your answer: a) Unauthorised access to system or organisational data b) Misuse, loss or unauthorised disclosure of confidential information c) Infringement of laws or regulations
11	What are your thoughts about extending the architecture for live traffic analysis?
12	What do you think are the strengths and weaknesses of the demonstrated tool?
13	Do you suggest any other feature(s) that the dashboard could incorporate?
14	Can you think of any other organisational issue (apart from the insider threat/misuse) that this tool could assist in resolving?

The main criterion for designing the questions was that each question should focus on a different facet of the research. The first two questions were aimed directly at evaluating the approach that was used by the researcher in order to tackle the research problem. Questions were included about the capability and effectiveness of the developed system in accomplishing specific features, ability of the prototype to meet the system requirements, usefulness of the system in assisting in insider threat investigation, strengths and weaknesses of the developed prototype and the possibility of future extensions of the architecture. Including questions about the developed prototype aided the researcher to provide the experts with a clear medium to evaluate the UO-NFAT system, which is being represented by the prototype.

During the interview sessions, the participating experts would be able to answer the questions based on their knowledge and experience in the respective domains. The information can be extracted from their feedback in an appropriate manner at a later point (i.e. the interview phase) in the evaluation process.

Before proceeding to the next phase of evaluation, request for ethical approval was put in place, which was approved by the ethical approval committee. The accepted form is included within the index section.

7.2.2 The Video Podcast Production Phase

Finding a suitable approach for presenting the novel research content and the implemented prototype to the expert evaluators was an important step in the evaluation process. Factors such as the geographical location and the limited availability of the expert evaluators were taken into account before selecting the right approach. It was only logical to assume that the expert evaluators are going to be situated in different geographical regions. This nullified the feasibility of an approach of presenting the research, including the prototype, to the experts in person. Another approach was to present the work and demonstrate the prototype to the evaluators via a live stream. But, this option would have been difficult to undertake given the limited amount of time (i.e. 15 to 20 minutes) available for presenting the entire work during a 45 minutes to 1 hour long interview session. Hence, an alternative approach of presenting the research and demonstrating the prototype through a video podcast that was prepared prior to the interview was selected as the appropriate one. The researcher decided to follow this approach as it seemed both practical and convenient in a time-sensitive situation. Thus the second phase of the evaluation process was set to create a video podcast that will portray the work achieved. The podcast content will provide the experts with a clear understanding of the research in question so that different aspects of the developed system can be evaluated.

The visual content for the video was prepared using slides in Microsoft PowerPoint. The content included an insight into the research problem, solution proposed by the researcher, the UO-NFAT system architecture, different features and functionalities of the developed prototype followed by the distinguishable features of the system and the future work of the research. The audio content (i.e. the narrations) were recorded separately, which were embedded into the respective slides. The PowerPoint file was then converted into a high-definition resolution video and was uploaded to Vimeo (a popular online video sharing platform). To ensure the safety of the unpublicised research information contained in the video, the status of the uploaded video was set to 'private' and the video was password protected. The URL and password to watch the video were given to the experts only prior to the interview. The URL to watch the video podcast is https://vimeo.com/213694792 and the viewer can gain access by using the password – aly195~?!WBO aly195~?!.

The main challenge in this phase was presenting the entire work in a 15 to 20 minutes long video without losing the entirety and quality of the content. A longer video could easily result in the potential participants losing their interest and choosing not to continue with the process. The final product of this phase was a 22 minutes long video podcast, which the expert evaluators will be requested to watch during the interview sessions.

As mentioned in the evaluation methodology, it was important to make sure that preparation of the interview questions and the video podcast went hand in hand so that the work presented in the video and the questions asked made sense when combined. As it was only through the video podcast that the experts gained an understanding of the work, any mismatch between the podcast content and the interview questions would have created ambiguity and confusion about both the research and the researcher. Completing the first two phases in a synchronised fashion benefited the researcher in creating questions that resonated with the content of the podcast. This step has not only helped the evaluators to relate the interview questions to the research and answer them accordingly but has also contributed towards providing a better experience for both parties involved in the interview.

7.2.3 The Selection Phase

Next up in the evaluation process was the selection phase which involved identifying the ideal group of people that are eligible to participate in the evaluation followed by selecting a sample group containing the potential participants. Academics that had exposure to the digital forensics domain were identified as the ideal population from which a group of academics, including professors and doctors, was selected. The sample group contained twenty nine members and the aim was to have a minimum of twelve participants.

Each and every person in the selected sample group was contacted individually via e-mail containing a brief introduction of the researcher, the purpose of them being contacted and the invitation to participate in the evaluation process. All members were provided with the information sheet and consent form. Members that responded positively to the invitation were contacted further in order to make an arrangement for a convenient date and time to conduct the interviews.

Acquiring a positive response from the contacted individuals was the main challenge in this phase. Based on the response to the e-mail invitation, the individuals can be divided in to four categories – members who never gave a response, members who gave a negative response, members who gave a positive response yet never carried on and members who gave a positive response and continued with the process.

7.2.4 The Interview Phase

The interview phase consisted of the actual interview sessions with the participants who accepted the invitation and agreed to proceed. The participating evaluators were from different geographical areas. Therefore, the interview sessions were conducted via Skype as face to face interviews were not a practical option.

As mentioned in the evaluation methodology, the researcher chose to follow an approach of booking a single session in order to fulfil the task of the evaluators watching the podcast followed by the researcher gathering their feedback. An alternative approach of providing the evaluators with the prepared video podcast a few days in advance of the interview would have been suitable. This will give the experts ample time to evaluate the work done by the researcher and take part in the interview at a later point in time in order to discuss their thoughts and give feedback. However, the chances are high that the mental faculties of the participating experts are going to be simultaneously occupied by various other tasks. As a result, retrieving every single detail of the research in question that was introduced a few days in advance in its entirety can be difficult. Therefore, the researcher opted for the former approach. The interview sessions were 45 minutes to an hour long Skype sessions with each session consisting of two parts. During the initial part of the interview session, each participant was requested to watch the prepared video podcast. Prior to the interview session, the link and password to the podcast were given to the participant. After finishing watching the podcast, the participant's feedback about the interview questions was collected. While some of the participants preferred a conventional interview style to give the feedback, some other participants went for a discussion based approach.

With the participants' permission, all interview sessions were recorded so that no information was lost due to any human error and the data can be carefully analysed afterwards in order to summarise the feedback provided. The recordings of the interview sessions were stored in an encrypted external hard drive and the files were destroyed once the feedbacks were analysed in order to ensure the data privacy of the participants.

7.3 Expert Evaluators

The stakeholder community, eligible to participate in the evaluation process, was identified as academics from around the world that have proven their expertise and keen interest within the field of digital forensics. As mentioned in the selection phase, a sample group of evaluators containing 29 experts was selected and each member was contacted with a personal invitation to participate in the evaluation process. These experts were from parts of the United Kingdom, Europe, South Africa, Australia and the United States of America. Along with the proficiency of the experts in the field, factors such as their willingness to participate and their availability played important roles in the selection process. As stated in section 7.2.3, the aim was to gather expert feedback from a minimum of twelve participants. However, only 10 invitations were acknowledged with 8 of the invitees responding positively and 2 invitations being turned down. Even though 8 experts agreed to take part, eventually, only 5 members of the group completed the process while the remaining 3 members, despite being followed up, decided to exit the process with no notification. Considering the fact that a specific part of the process (i.e. from sending out the invitations till the interviews) has taken 40 days to complete, the researcher decided to pause the search for more participants, examine the obtained set of feedbacks and continue only if the results are lacking in quality and richness of the content and are unsatisfactory. Review of the obtained feedback showed that the comments and suggestions provided by the experts who completed the process have made the evaluation sufficiently strong. Considering the three factors mentioned above – the low success rate in receiving a positive acknowledgement from the invitees, the huge amount of time it took to complete the process thus far and the promising feedback received from the experts – the researcher decided not to continue with the search for additional participants.

This section talks about the five experts that participated in the evaluation and completed the process by assessing the work done by the researcher. Each subsection describes the experts individually through their specific academic background and the justification for choosing them as participants. The subsections are titled in an order in which the expert evaluators were interviewed.

7.3.1 Prof Marcus K Rogers – Purdue Polytechnic, United States of America

Prof Marcus K Rogers is a Professor and the Department Head of the Purdue Polytechnic, located in Indiana, USA. He was Skype interviewed on 18 May 2017 and his contact email is rogersmk@purdue.edu.

Alongside being the Professor and Department Head, Marcus Rogers is also a Fellow of the American Academy of Forensic Sciences (AAFS), University Faculty Scholar and Fellow of Center for Education and Research in Information Assurance and Security (CERIAS). Prof Rogers is the Chair of the Digital and Multimedia Sciences section of the American Academy of Forensic Sciences (AAFS), past International Chair of the Law, Compliance and Investigation Domain of the Common Body of Knowledge (CBK) committee, Co-Editor in the Cybercrime Department of IEEE Security & Privacy, NIST-OSAC DE Member and Chair of the Education Task Group for DE. He is a former Police officer who worked in the area of fraud and computer crime investigations. Prof Rogers sits on the editorial board for several professional journals. He is also a member of other various national and international committees focusing on digital forensic science and digital evidence. Prof Rogers is the author of books, book chapters, and journal publications in the field of forensic psychology, digital forensics and applied psychological analysis. His research interests include applied cyber forensics, psychological digital crime scene analysis, and cyber terrorism. He is a frequent speaker at international and national information assurance and security conferences, and guest lectures at various universities throughout the world.

Although Prof Rogers was able to watch the prepared video podcast and provide his overall feedback about the novelty of the research and the prototype during the Skype session, he was not able to answer the interview questions due to time limitations. However, Prof Rogers agreed to return the completed questionnaire by the following weekend which was received as promised.

7.3.2 Dr Christos Kalloniatis – University of Aegean, Greece

Dr Christos Kalloniatis is an Associate Professor in the Department of Cultural Technology and Communication of the University of Aegean in Greece and was interviewed via Skype on 22 May 2017. His contact email is chkallon@aegean.gr.

Dr Christos Kalloniatis holds a PhD from the Department of Cultural Technology and Communication of the University of Aegean and a Masters degree (in Computer Science) from the University of Essex in UK. He is also a deputy member of the board of the Hellenic Authority for Communication Privacy and Security. Dr Christos' main research interests are the elicitation, analysis and modelling of security and privacy requirements in traditional and cloud-based systems, privacy enhancing technologies and the design of information system security and privacy in cultural informatics. He is an author of several refereed papers in international scientific journals and conferences and has served as a visiting professor in many European institutions. Prior to his academic career he has served at various places on the Greek public sector including the North Aegean Region and Ministry of Interior, Decentralisation and e-Governance. He is a lead-member of the cultural informatics research group as well as the privacy requirements research group in the Department of Cultural Technology and Communication of the University of the Aegean and has a close collaboration with the Laboratory of Information & Communication Systems Security of the University of the Aegean. He has served as a member of various development and research projects.

The expert's profound knowledge about privacy in multiple environments provided the researcher with an insight of some privacy aspects of the prototype.

7.3.3 Dr Olga Angelopoulou – University of Hertfordshire, United Kingdom

Dr Olga Angelopoulou is a Senior Lecturer in School of Computer Science of the University of Hertfordshire in Hatfield, UK. She was Skype interviewed on 23 May 2017 and her contact email is o.angelopoulou@herts.ac.uk.

Alongside being a member of the Centre for Computer Science and Informatics Research (CCSIR) of the University of Hertfordshire, Dr Olga is also a member of the Cyber Security Centre that provides training, teaching and research in cyber security and digital forensics domains. Her main research interests are in the areas of digital forensics, identity theft and online fraud and information security. In the recent years, Dr Olga has published multiple papers in conferences as well as journals including the European Conference on Cyber Warfare and Security (ECCWS), the Information Security Journal and the Journal of Information Warfare.

7.3.4 Prof Hein S Venter – University of Pretoria, South Africa

Professor Hein S Ventor is a Professor in Department of Computer Science and the Head of the Information and Computer Security Architectures (ICSA) research group at University of Pretoria in South Africa. A Skype interview session was conducted with him on 12 June 2017 and his contact email is hventer@cs.up.ac.za.

Prof Venter's areas of research interest include digital forensics, information privacy in medical and banking environments, wireless security systems with special interest in efficiently securing wireless sensor networks, network security and vulnerability scanning and intrusion detection in distributed systems. Prof Venter's interest in network security domain includes in network security health checking, web services security and intelligent network reconfiguration. One of his interesting works involves gaining ISO level standardisation for the digital forensic investigation process model.

Prof Venter has produced publications for nearly two decades and has published papers in the field of digital forensics since 2004. This shows the wealth of knowledge that Prof Venter possess, which was extremely helpful in acquiring a fruitful evaluation of the research.

After watching the video podcast Prof Hein showed interest in understanding more about the system architecture. Hence, the remaining part of the Skype session was spent discussing the architecture. Due the limited availability of time, Prof Hein suggested to provide his feedback to the interview questions in writing and to return the document the following day. The completed document was received on 14 June 2017.

7.3.5 Dr Stilianos Vidalis – University of Hertfordshire, United Kingdom

Dr Stilianos Vidalis is a Senior Lecturer of Cyber Security in School of Computer Science of the University of Hertfordshire in Hartfield, UK. He was interviewed via Skype on 15 June 2017 and his contact email is s.vidalis@herts.ac.uk.

Dr Stilianos is a member of the Centre for Computer Science and Informatics Research (CCSIR) and the Cyber Security Centre that focuses in the areas of cyber security and digital forensics at the University of Hertfordshire. His research interests fall in the areas of digital forensics, cyber-criminal profiling, threat assessment, cyber operations and effective computer defence mechanisms. Dr Stilianos has more than a decade worth of experience and during these years he has published multiple papers in conferences as well as journals such as Information Security Journal, Journal of Information Warfare and Journal of IT Governance Practice. His knowledge and understanding of the domain along with his experience helped greatly in the evaluation process.

7.4 Evaluation Results

This section presents the answers provided by all experts that participated in the interview and is divided in to fourteen subsections. Each subsection is titled based on the questions raised in the evaluation questionnaire along with the corresponding responses given by the participants. Grouping the responses based on questions instead of the participants seemed a more sensible approach as it would provide the reader with a better understanding of the entire response scenario.

7.4.1 Thoughts on the Novelty of the Research Contribution

The designed UO-NFAT system is aimed to reduce the cognitive load on the investigator by generating user interactions from low-level network metadata, which was implemented through a novel approach (i.e. the interaction based approach). The first question was formed with the intention of grasping the expert evaluators' thoughts on the innovativeness of the research contribution.

Prof Rogers started off by saying that the research idea presented was excellent. He thought that the interaction based approach for analysing the network metadata introduced by the researcher was innovative and that the research direction was really interesting. Dr Christos thought that there is good value and novelty to the approach and that the developed system clearly has more practical aspects to it for assisting the investigator in conducting the forensic investigation. Dr Olga thought that the researcher's approach of utilising network meta-data for creating user interactions is innovative. Prof Venter also thought that the research idea is quite novel. Dr Vidalis commented that understanding the user activities is crucial speaking from an investigation perspective and was pleased that the tool (UO-NFAT) has managed to accomplish the same. He specifically wanted to give credit to the novel approach implemented by the researcher. Based upon the responses received for question 1, it is safe to say that all expert evaluators were fully satisfied with the novel concept of a system that reduces the cognitive load on the investigator by identifying the online user interactions and is built around a case management premise exhibiting features such as hashing, visualisation and reporting, put forth by the researcher.

7.4.2 System's Capability to deal with Encryption and Enormous Data Volume

The system utilises low level network meta-data in order to find a way around the issues of encryption and processing of large volumes of payload data. The capability of the developed system to achieve this was addressed in the second question.

Prof Rogers thought that the aforementioned approach was good. However, he suggested being careful with the "getting around the encryption problem" in a sense that the tool still does not read the encrypted data. Having seen the prototype demonstration, Dr Christos thought that the idea of using low level meta-data to get around the issues of encryption and data volume itself is interesting and was pleased to see the actual implementation of this idea through the screenshots of the UO-NFAT dashboard. Both Dr Olga and Prof Venter did address the issues of increasing volume of network traffic and the encrypted nature of data that requires analysis that are faced by existing forensic analysis tool and how it affects the time taken for the investigation. Both evaluators appreciated the developed system's capability to work its way around these issues by utilising low-level metadata. Dr Vidalis said that the system's capability to deal with the issues of encryption and large data volume is interesting. The evaluator also added that the system utilising low level meta-data reminded him of the tools used in the intelligence sector and pointed it out as a positive comment.

7.4.3 System's Capability to Reduce the Investigator's Cognitive Load

One of the goals the system trying to achieve is to reduce the cognitive load placed on the investigator by extracting the online user activities within the organisational environment and by providing a dashboard that allows the investigator to interact with the tool and to examine the extracted user activities while using functionalities such as case management, visualisation and reporting offered by the tool. The purpose of the question was to find out how effective, according to the expert evaluators, the developed prototype is in achieving this goal.

To this question, Prof Rogers responded by saying that the cognitive load reduction implemented by the tool is good. However, the evaluator pointed out that the user interface of the tool needs to be tweaked to the 'Human Computer Interaction (HCI) standards and suggestions', which could help in a further reduction of the investigator's cognitive load. Dr Christos totally agreed with the capability of the tool to reduce the investigator's cognitive load. While discussing this question, the evaluator wanted to draw the researcher's attention towards a privacy related issue as the evaluator is an academic whose main research goal is developing effective privacy-aware systems for various environments including digital forensics. Further explanation of the suggestion put forth by the evaluator is provided in the discussion section of this chapter. Dr Olga thought that whether or not the tool can reduce the burden on the investigator would depend on the actual case that is being investigated. The evaluator preferred to view the tool as a point of reference as other tools are going to be involved in the investigation. Dr Olga said that the tool can provide some quick results that could help the investigator. Prof Venter was fully convinced about the capability of the tool to extract the user activities. The evaluator said that the approach is quite novel and effective in benefiting the investigator by reducing the cognitive load. Prof Venter also suggested that it would be better to prove that the tool indeed did correctly find all the activities; perhaps by using a tool like Wireshark to manually display the activities on a small dataset and then show that the developed tool gives the same results.

Dr Vidalis pointed out that, in his opinion, the system should be able to reduce the burden on the investigator as much as possible by providing useful information. The evaluator was pleased with the UO-NFAT system's capability to achieve this goal.

7.4.4 Thoughts on the Case Management Concept put forth by the Tool

The developed prototype incorporates the idea of case management, a concept that is well executed by established computer forensic tools such as FTK and EnCase. Thus, the tool is capable of providing a full analysis, reporting along with the case management functionality. The purpose of the question was to obtain the experts' thoughts about the system adding in the aforementioned concept.

Prof Rogers thought that this is a good idea. However he did prefer the tool's user interface to look more similar to the standard digital forensic tools. Dr Christos totally agreed with the case management concept based on which the tool is designed. Although there are improvements that can be made on the graphical interface part of the tool, he said he would not consider them as something that reduces the research contribution but would rather consider it as a part of future development. Dr Olga also liked the case management concept as the evaluator thought it would help with the chain of custody. Prof Venter responded to this question by saying that case management concept it handy. The evaluator said that having one tool to do the case management providing a full analysis and reporting makes it so much easier for the investigator. Dr Vidalis said that the developed system incorporating the concepts of case management and reporting along with a full analysis of the network meta-data is an indication of a good digital forensic analysis tool.

7.4.5 Comments on the Reporting Feature and its Usefulness

Question 5 was intended to get the experts' thoughts on the usefulness of the reporting feature of the prototype when it comes to presenting meaningful information of possible evidential value.

Prof Rogers thought that although the reporting part of the user interface makes sense to an investigator who is a network specialist, it is too busy and non-intuitive for an investigator that comes from a law enforcement background. Dr Christos responded by saying that the reporting feature is necessary in a forensic analysis tool. The evaluator added that the reporting feature presented in the UO-NFAT prototype was much liked. Dr Olga was pleased with the reporting functionality of the tool and the way the information is presented. The evaluator thought that although the report generated by the tool is not a completed case report speaking from an investigation perspective, it surely can provide useful information and is something that can become a part of the final case report. Prof Venter said that the reporting feature implemented in the UO-NFAT prototype is a well thought one and that it blends right in with the rest of the dashboard features. Dr Vidalis was pleased with the information provided by the reporting feature and for following the guidelines for a good digital forensic tool. The evaluator thought that the timeline plots, user comments, filter options selected and the database queries displayed in the 'case report' would be beneficial as a supplementary evidence.

7.4.6 Effectiveness of the Dashboard Functionalities

The purpose of question 6 was to understand what the expert evaluators thought about the effectiveness of different functionalities offered by the dashboard such as the chronological display of activities, the filter options (i.e. user activities, date & time and IP addresses) and the highlighted raw traffic display.

To this question, Prof Rogers responded by saying that the user interface of the dashboard looked too busy and suggested that, as a future work, the researcher needs to work with someone that specialises in 'Human Computer Interaction (HCI) standards' in order to make some tweaks to the dashboard interface. Dr Christos thought that the functionalities provided by the UO-NFAT dashboard are important and completely agreed with the necessity of having them. According to Dr Olga, all functionalities presented in the UO-NFAT dashboard were interesting and well thought out. The evaluator believes that the chronology of user activities is a crucial part of the investigation and pointed out that the timeline based visualisation of the activities presented in the dashboard is extremely useful. Prof Venter responded to this question by saying that the timelines, the ability to filter the timelines and the ability to view the raw data are always important in network forensic tools. Dr Venter continued by saying that the UO-NFAT dashboard has utilised those features effectively. Dr Vidalis thought that all functionalities offered by the dashboard were essential for an effective digital forensic tool and that the UO-NFAT dashboard is successful in integrating the functionalities.

7.4.7 Effectiveness of the TimeLine Based Display of User Activities

Question 7 is meant to gather the experts' thoughts about the effectiveness of the timeline based display of the user activities is, speaking from an insider threat and misuse investigation perspective.

Prof Rogers agreed that the current timeline based display used in the UO-NFAT dashboard is effective to a certain extend as the timeline display can be drilled down on by the use of filter options. However, the evaluator thought that it would be more effective if the timeline display allows the investigator to zoom in to and out of particular sessions from the timeline itself. Dr Christos replied that timeline based display of user activities is important for a forensic analysis tool and the prototype has adapted it well. However, the evaluator did mention his concern about the possibility of displaying all user activities for the entire period of time on a computer screen. Dr Christos suggested that it would be better if the activities are presented in a more filtered manner at the start (say, for a smaller time interval) and then the investigator can use the filter options to view the timeline of activities for a bigger time interval. According to the evaluator, this would help to make the timeline of activities more readable. Dr Olga said that presenting the user activities in a timeline based manner benefits the investigator greatly. Prof Venter thought that the system really bodes well for identifying the insider threat. The evaluator said that because of the continuous and more difficult to ascertain nature of the insider threat, the developed model will add a lot of value in terms of identifying the insider threat. Prof Venter also added that one being able to obtain the data much more readily from the inside rather than from the outside makes this case even stronger. Dr Vidalis identified the timeline based display of activities as the most important outcome of the investigation process as it will be the timeline of activities that is going to be presented in front of the authoritative decision maker at the end of the investigation. The evaluator agreed that incorporating the visualisation of timeline into the UO-NFAT is commendable and effective.

7.4.8 Need for More Filter Options

Understanding what the evaluators thought about the need for more filter options apart from the ones that are already present in the dashboard interface (i.e. user activities, date & time and the IP addresses) was the purpose of question eight.

Both Prof Rogers and Prof Venter were the advocates of the notion of incorporating protocols and ports into the existing set of filter options. Prof Rogers suggested that adding application layer protocols such as HTTP and FTP would be a good idea. Prof Venter continued by saying that including ports and protocols often give the investigator a “bigger picture” or other trends that might be deduced not necessarily on user level only. An example scenario Prof Venter used was how the investigator can establish trends about the “typical” protocols that are utilised on the particular network. Although Dr Christos did not suggest any other filter options, the evaluator did mention that consulting with a forensic investigation practitioner might be beneficial in gathering some valuable input regarding the filter options. In contrast, Dr Olga and Dr Vidalis were both pleased with the current filter options provided in the visualisation configuration window and did not suggest a need for any more of them. Both evaluators thought that the filter options were adequate and that they will serve well in helping the investigator to filter the visualisation charts.

7.4.9 Thoughts on the Feature that Compares User IPs based on the Number of Packets

Along with the chronological display of user activities, the visualisation window of the dashboard plots charts that compare the analysed user IP addresses against each other based on the number of packets that belong to the raw traffic, services and the activities. Question 9 intends to gather the experts’ thoughts regarding this idea.

To this question, Prof Rogers responded by saying that it would be difficult to determine whether an anomaly exists with the user's IP address in question without putting the raw traffic into context. Dr Christos thought that the idea of comparing the user IP addresses against the number of packets on different levels (i.e. raw traffic, services and activities) was good as it will help the investigator to view a general comparison between the user IP addresses based on which useful information can be gathered. Similar to Dr Christos, Dr Olga and Prof Venter thought that the feature mentioned in the respective question was a useful one. Dr Vidalis thought that comparing user IP addresses against each other based on the number of packets can be useful to the investigator in understanding the usage trends among the users.

7.4.10 Usefulness of the Tool in Different Insider Crime Activities

The UO-NFAT system is designed to identify online user interactions within the organisational environment by analysing the raw network metadata. Insider threat/misuse investigation is an example of the domain in which the investigator could benefit from using the proposed UO-NFAT. In question 10, certain types of insider crimes within organisations such as unauthorised access to system or organisational data, misuse, loss or unauthorised disclosure of confidential information and infringement of organisational laws and regulations were listed. The purpose of this question was to understand how useful the experts thought the tool would be in dealing with each of the aforementioned situations.

Prof Rogers was fully convinced that the developed tool will be extremely useful in assisting in the investigation of the listed insider crime activities. According to Dr Christos, combined with the investigator's subjective input, UO-NFAT can be a useful tool in finding a correlation between an insider crime and the culprit. The evaluator pointed out that the investigator's experience and intelligence is required in order to make use of the information that the tool provides. Dr Christos also added that the tool can be utilised as more than a forensic tool to assist in insider threat/misuse investigation. Dr Olga responded by saying that the tool can be useful in providing valuable information regarding all listed activities of insider crimes. The evaluator added that perhaps the same information could be utilised for investigating external attacks as well. Similar to the previous participants, Prof Venter also thought that the tool is quite useful in dealing with different activities of insider crimes. The evaluator wanted to reiterate the suggestion that was made about adding protocols and ports as filter options so that the system could reveal network trends (other than the user activity trends) that are not necessarily visible on a user level. Incorporating this feature will contribute more towards the investigation. Dr Vidalis identified UO-NFAT as a helpful tool in the investigation of insider threat/misuse. The evaluator also pointed out that there is no need to limit the application of the tool just to insider threat/misuse investigation. According to Dr Vidalis, the tool can be utilised for auditing and compliance purposes as well.

7.4.11 Thoughts on Extending the Architecture for Live (next-to-real-time) Traffic Analysis

Question 11 was aimed at knowing what the experts thought about extending the architecture for live traffic analysis.

Prof Rogers raised his concern of whether the tool can keep up with the volume of the live traffic. The evaluator wondered what the situation would be like if the tool got overwhelmed by the live traffic volume and captured only partial information. Dr Christos fully agreed with the idea of extending the architecture for live traffic analysis and thought that it would be beneficial as well. While discussing this question, the evaluator reiterated about the importance of implementing privacy policy and the downside of not having one for both the existing prototype as well as the future versions of the tool. Dr Olga said that the architecture should be extended for live traffic analysis as doing so would enable the tool to act as a monitoring as well as an incident response tool. The evaluator said that architecture extension will make the tool offer more than one application. Prof Venter thought it would be even more beneficial if the expanded system is capable of performing a live traffic analysis. The evaluator added that even if it is a near-real-time analysis, it would greatly benefit the organisation. While answering the question, Dr Vidalis suggested considering appliances that may already exist that are capable of collecting live network traffic. The evaluator was kind enough to show such an appliance, called Ditto Shark manufactured by WiebeTech, during the interview. Once placed in the network tunnel, the appliance would be able to make a copy of the network traffic that goes through the appliance and send the copy to databases. Such databases (that could be stored in the same web server where the UO-NFAT resides) can be used as the source of data for the UO-NFAT system which then can be analysed at regular time intervals. That way the goal of analysing next-to-real-time traffic can be achieved without actually extending the UO-NFAT architecture.

7.4.12 Strengths and Weaknesses of the Demonstrated Tool

The question was aimed to gather the strengths and weaknesses of the tool that was demonstrated in the video podcast.

According to Prof Rogers, capability of the tool to identify the user activities and filter them in a quick manner according to the investigator's needs is its main strength. The evaluator thought of the user interface of the dashboard being too busy and non-intuitive to investigators that are not network specialists as the weakness. Dr Christos thought that the capability to assist in the investigation, the potential to be used for other organisational tasks such as employee productivity and monitoring and the ability to reduce the burden on the investigator are the main strengths of the tool. The current user interface of the dashboard not meeting the HCI standards was identified as a drawback of the prototype by the evaluator. The capability to identify the online user activities from raw network metadata is one of the tool's strengths, according to Dr Olga. The interaction based approach that has been used to analyse the metadata was identified as strength of the system. The evaluator did not acknowledge any weakness in particular. Prof Venter identified the novelty and the fact that the demonstrated tool might address a lot of the insider issues as the strengths of UO-NFAT. The tool not being able to do a real-time analysis yet was pointed out as a weakness. Dr Vidalis pointed out that the evaluator will have to use the tool in order to give the most accurate comment on the strengths and weaknesses of the tool. However, based on the architecture, the evaluator identified the expandability of the architecture as the strength and the tool also running in an environment that cannot be trusted (considering the security of the traffic relationships that exist between different network nodes within the organisational network) as the weakness of the tool.

7.4.13 Suggestion for Any Other Dashboard Features

The purpose of question 13 was to obtain suggestions from the experts regarding any other feature(s) that the dashboard could incorporate.

Although Prof Rogers did not suggest any new features, he pointed out that in terms of appearance the UO-NFAT dashboard should look more similar to the dashboards of the traditional digital forensic tools. Dr Christos was happy with the features that the tool is currently exhibiting. The evaluator added that he would like to see the privacy policy that will give the user IP addresses anonymity incorporated into the future version of the tool. Similar to Dr Christos, Dr Olga was also happy with the current features demonstrated by the tool and did not suggest any additional ones. Prof Venter, however, suggested considering incorporating “a bit of Artificial Intelligence (AI)” so that the system becomes able to “learn” trends. Also, Prof Venter made another suggestion about trying to incorporate a more proactive nature (instead of the current reactive approach exhibited by the tool) by identifying certain trends automatically. Dr Vidalis was pleased with the existing features of the tool and did not suggest any new ones.

7.4.14 Scope of the Tool in Assisting in the Investigation of Other Organisational Issues

Question 14 was included to find the scope of the demonstrated tool (UO-NFAT), according to the experts, in assisting in the investigation of any organisational issues other than insider threat/misuse.

As a response to this question, Prof Rogers said that this tool could be used to detect rootkits and botnets. Dr Christos thought that the developed tool can do much more than assisting in a forensic investigation. The evaluator pointed out that the tool can have a broad range of applications within an organisational environment such as a traffic analysis tool for monitoring online user activities. Dr Olga said that the tool has the potential to be utilised as an intelligence tool that will allow learning more about the users (employees) as well as the network even if it is outside the scope of a digital forensic investigation. The evaluator also said that if the tool could deal with live traffic and analyse for external attacks then there is so much more the tool can offer (in terms of incident response and network monitoring) apart from assisting in insider threat/misuse investigation. As a response to this question, Prof Venter said that the developed system could be utilised for information or any other type of audits that might be necessary or required by the organisational policies. As an example, Prof Venter discussed about using the tool to confirm the presence of a user within a system, which might be linked to tracking the user’s behaviour so as to determine if the employee is really doing their work they are supposed to do. In this manner the tool could be used for productivity measures as well. However, Prof Venter did point out that such a type of utilisation might only work when the employees are expected to constantly conduct their work activities by continually utilising the organisational network. Dr Vidalis identified the tool’s ability to offer an opinion on the impact of an incident that occurs within the organisational environment as the scope of UO-NFAT other than assisting in the investigation of insider related issues.

7.5 Use Cases

Tools that are utilised in a digital forensic investigation are developed to be of assistance in conducting the investigation process in a smooth and efficient manner and deliver information of evidential value. In order to demonstrate how a particular tool will be performing an investigation task that has been assigned, use cases can be utilised. The use cases selected for this purpose could be real historical incidents that were investigated or scenarios that are crafted to exercise the forensic analysis tool. Network forensic analysis tools will be focusing on either the investigation of an incident that has already taken place or an ongoing monitoring of network traffic in anticipation of an incident.

As discussed through the previous chapters, the aim of the research was to develop a network forensic analysis tool that is built on a case management premise and that can reduce the investigator's effort by generating information about the user activities from low level network metadata. In order to demonstrate the ability of UO-NFAT to assist in the investigation, three different use cases are discussed in this section. Each of the three use cases covers a different type of incident that can occur in an organisation, which could affect the financial as well as reputational well-being of the organisation.

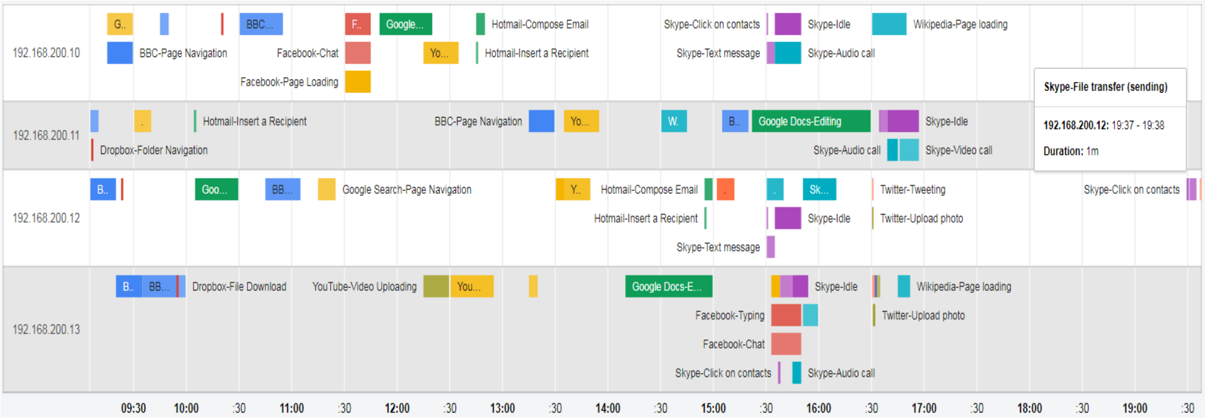
7.5.1 Use Case 1: Insider Misuse

In a medium scale organisation, the incident response team found out that confidential information has been leaked and been supplied to the competitors. There is no visible sign of any external attack on the organisational network, which led the team to suspect the involvement of one or more employees, classifying the incident as an insider threat. The investigator was assigned the task of resolving the case as quickly as possible.

- Who was involved in the leakage of the confidential data?
- When and how was the document leaked?
- Is there any other valuable information leaked that the incident response team is unaware of?

There are four managers who have legitimate access to the organisational data storage that holds all confidential information. The team had already made an assumption about the day on which the incident could have happened based on the date (2nd October, 2017) on which the leaked information was publicised by the competitors. Based on this information, the investigator decided to analyse the online traffic for the aforementioned individuals, who were thought to be the primary suspects, to see if any valuable information could be extracted.

The investigator used UO-NFAT to analyse the organisational network traffic metadata for the day before the incident occurred (i.e. 1st October, 2017).



Skype-File transfer (sending)

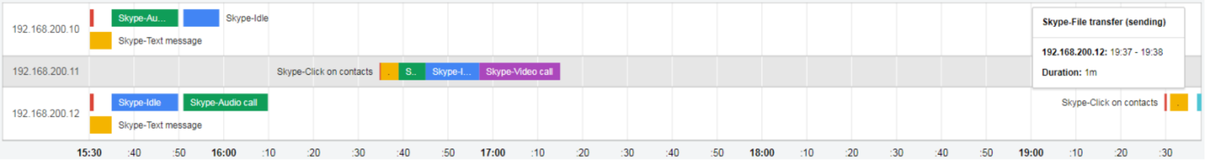
192.168.200.12: 19:37 - 19:38

Duration: 1m

Figure 7.2: Online activities of suspects on 1st October, 2017

The user IP addresses for the computers used by the managers who had access to the data are – 192.168.200.10, 192.168.200.11, 192.168.200.12 and 192.168.200.13. While examining the analysis result of the traffic that is shown in Figure 7.2, the investigator noticed that one of the managers was online after working hours on the 1st of October. The suspected manager seemed to be using Skype service and transferring a file around 7:37 PM on that day. The investigator looked for any file transfer or uploading activity (originating from any service other than Skype) initiated by these four employees, but was unable to find any.

To check further, the investigator applied the filter options to view the user activities only on the Skype service. The timeline generated by UO-NFAT is shown in Figure 7.3, which showed that only three out of the four managers used Skype on that particular day. Also, only one of them (with user IP address – 192.168.200.12) used Skype for transferring file, which was commenced at 7:37 PM on 1st of October, 2017.



Skype-File transfer (sending)

192.168.200.12: 19:37 - 19:38

Duration: 1m

Figure 7.3: Skype activities of suspects on 1st October, 2017

Although the aforementioned information provided by the UO-NFAT could indicate the possibility of an insider misuse, it still requires confirmation. In order to obtain confirmation, the investigator will have to gain rightful access to the data on the suspect's computer. Hence, at this point, the investigator decided to administer a computer forensic investigation on the local machine. By following the investigation procedure, the investigator was able to retrieve a copy of the document that was available locally, containing the leaked information. This confirmed that it was indeed one of the four suspects that committed the insider misuse and leaked the confidential information. Following this, the investigator used the UO-NFAT's reporting feature to produce the case report, which was submitted to the authorised personnel to take further actions.

This particular use case highlights the nature of the UO-NFAT, which is generating the online user activities from raw network traffic and helping the investigator through functionalities such as case management, visualisation and reporting. As seen in this case, the information provided by UO-NFAT was not a direct evidence to prove the crime. But, in fact, it helped the investigator to lead the inquiry in the right direction and to find the real evidence, which was obtained from the Skype archive.

7.5.2 Use Case 2: Data Leakage due to Accidental Policy Violation

On the 7th of November 2017, the product development manager of a large-scale organisation was informed that certain details about the company's latest product are already floating on the social media. The product launching was planned for the New Year in January 2018 and the product was anticipated to be one of the company's best so far and to play a role in raising the company's market share value. Despite taking all possible measures to keep the product details in between the relevant company staff, the situation left the product development manager wondering what went wrong.

- What caused the leakage of confidential information out in to the world?
- Was the incident a result of an inside job? If so, who is responsible for the act?

The investigator was allocated the task of finding the facts behind the incident and submitting the relevant information to the management. The team manager filled in the investigator on the events of that day. In the morning of the day the incident took place, the team manager had uploaded a documentation file containing the details about the new product and the product launching to Dropbox. The Dropbox account was accessible to all team members.

The team manager also recollected that there was a team meeting that morning from 10 AM till 12 PM and that all team members were present in the meeting. It was later in the evening on the same day when the team manager was notified about the leakage of the document.

Since only the team members are aware of the documentation that was prepared by the team manager, the members of the product development team were the primary suspects. The product development team contains five employees including the product development manager. The investigator started analysing the network traffic for all five members of the team in order to find the online user activities of the day when the product information was leaked. Going through the analysis results of that day which is shown in Figure 7.4, something peculiar caught the investigator's attention.

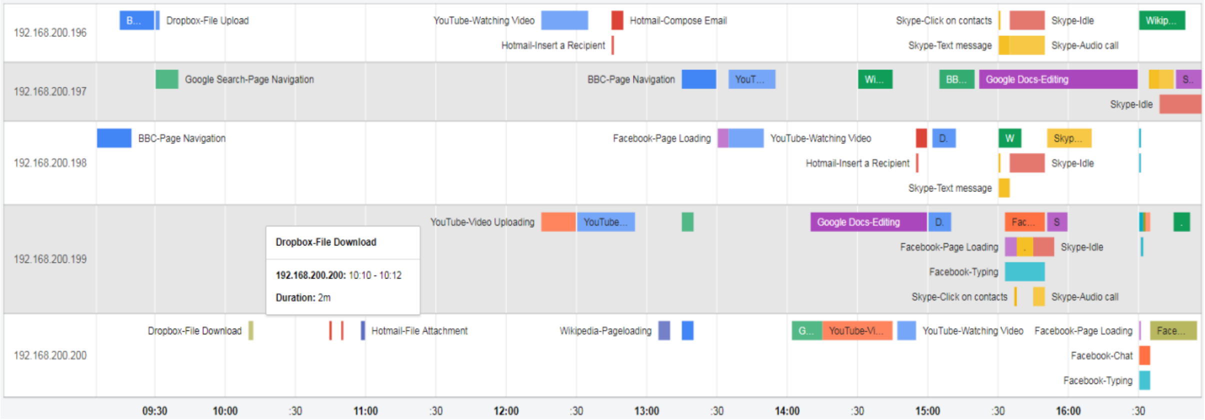


Figure 7.4: Online activities of suspects on 7th November, 2017

There were a few online activities detected for one of the team members' computer during when the team was having a meeting (i.e. between 10 and 12 o'clock). While all other members were inactive between 10 AM and 12:15 PM, the user with IP address 192.168.200.200 seemed to show online presence. This led the investigator to apply the UO-NFAT filters to zoom in on this user IP address for the aforementioned time duration, which is shown in Figure 7.5.

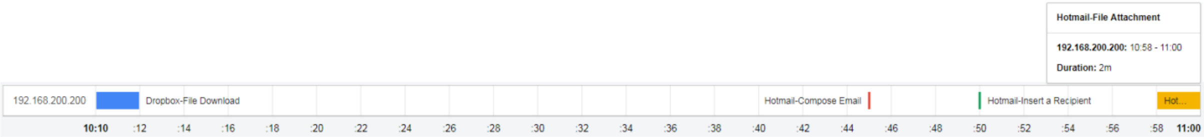


Figure 7.5: Suspicious online presence

The investigator discovered that there was a file downloaded from Dropbox at 10:10 AM followed by an attempt to compose an e-mail at 10:45 AM and then attaching a file in to the e-mail at 10:58 AM. Based on this result, the investigator was confident to confirm that someone had used the computer with the IP address 192.168.200.200, which belongs to one of the team members, who was in a team meeting at that time.

The investigator connected the ambiguous activities identified during when the computer was unattended. A possible scenario is – an employee other than the actual owner of the computer had succeeded in accessing the system to download the documentation from Dropbox and e-mailing it as an attachment. All relevant findings were bookmarked and the case report that the UO-NFAT produced was handed over to the management. On further investigation, it was found out that a colleague of the team member was the culprit. Workplace CCTV cameras revealed the actions. However, this plan was successful only because the actual user of the computer accidentally left the system unlocked, which gave the culprit access to the system. On questioning the user (i.e. the team member) confirmed that it was a one-time mistake. However, such policy violations due to accidental negligence had led to events that resulted in a massive trouble for the organisation.

7.5.3 Use Case 3: Workplace Productivity Monitoring

In a small scale business organisation containing 45 employees, the management decided to monitor the employees to analyse their worktime productivity. The management specifically wanted to understand the relationship between the employees' online activities and the workplace productivity during the working hours. In order to tackle the problem, it was necessary to know the daily online activities of the employees in the first place. The network administrator who is also trained to be the forensic investigator was called in and the task was assigned. The investigator was asked to find information that will help in answering certain questions raised by the assigned task, which were:

- How much online user activity occurred on the particular day?
- Are the observed activities beyond the expected level?
- Who is the most active employee during the work hours?

Since the UO-NFAT is a network forensic analysis tool, there was no need for the collection of data from system log files or application log files. The network traffic collected on a regular basis within the organisation was enough to provide a data feed for the tool. The investigator decided to extract the online activities of the users for the day (i.e. for 1st of

September 2017) from 9:00 AM till 5:00 PM. The tool plotted the resulting timeline chart displaying all extracted online user activities for the employees, which is shown in Figure 7.6.

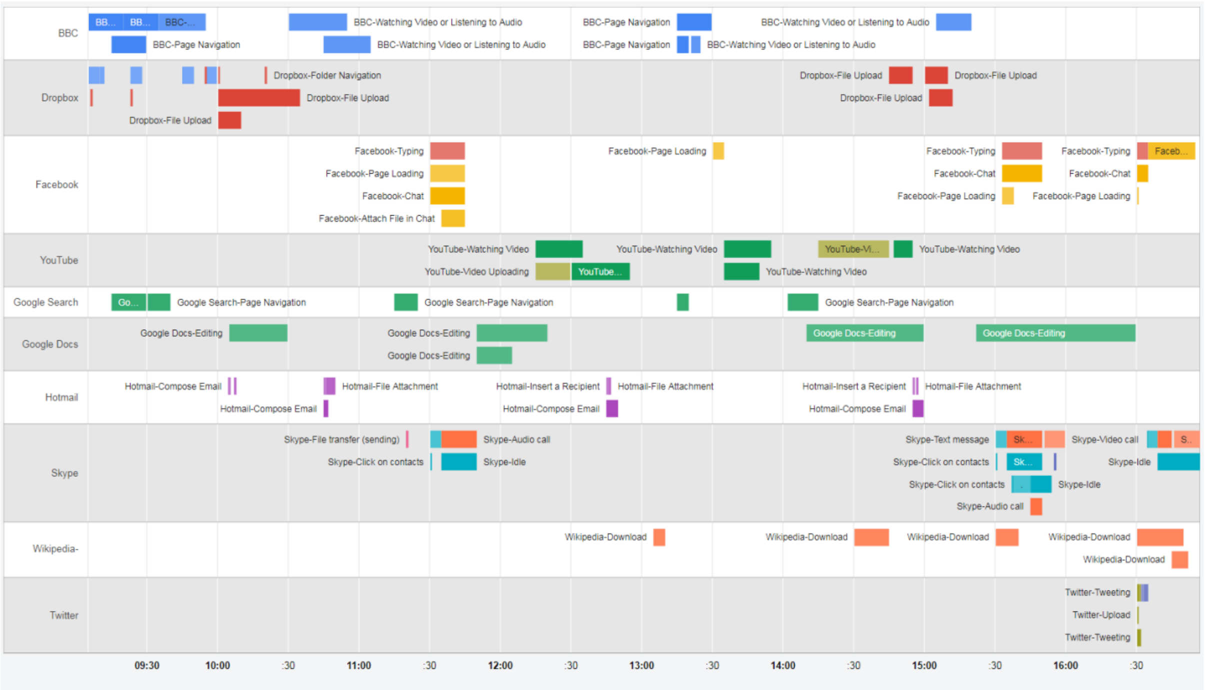


Figure 7.6: Daily online activities of users

From the chart it was clear that the employees were actually engaging in online activities during the work hours. While BBC website and Google Docs were used in between throughout the day, Skype, Facebook and YouTube activities were used mostly in the afternoon.

Following this, the investigator decided to see the daily online activities of the employees and the UO-NFAT gave the comparison chart that showed the top 5 employees who were the most active. The chart produced by the tool is displayed in Figure 7.7.

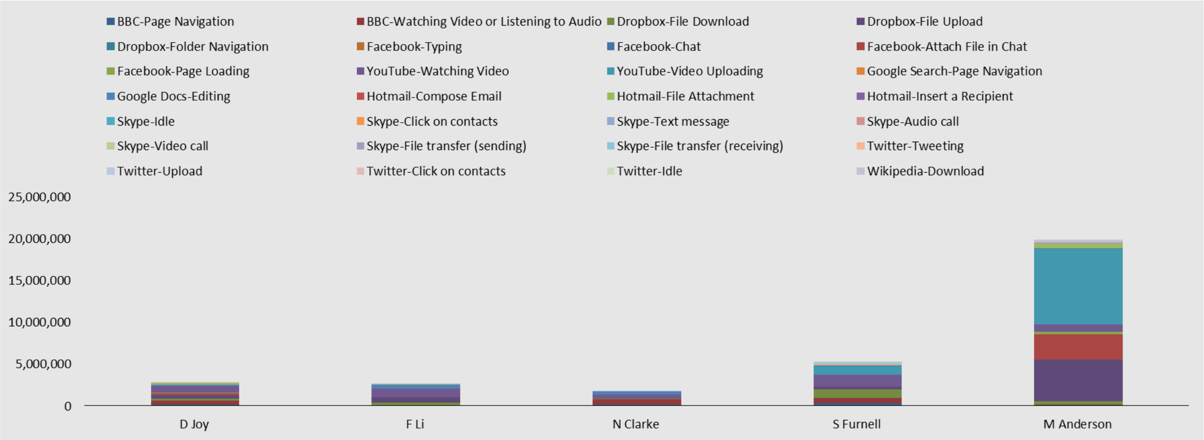


Figure 7.7: Comparison of users' daily online activities

The investigator was able to identify from the interactive visualisation chart that Mr. Anderson was the most active user online on September 1st generating a total of almost 19.9 million (19,898,622) packets by using various online activities identified by the tool. Mr. Anderson had engaged in majority of the identified activities on that day apart from video calling and receiving any file on Skype and any of the Twitter activities such as tweeting or uploading files. Major contribution by this user was via activities such as uploading video file to YouTube containing 9 million packets, uploading file to Dropbox generating about 5 million packets and attaching files in Facebook chat that produced about 3 million packets. Following Mr. Anderson was Mr. Furnell producing almost 5.28 million (5,278,066) packets most of which were as a result of activities such as watching video on YouTube (consisting of 1.4 million packets), uploading video to YouTube (containing 1 million packets) and downloading file from Dropbox (containing 1 million packets). The next two positions were occupied by Mr. Joy and Mr. Li with the main online activities of the day being a combination of watching video on YouTube, watching video or listening to audio on BBC, uploading file to Dropbox and document editing on Google Docs. While Mr. Joy did not engage in any of the Twitter activity, Mr. Li did not perform any of the Twitter or Facebook activities on the first day. Top online activities that Mr. Clarke participated in were watching video or listening to audio on BBC, watching video on YouTube and document editing on Google Docs. The investigator was able to make all these observations from the charts provided by the UO-NFAT. The investigator was able to make use of the reporting functionality of the tool to generate the report which was printed and handed over to the management.

Utilising the information provided by the UO-NFAT during the investigation, the authority was able to discover the online activity level in terms of man-hours, whether or not it is beyond the expected level and whether or not the employee productivity is affected by the situation. Based on the evidence provided by the investigator, with the help of UO-NFAT, the organization was able to take necessary steps to deal with the workplace productivity problem. This use case shows that even though the UO-NFAT is primarily a network forensic analysis tool, a peculiarity of the tool is its flexibility to be used as more than just a network forensic analysis tool. The information provided by UO-NFAT could be beneficial in more than one level within the organisation.

7.6 Discussion

It was important to find out whether the research that was put forth has contributed knowledge and added more value to the existing pool of knowledge within the field of digital forensics. An appropriate way to answer this question was by conducting an evaluation of the research in question. It was with this intention that the evaluation process was designed and executed, which involved gathering unbiased opinion from experts who have gained profound knowledge and experience in the digital forensics domain. The Discussion section begins with a review of the overall effectiveness of the evaluation process. Subsequently, both the suggestions and questions voiced by the participating experts during the interview sessions are addressed along with the researcher’s analysis in the best way possible.

Having completed the expert evaluation, it was worth assessing the effectiveness of the whole process and checking whether the process has fulfilled its purpose. The researcher believes that looking at elements that have affected the evaluation positively and adversely can give an insight into the effectiveness of the process. Factors such as the decision to opt for an interview instead of a questionnaire, decision to use an open-ended question format for the interview and the decision to combine the podcast viewing and the interview into a single session worked in favour of producing the desired result and improving the effectiveness of the evaluation process. The researcher believes that the audio/video content of the podcast conveyed the research concept clearly and helped the evaluators to create a mental image of what the fully developed version of the UO-NFAT would look like through various screenshots of the implemented prototype. Also, the decision to execute the first two phases in a synchronous fashion aided in extracting feedback that is of enriched quality during the interview phase thereby improving the overall effectiveness of the evaluation process.

Likewise, certain factors that have limited the evaluation process from attaining the maximum level of effectiveness were identified. The duration of a unique period in the process that was overlapped between the selection and the interview phases (i.e. from sending invitations till completing the interviews) was arguably one such factor that has limited the effectiveness of the evaluation process. An associated limiting factor was the total number of experts that the researcher was able to interview over the course of evaluation. Even though the aim was to include at least 12 experts in the evaluation process, as seen from section 7.3 of this chapter, the researcher decided to wrap up the process after interviewing 5 academic experts. The fact that the process was able to complete interviews with only 5 experts in a 40-days period had an unfavourable effect on the process. Although the feedback received was of high quality delivering a clean assessment of the system, the researcher assumes that the strength of the evaluation would have improved if more number of similar participants were interviewed within the same period of time. Another limiting factor was that the group of experts that was interviewed did not contain any practitioners. The academic experts did, indeed, provide a genuine and deep assessment of the research aspects of the UO-NFAT system that was presented, which might not have been possible for the counterpart (i.e. the practitioners) to provide. However, speaking from an implementation perspective, the practitioners would have been able to offer their insight into the design of the tool's user interface as well as the usability aspects of the UO-NFAT. Perhaps, based upon the prototype that was implemented, a professional perspective on factors such as user interface design and user experience based on the current industry standards would have come from the practitioners. In hindsight, the researcher thinks that the effectiveness of the evaluation process would have improved if the aforementioned limiting factors were absent.

The remaining part of this section will discuss suggestions and questions put forth by the experts during the interview sessions. To begin with, while answering the question on the effectiveness of the timeline based display of user activities (Section 7.4.7); a few suggestions were made by the evaluators. Prof Rogers suggested that it would be more effective if the timeline display allows the investigator to drill down particular sessions from the timeline itself. Currently the implemented prototype allows the investigator to achieve this by utilising the filter options in the visualisation configuration window. Prof Rogers also suggested that the dashboard interface of the tool needs to be tweaked to the HCI standards, which could help in making the dashboard even more user friendly thereby reducing a bit more of the investigator's effort. Dr Christos suggested starting displaying the timeline for a short time interval, and then the investigator can use the filter options to gradually expand the timeline. All of the aforementioned suggestions and modifications will be accomplished as part of the future work.

One thought expressed by Prof Hein Venter was that how much the interactions derived from the analysis of network metadata differs from the information that can be found by using a tool like Wireshark and that whether or not the same kind of information is accomplished with Wireshark. The primary difference is that the investigator will not be able to find the user activities from the raw network traffic displayed by a tool such as Wireshark. What Wireshark presents is the raw network packets with the entire header and payload data and it will require the investigator to spend both time and effort utilising multiple Wireshark filters in order to deduce any meaningful information. UO-NFAT, in contrast, provides a full chronological display of the online user activities. In that sense, the UO-NFAT differs from Wireshark both in terms of the level of meaningful information provided and the visual representation of that information that is easily understandable. Speaking from a point of analysis approach utilised by both tools, Wireshark uses a packet based network analysis approach whereas UO-NFAT operates using a novel interaction based network analysis approach. This allows the developed tool to dig deeper in to the traffic to find the presence of packets (single, multiple or stream) that are responsible for user activities. It is this approach that allows UO-NFAT to reduce the investigator’s burden of going through the raw traffic thereby assisting in the investigation process.

Another point that was highlighted by one of the expert evaluators – Dr Christos – was regarding the assignment of static IP addresses to the machines within the organisational network. In order for the UO-NFAT architecture to achieve its goal, it is assumed that all employee systems within an organisational subnet are assigned with static IP addresses. Dr Christos emphasized that this assumption has to be enforced so that the developed system can be utilised and to eliminate the situation of any employee using a dynamic IP address or making use of any tool to disguise the IP address while they are online.

Dr Christos raised a question about a privacy related issue that he thought the researcher needs to pay attention to. The intention behind this question was to show how the tool can be used in a bad way, which means that if fallen into the wrong hands it can result in privacy related violations regarding the users. As the tool is able to generate user activity patterns that are of privacy related nature (such as what the users were doing online at a specific time), he was concerned that the tool could be used in a malicious way. And he wanted the researcher to think about how the tool can handle the issue regarding user privacy. One way of addressing this issue would be by establishing a privacy policy for the tool. For instance, a policy that would enable the tool to hide the last three digits of the user IP addresses on a normal day and would only reveal any user IP address if and when a post-incident investigation occurs. Establishing such a privacy policy would be necessary as well as beneficial as the experts have identified the tool's potential to be used as more than a tool to assist in a network forensic investigation. The future work will include adding the adequate privacy policy so that the anonymity of the users (employees) is maintained unless a need for disclosure arises.

Dr Christos wanted to know how the tool identifying the online user activities is going to help an investigation involving an external attack such as malicious software. The UO-NFAT is designed to assist the investigation by identifying the online user activities within the organisational environment. The system is not designed with the investigation of an external attack including malicious software in mind. Hence it may not directly benefit an investigation of such a nature. However, it might help the investigator in finding out what the user(s) are up to while an external attack takes place, which may provide some valuable hints. Also, the identification of online user activities could be an indication of external attacks in a scenario where an intruder has gained access into an employee's computer and has started using the Internet based applications. This could result in traffic similar to that generated by a genuine user, which would be hard to detect. In such a circumstance, UO-NFAT could give the investigator helpful hints.

During the evaluation session, a suggestion made by Dr Christos was to consider renaming the framework so that the tool is not categorised only as a forensic analysis tool. The researcher believes that this suggestion was driven by two reasons. Firstly, the developed tool is something that 'assists' the investigator in a forensic investigation, not something that can carry out a completely independent investigation process. The identified activities may not serve as direct evidence, but they can become the supplementary information regarding the user activities at the time of the incident so that the investigator can do the correlation. Even though the tool can reduce the cognitive effort invested by the investigator, it still requires the investigator's intuition and intelligence. Secondly, Dr Christos thought that the tool is capable of being used as more than just a forensic analysis tool. The evaluator believes that it could also be utilised as a monitoring tool or in other aspects of an organisational environment. Even though both of the aforementioned factors has value to them, the researcher would like to categorise the tool as an NFAT due to its capability to perform forensic analysis and to exhibit functionalities such as visualisation and reporting, in a case management fashion.

Also, another suggestion made by Dr Christos was to evaluate the tool with an investigation practitioner, even if it is an informal one. He mentioned that it would be valuable knowing an investigator's perspective on matters such as whether the interface and the way the information presented is useful and practical, whether the investigator wants to modify any part of the interface etc.

Dr Olga pointed out that the online services used, the online activities performed and the incoming network traffic might be indicators of an external attack. And since UO-NFAT is capable of identifying the online services used, the online activities performed and the incoming traffic, the tool can be utilised to assist in the investigation external attacks as well. Also, Dr Olga made a suggestion regarding extending the architecture for live traffic analysis and analysing the traffic for external attacks. The evaluator believes that this would enable the tool to offer assistance beyond the limits of a forensic analysis tool, such as an incident response tool and a monitoring tool. Similar to this, Dr Vidalis also suggested that the tool has the capability to be utilised in auditing and compliances purposes as well.

While discussing question number 12 (Section 7.4.12), one of the expert evaluators (Prof Rogers) commented that the dashboard appeared a bit too busy and said that it could be overwhelming for an investigator from a law-enforcement background. However, another expert (Dr Vidalis) commented that any investigator will have to go through a learning curve while beginning to use a new tool or technology despite the investigator's background. The evaluator also said that the complexity will become less of a concern as the investigator will get used to the tool and that in time the investigator would prefer certain level of complexity. The researcher fully agrees with the latter comment and thinks that any investigator, despite the background, will need certain amount of time in order to get familiarised and eventually get used to the new tool. However, as part of the future work, the researcher would like to find out whether the issue that was pointed out in the former comment is a real problem and if so, explore the possibilities of any modifications that can reduce the effect of the problem.

Dr Vidalis suggested considering incorporating sentiment analysis into the system, which can help the system to offer an implication on the state of the user or on the type of interaction. The assumption is that each user is different; and even the same user on different days might do things slightly different because the user might be affected by the events that have happened in the recent life. The evaluator thought that it would really interesting if the researcher could integrate an analysis technique that will enable the system to suggest whether a change in the interactions has resulted from some kind of a threat or from a change in the user's state/behaviour.

Finally, majority of the evaluators suggested integrating live (next-to-real-time) traffic analysis into the tool. While a part of the evaluators preferred extending the architecture for this purpose, one evaluator opted for utilising appliances such as 'Ditto Shark' that can provide traffic data copied from the network instead of extending the architecture. The bottom line is that the evaluator thought that the idea of implementing next-to-real-time traffic analysis would be beneficial and the task will be accomplished in the future.

7.7 Conclusion

The expert evaluation of the work was a necessary and important stage of the research without which the research will not be whole. The evaluation was necessary because the assessment done by the external group of academic experts will strengthen the quality of the research and the thesis equally. Unbiased opinions on different aspects of the research such as the novelty of the research contribution, capability and effectiveness of the developed UO-NFAT system in accomplishing specific features, ability of the prototype to meet the system requirements, strengths and weaknesses of the developed prototype, and the possibility of future extension of the architecture were collected from the experts. The insightful feedback given by the experts was helpful in understanding the practicality of the UO-NFAT as a tool to assist the investigator. Discussions with the expert evaluators helped the researcher to recognise areas such as monitoring, user intelligence and auditing and compliance in which the tool can be utilised apart from in the investigation of insider threat/misuse. Also, the researcher was provided with valuable suggestions for improving some of the current features of the tool and was supplied with inputs to ponder for the future work of the research. Since the input provided by the five experts that participated in the evaluation was rich in its content and quality, the researcher chose not to go for more evaluators. Even though this decision was made due to the time constraint, the limited number of participants is still a weakness of the evaluation process. Once all comments, suggestions for modifications and criticisms made by the experts were documented in to the chapter, the evaluation stage of the research was successfully completed. As far as the future work of UO-NFAT is concerned, the researcher would like to improve the tool based upon the modifications suggested by the experts; incorporating the HCI standards, analysing live network traffic and establishing a privacy policy for the tool to name a few.

8 Conclusion and Future work

Through highlighting the achievements of the research and identifying the limitations, this chapter brings a conclusion to the research. The research goal was set to develop a novel web-based user oriented network forensic analysis tool (UO-NFAT) that can reduce the cognitive load of the investigator and the time of investigation. The tool was aimed to achieve this goal by generating online user interactions from the analysis of low-level network traffic metadata and by empowering the tool to integrate the functionalities such as case management, visualisation and reporting exerted by established computer forensic tools. A path was set by beginning learning about the network forensic domain and investigating the current state of the art in order to identify the research problem. Following the literature review, a novel solution to tackle the problem was hypothesised, which was tested for its feasibility. After proving the practicality of the hypothesis, the research went on to design a novel architecture based on which a prototype for the proposed tool was implemented. In the final stage of the research, the work done was evaluated by expert academics within the field.

8.1 Research Achievements

Core tasks involved in each stage of the research were conducting a critical review of the literature, devising a novel approach to identify user activities from network metadata, designing a novel architecture, implementation of a prototype and the evaluation of the research. Main achievements made during different stages of this research are listed below:

1. An understanding for the current state of the art was the necessary in the primary stage of the research. Starting from reviewing the different branches of digital forensics and narrowing it down to the network forensics domain, the literature review stage allowed the researcher to identify the research problem that needed to be addressed. Identifying that there is a need for a network forensic analysis tool that can reduce the investigator’s burden and the time of investigation while meeting the system requirements was the main achievement of the first stage of the research.
2. Designing a novel approach to extract online user interactions from the analysis of low-level network traffic metadata was a significant achievement of the second stage of the research. A novel interaction based approach to analyse the network metadata that resulted while users interacting with popular Internet based services was devised in this stage, which was capable of dealing with the issue of the enormous volume and the encrypted nature of network traffic.

3. Upon proving the hypothesis (i.e. the interaction based approach), the next stage of the research involved designing a novel architecture for the proposed UO-NFAT that is capable of incorporating functionalities such as case management, visualisation and reporting along with extracting the online user activities. Being able to design an architecture that accomplishes the aforementioned functionalities was the achievement made in stage three of the research.
4. Following the successful designing of the architecture, it was necessary to implement a prototype that can exhibit a subset of functionalities that are endorsed by the architecture. Successful implementation of a functional UO-NFAT prototype was the main achievement of the fourth stage of the research. The prototype helped in proving that feasibility of developing the full version of the tool that was proposed via the designed architecture.
5. In order to strengthen the research, the work done was in need of evaluation from the experts within the domain of digital forensics. The evaluators' perspective regarding the novelty of the research contribution, system's effectiveness in accomplishing specific features, ability to meet the system requirements, strengths and weaknesses of the prototype and possibilities of future extensions of the architecture was collected from the expert evaluators. Being able to collect the feedback that was rich in its content added value to the research and was the achievement of the fifth stage of the research.

8.2 Limitations of Research

Along with the achievements made, there were certain limitations that are identified.

1. Lack of availability of the public datasets containing network metadata that are suitable for the analysis in order to identify the user activities was one of the limitations. As a result, the researcher had to collect a new dataset that served the purpose of identifying the user interactions.
2. Another limitation of the research was the inability to test the implemented prototype using a larger dataset containing more number of users and more network metadata. Although the dataset that was utilised was enough to prove the concept, analysing a larger dataset would have revealed unseen issues related to the prototype, which could help to improve the UO-NFAT architecture.

3. Limited number of online user interactions that were identified utilising the interaction-based approach was another drawback of the research. During the experiment, the researcher was able to extract identifiable signatures for twenty-eight different interactions that are from the ten selected services. However, there were other user interactions that did not have any identifiable signatures. For instance, activities such as replying to an email in Hotmail, posting/commenting on Facebook and creation/sharing/deletion of document in Google Docs did not deliver any pattern that could help in identifying those interaction. Also, it is possible that services that were not included in the experiment may contain user interactions with identifiable signatures, which can only be confirmed with further exploration.
4. Limitation in time and development skills did put certain boundaries to the prototype-development phase of the research. Being a novice in developing the web based prototype, the task demanded more of the researcher's time. Even though, an extensive amount of time and effort was spent on developing a functional prototype that is based on the UO-NFAT architecture, the final product was not something that matches the commercial forensic tools when it comes to design quality. If the depth of skills within the field of web application development was at an expert level, then the researcher would have been able to improve the usability design of the prototype's interfaces thereby offering a better user experience.
5. The limited number of experts participated in the interviews that were conducted during the evaluation phase was another drawback of the research. Feedback from more experts would definitely have improved the quality of the evaluation process.

8.3 Future Work

The research identified a gap that existed within the domain of network forensics and successfully executed the best effort within the researcher's capacity in proposing a novel solution to the problem (i.e. a novel user oriented network forensic analysis tool) followed by the development and evaluation of the prototype. However, there are areas that the current research was not able to include within the PhD phase due to the time constraints. These unfulfilled stages can be taken further into the future work of the research.

During the evaluation stage of the research, valuable suggestions for modifications were provided by the expert evaluators. Suggestions including modifying the dashboard according to the Human Computer Interface (HCI) standards, start displaying the timeline for a short time duration to start with, incorporating the zoom in and zoom out feature into the timeline itself and improving the security feature of the tool by hiding the user IP addresses in order to maintain the anonymity of the employees until the investigation demands for any identity to be revealed. All of the aforementioned suggestions will certainly be undertaken as part of the future work. Apart from these modifications, certain tasks are set to be fulfilled as part of the future research, which are briefed in this Section.

8.3.1 Automated Signature Extraction

In this research, signatures for the online user activities (for the 9 selected Internet services) were extracted through manual analysis. Developing a methodology that would make the tool capable of extracting the user activity signatures for existing as well as new Internet based applications is a part of the future work. The automated signature extraction will result in the modification of the UO-NFAT architecture incorporating a module for the automated extraction of signatures. This can make the tool become both fully automated and dynamic in nature and improve the overall performance.

8.3.2 Real-time Processing of Network Metadata

Improving the scalability and performance of the tool is another part of the future work. This can be achieved by making the tool capable of analysing live (next-to-real-time) network traffic metadata thereby providing a picture of what the users are up to in real-time. A notion for making the real-time processing of network metadata was suggested by one of the expert evaluators during the evaluation stage. The idea involved utilising a hardware appliance that is capable of creating a copy of the network traffic for every 5 or 10 minutes duration that could be saved into a database and then the tool can analyse the data for identifying the user activities.

8.3.3 Facilitation of Integrated Visualisation

The current architecture is dependent on the utilisation of third party applications, such as Google Charts and other visualisation APIs for enabling the tool’s visualisation feature. The future work of the research would be seeking interest in integrating visualisation techniques into the UO-NFAT that would enable the tool to generate the visualisation functionality in an independent manner. This will involve expanding the tool with the built-in capability to generate the visualisation charts on request. Enabling UO-NFAT with integrated visualisation will improve the security of the tool as the tool will not be relying upon any external visualisation application.

8.4 The Final Word

The challenges, such as the enormous volume and the encrypted nature of network traffic, that were identified during the course of this research will continue to exist. Perhaps the advancements in domains such as Big Data, Internet of Things (IoT), Security and Privacy will increase the intensity of those challenges. As a result, it is possible that the network forensic investigation domain is going to be in need of innovative ideas to minimise the effects of the aforementioned challenges, and is going to thrive in the future.

References

1. Cybersecurity Ventures (2017) *2017 Cybercrime Report.* Available at: https://cybersecurityventures.com/2015-wp/wp-content/uploads/2017/10/2017-Cybercrime-Report.pdf (Accessed: 11 May 2018)

2. Goodman, M. D. (1997) 'Why the Police Don't Care About Computer Crime', *Harvard Journal of Law & Technology,* **10**(3), pp. 466-495

3. Commission of the European Communities (2001) *Communication from the Commission to the Council, the European Parliament, the Economic and Social Committee and the Committee of the Regions.* Brussels

4. Verizon (2017) *2017 Data Breach Investigations Report.* Available at: https://enterprise.verizon.com/resources/reports/2017_dbir.pdf (Accessed: 2 August 2018)

5. PwC (2018) *The Global State of Information Security Survey 2018.* Available at: https://www.pwc.com/us/en/services/consulting/cybersecurity/library/information-security-survey.html (Accessed: 5 March 2019)

6. United Nations Crime and Justice Information Network (1994) *International review of criminal policy - United Nations Manual on the prevention and control of computer-related crime*. Available at: https://www.ncjrs.gov/App/Publications/abstract.aspx?ID=152325 (Accessed: 18 March 2014)

7. Evans, D. (2011) 'The Internet of Things – How the next evolution of the Internet is changing everything'. *Cisco Internet Business Solutions Group (IBSG) white paper*, 1-11

8. Internet Crime Complaint Centre (2018) *2018 Internet crime report.* Available at: https://pdf.ic3.gov/2018_IC3Report.pdf (Accessed: 3 January 2019)

9. Information Security and Forensics Society (2004) *An Introduction to Computer Forensics.* Available at: http://www.isfs.org.hk/publications/ComputerForensics_part1.pdf (Accessed: 6 July 2013)

10. Garfinkel, S L. (2010) ‘Digital Forensics Research: The next 10 years’, *Digital Investigation: The International Journal of Digital Forensics & Incident Response,* **7**, pp. 64-73

11. Reith, M. *et al.* (2002) ‘An Examination of Digital Forensic Models’, *International Journal of Digital Evidence,* **1**(3), pp. 1-12

12. US-CERT (2008) *Computer Forensics.* Available at: https://www.us-cert.gov/sites/default/files/publications/forensics.pdf (Accessed: 13 February, 2013)

13. Ruan, K., Carthy, J., Kechadi, T. and Crosbie, M. (2011) ‘Cloud Forensics: An Overview’ *Advances in Digital Forensics,* 7, pp. 15-26

14. Mandia, K. and Procise C. (2003) *Incident Response and Computer Forensics,* New York: Osborne McGraw-Hill

15. Clarke, N. (2010) *Computer Forensics – A Pocket Guide.* Cambridgeshire: IT Governance Publishing

16. Farmer, D. and Venema, W. (2005) *Forensic Discovery,* Massachusetts: Addison-Wesley

17. Pollitt, M. (1995) ‘Computer Forensics: An Approach to Evidence in Cyberspace’, *Proceeding of the National Information Systems Security Conference, Baltimore*, **2**, pp. 487-491

18. Carrier, B. and Spafford, E.H. (2003) ‘Getting Physical with the Digital Investigation Process’, *International Journal of Digital Evidence* **2**(2), pp. 1-20

19. Stephenson, P. (2003) ‘A comprehensive Approach to Digital Incident Investigation’, *Elsevier Information Security Technical Report*, **8**(2), pp. 42-54

20. Baryamureeba, V. and Tushabe, F. (2006) ‘The Enhanced Digital Investigation Process Model’, *Asian Journal of Information Technology,* **5**(7), pp. 790-794

21. Ciardhuain, S. (2004) ‘An Extended model of Cybercrime Investigations’, *International Journal of Digital Evidence,* **3**(1), pp. 1-22

22. Rogers, M. *et al.* (2006) ‘Computer Forensics Field Triage Process Model’, *Conference on Digital Forensics, Security and Law*

23. Freiling, F.C. and Schwittay, B. (2007) ‘A Common Process Model for Incident Response and Computer Forensics’, *Proceedings of the Conference on IT Incident Management and IT Forensics,* pp. 1-21

24. Perumal (2009) ‘Digital Forensic Model Based On Malaysian Investigation Process’, *International Journal of Computer Science and Network Security*, **9**(8), pp. 38-44

25. Agarwal, A. *et al.* (2011) ‘Systematic Digital Forensic Investigation Model’, *International Journal of Computer Science and Security (IJCSS),***5**(1), pp. 118-131

26. James, J. I. and Gladyshev, P. (2013) ‘Challenges with Automation in Digital Forensic Investigations’, *Computers and Society,* pp. 1-17

27. Aminnezhad, A., Dehghantanha, A. and Abdullah, M T. (2012) ‘A Survey on Privacy Issues in Digital Forensics’, *International Journal of Cyber-Security and Digital Forensics (IJCSDF)*, **1**(4), pp. 311-323

28. Mercuri, R. (2010) ‘Criminal Defence Challenges in Computer Forensics’, *International Conference on Digital Forensics and Cyber Crime,* 31, pp 132-138 pp.

29. Hoelz, B.W., Ralha, C.G. and Geeverghese, R. (2009) ‘Artificial Intelligence Applied to Computer Forensics’, *Proceedings of the 2009 ACM symposium on Applied Computing*, 883-888

30. Arthur, K. K. and Olivier, M.S. (2009) ‘Processing Algorithms for Components within a Forensic Evidence Management System’, *Proceedings of the Fourth International Workshop on Digital Forensics & Incident Analysis (WDFIA)*, pp. 53-62

31. Waits, C., Akinyele, J. A., Nolan, R. and Rogers, L. (2008) ‘Computer Forensics: Results of Live Response Inquiry vs. Memory Image Analysis’, *Technical Note Software Engineering Institute*, pp. 1-31

32. Ieong, R. S. (2006) ‘FORZA – Digital forensics investigation framework that incorporate legal issues’, *Digital Investigation: The International Journal of Digital Forensics & Incident Response,* **3**, pp. 29-36

33. Rogers, M. K. *et al.* (2006) ‘Computer Forensics Field Triage Process Model’. *Conference on Digital Forensics, Security and Law,* **1**(2), pp. 27-40

34. International Telecommunication Union (2017) *ICT Facts and Figures.* Available at: https://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2017.pdf (Accessed: 5 December 2018)

35. Curran, K. *et al.* (2010) 'Mobile Phone Forensic Analysis', *International Journal of Digital Crime and Forensics,* **2**(2), pp. 1-11

36. Mell, P. and Grance, T. (2011) *The NIST Definition of Cloud Computing.* Available at: https://csrc.nist.gov/publications/detail/sp/800-145/final (Accessed: 10 December 2013)

37. Khajeh-Hosseini, A., Greenwood, D. and Sommerville, I. (2010) 'Cloud Migration: A Case Study of Migrating an Enterprise IT System to IasS', *IEEE 3rd International Conference on Cloud Computing (CLOUD 2010)*

38. Market Research Media *Global Cloud Computing Market Forcast 2019-2024.* Available at: https://www.marketresearchmedia.com/?p=839 (Accessed: 3 June 2018)

39. Zawoad, S. and Hasan, R. (2013) 'Cloud Forensics: A Meta-Study of Challenges, Approaches and Open Problems', *arXiv:1302.6312v1.* Available at: https://arxiv.org/pdf/1302.6312.pdf (Accessed: 13 December 2013)

40. Huth, A. and Cebula, J. (2013) *United States Computer Emergency Readiness Team – The Basics of Cloud Computing.* Available at: https://www.us-cert.gov/security-publications/basics-cloud-computing (Accessed: 17 November, 2013)

41. Ruan, K. and Carthy, J. (2011) 'Cloud Forensics Maturity Model', *Digital Forensics and Cybercrime: 4th International Conference ICDF2C 2012.* pp. 22-41

42. Paraben Corporation (2014) *Computer Forensic Tools Comparison Chart - 2013.* Available at: https://www.paraben.com/downloads/p2c-comparison.pdf (Accessed: 19 February, 2014)

43. Paraben Corporation (2016) *Capabilitites Brief 2016.* Available at: https://cdn.website-editor.net/7ea03a3a8f4c4ee38e44600303d3124e/files/uploaded/SEG-%2520Paraban-Digital-Forensics.pdf (Accessed: 19 October, 2020)

44. Garfinkel, S. (2007) ‘Anti-Forensics: Technique, Detection and Countermeasures’, *2^{nd} International Conference on i-Warfare and Security.* pp. 77-84

45. Parziale, L., Britt, D.T., Davis, C., Forrester, J., Liu, W., Mathews, C., and Rosselot, N. (2006) *TCP/IP Tutorial and Technical Overview.* Available at: http://www.redbooks.ibm.com/redbooks/pdfs/gg243376.pdf (Accessed: 5 November 2013)

46. Cisco (2019) *Internetworking Basics.* Available at: https://www.cisco.com/E-Learning/bulk/public/tac/cim/cib/using_cisco_ios_software/linked/tcpip.htm (Accessed: 12 February 2014)

47. Kozierok, C. (2005) *TCP/IP Guide – A Comprehensive, Illustrated Internet Protocols Reference* United States of America: William Pollock

48. Cisco (2019) *TCP/IP Overview.* Available at: https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13769-5.html (Accessed: 12 February 2014)

49. Internet World Stats (2019) *World Internet Users Statistics and 2019 World Population Stats.* Available at: https://www.internetworldstats.com/stats.htm (Accessed: 10 March 2014)

50. Joshi, R C., Niyogi, R. and Pilli, E S. (2010) ‘Network Forensic Frameworks: Survey and research challenges’, *Digital Investigation,* **7**, pp. 14-27

51. Pilli, E.S., Joshi, R.C. and Niyogi, R. (2010) 'A Generic Framework for Network Forensics', *International Journal of Computer Applications.* 1(11), pp. 1-6

52. Yasinsac, A. and Manzano, Y. (2002) ‘Honeytraps: A Network Forensic Tool’. *Proceedings of the 6^{th} multi-conference on systemics, cybernetics and informatics*

53. Ren, W. and Jin, H. (2005) ‘Modelling the Network Forensics Behaviour’, *Proceedings of the first International Conference on Security and Privacy for Emerging Areas in Communication Networks*, pp. 1-8

54. Microsoft (2014) *Common Types of Network Attacks.* Available at: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-2000-server/cc959354(v=technet.10) (Accessed: 14 March 2014)

55. Lamis T. (2010) ‘A Forensic Approach to Incident Response’, *Proceedings of 2010 Information Security Curriculum Development Conference.* pp. 177-185

56. Khurana, H. *et al.* (2009) ‘Palantir: A Framework for Collaborative Incident Response and Investigation’, *Proceedings of the 8th Symposium on Identity and Trust on the Internet,* pp. 38-51

57. Corey, V. *et al.* (2002) ‘Network Forensic Analysis’, *IEEE Internet Computing*. **6**(6), pp. 60-66

58. SC Magazine (2013) *NIKSUN NetDetectorLive.* Available at: https://www.scmagazine.com/home/reviews/first-looks/niksun-supreme-eagle-netdetectorlive/ (Accessed: 23 September, 2013)

59. NIKSUN *NetDetector.* Available at: https://www.niksun.com/c/1/ds/NIKSUNDatasheet_NetDetector.pdf (Accessed: 23 September, 2013)

60. RSA (2019) *RSA Netwitness Platform.* Available at: https://www.rsa.com/content/dam/en/infographic/rsa-netwitness-11-reasons-to-upgrade.pdf (Accessed: 29 May 2019)

61. AccessData Corp. (2013) *SilentRunner Sentinel.* Available at: https://accessdata.com/products/cyber-security/silent-runner (Accessed: 13 July, 2013)

62. LiveAction (2019) *OmniPeek Network Analyzer.* Available at : https://www.liveaction.com/products/omnipeek-network-protocol-analyzer/?__hstc=43204730.1c4d878a9f8687d8b9770e6c6a9f5afc.1563169726498.1563169726498.1563169726498.1&__hssc=43204730.1.1563169726500&__hsfp=1939929211 (29 May, 2019)

63. Sourceforge (2005) *PyFlag Tutorials.* Available at: http://pyflag.sourceforge.net/Documentation/tutorials/ (29 April, 2013)

64. NETRESEC (2007) *NetworkMiner.* Available at: https://www.netresec.com/?page=NetworkMiner (8 July, 2013)

65. Xplico (2018) *Xplico Features.* Available at: http://www.xplico.org/about (10 September, 2013)

66. Sira, R. (2003) ‘Network Forensic Analysis Tools: An Overview of an Emerging Technology’, *Global Information Assurance Certification Paper Directory*. pp. 1-12

67. Cohen, M. I. (2008) ‘PyFlag – An advanced network forensic framework’, *Digital Investigation: The International Journal of Digital Forensics & Incident Response.* **5**, pp. 112-120

68. Cohen, M. and Collet, D. (2007) ‘PyFlag Forensic and Log Analysis GUI – Presentation’, *AusCERT Asia Pacific Information Technology Security Conference.* Available at: http://pyflag.sourceforge.net/Presentations/PyFlag_Auscert2007/ (Accessed: 27 July, 2013)

69. Casey, E. (2004) ‘Tool review: Network traffic as a source of evidence: tool strengths, weaknesses and future needs’. *Digital Investigation: The International Journal of Digital Forensics and Incident Response,* **1**(1), 28-43

70. Casey, E. (2011) Digital evidence and computer crime: Forensic science, computers, and the internet. USA: Academic press

71. Merkle, L.D. (2008) ‘Automated Network Forensics’, *Proceedings of the conference on genetic and evolutionary computation (GECCO 2008).* pp. 1929-1932

72. Stoffel, K., Cotofrei, P. and Han, D. (2010) ‘Fuzzy Methods for Forensic Data Analysis’, *2010 International Conference of Soft Computing and Pattern Recognition (SoCPar).* pp. 23-28

73. Mukkamala, S. and Hung, A.W. (2003) ‘Identifying Significant Features for Network Forensic Analysis using Artificial Intelligence Techniques’, *International Journal of Digital Evidence.* **1**(4), pp. 1-17

74. Tsai, C., Hsu, Y., Lin, C. and Lin, W. (2009) ‘Intrusion Detection by Machine Learning: A Review’, *International Journal of Expert System with Applications.* **36**(10), pp. 11994-12000

75. ZOHO Corp (2007) *Analysing Logs for Security Information Event Management.* Available at: https://download.manageengine.com/products/eventlog/Analyzing-Logs-for-SIEM-Whitepaper.pdf (Accessed: 18 May, 2014)

76. Cisco (2010) *Cisco Security Information Event Management Deployment Guide.* Available at: https://www.cisco.com/c/dam/en/us/solutions/collateral/enterprise/design-zone-security-technology-partners/bn_cisco_siem.pdf (15 March 2014)

77. Gartner (2012) *Security Information and Event Management 2012.* Available at: https://www.gartner.com/it-glossary/security-information-and-event-management-siem (Accessed: 6 May 2014)

78. McAfee (2013) *SIEM: Keeping Pace with Big Security Data.* Available at: https://www.mcafee.com/enterprise/en-gb/security-awareness/operations.html (Accessed: May, 2014)

79. Butler, M. (2009) *Benchmarking Security Information Event Management.* Available at: https://www.sans.org/reading-room/whitepapers/analyst/membership/34755 (Accessed: 23 May, 2014)

80. Francis, D. (2012) *InformationWeek reports - IT Pro Ranking: SIEM vendors survey 2012.* Available at: http://eval.symantec.com/mktginfo/enterprise/other_resources/b-informationweek-it-pro-ranking-siem_june_2012.en-us.pdf (Accessed: 14 May, 2014)

81. RSA (2012) *Transforming Traditional Security Strategies into An Early Warning System for Advanced Threats.* Available at: https://uk.emc.com/collateral/software/solution-overview/h11031-transforming-traditional-security-strategies-so.pdf (Accessed: 24 May, 2014)

82. Splunk (2012) *Splunk, Big Data and the Future of Security.*Available at: https://www.ethicalhat.com/wp-content/uploads/2016/05/Splunk-Big-Data-and-the-Future-of-Security-US-Format-1.pdf (Accessed: 25 May, 2014)

83. Nikkel, B.J. (2005) ‘Generalizing Sources of Live Network Evidence’, *Digital Investigation: The International Journal of Digital Forensics and Incident Response,* **2**(3), pp. 193-200

84. Palmer, G. (2001). ‘A road map for digital forensic research’, *Proceedings of The Digital Forensic Research Conference*, pp. 27-30. Available at: https://www.dfrws.org/sites/default/files/session-files/a_road_map_for_digital_forensic_research.pdf (Accessed: 1 August 2018)

85. Lutes, K. D., and Mislan, R. P. (2008) 'Challenges in mobile phone forensics', *Proceeding of the 5th International Conference on Cybernetics and Information Technologies, Systems and Applications (CITSA)*. Available at: http://www.iiis.org/cds2008/cd2008sci/citsa2008/paperspdf/i649ok.pdf (Accessed: 10 September 2018)

86. Ibrahim, N. *et al.* (2016) 'SIM Card Forensics: Digital Evidence', *Annual ADFSL Conference on Digital Forensics, Security and Law*, pp. 219-234. Available at: https://commons.erau.edu/cgi/viewcontent.cgi?article=1367&context=adfsl (Accessed: 15 September 2018)

87. Yasinsac, A. and Manzano, Y. (2001) 'Policies to Enhance Computer and Network Forensics', *Proceedings of the 2001 IEEE Workshop on Information Assurance and Security. pp. 289-295*

88. Cahyani, N. D. W. *et al.* (2017) 'Forensic data acquisition from cloud-of-things devices: windows Smartphones as a case study', *Concurrency and Computation: Practice and Experience*, *29*(14). Available at: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpe.3855 (Accessed: 20 September 2018)

89. Anobah, M. *et al.* (2014) 'Testing framework for mobile device forensics tools', *Journal of Digital Forensics, Security and Law*, *9*(2), pp. 221-234. Available at: https://pdfs.semanticscholar.org/9dc5/01f4a48957cdc8473367d25034fc3251ec10.pdf (Accessed: 5 October 2018)

90. Barmpatsalou, K. *et al.* (2018) 'Current and future trends in mobile device forensics: A survey', *ACM Computing Surveys (CSUR)*, *51*(3). Available at: https://dl.acm.org/citation.cfm?id=3177847 (Accessed: 12 October 2018)

91. Bednar, P. and Katos, V. (2011) 'SSD: New challenges for digital forensics', *VIII Conference of the Italian Chapter of AIS Information Systems: a crossroads for Organization, Management, Accounting and Engineering Proceedings ISBN: 978-88-6105-063*. Available at: https://www.researchgate.net/publication/264001774_SSD_New_Challenges_for_Digital_Forensics (Accessed: 23 October 2018)

92. Joshi, B.R. and Hubbard, R. (2016) 'Forensics analysis of solid state drive (SSD)', *Proceedings of 2016 Universal Technology Management Conference (UTMC)*. Available at: https://pdfs.semanticscholar.org/3220/91e946997199b14713cf52eb295eb2f8be6c.pdf?_ga=2.74473810.1363913891.1562899997-1574940338.1562899997 (Accessed: 29 October 2018)

93. Bell, G.B. and Boddington, R. (2010) 'Solid state drives: the beginning of the end for current practice in digital forensic recovery?', *Journal of Digital Forensics, Security and Law*, *5*(3). Available at: https://www.researchgate.net/publication/228662565_Solid_State_Drives_The_Beginning_of_the_End_for_Current_Practice_in_Digital_Forensic_Recovery (Accessed: 3 November 2018)

94. Council of Europe (2001) *Convention on Cybercrime.* Budapest

95. Dadax (2019) *Internet Users in Real Time - Internet Live Stats*. Available at: https://www.internetlivestats.com/watch/internet-users/ (Accessed: 17 April 2018)

96. Cisco (2019) *Cisco Visual Networking Index: Forecast and Trends, 2017-2022.* Available at: https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/white-paper-c11-741490.html#_Toc484813982 (Accessed: 20 January 2019)

97. NewZoo (2019) *Global Mobile Market Report 2018.* Available at: https://resources.newzoo.com/hubfs/Factsheets/Newzoo_The_Global_Mobile_Market_Report_Fact_Sheet.pdf?hsCtaTracking=42c30a40-0d26-4ab1-9b15-600d1203b1c4%7C1615e0db-57f0-4a8c-a537-e09ac65e41f8 (Accessed: 25 January 2019)

98. Statista (2019) *Global Digital Population as of April 2019.* Available at: https://www.statista.com/statistics/617136/digital-population-worldwide/ (Accessed: 2 February 2019)

99. Datareportal (2019) *Digital 2019: Global Digital Overview.* Available at: https://datareportal.com/reports/digital-2019-global-digital-overview (Accessed: 3 March 2019)

100.McAfee (2019) *McAfee Labs Threats Report December 2018.* Available at: https://www.mcafee.com/enterprise/en-us/assets/reports/rp-quarterly-threats-dec-2018.pdf (Accessed: 5 March 2019)

101.AO Kaspersky Lab (2019) *IT Threat Evolution Q3 2018 Statistics.* Available at: https://securelist.com/it-threat-evolution-q3-2018-statistics/88689/ (Accessed: 10 March 2019)

102.Google (2019) *Android Developers: Data and File Storage Overview.* Available at: https://developer.android.com/guide/topics/data/data-storage (Accessed: 11 March 2019)

103.Das, R. (2017) 'Evidence Acquisition in Mobile Forensics', *Infosec*, 24 November. Available at: https://resources.infosecinstitute.com/evidence-acquisition-mobile-forensics-2/ (Accessed: 15 March 2019)

104.Microsoft (2016) *Cryptography and Encryption in Office 2016.* Available at: https://docs.microsoft.com/en-us/deployoffice/security/cryptography-and-encryption-in-office (Accessed: 20 April 2018)

105.NIST (2011) *The NIST Definition of Cloud Computing.* Available at: https://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-145.pdf (Accessed: 25 March 2019)

106.NIST (2014) *NIST Cloud Computing Forensic Science Challenges.* Available at: https://csrc.nist.gov/csrc/media/publications/nistir/8006/draft/documents/draft_nistir_8006.pdf (Accessed: 25 March 2019)

107.Statista (2019) *Global Hard Disk Drive (HDD) Shipments 1976-2022.* Available at: https://www.statista.com/statistics/398951/global-shipment-figures-for-hard-disk-drives/ (Accessed: 1 April 2019)

108.Statista (2019) *Enterprise HDDs and SSDs: revenue comparison worldwide 2012-2019.* https://www.statista.com/statistics/619653/worldwide-enterprise-hdd-and-ssd-industry-revenue/ (Accessed: 4 April 2019)

109.Statista (2019) *Number of apps available in leading app stores as of first quarter of 2019.* Available at: https://www.statista.com/statistics/276623/number-of-apps-available-in-leading-app-stores/ (Accessed: 5 April 2019)

110.Statista (2019) *Cloud Computing - Statistics and Facts.* Available at: https://www.statista.com/topics/1695/cloud-computing/ (Accessed: 7 Jun 2018)

111.Hruska, J. (2019) 'How Do SSDs Work', *ExtremeTech,* 6 February. Available at: https://www.extremetech.com/extreme/210492-extremetech-explains-how-do-ssds-work (Accessed: 5 April 2019)

112.Mohsin, T. (2019) 'Free and Open Source Computer Forensic Tools', *Infosec*, 27 September. Available at: https://resources.infosecinstitute.com/category/computerforensics/introduction/free-open-source-tools/ (Accessed: 15 March 2019)

113.Barraco, L. (2014) 'Top 5 Problems with Traditional SIEM', *AT&T Cybersecurity*, 15 April. Available at: https://www.alienvault.com/blogs/security-essentials/top-5-problems-with-traditional-siem-infographic (Accessed: 5 July 2019)

114.Symantec Corporation (2019) *What is SSL, TLS and HTTPS?*. Available at: https://www.websecurity.symantec.com/en/uk/security-topics/what-is-ssl-tls-https (Accessed: 1 July 2019)

115.IETF (2019) *RFC 2818 – HTTP over TLS*. Available at: https://tools.ietf.org/html/rfc2818 (Accessed: 1 July 2019)

116.Wireshark (2019) *TLS – The Wireshark Wiki*. Available at: https://wiki.wireshark.org/TLS (Accessed: 3 July 2019)

117.Drehel, N. (2019) 'Rethinking the Silo Environment: Let us collaborate on Digital Forensics Investigations', *AccessData*, 25 April. Available at: https://accessdata.com/blog/rethinking-the-silo-environment-lets-collaborate-on-digital-forensics-inves (Accessed: 5 April 2018)

118.AccessData (2017) *AD Lab: When Investigation Extend Past Your Organisation or Agency – Divide and Conquer*. Available at:

https://accessdata.com/assets/images/misc-content/LIT_AD_Lab-6.3-WEB_.pdf (Accessed: 8 April 2018)

119.Miller, C. (2019) ‘Making Smart Technology Decisions to Improve Case Collaboration’, *Forensic Focus*, 1 November. Available at: https://articles.forensicfocus.com/2017/11/01/making-smart-technology-decisions-to-improve-case-collaboration/ (Accessed: 9 April 2018)

120.Google (2019) *HTTPS Encryption on the Web – Google Transparency Report.* Available at: https://transparencyreport.google.com/https/overview?hl=en (Accessed: 14 April 2018)

121.Mozilla (2019) *Internet Health Report 2018.* Available at: https://d20x8vt12bnfa2.cloudfront.net/2018/ShortVersionInternetHealthReport_2018.pdf (Accessed: 15 April 2018)

122.Lofberg, M and Molin, P. (2005) *Web vs Standalone Applications – A Maintenance Application for Business Intelligence.* Master Thesis. Blekinge Institute of Technology. Available at: http://www.diva-portal.org/smash/get/diva2:828988/FULLTEXT01.pdf (Accessed: 18 April 2018)

123.Dadax (2019) *Total Number of Websites - Internet Live Stats.* Available at: https://www.internetlivestats.com/total-number-of-websites/ (Accessed: 17 April 2018)

124.Dadax (2019) *Emails sent in 1 second - Internet Live Stats.* Available at: https://www.internetlivestats.com/one-second/#email-band (Accessed: 17 April 2018)

125.Dadax (2019) *Google searches in 1 second - Internet Live Stats.* Available at: https://www.internetlivestats.com/one-second/#google-band (Accessed: 17 April 2018)

126.Dadax (2019) *Tweets sent in 1 second - Internet Live Stats.* Available at: https://www.internetlivestats.com/one-second/#tweets-band (Accessed: 17 April 2018)

127.Dadax (2019) *YouTube videos viewed in 1 second - Internet Live Stats.* Available at: https://www.internetlivestats.com/one-second/#youtube-band (Accessed: 17 April 2018)

128.Dadax (2019) *Skype calls in 1 second - Internet Live Stats.* Available at: https://www.internetlivestats.com/one-second/#skype-band (Accessed: 17 April 2018)

129.Papagelis, M. (2010) *Web App Architectures.* Available at: http://www.cs.toronto.edu/~mashiyat/csc309/Lectures/Web%20App%20Architectures.pdf (Accessed: 23 April 2018)

130.IBM (2019) *Three-tier Architectures.* Available at: https://www.ibm.com/support/knowledgecenter/en/SSEQTP_8.5.5/com.ibm.websphere.base.iseries.doc/ae/covr_3-tier.html (Accessed: 27 April 2018)

131.Gartner (2019) *Retire the Three-Tier Architecture to move toward Digital Business.* Available at: https://www.gartner.com/binaries/content/assets/events/keywords/applications/apps20i/retire_the_threetier_applica_308298.pdf (Accessed: 29 April 2018)

132.JReport (2019) *3-Tier Architecture: A Complete Overview.* Available at: https://www.jinfonet.com/resources/bi-defined/3-tier-architecture-complete-overview/ (Accessed: 29 April 2018)

133.Strickland, J. (2019) 'Anti-Forensics', *Anti-Forensics – HowStuffWorks,* 25 February. Available at: https://computer.howstuffworks.com/computer-forensic3.htm (Accessed: 1 May 2018)

134.De Beer, R. *et al.* (2014) 'Anti-Forensic Tool Use and Their Impact on Digital Forensic Investigations: A South African Perspective', *The Proceedings of the International Conference in Information Security and Digital Forensics,* Thessaloniki, Greece. Available at: https://www.researchgate.net/publication/269400823_Anti-Forensic_Tool_Use_and_Their_Impact_on_Digital_Forensic_Investigations_A_South_African_Perspective (Accessed: 6 May 2018)

135.Convery, S. (2007) 'Network Authentication, Authorisation, and Accounting Part One: Concepts, Elements, and Approaches', *The Internet Protocol Journal,* 10(1), pp. 2-12. Available at: https://www.cisco.com/c/dam/en_us/about/ac123/ac147/archived_issues/ipj_10-1/ipj_10-1.pdf (Accessed: 9 May 2018)

136.Laravel (2019) *Hashing – Laravel – The PHP Framework for Web Artisans.* Available at: https://laravel.com/docs/5.0/hashing (Accessed: 14 May 2018)

137.PHP (2019) *PHP: password_hash – Manual.* Available at: https://www.php.net/manual/en/function.password-hash.php (Accessed: 14 May 2018)

138.The Interaction Design Foundation (2019) *What is Human-Computer Interaction?.* Available at: https://www.interaction-design.org/literature/topics/human-computer-interaction (Accessed: 3 June 2018)

139.Dix, A. (2010) 'Human-Computer Interacion: A stable discipline, a nascent science, and the growth of the long tail', *Interacting with Computers,* 22(1), pp. 13-27. doi: 10.1016/j.intcom.2009.11.007

140.ISO (2018) *ISO 9241-11:2018(en) Ergonomics of Human System Interaction – Part 11: Usability: Definitions and Concepts.* Available at: https://www.iso.org/obp/ui/#iso:std:iso:9241:-11:ed-2:v1:en (Accessed: 6 July 2018)

141.ISO (2018) *ISO/IEC 25010:2011(en) Systems and software engineering — Systems and software Quality Requirements and Evaluation (SQuaRE) — System and software quality models.* Available at: https://www.iso.org/obp/ui/#iso:std:iso-iec:25010:ed-1:v1:en (Accessed: 7 July 2018)

142.Alotibi, G. (2017) *Behavioural Monitoring via Network Communications.* PhD Thesis. Plymouth University. Available at: https://pearl.plymouth.ac.uk/handle/10026.1/9964 (Accessed: 3 May 2018)

143.Clarke, N. *et al.* (2017) 'A novel privacy preserving user identification approach for network traffic', *Computers & Security* http://dx.doi.org/doi:10.1016/j.cose.2017.06.012.

144.Claffy, K.C., Braun, H.W., and Polyzos G.C. (1995) 'A parameterizable methodology for Internet traffic flow profiling', *IEEE Journal on Selected Areas of Communication,* 13 (8), pp. 1481-1494

145.H. Debar, H., Dacier, M., and Wespi, A. (1999) 'Towards a taxonomy of intrusion-detection systems', *Computer Networks,* 31, pp. 805-822

146.Mahoney, M.V. and Chan, P.K. (2001) *PHAD: Packet header anomaly detection for identifying hostile network traffic*. Available at: https://cs.fit.edu/~mmahoney/paper3.pdf (Accessed: 10 February 2018)

147.Wang, K., and Stolfo, S.J. (2004) 'Anomalous payload-based network intrusion detection', *In International Workshop on Recent Advances in Intrusion Detection,* pp. 203-222

148.Zanero, S. (2005) 'Analyzing TCP traffic patterns using self-organizing maps', *In International Conference on Image Analysis and Processing,* pp. 83-90

149.Wang, K., Cretu, G., and Stolfo, S.J. (2005) 'Anomalous payload-based worm detection and signature generation', *In International Workshop on Recent Advances in Intrusion Detection,* pp. 227-246

150.Bolzoni, D., Etalle, S., and Hartel, P. (2006) 'Poseidon: a 2-tier anomaly-based network intrusion detection system', *In Fourth IEEE International Workshop on Information Assurance (IWIA'06).*

151.Wang, X. *et al.* (2008) 'Sigfree: A signature-free buffer overflow attack blocker', *IEEE transactions on dependable and secure computing,* 7(1), pp. 65-79

152.Ahmed, I., and Lhee, K.S. (2011) 'Classification of packet contents for malware detection', *Journal in computer Virology,* 7(4), pp. 279-295

153.Al-Bataineh, A., and White, G. (2012) 'Analysis and detection of malicious data exfiltration in web traffic', *In 2012 7th International Conference on Malicious and Unwanted Software,* pp. 26-31

154.He, G. *et al.* (2014) 'A novel method to detect encrypted data exfiltration', *In 2014 Second International Conference on Advanced Cloud and Big Data,* pp. 240-246

155.Jurga, R.E., and Hulboj, M.M. (2007) *Packet sampling for network monitoring.* Available at: https://openlab-mu-internal.web.cern.ch/openlab-mu-internal/03_documents/3_technical_documents/technical_reports/2007/rj-mm_samplingreport.pdf (Accessed: 3 February 2018)

156.Cho, Y.H., Navab, S., and Mangione-Smith, W.H. (2002) 'Specialized hardware for deep network packet filtering'. *In International Conference on Field Programmable Logic and Applications,* pp. 452-461

157.Dharmapurikar, S. and Lockwood, J.W. (2006) 'Fast and scalable pattern matching for network intrusion detection systems', *IEEE Journal on Selected Areas in communications.* 24(10), pp.1781-1792

158.Sourdis, I. *et al.* (2005) 'A reconfigurable perfect-hashing scheme for packet inspection', *In International Conference on Field Programmable Logic and Applications,* pp. 644-647

159.Smith, R. *et al.* (2009) 'Evaluating GPUs for network packet signature matching', *In 2009 IEEE International Symposium on Performance Analysis of Systems and Software,* pp. 175-184

160.Liu, R.T., Huang, N.F., Kao, C.N., Chen, C.H. and Chou, C.C. (2004) 'A fast pattern-match engine for network processor-based network intrusion detection system', *In International Conference on Information Technology: Coding and Computing,* 1, pp. 97-101

161.Yu, F., Chen, Z., Diao, Y., Lakshman, T.V., and Katz, R.H. (2006) 'December. Fast and memory-efficient regular expression matching for deep packet inspection', *In Proceedings of the 2006 ACM/IEEE symposium on Architecture for networking and communications systems,* pp. 93-102

162.Smallwood, D., and Vance, A. (2011) 'Intrusion analysis with deep packet inspection: increasing efficiency of packet based investigations', *In 2011 International Conference on Cloud and Service Computing,* pp. 342-347

163.Sun, Y., Valgenti, V.C., and Kim, M.S. (2011) 'NFA-based pattern matching for deep packet inspection', *In 2011 proceedings of 20th international conference on computer communications and networks (ICCCN),* pp. 1-6

164.Claise (2008) *Specification of the IP flow information export (IPFIX) protocol for the exchange of IP traffic flow information.* Available at: https://tools.ietf.org/html/rfc5101 (Accessed: 11 February 2018)

165.Sflow (2015) *Making the network visible.* Available at https://sflow.org/ (Accessed: 12 February 2018)

166.Juniper (2011) *Juniper Flow Monitoring.* Available at: https://www.juniper.net/us/en/local/pdf/app-notes/3500204-en.pdf (Accesed: 12 February 2018)

167.Estan, C., and Varghese, G. (2003) 'New directions in traffic measurement and accounting: Focusing on the elephants, ignoring the mice', *ACM Transactions on Computer Systems (TOCS),* 21(3), pp. 270-313

168.Duffield, N., Lund, C., and Thorup, M. (2004) 'Flow sampling under hard resource constraints', *Proceedings of the joint international conference on Measurement and modeling of computer systems,* pp. 85-96. doi: 10.1145/1005686.1005699

169.Duffield, N., Lund, C., and Thorup, M. (2005) 'Learn more, sample less: control of volume and variance in network measurement', *IEEE Transactions of Information Theory*, 51 (5), pp. 1756-1775. doi: 10.1109/TIT.2005.846400

170.Androulidakis, G., Chatzigiannakis, V., and Papavassiliou, S. (2007) 'Using selective sampling for the support of scalable and efficient network anomaly detection' *2007 IEEE Globecom workshops,* pp. 1-5

171.Canini, M., Fay, D., Miller, D.J., Moore, A.W., and Bolla, R. (2009) 'Per flow packet sampling for high-speed network monitoring', *First international Communication systems and networks and workshops,* pp. 1-10. doi: 10.1109/COMSNETS.2009.4808888

172.Kim, M.S., Kong, H.J., Hong, S.C., Chung, S.H., and Hong, J.W. (2004) 'A flow-based method for abnormal network traffic detection', *In 2004 IEEE/IFIP network operations and management symposium,* 1, pp. 599-612

173.Pao, T.L., and Wang, P.W. (2004) 'Netflow based intrusion detection system' *In IEEE International Conference on Networking, Sensing and Control*, 2, pp. 731-736

174.Crotti, M., Gringoli, F., Pelosato, P., and Salgarelli, L. (2006) 'A statistical approach to IP-level classification of network traffic', *In 2006 IEEE International Conference on Communications,* 1, pp. 170-176

175.Song, J., Takakura, H., Okabe, Y., Eto, M., Inoue, D., and Nakao, K. (2011) 'Statistical analysis of honeypot data and building of Kyoto 2006+ dataset for NIDS evaluation', *In Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security,* pp. 29-36

176.Muraleedharan, N., Parmar, A., and Kumar, M. (2010) 'A flow based anomaly detection system using chi-square technique', *In 2010 IEEE 2nd international Advance computing conference (IACC),* pp. 285-289

177.Braga, R., de Souza Mota, E., and Passito, A. (2010) 'Lightweight DDoS flooding attack detection using NOX/OpenFlow', *In 2010 IEEE 35th Conference on Local Computer Networks (LCN)*, 10, pp. 408-415

178.Winter, P., Hermann, E., and Zeilinger, M. (2011) 'Inductive intrusion detection in flow-based network data using one-class support vector machines', *In 2011 4th IFIP international conference on new technologies, mobility and security,* pp. 1-5

179.Tegeler, F., Fu, X., Vigna, G., and Kruegel, C. (2012) 'Botfinder: Finding bots in network traffic without deep packet inspection', *In Proceedings of the 8th international conference on Emerging networking experiments and technologies,* pp. 349-360

180.Jadidi, Z., Muthukkumarasamy, V., Sithirasenan, E., and Sheikhan, M. (2013) 'Flow-based anomaly detection using neural network optimized with GSA algorithm', *In 2013 IEEE 33rd International Conference on Distributed Computing Systems Workshops,* pp. 76-81

181.Hofstede, R., Bartoš, V., Sperotto, A., and Pras, A. (2013) 'Towards real-time intrusion detection for NetFlow and IPFIX', *In Proceedings of the 9th International Conference on Network and Service Management (CNSM 2013),* pp. 227-234

182.Alexa (2017) *The top 500 sites on the web.* Available at: https://www.alexa.com/topsites (Accessed: 2 June 2017)

183.W3Techs (2018) *Usage Statistics of IPv6 for Websites.* Available at: https://w3techs.com/technologies/details/ce-ipv6/all/all (Accessed: 3 June 2018)

184.SimilarWeb (2018) *Bbc.co.uk Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/bbc.co.uk (Accessed: 4 June 2018)

185.SimilarWeb (2018) *Dropbox.com Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/dropbox.com (Accessed: 4 June 2018)

186.Dropbox (2018) *Tour - Dropbox Business.* Available at: https://www.dropbox.com/business/tour (Accessed: 4 June 2018)

187.SimilarWeb (2018) *Facebook.com Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/facebook.com (Accessed: 4 June 2018)

188.SimilarWeb (2018) *Google.co.uk Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/google.co.uk (Accessed: 4 June 2018)

189.Alexa (2018) *Alexa - Twitter Competitve Analysis, Marketing Mix and Traffic.* Available at: https://www.alexa.com/siteinfo/twitter.com (Accessed: 4 June 2018)

190.SimilarWeb (2018) *Twitter.com Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/twitter.com (Accessed: 4 June 2018)

191.Alexa (2018) *Alexa - Wikipedia Competitve Analysis, Marketing Mix and Traffic.* Available at: https://www.alexa.com/siteinfo/wikipedia.org (Accessed: 5 June 2018)

192.SimilarWeb (2018) *Wikipedia.org Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/wikipedia.org (Accessed: 5 June 2018)

193.Alexa (2018) *Alexa - YouTube Competitve Analysis, Marketing Mix and Traffic.* Available at: https://www.alexa.com/siteinfo/youtube.com (Accessed: 5 June 2018)

194.SimilarWeb (2018) *YouTube.com Analytics - Market share stats and Traffic ranking.* Available at: https://www.similarweb.com/website/youtube.com (Accessed: 5 June 2018)

195.W3C (2018) *Accessibility, Usability and Inclusion - Web Accessibility Initiative (W3C).* Available at: https://www.w3.org/WAI/fundamentals/accessibility-usability-inclusion/ (Accessed: 7 September 2018)

196.Ashraf, I. (2014) 'An overview of service models of cloud computing' *International Journal of Multidisciplinary and Current Research*, *2*(1), pp.779-783.

Appendix A: Publications

1. **Forensic Investigation of Network Traffic: A Study into the Derivation of Application-Level features from Network-Level Metadata**
   Li F, Clarke NL, Alotibi G, Joy D 6th Annual International Conference on ICT: Big data, Could and Security (ICT-BDCS 2015), 27-28 July, ISSN: 2382-5669, pp68-73, 2015, DOI: 10.5176/2382-5669_ICT-BDCS15.28
2. **A User-oriented Network Forensic Analyser: The Design of a High-Level Protocol Analyser**
   Joy D, Li F, Clarke NL, Furnell SM Proceedings of the 12th Australian Digital Forensics Conference, 1-3 December, ECU Joondalup Campus, Perth, Western Australia, pp 84-93, ISBN 978-0-7298-0719-7, 2014, DOI: 10.4225/75/57b3e511fb87f

Appendix B: Script for Online User Activity Extraction

BBC

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_bbc_version_1_fn(dbpath,user_num,case_name_fldr,primary_dbpath )


i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
bbc=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,'','','org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,'','','org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


 %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR
 %   THIS USER INTO THE
 %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
 %   THE CHARTS

count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),'','','org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=','''',user_num,'''' );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);



start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

        query_base = 'select count (*) from IP_logs where S_IP like "212.58.244.%" or D_IP like
"212.58.244.%" or S_IP like "212.58.246.%" or D_IP like "212.58.246.%"';
        limit_base = exec(conn, query_base);
        limit_base = fetch(limit_base);
        limit = cell2mat(limit_base.Data);
if (limit>0)
```

```
  while (i<=row)

%        if (start_a>count)
%          break;
%        end
       query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "212.58.244.%" or D_IP like "212.58.244.%" or S_IP like
"212.58.246.%" or D_IP like "212.58.246.%")');
       limit1_base = exec(conn, query_base);
       limit1_base = fetch(limit1_base);
       limit1 = cell2mat(limit1_base.Data);
       limit2= limit2+limit1;

      query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "212.58.244.%" or D_IP like "212.58.244.%" or S_IP like
"212.58.246.%" or D_IP like "212.58.246.%")');
      bbc_rows = exec(conn, query_base1);
      bbc_rows = fetch(bbc_rows);

     if (limit1 ~= 0)
      for j=k+1:limit2
        if (limit2 > limit)
          break;
        end
        bbc(x,:) = bbc_rows.Data(p,:);
        p=p+1;
        if (x==q)
          user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\');
          if exist(user_service_folder_loc,'dir')
            user_service_matfile_loc = strcat(user_service_folder_loc, 'BBC', num2str(matfile_count));
            save(user_service_matfile_loc,'bbc');
          else
            mkdir(user_service_folder_loc);
            user_service_matfile_loc = strcat(user_service_folder_loc, 'BBC', num2str(matfile_count));
            save(user_service_matfile_loc,'bbc');
          end
          matfile_count=matfile_count+1;
          clear bbc;
          x=0;

        elseif (x==limit-((matfile_count-1)*q))
            user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\');
          if exist(user_service_folder_loc,'dir')
            user_service_matfile_loc = strcat(user_service_folder_loc, 'BBC', num2str(matfile_count));
            save(user_service_matfile_loc,'bbc');
          else
            mkdir(user_service_folder_loc);
            user_service_matfile_loc = strcat(user_service_folder_loc, 'BBC', num2str(matfile_count));
            save(user_service_matfile_loc,'bbc');
          end
          clear bbc;
        end
%          if (p>limit1)
%            break;
%          end
        x=x+1;
```

```
            end
          end
            k=limit2;
            p=1;
            start_a = start_b+1;
            start_b = start_b+10000;
            i=i+1;

    end
%     stat(user_num+1,9)={'1'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');

    bbc_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);
%     bbc_https_interactions_final(x,matfile_count,user_num);
% else
%     stat(user_num+1,9)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
bbc_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)


iii=1;

bbchttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC',num2str(i),'.mat'), 'bbc');
        for ii=1:500000
            if (strcmp(cellstruct.bbc(ii,3),'80')==1 || strcmp(cellstruct.bbc(ii,5),'80')==1)
                  bbchttpinteractions(iii,:)=cellstruct.bbc(ii,:);
                  iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC',num2str(i),'.mat'), 'bbc');
```

```
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.bbc(ii,3),'80')==1 || strcmp(cellstruct.bbc(ii,5),'80')==1)
                    bbchttpinteractions(iii,:)=cellstruct.bbc(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.bbc(ii,3),'80')==1 || strcmp(cellstruct.bbc(ii,5),'80')==1)
                    bbchttpinteractions(iii,:)=cellstruct.bbc(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

for ii=1:(iii-1)
    bbchttpinteractions(ii,8)={'0'};
end


%               FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(bbchttpinteractions,1);
if (iii>1)
for n=1:size(bbchttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(bbchttpinteractions(n,8),'0')==1)
    tag = bbchttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = bbchttpinteractions{n,1};
        interaction_temp{counter1, 3} = bbchttpinteractions{n,2};
        interaction_temp{counter1, 4} = bbchttpinteractions{n,3};
        interaction_temp{counter1, 5} = bbchttpinteractions{n,4};
        interaction_temp{counter1, 6} = bbchttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
       interaction_temp{counter1, 8} = 0; %data volume

        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = bbchttpinteractions{n,4};
        interaction_temp{counter2, 4} = bbchttpinteractions{n,5};
        interaction_temp{counter2, 5} = bbchttpinteractions{n,2};
        interaction_temp{counter2, 6} = bbchttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume

        flag1=1;
        counter1=counter1+2;
        counter2=counter2+2;

    end
    if (flag1>0)
     for j=1:size(bbchttpinteractions,1)
           s_ip = bbchttpinteractions{j,2};
           s_port = bbchttpinteractions{j,3};
           d_ip = bbchttpinteractions{j,4};
           d_port = bbchttpinteractions{j,5};
           tag = bbchttpinteractions{j,7};
      if (strcmp(bbchttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


        interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
        interaction_temp{counter1-2, 2} = bbchttpinteractions{j,1}; %end time
        interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(bbchttpinteractions{j,6}); %total data volume
        bbchttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
        if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
           if (counter3==0)
              interaction_temp{counter2-2, 1} = bbchttpinteractions{j,1};
              counter3=1;
           end
        interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
        interaction_temp{counter2-2, 2} = bbchttpinteractions{j,1}; %end time
        interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(bbchttpinteractions{j,6}); %total data volume
        bbchttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
      end
     end
     counter3=0;

    end



end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\');
   mkdir(dir_path);
   save (strcat(dir_path,'BBC HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;    % If HTTP traffic does exist
   bbc_http_interactions_final(x, matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else          %If HTTP traffic doesn't exist
   existsornot = 0;
   bbc_http_interactions_final(x, matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
bbc_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath)
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
   interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
   if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
      interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
   else
   interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
   end
   interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
   interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
   interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
   interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
   interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
   interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
   interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
   interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
   ii=ii+1;
%     i=i+2;
end
   save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\BBC HTTP Interactions - Final Table.mat'), 'interaction_final');

bbc_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,pri
mary_dbpath);
else          %No HTTP traffic exists
```

```
    interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

bbc_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,pri
mary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
bbc_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,pri
mary_dbpath )
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       ------------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC',num2str(i),'.mat'), 'bbc');
        for ii=1:500000
            if (strcmp(cellstruct.bbc(ii,3),'443')==1 || strcmp(cellstruct.bbc(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.bbc(ii,:);
                    iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC',num2str(i),'.mat'), 'bbc');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.bbc(ii,3),'443')==1 || strcmp(cellstruct.bbc(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.bbc(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.bbc(ii,3),'443')==1 || strcmp(cellstruct.bbc(ii,5),'443')==1)
                httpsinteractions(iii,:)=cellstruct.bbc(ii,:);
                iii=iii+1;
            end
        end
    end
  end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%         for ii=1:500000
%             if (bbc(ii,7)== 'S')
% %                 facebookinteractions()
%             end
%         end
%     else
%         for ii=1:(x-1)
%             if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                 httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                 iii=iii+1;
%             end
%         end
% %         for ii=1:(iii-1)
% %             httpsinteractions(ii,8)={'0'};
% %         end
%
%     end
% end




%                     FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                     ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%             interaction_temp{counter1, 7} = 0;
% %             interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%             interaction_temp{counter1, 8} = 0;
%             interaction_temp{counter1, 9} = 0;
% %             interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%             interaction_temp{counter1, 10} = 0;
```

```
    if (timestamp2-timestamp1 < 6.9444e-06)

            s_ip = httpsinteractions{n,2};
            s_port = httpsinteractions{n,3};
            d_ip = httpsinteractions{n,4};
            d_port = httpsinteractions{n,5};
            tag = httpsinteractions{n,7};

%           if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
              fprintf('yes');
%                if isempty (interaction_temp{counter1, 1}==1)
%                interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%                end
              interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

              if (strcmp(counter_increment_status,'yes')==1)
                 if (strcmp(s_port,'443')==1)
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                 else
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                 end
              end

              if (isempty (interaction_temp{counter1, 7})==1)
              interaction_temp{counter1, 7} = 0;
              end
% %              interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
              if (isempty (interaction_temp{counter1, 8})==1)
              interaction_temp{counter1, 8} = 0;
              end
              if (isempty (interaction_temp{counter1, 9})==1)
              interaction_temp{counter1, 9} = 0;
              end
% %              interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
              if (isempty (interaction_temp{counter1, 10})==1)
              interaction_temp{counter1, 10} = 0;
              end

              if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%                interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%                interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%                interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%                interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                 if(strcmp(s_port,'443')==1)
```

```
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (User to Service)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                        if (n == (size(httpsinteractions,1)-1))
                            interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                            interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                        end
                    else
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                        if (n == (size(httpsinteractions,1)-1))
                            interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                            interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                        end
                    end
                end

                if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%                   interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%                   interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                    if(strcmp(s_port,'443')==1)
                                            fprintf('\na');
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                        if (n == (size(httpsinteractions,1)-1))
                            interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                            interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                        end
                    else
                                            fprintf('\nb');
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                        if (n == (size(httpsinteractions,1)-1))
                            interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                            interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                        end
                    end
                end
%                   flag1=1;

              else
                    if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                                            fprintf('\na');
```

```
               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (Service to User)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
               counter1=counter1+1;
               counter_increment_status='yes';
               elseif (strcmp(counter_increment_status,'no')==1)
                                    fprintf('\nb');
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
               counter1=counter1+1;
               counter_increment_status='yes';
               end

          end

   else
      if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                          fprintf('\na');
      interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of Packets
(Service to User)
      interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
      counter1=counter1+1;
      counter_increment_status='yes';
      elseif (strcmp(counter_increment_status,'no')==1)
                          fprintf('\nb');
      interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of Packets
(User to Service)
      interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
      counter1=counter1+1;
      counter_increment_status='yes';
      end

   end
%  end
end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\');
   mkdir(dir_path);
   save (strcat(dir_path,'BBC HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
   if (existsornot==1)     %Both HTTP and HTTPS traffic exist
      http_https_interactions_combined = [interaction_final;interaction_temp];
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'BBC TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      bbc_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%        statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%        finalstat = statload1.finalstat;
%        finalstat(user_num+1,9)={'1'};
%        save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   else                  %Only HTTPS traffic exists; no HTTP
      http_https_interactions_combined = interaction_temp;
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
```

```
        save (strcat(dir_path,'BBC TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
        bbc_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%           statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%           finalstat = statload1.finalstat;
%           finalstat(user_num+1,9)={'1'};
%           save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    end
else                    % No HTTPS traffic exists
%           http_https_interactions_combined = interaction_final;
%           tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%           tcp_interactions_final = interaction_final;
        if (existsornot==1)     % No HTTPS but HTTP exists
            dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\News\BBC\');
            mkdir(dir_path);
            http_https_interactions_combined = interaction_final;
            tcp_interactions_final = sortrows (http_https_interactions_combined,1);
            save (strcat(dir_path,'BBC TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
            bbc_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%               save (strcat(dir_path,'BBC TCP Interactions - Final Table.mat'), 'interaction_final');

%               statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%               finalstat = statload1.finalstat;
%               finalstat(user_num+1,9)={'1'};
%               save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ bbc_user_activity_final ] =
bbc_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here



        %   CODE FOR FINDING OUT THE BBC USER ACTIVITIES BASED ON THE EXTRACTED USER
INTERACTIONS
        %   ------------------------------------------------------------------------------------

j=1;
k=1;
m=1;
count=0;
count1=0;
bbc_user_activity_temp = cell(1,10);
bbc_user_activity_temp_1 = cell(1,18);
bbc_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code - All services - Users 10 to 20\User_services_All_Interactions1\user19\News\BBC\BBC TCP Interactions - Final Table.mat', 'tcp_interactions_final');
```

```
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\News\BBC\BBC TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
   if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 || cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
      bbc_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
      j=j+1;
   end
end

for i=1:size(bbc_user_activity_temp,1)
   usr_to_srvr_avg_pkt_size =
ceil(cell2mat(bbc_user_activity_temp(i,8))/cell2mat(bbc_user_activity_temp(i,7)));
   bbc_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user to BBC
server)

   srvr_to_usr_avg_pkt_size =
ceil(cell2mat(bbc_user_activity_temp(i,10))/cell2mat(bbc_user_activity_temp(i,9)));
   bbc_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size (BBC server
to user)

   t1 = datevec(cell2mat(bbc_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating the time
duration of each interaction
   t2 = datevec(cell2mat(bbc_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
   duration = etime(t2,t1);
   bbc_user_activity_temp(i,13) = {duration};      %Column 13 --> Duration
   if (cell2mat(bbc_user_activity_temp(i,7))>=10 && cell2mat(bbc_user_activity_temp(i,9))>=10)
%        if (cell2mat(bbc_user_activity_temp(i,10)) >= 4*cell2mat(bbc_user_activity_temp(i,8)))
         bbc_user_activity_temp_1(k,1:13) = bbc_user_activity_temp(i,1:13);
         k=k+1;
%        end
   end
end


j=1;
if (size(bbc_user_activity_temp_1,1)>=1)
   for i=1:size(bbc_user_activity_temp_1,1)
      %   CHECKING FOR WATCHING VIDEO OR LISTENING AUDIO
      if (cell2mat(bbc_user_activity_temp_1 (i,12))>=1000 )%&& cell2mat(wikipedia_user_activity_temp_1
(k,12))<=1400)
        if (cell2mat(bbc_user_activity_temp_1 (i,13))>=20) %Assuming if duration is greater than 20s, it might be
watching video/listening to audio
           if (cell2mat(bbc_user_activity_temp_1(i,7))>=30 && cell2mat(bbc_user_activity_temp_1(i,9))>=30)
              if (cell2mat(bbc_user_activity_temp_1(i,7))/cell2mat(bbc_user_activity_temp_1(i,9))>=0.5 &&
cell2mat(bbc_user_activity_temp_1(i,7))/cell2mat(bbc_user_activity_temp_1(i,9))<=1.3)
                 bbc_user_activity_temp_1(i,14) = {'BBC-Watching Video or Listening to Audio'};

                    bbc_user_activity_temp_1{i,15} = 'BBC';
                    bbc_user_activity_temp_1{i,16} = bbc_user_activity_temp_1{i,14};
                    stime = bbc_user_activity_temp_1{i,1};
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = bbc_user_activity_temp_1{i,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    bbc_user_activity_temp_1{i,17} = stime_formatted;
                    bbc_user_activity_temp_1{i,18} = etime_formatted;
                    bbc_user_activity_final(j,1:18) = bbc_user_activity_temp_1(i,1:18);
                    j=j+1;
                end
            end
        end
       end

       %   CHECKING FOR PAGE NAVIGATION
       if (cell2mat(bbc_user_activity_temp_1 (i,12))>=500 )%&& cell2mat(wikipedia_user_activity_temp_1
(k,12))<=1400)
        if (cell2mat(bbc_user_activity_temp_1 (i,13))<20) %Assuming if duration is less than 20s, it might be
page navigation
           if (cell2mat(bbc_user_activity_temp_1(i,7))>=25 && cell2mat(bbc_user_activity_temp_1(i,9))>=25)
              if (cell2mat(bbc_user_activity_temp_1(i,7))/cell2mat(bbc_user_activity_temp_1(i,9))>=0.5 &&
cell2mat(bbc_user_activity_temp_1(i,7))/cell2mat(bbc_user_activity_temp_1(i,9))<=1.3)
                 bbc_user_activity_temp_1(i,14) = {'BBC-Page Navigation'};

                    bbc_user_activity_temp_1{i,15} = 'BBC';
                    bbc_user_activity_temp_1{i,16} = bbc_user_activity_temp_1{i,14};
                    stime = bbc_user_activity_temp_1{i,1};
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = bbc_user_activity_temp_1{i,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    bbc_user_activity_temp_1{i,17} = stime_formatted;
                    bbc_user_activity_temp_1{i,18} = etime_formatted;

                    bbc_user_activity_final(j,1:18) = bbc_user_activity_temp_1(i,1:18);
                    j=j+1;
                end
            end
        end
       end
   end
%     bbc_user_activity_final(i,1:18) = bbc_user_activity_temp_1(i,1:18);
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Ne
ws\BBC\');
   mkdir(dir_path);
   save (strcat(dir_path,'BBC_User_Activities.mat'), 'bbc_user_activity_final');


        %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
   matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
   matlab_table1 = cell2table(bbc_user_activity_final,'VariableNames',matlab_table_col_names);

       %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
   conn_db_1 = database(primary_dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
   sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
   exec(conn_db_1,sql_query_1);
```

```
    insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Dropbox

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_dropbox_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here



%               CODE FOR EXTRACTING TRAFFIC - SERVICE --> DROPBOX
%               ------------------------------------------------


i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
dropbox=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,'','','org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,'','','org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


    %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
    %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
    %   THE CHARTS
    count1=num2str(count);
    conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),'','','org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
    count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=','''',user_num,'''' );
    User_tot_num_pkts = exec(conn_1, count_sql_query_1);
    User_tot_num_pkts = fetch(User_tot_num_pkts);


start_a=1;
start_b=10000;
row=count/start_b;
```

```
row=ceil(row);

            query_base = 'select count (*) from IP_logs where S_IP like "108.160.172.%" or D_IP like
"108.160.172.%" or S_IP like "108.160.165.%" or D_IP like "108.160.165.%"';
            limit_base = exec(conn, query_base);
            limit_base = fetch(limit_base);
            limit = cell2mat(limit_base.Data);
if (limit>0)
    while (i<=row)

%            if (start_a>count)
%                break;
%            end
             query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "108.160.172.%" or D_IP like ''108.160.172.%" or S_IP like
"108.160.165.%" or D_IP like "108.160.165.%")');
             limit1_base = exec(conn, query_base);
             limit1_base = fetch(limit1_base);
             limit1 = cell2mat(limit1_base.Data);
             limit2= limit2+limit1;

           query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "108.160.172.%" or D_IP like "108.160.172.%" or S_IP like
"108.160.165.%" or D_IP like "108.160.165.%")');
           dropbox_rows = exec(conn, query_base1);
           dropbox_rows = fetch(dropbox_rows);

          if (limit1 ~= 0)
           for j=k+1:limit2
               if (limit2 > limit)
                   break;
               end
               dropbox(x,:) = dropbox_rows.Data(p,:);
               p=p+1;
               if (x==q)
                   user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Dropbox', num2str(matfile_count));
                       save(user_service_matfile_loc,'dropbox');
                   else
                       mkdir(user_service_folder_loc);
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Dropbox', num2str(matfile_count));
                       save(user_service_matfile_loc,'dropbox');
                   end
                   matfile_count=matfile_count+1;
                   clear dropbox;
                   x=0;

               elseif (x==limit-((matfile_count-1)*q))
                       user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Dropbox', num2str(matfile_count));
                       save(user_service_matfile_loc,'dropbox');
                   else
                       mkdir(user_service_folder_loc);
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Dropbox', num2str(matfile_count));
```

```
                save(user_service_matfile_loc,'dropbox');
              end
              clear dropbox;
            end
%             if (p>limit1)
%                break;
%             end
            x=x+1;
          end
         end
          k=limit2;
          p=1;
          start_a = start_b+1;
          start_b = start_b+10000;
          i=i+1;

   end
   %stat(user_num+1,10)={'1'};
   %save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
   dropbox_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);

% else
   %stat(user_num+1,10)={'0'};
   %save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
dropbox_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       -------------------------------------------------------------------------

iii=1;

dropboxhttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
   for i=1:(matfile_count-1)
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox',num2str(i),'.mat'), 'dropbox');
```

```
      for ii=1:500000
         if (strcmp(cellstruct.dropbox(ii,3),'80')==1 || strcmp(cellstruct.dropbox(ii,5),'80')==1)
               dropboxhttpinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
         end
      end
   end
else
   for i=1:matfile_count
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox',num2str(i),'.mat'), 'dropbox');
      if (i<matfile_count)
         for ii=1:500000
            if (strcmp(cellstruct.dropbox(ii,3),'80')==1 || strcmp(cellstruct.dropbox(ii,5),'80')==1)
               dropboxhttpinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
            end
         end
      else
         for ii=1:(x-1)
            if (strcmp(cellstruct.dropbox(ii,3),'80')==1 || strcmp(cellstruct.dropbox(ii,5),'80')==1)
               dropboxhttpinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
            end
         end
      end
   end
end

for ii=1:(iii-1)
   dropboxhttpinteractions(ii,8)={'0'};
end



%                FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(dropboxhttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
for n=1:size(dropboxhttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)
```

```
if (strcmp(dropboxhttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
    tag = dropboxhttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = dropboxhttpinteractions{n,1};
        interaction_temp{counter1, 3} = dropboxhttpinteractions{n,2};
        interaction_temp{counter1, 4} = dropboxhttpinteractions{n,3};
        interaction_temp{counter1, 5} = dropboxhttpinteractions{n,4};
        interaction_temp{counter1, 6} = dropboxhttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
%         interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
        interaction_temp{counter1, 8} = 0; %data volume

%         interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = dropboxhttpinteractions{n,4};
        interaction_temp{counter2, 4} = dropboxhttpinteractions{n,5};
        interaction_temp{counter2, 5} = dropboxhttpinteractions{n,2};
        interaction_temp{counter2, 6} = dropboxhttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume

        flag1=1;
        counter1=counter1+2;
        counter2=counter2+2;

    end

    if (flag1>0)
     for j=1:size(dropboxhttpinteractions,1)
            s_ip = dropboxhttpinteractions{j,2};
            s_port = dropboxhttpinteractions{j,3};
            d_ip = dropboxhttpinteractions{j,4};
            d_port = dropboxhttpinteractions{j,5};
            tag = dropboxhttpinteractions{j,7};
      if (strcmp(dropboxhttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


        interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
        interaction_temp{counter1-2, 2} = dropboxhttpinteractions{j,1}; %end time
        interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(dropboxhttpinteractions{j,6}); %total data volume
        dropboxhttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
%      end
%
```

```
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
        if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
            if (counter3==0)
                interaction_temp{counter2-2, 1} = dropboxhttpinteractions{j,1};
                counter3=1;
            end
        interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
        interaction_temp{counter2-2, 2} = dropboxhttpinteractions{j,1}; %end time
        interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(dropboxhttpinteractions{j,6}); %total data volume
        dropboxhttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
      end
     end
     counter3=0;

   end




end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
   mkdir(dir_path);
   save (strcat(dir_path,'Dropbox HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;    % If HTTP traffic does exist
   dropbox_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else          %If HTTP traffic doesn't exist
   existsornot = 0;
   dropbox_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
dropbox_http_interactions_final(x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath)
```

```
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
   interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
   if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
      interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
   else
   interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
   end
   interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
   interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
   interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
   interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
   interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
   interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
   interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
   interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
   ii=ii+1;
%     i=i+2;
end
   save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\On
line Storage\Dropbox\Dropbox HTTP Interactions - Final Table.mat'), 'interaction_final');

dropbox_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpat
h,primary_dbpath);
else            %No HTTP traffic exists
   interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

dropbox_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpat
h,primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
dropbox_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpat
h,primary_dbpath)
%UNTITLED6 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%        ------------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
   for i=1:(matfile_count-1)
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox',num2str(i),'.mat'), 'dropbox');
      for ii=1:500000
         if (strcmp(cellstruct.dropbox(ii,3),'443')==1 || strcmp(cellstruct.dropbox(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
         end
      end
   end
else
   for i=1:matfile_count
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox',num2str(i),'.mat'), 'dropbox');
      if (i<matfile_count)
         for ii=1:500000
            if (strcmp(cellstruct.dropbox(ii,3),'443')==1 || strcmp(cellstruct.dropbox(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
            end
         end
      else
         for ii=1:(x-1)
            if (strcmp(cellstruct.dropbox(ii,3),'443')==1 || strcmp(cellstruct.dropbox(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.dropbox(ii,:);
               iii=iii+1;
            end
         end
      end
   end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%           for ii=1:500000
%              if (bbc(ii,7)== 'S')
% %                    facebookinteractions()
%              end
%           end
%     else
%           for ii=1:(x-1)
%               if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                   httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                   iii=iii+1;
%               end
%           end
% %           for ii=1:(iii-1)
% %                   httpsinteractions(ii,8)={'0'};
% %           end
%
%     end
% end




%                         FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                         ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%               interaction_temp{counter1, 7} = 0;
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%               interaction_temp{counter1, 8} = 0;
%               interaction_temp{counter1, 9} = 0;
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%               interaction_temp{counter1, 10} = 0;


    if (timestamp2-timestamp1 < 6.9444e-06)

          s_ip = httpsinteractions{n,2};
          s_port = httpsinteractions{n,3};
          d_ip = httpsinteractions{n,4};
          d_port = httpsinteractions{n,5};
          tag = httpsinteractions{n,7};

%           if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
              fprintf('yes');
%               if isempty (interaction_temp{counter1, 1}==1)
%               interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%               end
              interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

              if (strcmp(counter_increment_status,'yes')==1)
                  if (strcmp(s_port,'443')==1)
                      interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                      interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
```

```
                interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                counter_increment_status='no';
            else
                interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                counter_increment_status='no';
            end
        end

        if (isempty (interaction_temp{counter1, 7})==1)
        interaction_temp{counter1, 7} = 0;
        end
% %          interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
        if (isempty (interaction_temp{counter1, 8})==1)
        interaction_temp{counter1, 8} = 0;
        end
        if (isempty (interaction_temp{counter1, 9})==1)
        interaction_temp{counter1, 9} = 0;
        end
% %          interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
        if (isempty (interaction_temp{counter1, 10})==1)
        interaction_temp{counter1, 10} = 0;
        end

        if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%             interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%             interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%             interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%             interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
            if(strcmp(s_port,'443')==1)

            interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;              % Number of
Packets (User to Service)
            interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                if (n == (size(httpsinteractions,1)-1))
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;              % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                end
            else
            interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;              % Number of
Packets (User to Service)
            interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                if (n == (size(httpsinteractions,1)-1))
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;              % Number of
Packets (Service to User)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                end
            end
        end
```

```
           if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%             interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%             interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
              if(strcmp(s_port,'443')==1)
                                  fprintf('\na');
              interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
              interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                 if (n == (size(httpsinteractions,1)-1))
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                 end
              else
                                  fprintf('\nb');
              interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
              interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                 if (n == (size(httpsinteractions,1)-1))
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                 end
              end
           end
%             flag1=1;

          else
              if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                                  fprintf('\na');
              interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
              interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
              counter1=counter1+1;
              counter_increment_status='yes';
              elseif (strcmp(counter_increment_status,'no')==1)
                                  fprintf('\nb');
              interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
              interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
              counter1=counter1+1;
              counter_increment_status='yes';
              end

          end

   else
      if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                         fprintf('\na');
      interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of Packets
(Service to User)
```

```
      interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
      counter1=counter1+1;
      counter_increment_status='yes';
      elseif (strcmp(counter_increment_status,'no')==1)
                         fprintf('\nb');
      interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;              % Number of Packets
(User to Service)
      interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
      counter1=counter1+1;
      counter_increment_status='yes';
      end

   end
%  end
end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
   mkdir(dir_path);
   save (strcat(dir_path,'Dropbox HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
   if (existsornot==1)     %Both HTTP and HTTPS traffic exist
      http_https_interactions_combined = [interaction_final;interaction_temp];
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'Dropbox TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      dropbox_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

      %%statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
      %%finalstat = statload1.finalstat;
      %%finalstat(user_num+1,10)={'1'};
      %%save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   else                  %Only HTTPS traffic exists; no HTTP
%         http_https_interactions_combined = interaction_temp;
%         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      tcp_interactions_final = interaction_temp;
      save (strcat(dir_path,'Dropbox TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      dropbox_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
      %%statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
      %%finalstat = statload1.finalstat;
      %%finalstat(user_num+1,10)={'1'};
      %%save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   end
else                      % No HTTPS traffic exists
%         http_https_interactions_combined = interaction_final;
%         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%         tcp_interactions_final = interaction_final;
      if (existsornot==1)     % No HTTPS but HTTP exists
         dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
         mkdir(dir_path);
%           tcp_interactions_final = interaction_final;
         http_https_interactions_combined = interaction_final;
         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
         save (strcat(dir_path,'Dropbox TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
         dropbox_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
         %%statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
         %%finalstat = statload1.finalstat;
         %%finalstat(user_num+1,10)={'1'};
         %%save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
      end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ dropbox_user_activity_final ] =
dropbox_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED6 Summary of this function goes here
%   Detailed explanation goes here


      %   CODE FOR FINDING OUT THE DROPBOX USER ACTIVITIES BASED ON THE EXTRACTED
USER INTERACTIONS
      %   -----------------------------------------------------------------------------------------

j=1;
k=1;
count=0;
dropbox_user_activity_temp = cell(1,10);
dropbox_user_activity_temp_1 = cell(1,18);
dropbox_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user20\Online
Encyclopedia\Wikipedia\Wikipedia TCP Interactions - Final Table.mat', 'tcp_interactions_final');
% cellstructb = load('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions1\user4\Online
Storage\Dropbox\Dropbox TCP Interactions - Final Table', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Storage\Dropbox\Dropbox TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
   if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 && cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
      dropbox_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
      j=j+1;
   end
end

for i=1:size(dropbox_user_activity_temp,1)

      usr_to_srvr_avg_pkt_size =
ceil(cell2mat(dropbox_user_activity_temp(i,8))/cell2mat(dropbox_user_activity_temp(i,7)));
      dropbox_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user
to Dropbox server - upload)

      srvr_to_usr_avg_pkt_size =
ceil(cell2mat(dropbox_user_activity_temp(i,10))/cell2mat(dropbox_user_activity_temp(i,9)));
      dropbox_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size
(Dropbox server to user - download)

      t1 = datevec(cell2mat(dropbox_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating
the time duration of each interaction
      t2 = datevec(cell2mat(dropbox_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
      duration = etime(t2,t1);
```

```
        dropbox_user_activity_temp(i,13) = {duration};

    if (cell2mat(dropbox_user_activity_temp(i,7))>=40 && cell2mat(dropbox_user_activity_temp(i,9))>=40)
            dropbox_user_activity_temp_1(k,1:13) = dropbox_user_activity_temp(i,1:13);
            k=k+1;
    end
end



k=1;
j=1;
if (size(dropbox_user_activity_temp_1,1)>=1)
    for k=1:size(dropbox_user_activity_temp_1,1)
        %   FOR FILE DOWNLOAD
        if (cell2mat(dropbox_user_activity_temp_1 (k,12))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
             if (cell2mat(dropbox_user_activity_temp_1 (k,11))>=45 && cell2mat(dropbox_user_activity_temp_1
(k,11))<=65)
                 if (cell2mat(dropbox_user_activity_temp_1(k,7))>=30 &&
cell2mat(dropbox_user_activity_temp_1(k,9))>=30)
                     if
(cell2mat(dropbox_user_activity_temp_1(k,7))/cell2mat(dropbox_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(dropbox_user_activity_temp_1(k,7))/cell2mat(dropbox_user_activity_temp_1(k,9))<=1.3)
                        dropbox_user_activity_temp_1(k,14) = {'Dropbox-File Download'};

                          dropbox_user_activity_temp_1{k,15} = 'Dropbox';
                          dropbox_user_activity_temp_1{k,16} = dropbox_user_activity_temp_1{k,14};
                          stime = dropbox_user_activity_temp_1{k,1};
                          stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                          endtime = dropbox_user_activity_temp_1{k,2};
                          etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                          dropbox_user_activity_temp_1{k,17} = stime_formatted;
                          dropbox_user_activity_temp_1{k,18} = etime_formatted;

                          dropbox_user_activity_final(j,1:18) = dropbox_user_activity_temp_1(k,1:18);
                          j=j+1;
                     end
                 end
             end
        end

        %   FOR FILE UPLOAD
        if (cell2mat(dropbox_user_activity_temp_1 (k,11))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
             if (cell2mat(dropbox_user_activity_temp_1 (k,12))>=45 && cell2mat(dropbox_user_activity_temp_1
(k,12))<=65)
                 if (cell2mat(dropbox_user_activity_temp_1(k,7))>=30 &&
cell2mat(dropbox_user_activity_temp_1(k,9))>=30)
                     if
(cell2mat(dropbox_user_activity_temp_1(k,7))/cell2mat(dropbox_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(dropbox_user_activity_temp_1(k,7))/cell2mat(dropbox_user_activity_temp_1(k,9))<=1.3)
                        dropbox_user_activity_temp_1(k,14) = {'Dropbox-File Upload'};

                          dropbox_user_activity_temp_1{k,15} = 'Dropbox';
                          dropbox_user_activity_temp_1{k,16} = dropbox_user_activity_temp_1{k,14};
                          stime = dropbox_user_activity_temp_1{k,1};
                          stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
```

```
              endtime = dropbox_user_activity_temp_1{k,2};
              etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
              dropbox_user_activity_temp_1{k,17} = stime_formatted;
              dropbox_user_activity_temp_1{k,18} = etime_formatted;

              dropbox_user_activity_final(j,1:18) = dropbox_user_activity_temp_1(k,1:18);
              j=j+1;
          end
        end
      end
    end

    %   FOR FOLDER NAVIGATION
        if (cell2mat(dropbox_user_activity_temp_1(k,7))==2)
          if (cell2mat(dropbox_user_activity_temp_1(k,8))==155)% &&
cell2mat(dropbox_user_activity_temp_1(k,8))<=155)
            dropbox_user_activity_temp_1(k,14) = {'Dropbox-Folder Navigation'};

              dropbox_user_activity_temp_1{k,15} = 'Dropbox';
              dropbox_user_activity_temp_1{k,16} = dropbox_user_activity_temp_1{k,14};
              stime = dropbox_user_activity_temp_1{k,1};
              stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
              endtime = dropbox_user_activity_temp_1{k,2};
              etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
              dropbox_user_activity_temp_1{k,17} = stime_formatted;
              dropbox_user_activity_temp_1{k,18} = etime_formatted;

              dropbox_user_activity_final(j,1:18) = dropbox_user_activity_temp_1(k,1:18);
              j=j+1;
          end
        end
  end
  dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Storage\Dropbox\');
  mkdir(dir_path);
  save (strcat(dir_path,'Dropbox_User_Activities.mat'), 'dropbox_user_activity_final');


     %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
  matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
  matlab_table1 = cell2table(dropbox_user_activity_final,'VariableNames',matlab_table_col_names);

    %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
  conn_db_1 = database(primary_dbpath,'','','org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
  sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
  exec(conn_db_1,sql_query_1);
  insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);
```

```
end
end
```

Facebook

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_facebook_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here



%               CODE FOR EXTRACTING TRAFFIC - SERVICE --> FACEBOOK
%               -------------------------------------------------


i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
facebook=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=','"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);


start_a=1;
start_b=10000;
row=count/start_b;
```

```
row=ceil(row);

          query_base = 'select count (*) from IP_logs where S_IP like "31.13.90.%" or D_IP like "31.13.90.%"';
          limit_base = exec(conn, query_base);
          limit_base = fetch(limit_base);
          limit = cell2mat(limit_base.Data);
if (limit>0)
   while (i<=row)

%            if (start_a>count)
%               break;
%            end
           query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "31.13.90.%" or D_IP like "31.13.90.%")');
           limit1_base = exec(conn, query_base);
           limit1_base = fetch(limit1_base);
           limit1 = cell2mat(limit1_base.Data);
           limit2= limit2+limit1;

         query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "31.13.90.%" or D_IP like "31.13.90.%")');
         facebook_rows = exec(conn, query_base1);
         facebook_rows = fetch(facebook_rows);

        if (limit1 ~= 0)
         for j=k+1:limit2
            if (limit2 > limit)
               break;
            end
            facebook(x,:) = facebook_rows.Data(p,:);
            p=p+1;
            if (x==q)
               user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
               if exist(user_service_folder_loc,'dir')
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Facebook', num2str(matfile_count));
                  save(user_service_matfile_loc,'facebook');
               else
                  mkdir(user_service_folder_loc);
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Facebook', num2str(matfile_count));
                  save(user_service_matfile_loc,'facebook');
               end
               matfile_count=matfile_count+1;
               clear facebook;
               x=0;

            elseif (x==limit-((matfile_count-1)*q))
                  user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
               if exist(user_service_folder_loc,'dir')
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Facebook', num2str(matfile_count));
                  save(user_service_matfile_loc,'facebook');
               else
                  mkdir(user_service_folder_loc);
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Facebook', num2str(matfile_count));
                  save(user_service_matfile_loc,'facebook');
               end
               clear facebook;
            end
%               if (p>limit1)
%                   break;
%               end
            x=x+1;
        end
       end
        k=limit2;
        p=1;
        start_a = start_b+1;
        start_b = start_b+10000;
        i=i+1;

    end
% %       UPDATING THE 'STAT' CELL-ARRAY TO SHOW IF THE USER CONTAINS ANY
FACEBOOK TRAFFIC
%     stat(user_num+1,3)={'1'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
    facebook_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);

% else
% %       UPDATING THE 'STAT' CELL-ARRAY TO SHOW IF THE USER CONTAINS ANY
FACEBOOK TRAFFIC
%     stat(user_num+1,3)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
facebook_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED10 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       -------------------------------------------------------------------------

iii=1;

facebookhttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
    for i=1:(matfile_count-1)
```

```
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
        for ii=1:500000
            if (strcmp(cellstruct.facebook(ii,3),'80')==1 || strcmp(cellstruct.facebook(ii,5),'80')==1)
                    facebookhttpinteractions(iii,:)=cellstruct.facebook(ii,:);
                    iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.facebook(ii,3),'80')==1 || strcmp(cellstruct.facebook(ii,5),'80')==1)
                    facebookhttpinteractions(iii,:)=cellstruct.facebook(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.facebook(ii,3),'80')==1 || strcmp(cellstruct.facebook(ii,5),'80')==1)
                    facebookhttpinteractions(iii,:)=cellstruct.facebook(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

for ii=1:(iii-1)
    facebookhttpinteractions(ii,8)={'0'};
end



%               FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(facebookhttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
```

```
for n=1:size(facebookhttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(facebookhttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
   tag = facebookhttpinteractions{n,7};

   if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
      interaction_temp{counter1, 1} = facebookhttpinteractions{n,1};
      interaction_temp{counter1, 3} = facebookhttpinteractions{n,2};
      interaction_temp{counter1, 4} = facebookhttpinteractions{n,3};
      interaction_temp{counter1, 5} = facebookhttpinteractions{n,4};
      interaction_temp{counter1, 6} = facebookhttpinteractions{n,5};
      interaction_temp{counter1, 7} = 0;% number of packet
%        interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
      interaction_temp{counter1, 8} = 0; %data volume

%        interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
      interaction_temp{counter2, 1} = '0';
      interaction_temp{counter2, 2} = '0';
      interaction_temp{counter2, 3} = facebookhttpinteractions{n,4};
      interaction_temp{counter2, 4} = facebookhttpinteractions{n,5};
      interaction_temp{counter2, 5} = facebookhttpinteractions{n,2};
      interaction_temp{counter2, 6} = facebookhttpinteractions{n,3};
      interaction_temp{counter2, 7} = 0;% number of packet
      interaction_temp{counter2, 8} = 0; %data volume

      flag1=1;
      counter1=counter1+2;
      counter2=counter2+2;

   end

   if (flag1>0)
    for j=1:size(facebookhttpinteractions,1)
         s_ip = facebookhttpinteractions{j,2};
         s_port = facebookhttpinteractions{j,3};
         d_ip = facebookhttpinteractions{j,4};
         d_port = facebookhttpinteractions{j,5};
         tag = facebookhttpinteractions{j,7};
     if (strcmp(facebookhttpinteractions(j,8),'0')==1)
      if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


      interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
      interaction_temp{counter1-2, 2} = facebookhttpinteractions{j,1}; %end time
      interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(facebookhttpinteractions{j,6}); %total data volume
      facebookhttpinteractions(j,8) = {'1'};
                 sum_col_eight = sum_col_eight+1;
%        new(m,:)=bbchttpinteractions(n,:);
      m=m+1;
      a=n;
      loopcounter=loopcounter-1;
%             loopcounter1=loopcounter1+1;
```

```
      end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
          if (counter3==0)
             interaction_temp{counter2-2, 1} = facebookhttpinteractions{j,1};
             counter3=1;
          end
       interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
       interaction_temp{counter2-2, 2} = facebookhttpinteractions{j,1}; %end time
       interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(facebookhttpinteractions{j,6}); %total data volume
       facebookhttpinteractions(j,8) = {'1'};
                   sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
       m=m+1;
       a=n;
       loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

      end
     end
    end
    counter3=0;

   end




end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
   mkdir(dir_path);
   save (strcat(dir_path,'Facebook HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;    % If HTTP traffic does exist
   facebook_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else          %If HTTP traffic doesn't exist
   existsornot = 0;
   facebook_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
facebook_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath)
%UNTITLED11 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
    interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
    if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
        interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
    else
    interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
    end
    interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
    interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
    interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
    interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
    interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
    interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
    interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
    interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
    ii=ii+1;
%     i=i+2;
end
    save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\So
cial Networking\Facebook\Facebook HTTP Interactions - Final Table.mat'), 'interaction_final');

facebook_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpat
h,primary_dbpath);
else            %No HTTP traffic exists
    interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

facebook_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpat
h,primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
facebook_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpa
th,primary_dbpath)
%UNTITLED12 Summary of this function goes here
%   Detailed explanation goes here

%         TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       -----------------------------------------------------------------------

% x=2949;  %for user16
% x=8080; %for user14
% x=187481;  %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);   %All https rows (input for the main part of code)
interaction_temp=cell(1,10);   %All https interactions (output)
tcp_interactions_final=cell(1,10);  %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
   for i=1:(matfile_count-1)
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
      for ii=1:500000
         if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
               iii=iii+1;
         end
      end
   end
else
   for i=1:matfile_count
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
      if (i<matfile_count)
         for ii=1:500000
            if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
               iii=iii+1;
            end
         end
      else
         for ii=1:(x-1)
            if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
               iii=iii+1;
            end
         end
      end
   end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%    if (i<matfile_count)
%          for ii=1:500000
%             if (bbc(ii,7)== 'S')
% %                    facebookinteractions()
%               end
%           end
%     else
%           for ii=1:(x-1)
%               if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                   httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                   iii=iii+1;
%               end
%           end
% %           for ii=1:(iii-1)
% %                  httpsinteractions(ii,8)={'0'};
% %           end
%
%     end
% end




%                           FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                           ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%                interaction_temp{counter1, 7} = 0;
% %                 interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%               interaction_temp{counter1, 8} = 0;
%               interaction_temp{counter1, 9} = 0;
% %                 interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%               interaction_temp{counter1, 10} = 0;


    if (timestamp2-timestamp1 < 6.9444e-06)

            s_ip = httpsinteractions{n,2};
            s_port = httpsinteractions{n,3};
            d_ip = httpsinteractions{n,4};
            d_port = httpsinteractions{n,5};
            tag = httpsinteractions{n,7};

%           if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
              fprintf('yes');
%               if isempty (interaction_temp{counter1, 1}==1)
%               interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%               end
              interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time
            if (strcmp(counter_increment_status,'yes')==1)
                if (strcmp(s_port,'443')==1)
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                else
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                end
            end

            if (isempty (interaction_temp{counter1, 7})==1)
            interaction_temp{counter1, 7} = 0;
            end
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
            if (isempty (interaction_temp{counter1, 8})==1)
            interaction_temp{counter1, 8} = 0;
            end
            if (isempty (interaction_temp{counter1, 9})==1)
            interaction_temp{counter1, 9} = 0;
            end
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
            if (isempty (interaction_temp{counter1, 10})==1)
            interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%               interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%               interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                if(strcmp(s_port,'443')==1)

                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of
Packets (User to Service)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                    if (n == (size(httpsinteractions,1)-1))
                        interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of
Packets (Service to User)
                        interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                    end
                else
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                    % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                    if (n == (size(httpsinteractions,1)-1))
                        interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                    % Number of
Packets (Service to User)
```

```
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
              end
            end
          end

          if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%           interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%           interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
            if(strcmp(s_port,'443')==1)
                          fprintf('\na');
            interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;            % Number of
Packets (Service to User)
            interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
              if (n == (size(httpsinteractions,1)-1))
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;            % Number of
Packets (Service to User)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
              end
            else
                          fprintf('\nb');
            interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;            % Number of
Packets (User to Service)
            interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
              if (n == (size(httpsinteractions,1)-1))
                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;            % Number of
Packets (Service to User)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
              end
            end
          end
%         flag1=1;

         else
            if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                          fprintf('\na');
            interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;            % Number of
Packets (Service to User)
            interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
            counter1=counter1+1;
            counter_increment_status='yes';
            elseif (strcmp(counter_increment_status,'no')==1)
                          fprintf('\nb');
            interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;            % Number of
Packets (User to Service)
            interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
            counter1=counter1+1;
            counter_increment_status='yes';
            end

         end

  else
```

```
      if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                        fprintf('\na');
      interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of Packets
(Service to User)
      interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
      counter1=counter1+1;
      counter_increment_status='yes';
      elseif (strcmp(counter_increment_status,'no')==1)
                        fprintf('\nb');
      interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of Packets
(User to Service)
      interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
      counter1=counter1+1;
      counter_increment_status='yes';
      end

   end
%  end
end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
   mkdir(dir_path);
   save (strcat(dir_path,'Facebook HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
   if (existsornot==1)     %Both HTTP and HTTPS traffic exist
      http_https_interactions_combined = [interaction_final;interaction_temp];
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'Facebook TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      facebook_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%         statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%         finalstat = statload1.finalstat;
%         finalstat(user_num+1,3)={'1'};
%         save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   else                 %Only HTTPS traffic exists; no HTTP
      http_https_interactions_combined = interaction_temp;
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'Facebook TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      facebook_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%         statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%         finalstat = statload1.finalstat;
%         finalstat(user_num+1,3)={'1'};
%         save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   end
else                    % No HTTPS traffic exists
%         http_https_interactions_combined = interaction_final;
%         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%         tcp_interactions_final = interaction_final;
      if (existsornot==1)     % No HTTPS but HTTP exists
         dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
         mkdir(dir_path);
         http_https_interactions_combined = interaction_final;
         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
         save (strcat(dir_path,'Facebook TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
```

```
        facebook_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%           statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%           finalstat = statload1.finalstat;
%           finalstat(user_num+1,3)={'1'};
%           save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ facebook_user_activity_final ] =
facebook_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here



    %   CODE FOR FINDING OUT THE FACEBOOK USER ACTIVITIES BASED ON THE
EXTRACTED USER INTERACTIONS
    %   ------------------------------------------------------------------------------------------

j=1;
k=1;
count=0;
facebook_user_activity_temp = cell(1,10);
facebook_user_activity_temp_1 = cell(1,18);
facebook_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user20\Online
Encyclopedia\Wikipedia\Wikipedia TCP Interactions - Final Table.mat', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Social Networking\Facebook\Facebook TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
  if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 && cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
    facebook_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
    j=j+1;
  end
end

for i=1:size(facebook_user_activity_temp,1)

    usr_to_srvr_avg_pkt_size =
ceil(cell2mat(facebook_user_activity_temp(i,8))/cell2mat(facebook_user_activity_temp(i,7)));
    facebook_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user
to Facebook server)

    srvr_to_usr_avg_pkt_size =
ceil(cell2mat(facebook_user_activity_temp(i,10))/cell2mat(facebook_user_activity_temp(i,9)));
    facebook_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size
(Facebook server to user)

    t1 = datevec(cell2mat(facebook_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating
the time duration of each interaction
    t2 = datevec(cell2mat(facebook_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
```

```
        duration = etime(t2,t1);
        facebook_user_activity_temp(i,13) = {duration};     %Column 13 --> Duration

    if (cell2mat(facebook_user_activity_temp(i,7))>=1 && cell2mat(facebook_user_activity_temp(i,9))>=1)
            facebook_user_activity_temp_1(k,1:13) = facebook_user_activity_temp(i,1:13);
            k=k+1;
    end
end


k=1;
j=1;
if (size(facebook_user_activity_temp_1,1)>=1)
    for k=1:size(facebook_user_activity_temp_1,1)
        %   CHECK FOR TYPING
%         if (cell2mat(facebook_user_activity_temp_1 (k,12))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
%          if (cell2mat(facebook_user_activity_temp_1 (k,11))>=45 && cell2mat(facebook_user_activity_temp_1
(k,11))<=65)
            if (cell2mat(facebook_user_activity_temp_1(k,7))==2 )
                if (cell2mat(facebook_user_activity_temp_1(k,8))==1502 )
%                 if
(cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))<=1.3)
                    facebook_user_activity_temp_1(k,14) = {'Facebook-Typing'};

                        facebook_user_activity_temp_1{k,15} = 'Facebook';
                        facebook_user_activity_temp_1{k,16} = facebook_user_activity_temp_1{k,14};
                        stime = facebook_user_activity_temp_1{k,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = facebook_user_activity_temp_1{k,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        facebook_user_activity_temp_1{k,17} = stime_formatted;
                        facebook_user_activity_temp_1{k,18} = etime_formatted;

                        facebook_user_activity_final(j,1:18) = facebook_user_activity_temp_1(k,1:18);
                        j=j+1;
%                   end
                end
            end
%           end
%         end




        %   CHECK FOR CHAT
%         if (cell2mat(facebook_user_activity_temp_1 (k,12))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
%          if (cell2mat(facebook_user_activity_temp_1 (k,11))>=45 && cell2mat(facebook_user_activity_temp_1
(k,11))<=65)
            if (cell2mat(facebook_user_activity_temp_1(k,7))==2 )
                if (cell2mat(facebook_user_activity_temp_1(k,8))>=2625 &&
cell2mat(facebook_user_activity_temp_1(k,8))<=2725)  % Base value for chat is 2625; checking for upto 100
characters
%                 if
(cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))<=1.3)
                    facebook_user_activity_temp_1(k,14) = {'Facebook-Chat'};
                        facebook_user_activity_temp_1{k,15} = 'Facebook';
                        facebook_user_activity_temp_1{k,16} = facebook_user_activity_temp_1{k,14};
                        stime = facebook_user_activity_temp_1{k,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = facebook_user_activity_temp_1{k,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        facebook_user_activity_temp_1{k,17} = stime_formatted;
                        facebook_user_activity_temp_1{k,18} = etime_formatted;

                        facebook_user_activity_final(j,1:18) = facebook_user_activity_temp_1(k,1:18);
                        j=j+1;
%                 end
                end
            end
%         end
%        end



        %   CHECK FOR ATTACH FILE IN CHAT WINDOW
%        if (cell2mat(facebook_user_activity_temp_1 (k,12))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
%         if (cell2mat(facebook_user_activity_temp_1 (k,11))>=45 && cell2mat(facebook_user_activity_temp_1
(k,11))<=65)
            if (cell2mat(facebook_user_activity_temp_1(k,9))==1 )
                if (cell2mat(facebook_user_activity_temp_1(k,12))>=1000)
%                 if
(cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))<=1.3)
                    facebook_user_activity_temp_1(k,14) = {'Facebook-Attach File in Chat'};

                        facebook_user_activity_temp_1{k,15} = 'Facebook';
                        facebook_user_activity_temp_1{k,16} = facebook_user_activity_temp_1{k,14};
                        stime = facebook_user_activity_temp_1{k,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = facebook_user_activity_temp_1{k,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        facebook_user_activity_temp_1{k,17} = stime_formatted;
                        facebook_user_activity_temp_1{k,18} = etime_formatted;

                        facebook_user_activity_final(j,1:18) = facebook_user_activity_temp_1(k,1:18);
                        j=j+1;
%                 end
                end
            end
%         end
%        end



        %   CHECK FOR PAGE LOADING
        if (cell2mat(facebook_user_activity_temp_1 (k,12))>=600)
%         if (cell2mat(facebook_user_activity_temp_1 (k,11))>=45 && cell2mat(facebook_user_activity_temp_1
(k,11))<=65)
```

```
            if (cell2mat(facebook_user_activity_temp_1(k,7))>=30 &&
cell2mat(facebook_user_activity_temp_1(k,9))>=30)
%                if
(cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(facebook_user_activity_temp_1(k,7))/cell2mat(facebook_user_activity_temp_1(k,9))<=1.3)
                facebook_user_activity_temp_1(k,14) = {'Facebook-Page Loading'};

                    facebook_user_activity_temp_1{k,15} = 'Facebook';
                    facebook_user_activity_temp_1{k,16} = facebook_user_activity_temp_1{k,14};
                    stime = facebook_user_activity_temp_1{k,1};
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = facebook_user_activity_temp_1{k,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    facebook_user_activity_temp_1{k,17} = stime_formatted;
                    facebook_user_activity_temp_1{k,18} = etime_formatted;

                    facebook_user_activity_final(j,1:18) = facebook_user_activity_temp_1(k,1:18);
                    j=j+1;
%                end
            end
%          end
        end


    end



    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Soci
al Networking\Facebook\');
    mkdir(dir_path);
    save (strcat(dir_path,'Facebook_User_Activities.mat'), 'facebook_user_activity_final');


        %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
    matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
    matlab_table1 = cell2table(facebook_user_activity_final,'VariableNames',matlab_table_col_names);

        %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
    conn_db_1 = database(primary_dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
    sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
    exec(conn_db_1,sql_query_1);
    insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Hotmail

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_hotmail_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here



%               CODE FOR EXTRACTING TRAFFIC - SERVICE --> HOTMAIL and BING
%               -------------------------------------------------------



i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
hotmailbing=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=','"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);



start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

        query_base = 'select count (*) from IP_logs where S_IP like "131.253.61.%" or D_IP like
"131.253.61.%" or S_IP like "157.56.122.%" or D_IP like "157.56.122.%" or S_IP like "204.79.197.%" or D_IP
like "204.79.197.%"';
        limit_base = exec(conn, query_base);
```

```
            limit_base = fetch(limit_base);
            limit = cell2mat(limit_base.Data);
if (limit>0)
    while (i<=row)

%            if (start_a>count)
%                break;
%            end
            query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "131.253.61.%" or D_IP like "131.253.61.%" or S_IP like
"157.56.122.%" or D_IP like "157.56.122.%" or S_IP like "204.79.197.%" or D_IP like "204.79.197.%")');
            limit1_base = exec(conn, query_base);
            limit1_base = fetch(limit1_base);
            limit1 = cell2mat(limit1_base.Data);
            limit2= limit2+limit1;

           query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "131.253.61.%" or D_IP like "131.253.61.%" or S_IP like
"157.56.122.%" or D_IP like "157.56.122.%" or S_IP like "204.79.197.%" or D_IP like "204.79.197.%")');
           hotmailbing_rows = exec(conn, query_base1);
           hotmailbing_rows = fetch(hotmailbing_rows);

          if (limit1 ~= 0)
           for j=k+1:limit2
               if (limit2 > limit)
                   break;
               end
               hotmailbing(x,:) = hotmailbing_rows.Data(p,:);
               p=p+1;
               if (x==q)
                   user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'HotmailBing',
num2str(matfile_count));
                       save(user_service_matfile_loc,'hotmailbing');
                   else
                       mkdir(user_service_folder_loc);
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'HotmailBing',
num2str(matfile_count));
                       save(user_service_matfile_loc,'hotmailbing');
                   end
                   matfile_count=matfile_count+1;
                   clear hotmailbing;
                   x=0;

               elseif (x==limit-((matfile_count-1)*q))
                       user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'HotmailBing',
num2str(matfile_count));
                       save(user_service_matfile_loc,'hotmailbing');
                   else
                       mkdir(user_service_folder_loc);
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'HotmailBing',
num2str(matfile_count));
                       save(user_service_matfile_loc,'hotmailbing');
```

```
                end
                clear hotmailbing;
            end
%               if (p>limit1)
%                   break;
%               end
            x=x+1;
        end
       end
        k=limit2;
        p=1;
        start_a = start_b+1;
        start_b = start_b+10000;
        i=i+1;

    end
%     stat(user_num+1,4)={'1'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
    hotmailbing_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);

% else
%     stat(user_num+1,4)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
hotmailbing_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED16 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%        ------------------------------------------------------------------------

iii=1;

hotmailbinghttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
    for i=1:(matfile_count-1)
    cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing',num2str(i),'.mat'), 'hotmailbing');
    for ii=1:500000
      if (strcmp(cellstruct.hotmailbing(ii,3),'80')==1 || strcmp(cellstruct.hotmailbing(ii,5),'80')==1)
          hotmailbinghttpinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
          iii=iii+1;
      end
    end
  end
else
  for i=1:matfile_count
    cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing',num2str(i),'.mat'), 'hotmailbing');
    if (i<matfile_count)
      for ii=1:500000
        if (strcmp(cellstruct.hotmailbing(ii,3),'80')==1 || strcmp(cellstruct.hotmailbing(ii,5),'80')==1)
          hotmailbinghttpinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
          iii=iii+1;
        end
      end
    else
      for ii=1:(x-1)
        if (strcmp(cellstruct.hotmailbing(ii,3),'80')==1 || strcmp(cellstruct.hotmailbing(ii,5),'80')==1)
          hotmailbinghttpinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
          iii=iii+1;
        end
      end
    end
  end
end

for ii=1:(iii-1)
  hotmailbinghttpinteractions(ii,8)={'0'};
end



%            FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(hotmailbinghttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%        break;
%     end
if (iii>1)
```

```
for n=1:size(hotmailbinghttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(hotmailbinghttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
   tag = hotmailbinghttpinteractions{n,7};

   if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
       interaction_temp{counter1, 1} = hotmailbinghttpinteractions{n,1};
       interaction_temp{counter1, 3} = hotmailbinghttpinteractions{n,2};
       interaction_temp{counter1, 4} = hotmailbinghttpinteractions{n,3};
       interaction_temp{counter1, 5} = hotmailbinghttpinteractions{n,4};
       interaction_temp{counter1, 6} = hotmailbinghttpinteractions{n,5};
       interaction_temp{counter1, 7} = 0;% number of packet
%        interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
       interaction_temp{counter1, 8} = 0; %data volume

%        interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
       interaction_temp{counter2, 1} = '0';
       interaction_temp{counter2, 2} = '0';
       interaction_temp{counter2, 3} = hotmailbinghttpinteractions{n,4};
       interaction_temp{counter2, 4} = hotmailbinghttpinteractions{n,5};
       interaction_temp{counter2, 5} = hotmailbinghttpinteractions{n,2};
       interaction_temp{counter2, 6} = hotmailbinghttpinteractions{n,3};
       interaction_temp{counter2, 7} = 0;% number of packet
       interaction_temp{counter2, 8} = 0; %data volume

       flag1=1;
       counter1=counter1+2;
       counter2=counter2+2;

   end

   if (flag1>0)
    for j=1:size(hotmailbinghttpinteractions,1)
          s_ip = hotmailbinghttpinteractions{j,2};
          s_port = hotmailbinghttpinteractions{j,3};
          d_ip = hotmailbinghttpinteractions{j,4};
          d_port = hotmailbinghttpinteractions{j,5};
          tag = hotmailbinghttpinteractions{j,7};
     if (strcmp(hotmailbinghttpinteractions(j,8),'0')==1)
      if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


       interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
       interaction_temp{counter1-2, 2} = hotmailbinghttpinteractions{j,1}; %end time
       interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(hotmailbinghttpinteractions{j,6}); %total data volume
       hotmailbinghttpinteractions(j,8) = {'1'};
                   sum_col_eight = sum_col_eight+1;
%        new(m,:)=bbchttpinteractions(n,:);
       m=m+1;
       a=n;
       loopcounter=loopcounter-1;
%              loopcounter1=loopcounter1+1;
```

```
      end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
          if (counter3==0)
              interaction_temp{counter2-2, 1} = hotmailbinghttpinteractions{j,1};
              counter3=1;
          end
       interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
       interaction_temp{counter2-2, 2} = hotmailbinghttpinteractions{j,1}; %end time
       interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(hotmailbinghttpinteractions{j,6}); %total data volume
       hotmailbinghttpinteractions(j,8) = {'1'};
                   sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
       m=m+1;
       a=n;
       loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

      end
     end
    end
    counter3=0;

   end




end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
   mkdir(dir_path);
   save (strcat(dir_path,'HotmailBing HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;    % If HTTP traffic does exist
   hotmailbing_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else          %If HTTP traffic doesn't exist
   existsornot = 0;
   hotmailbing_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
hotmailbing_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbp
ath)
%UNTITLED17 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing HTTP Interactions - Intial Table.mat'),
'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
    interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
    if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
        interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
    else
    interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
    end
    interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
    interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
    interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
    interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
    interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
    interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
    interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
    interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
    ii=ii+1;
%     i=i+2;
end
    save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mi
crosoft Services - Info Search & Email\HotmailBing\HotmailBing HTTP Interactions - Final Table.mat'),
'interaction_final');

hotmailbing_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,db
path,primary_dbpath);
else            %No HTTP traffic exists
    interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

hotmailbing_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,db
path,primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
hotmailbing_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,db
path,primary_dbpath)
%UNTITLED18 Summary of this function goes here
```

```
%   Detailed explanation goes here

%       TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       ----------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
   for i=1:(matfile_count-1)
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing',num2str(i),'.mat'), 'hotmailbing');
      for ii=1:500000
         if (strcmp(cellstruct.hotmailbing(ii,3),'443')==1 || strcmp(cellstruct.hotmailbing(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
               iii=iii+1;
         end
      end
   end
else
   for i=1:matfile_count
      cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing',num2str(i),'.mat'), 'hotmailbing');
      if (i<matfile_count)
         for ii=1:500000
            if (strcmp(cellstruct.hotmailbing(ii,3),'443')==1 || strcmp(cellstruct.hotmailbing(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
               iii=iii+1;
            end
         end
      else
         for ii=1:(x-1)
            if (strcmp(cellstruct.hotmailbing(ii,3),'443')==1 || strcmp(cellstruct.hotmailbing(ii,5),'443')==1)
               httpsinteractions(iii,:)=cellstruct.hotmailbing(ii,:);
               iii=iii+1;
            end
         end
      end
   end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
```

```
%     if (i<matfile_count)
%           for ii=1:500000
%              if (bbc(ii,7)== 'S')
% %                   facebookinteractions()
%              end
%           end
%     else
%           for ii=1:(x-1)
%              if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                 httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                 iii=iii+1;
%              end
%           end
% %           for ii=1:(iii-1)
% %                 httpsinteractions(ii,8)={'0'};
% %           end
%
%     end
% end




%                          FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                          ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%               interaction_temp{counter1, 7} = 0;
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%              interaction_temp{counter1, 8} = 0;
%              interaction_temp{counter1, 9} = 0;
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%              interaction_temp{counter1, 10} = 0;


    if (timestamp2-timestamp1 < 6.9444e-06)

          s_ip = httpsinteractions{n,2};
          s_port = httpsinteractions{n,3};
          d_ip = httpsinteractions{n,4};
          d_port = httpsinteractions{n,5};
          tag = httpsinteractions{n,7};

%           if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
              fprintf('yes');
%              if isempty (interaction_temp{counter1, 1}==1)
%              interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%              end
            interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

            if (strcmp(counter_increment_status,'yes')==1)
               if (strcmp(s_port,'443')==1)
                  interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                  interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                  interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                  interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                  interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                  counter_increment_status='no';
               else
                  interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                  interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                  interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                  interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                  interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                  counter_increment_status='no';
               end
            end

            if (isempty (interaction_temp{counter1, 7})==1)
            interaction_temp{counter1, 7} = 0;
            end
% %              interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
            if (isempty (interaction_temp{counter1, 8})==1)
            interaction_temp{counter1, 8} = 0;
            end
            if (isempty (interaction_temp{counter1, 9})==1)
            interaction_temp{counter1, 9} = 0;
            end
% %              interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
            if (isempty (interaction_temp{counter1, 10})==1)
            interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%              interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%              interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%              interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%              interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)

               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (User to Service)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (Service to User)
                     interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               else
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
```

```
              interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;             % Number of
Packets (Service to User)
              interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
            end
          end
        end

        if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%          interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%          interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
          if(strcmp(s_port,'443')==1)
                        fprintf('\na');
          interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;             % Number of
Packets (Service to User)
          interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
            if (n == (size(httpsinteractions,1)-1))
              interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;             % Number of
Packets (Service to User)
              interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
            end
          else
                        fprintf('\nb');
          interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;             % Number of
Packets (User to Service)
          interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
            if (n == (size(httpsinteractions,1)-1))
              interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;             % Number of
Packets (Service to User)
              interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
            end
          end
        end
%          flag1=1;

       else
          if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                        fprintf('\na');
          interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;             % Number of
Packets (Service to User)
          interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
          counter1=counter1+1;
          counter_increment_status='yes';
          elseif (strcmp(counter_increment_status,'no')==1)
                        fprintf('\nb');
          interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;             % Number of
Packets (User to Service)
          interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
          counter1=counter1+1;
          counter_increment_status='yes';
          end

       end
```

```
  else
      if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                          fprintf('\na');
      interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;              % Number of Packets
(Service to User)
      interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
      counter1=counter1+1;
      counter_increment_status='yes';
      elseif (strcmp(counter_increment_status,'no')==1)
                          fprintf('\nb');
      interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;              % Number of Packets
(User to Service)
      interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
      counter1=counter1+1;
      counter_increment_status='yes';
      end

  end
%  end
end
  dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
  mkdir(dir_path);
  save (strcat(dir_path,'HotmailBing HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
  if (existsornot==1)    %Both HTTP and HTTPS traffic exist
      http_https_interactions_combined = [interaction_final;interaction_temp];
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'HotmailBing TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      hotmailbing_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%        statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%        finalstat = statload1.finalstat;
%        finalstat(user_num+1,4)={'1'};
%        save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
  else                %Only HTTPS traffic exists; no HTTP
      http_https_interactions_combined = interaction_temp;
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'HotmailBing TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
      hotmailbing_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%        statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%        finalstat = statload1.finalstat;
%        finalstat(user_num+1,4)={'1'};
%        save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
  end
else                  % No HTTPS traffic exists
%        http_https_interactions_combined = interaction_final;
%        tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%        tcp_interactions_final = interaction_final;
      if (existsornot==1)    % No HTTPS but HTTP exists
          dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
          mkdir(dir_path);
          http_https_interactions_combined = interaction_final;
```

```
        tcp_interactions_final = sortrows (http_https_interactions_combined,1);
        save (strcat(dir_path,'HotmailBing TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
        hotmailbing_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%           save (strcat(dir_path,'HotmailBing TCP Interactions - Final Table.mat'), 'interaction_final');

%           statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%           finalstat = statload1.finalstat;
%           finalstat(user_num+1,4)={'1'};
%           save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ hotmailbing_user_activity_final ] =
hotmailbing_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here




%   CODE FOR FINDING OUT THE HOTMAIL-BING USER ACTIVITIES BASED ON THE
EXTRACTED USER INTERACTIONS
%   -----------------------------------------------------------------------------------------------

j=1;
k=1;
m=1;
count=0;
count1=0;
hotmailbing_user_activity_temp = cell(1,10);
hotmailbing_user_activity_temp_1 = cell(1,18);
hotmailbing_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user19\News\BBC\BBC TCP Interactions - Final
Table.mat', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - Info Search & Email\HotmailBing\HotmailBing TCP Interactions - Final
Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
   if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 || cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
      hotmailbing_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
      j=j+1;
   end
end

for i=1:size(hotmailbing_user_activity_temp,1)
   usr_to_srvr_avg_pkt_size =
ceil(cell2mat(hotmailbing_user_activity_temp(i,8))/cell2mat(hotmailbing_user_activity_temp(i,7)));
   hotmailbing_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user
to Hotmail server)

   srvr_to_usr_avg_pkt_size =
ceil(cell2mat(hotmailbing_user_activity_temp(i,10))/cell2mat(hotmailbing_user_activity_temp(i,9)));
```

```
    hotmailbing_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size
(Hotmail server to user)

    t1 = datevec(cell2mat(hotmailbing_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating
the time duration of each interaction
    t2 = datevec(cell2mat(hotmailbing_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
    duration = etime(t2,t1);
    hotmailbing_user_activity_temp(i,13) = {duration};      %Column 13 --> Duration
    if (cell2mat(hotmailbing_user_activity_temp(i,7))>=1 && cell2mat(hotmailbing_user_activity_temp(i,9))>=1)
            hotmailbing_user_activity_temp_1(k,1:13) = hotmailbing_user_activity_temp(i,1:13);
            k=k+1;
    end
end



j=1;
if (size(hotmailbing_user_activity_temp_1,1)>=1)
    for i=1:size(hotmailbing_user_activity_temp_1,1)
        % CHECKING FOR COMPOSE EMAIL (Clicking on 'New' button)
        if (cell2mat(hotmailbing_user_activity_temp_1 (i,9))==1 )
            if (cell2mat(hotmailbing_user_activity_temp_1 (i,12))==971 ||
cell2mat(hotmailbing_user_activity_temp_1 (i,12))==981)
                hotmailbing_user_activity_temp_1(i,14) = {'Hotmail-Compose Email'};
                        hotmailbing_user_activity_temp_1{i,15} = 'Hotmail';
                        hotmailbing_user_activity_temp_1{i,16} = hotmailbing_user_activity_temp_1{i,14};
                        stime = hotmailbing_user_activity_temp_1{i,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = hotmailbing_user_activity_temp_1{i,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        hotmailbing_user_activity_temp_1{i,17} = stime_formatted;
                        hotmailbing_user_activity_temp_1{i,18} = etime_formatted;

                        hotmailbing_user_activity_final(j,1:18) = hotmailbing_user_activity_temp_1(i,1:18);
                        j=j+1;
            end
        end


        % CHECKING FOR FILE ATTACHMENT
            if (cell2mat(hotmailbing_user_activity_temp_1 (i,11))>=1000)
                if (cell2mat(hotmailbing_user_activity_temp_1(i,7))>=30 &&
cell2mat(hotmailbing_user_activity_temp_1(i,9))>=30)
                    hotmailbing_user_activity_temp_1(i,14) = {'Hotmail-File Attachment'};
                        hotmailbing_user_activity_temp_1{i,15} = 'Hotmail';
                        hotmailbing_user_activity_temp_1{i,16} = hotmailbing_user_activity_temp_1{i,14};
                        stime = hotmailbing_user_activity_temp_1{i,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = hotmailbing_user_activity_temp_1{i,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        hotmailbing_user_activity_temp_1{i,17} = stime_formatted;
                        hotmailbing_user_activity_temp_1{i,18} = etime_formatted;

                        hotmailbing_user_activity_final(j,1:18) = hotmailbing_user_activity_temp_1(i,1:18);
                        j=j+1;
                end
            end
        % CHECKING FOR INSERT A RECEPIENT
        if (cell2mat(hotmailbing_user_activity_temp_1 (i,7))==1 )
            if (cell2mat(hotmailbing_user_activity_temp_1 (i,11))==971)     % NEED TO CONFIRM THE
CORRECT VALUE IN THIS LINE
                hotmailbing_user_activity_temp_1(i,14) = {'Hotmail-Insert a Recepient'};
                        hotmailbing_user_activity_temp_1{i,15} = 'Hotmail';
                        hotmailbing_user_activity_temp_1{i,16} = hotmailbing_user_activity_temp_1{i,14};
                        stime = hotmailbing_user_activity_temp_1{i,1};
                        stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                        endtime = hotmailbing_user_activity_temp_1{i,2};
                        etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                        hotmailbing_user_activity_temp_1{i,17} = stime_formatted;
                        hotmailbing_user_activity_temp_1{i,18} = etime_formatted;

                        hotmailbing_user_activity_final(j,1:18) = hotmailbing_user_activity_temp_1(i,1:18);
                        j=j+1;
            end
        end

    end

    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - Info Search & Email\HotmailBing\');
    mkdir(dir_path);
    save (strcat(dir_path,'HotmailBing_User_Activities.mat'), 'hotmailbing_user_activity_final');


         %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
    matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
    matlab_table1 = cell2table(hotmailbing_user_activity_final,'VariableNames',matlab_table_col_names);

        %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
    conn_db_1 = database(primary_dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
    sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
    exec(conn_db_1,sql_query_1);
    insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Google Docs, Google Search and YouTube

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_google_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED4 Summary of this function goes here
%   Detailed explanation goes here



%               CODE FOR EXTRACTING TRAFFIC - SERVICE --> GOOGLE SERVICES
%               (GOOGLE SEARCH, GOOGLE DOCS, YOUTUBE) AS ALL OF THEM
%               SHARE THE SAME RANGE OF IP ADDRESSES
%               ---------------------------------------------------------


i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
google=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=',"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);


start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

         query_base = 'select count (*) from IP_logs where S_IP like "216.58.208.%" or D_IP like
"216.58.208.%" or S_IP like "216.58.209.%" or D_IP like "216.58.209.%" or S_IP like "216.58.204.%" or D_IP
```

```
like "216.58.204.%" or S_IP like "216.58.198.%" or D_IP like "216.58.198.%" or S_IP like "74.125.133.%" or
D_IP like "74.125.133.%" or S_IP like "173.194.78.%" or D_IP like "173.194.78.%"';
            limit_base = exec(conn, query_base);
            limit_base = fetch(limit_base);
            limit = cell2mat(limit_base.Data);
if (limit>0)
    while (i<=row)

%           if (start_a>count)
%               break;
%           end
            query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "216.58.208.%" or D_IP like "216.58.208.%" or S_IP like
"216.58.209.%" or D_IP like "216.58.209.%" or S_IP like "216.58.204.%" or D_IP like "216.58.204.%" or S_IP
like "216.58.198.%" or D_IP like "216.58.198.%" or S_IP like "74.125.133.%" or D_IP like "74.125.133.%" or
S_IP like "173.194.78.%" or D_IP like "173.194.78.%")');
            limit1_base = exec(conn, query_base);
            limit1_base = fetch(limit1_base);
            limit1 = cell2mat(limit1_base.Data);
            limit2= limit2+limit1;

           query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "216.58.208.%" or D_IP like "216.58.208.%" or S_IP like
"216.58.209.%" or D_IP like "216.58.209.%" or S_IP like "216.58.204.%" or D_IP like "216.58.204.%" or S_IP
like "216.58.198.%" or D_IP like "216.58.198.%" or S_IP like "74.125.133.%" or D_IP like "74.125.133.%" or
S_IP like "173.194.78.%" or D_IP like "173.194.78.%")');
           google_rows = exec(conn, query_base1);
           google_rows = fetch(google_rows);

          if (limit1 ~= 0)
           for j=k+1:limit2
               if (limit2 > limit)
                   break;
               end
               google(x,:) = google_rows.Data(p,:);
               p=p+1;
               if (x==q)
                   user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Goo
gle Services\Google\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Google', num2str(matfile_count));
                       save(user_service_matfile_loc,'google');
                   else
                       mkdir(user_service_folder_loc);
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Google', num2str(matfile_count));
                       save(user_service_matfile_loc,'google');
                   end
                   matfile_count=matfile_count+1;
                   clear google;
                   x=0;

               elseif (x==limit-((matfile_count-1)*q))
                       user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Goo
gle Services\Google\');
                   if exist(user_service_folder_loc,'dir')
                       user_service_matfile_loc = strcat(user_service_folder_loc, 'Google', num2str(matfile_count));
                       save(user_service_matfile_loc,'google');
                   else
```

```
                mkdir(user_service_folder_loc);
                user_service_matfile_loc = strcat(user_service_folder_loc, 'Google', num2str(matfile_count));
                save(user_service_matfile_loc,'google');
              end
              clear google;
            end
%             if (p>limit1)
%                 break;
%             end
            x=x+1;
          end
        end
          k=limit2;
          p=1;
          start_a = start_b+1;
          start_b = start_b+10000;
          i=i+1;

  end
%     stat(user_num+1,2)={'1'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
  google_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);

% else
%     stat(user_num+1,2)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
google_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED13 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       ------------------------------------------------------------------------

iii=1;

googlehttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
  for i=1:(matfile_count-1)
```

```
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google',num2str(i),'.mat'), 'google');
        for ii=1:500000
            if (strcmp(cellstruct.google(ii,3),'80')==1 || strcmp(cellstruct.google(ii,5),'80')==1)
                googlehttpinteractions(iii,:)=cellstruct.google(ii,:);
                iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google',num2str(i),'.mat'), 'google');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.google(ii,3),'80')==1 || strcmp(cellstruct.google(ii,5),'80')==1)
                    googlehttpinteractions(iii,:)=cellstruct.google(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.google(ii,3),'80')==1 || strcmp(cellstruct.google(ii,5),'80')==1)
                    googlehttpinteractions(iii,:)=cellstruct.google(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

for ii=1:(iii-1)
    googlehttpinteractions(ii,8)={'0'};
end



%               FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(googlehttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
```

```
for n=1:size(googlehttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(googlehttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
    tag = googlehttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = googlehttpinteractions{n,1};
        interaction_temp{counter1, 3} = googlehttpinteractions{n,2};
        interaction_temp{counter1, 4} = googlehttpinteractions{n,3};
        interaction_temp{counter1, 5} = googlehttpinteractions{n,4};
        interaction_temp{counter1, 6} = googlehttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
%         interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
        interaction_temp{counter1, 8} = 0; %data volume

%         interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = googlehttpinteractions{n,4};
        interaction_temp{counter2, 4} = googlehttpinteractions{n,5};
        interaction_temp{counter2, 5} = googlehttpinteractions{n,2};
        interaction_temp{counter2, 6} = googlehttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume

        flag1=1;
        counter1=counter1+2;
        counter2=counter2+2;

    end

    if (flag1>0)
     for j=1:size(googlehttpinteractions,1)
            s_ip = googlehttpinteractions{j,2};
            s_port = googlehttpinteractions{j,3};
            d_ip = googlehttpinteractions{j,4};
            d_port = googlehttpinteractions{j,5};
            tag = googlehttpinteractions{j,7};
      if (strcmp(googlehttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


        interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
        interaction_temp{counter1-2, 2} = googlehttpinteractions{j,1}; %end time
        interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(googlehttpinteractions{j,6}); %total data volume
        googlehttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%             loopcounter1=loopcounter1+1;
```

```
      end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
        if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
            if (counter3==0)
                interaction_temp{counter2-2, 1} = googlehttpinteractions{j,1};
                counter3=1;
            end
        interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
        interaction_temp{counter2-2, 2} = googlehttpinteractions{j,1}; %end time
        interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(googlehttpinteractions{j,6}); %total data volume
        googlehttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%                loopcounter1=loopcounter1+1;

      end
     end
    end
    counter3=0;

   end




end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Goo
gle Services\Google\');
   mkdir(dir_path);
   save (strcat(dir_path,'Google HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;     % If HTTP traffic does exist
   google_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else           %If HTTP traffic doesn't exist
   existsornot = 0;
   google_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
google_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath)
%UNTITLED14 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
   interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
   if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
      interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
   else
   interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
   end
   interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
   interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
   interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
   interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
   interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
   interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
   interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
   interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
   ii=ii+1;
%     i=i+2;
end
   save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Go
ogle Services\Google\Google HTTP Interactions - Final Table.mat'), 'interaction_final');

google_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath);
else            %No HTTP traffic exists
   interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

google_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
google_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath)
%UNTITLED15 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       -------------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
   for i=1:(matfile_count-1)
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google',num2str(i),'.mat'), 'google');
       for ii=1:500000
           if (strcmp(cellstruct.google(ii,3),'443')==1 || strcmp(cellstruct.google(ii,5),'443')==1)
                 httpsinteractions(iii,:)=cellstruct.google(ii,:);
                 iii=iii+1;
           end
       end
   end
else
   for i=1:matfile_count
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google',num2str(i),'.mat'), 'google');
       if (i<matfile_count)
           for ii=1:500000
               if (strcmp(cellstruct.google(ii,3),'443')==1 || strcmp(cellstruct.google(ii,5),'443')==1)
                   httpsinteractions(iii,:)=cellstruct.google(ii,:);
                   iii=iii+1;
               end
           end
       else
           for ii=1:(x-1)
               if (strcmp(cellstruct.google(ii,3),'443')==1 || strcmp(cellstruct.google(ii,5),'443')==1)
                   httpsinteractions(iii,:)=cellstruct.google(ii,:);
                   iii=iii+1;
               end
           end
       end
   end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%           for ii=1:500000
%               if (bbc(ii,7)== 'S')
% %                facebookinteractions()
%            end
%          end
%    else
%          for ii=1:(x-1)
%             if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                iii=iii+1;
%            end
%          end
% %          for ii=1:(iii-1)
% %                httpsinteractions(ii,8)={'0'};
% %          end
%
%    end
% end




%                        FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                        ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%              interaction_temp{counter1, 7} = 0;
% %              interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%              interaction_temp{counter1, 8} = 0;
%              interaction_temp{counter1, 9} = 0;
% %              interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%              interaction_temp{counter1, 10} = 0;


   if (timestamp2-timestamp1 < 6.9444e-06) % 6.9444e-06 is 1 second time difference

         s_ip = httpsinteractions{n,2};
         s_port = httpsinteractions{n,3};
         d_ip = httpsinteractions{n,4};
         d_port = httpsinteractions{n,5};
         tag = httpsinteractions{n,7};

%          if (flag1==0 && strcmp(tag, 'S'))
           if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
             fprintf('yes');
%              if isempty (interaction_temp{counter1, 1}==1)
%              interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%              end
             interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time
```

```
            if (strcmp(counter_increment_status,'yes')==1)
                if (strcmp(s_port,'443')==1)
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                else
                    interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                    interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                    interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                    interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                    interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                    counter_increment_status='no';
                end
            end

            if (isempty (interaction_temp{counter1, 7})==1)
            interaction_temp{counter1, 7} = 0;
            end
% %                 interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
            if (isempty (interaction_temp{counter1, 8})==1)
            interaction_temp{counter1, 8} = 0;
            end
            if (isempty (interaction_temp{counter1, 9})==1)
            interaction_temp{counter1, 9} = 0;
            end
% %                 interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
            if (isempty (interaction_temp{counter1, 10})==1)
            interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%               interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%               interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                if(strcmp(s_port,'443')==1)

                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (User to Service)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                    if (n == (size(httpsinteractions,1)-1))
                        interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (Service to User)
                        interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                    end
                else
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                    if (n == (size(httpsinteractions,1)-1))
                        interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (Service to User)
```

```
                           interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                       end
                    end
                end

                if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%                   interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%                   interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                    if(strcmp(s_port,'443')==1)
                                               fprintf('\na');
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                       if (n == (size(httpsinteractions,1)-1))
                           interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                    % Number of
Packets (Service to User)
                           interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                       end
                    else
                                               fprintf('\nb');
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                    % Number of
Packets (User to Service)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                       if (n == (size(httpsinteractions,1)-1))
                           interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of
Packets (Service to User)
                           interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                       end
                    end
                end
%                   flag1=1;

              else
                    if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                                               fprintf('\na');
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                    counter1=counter1+1;
                    counter_increment_status='yes';
                    elseif (strcmp(counter_increment_status,'no')==1)
                                               fprintf('\nb');
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                    % Number of
Packets (User to Service)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                    counter1=counter1+1;
                    counter_increment_status='yes';
                    end

              end

    else
    if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                    fprintf('\na');
    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of Packets
(Service to User)
    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
    counter1=counter1+1;
    counter_increment_status='yes';
    elseif (strcmp(counter_increment_status,'no')==1)
                    fprintf('\nb');
    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of Packets
(User to Service)
    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
    counter1=counter1+1;
    counter_increment_status='yes';
    end

  end
%  end
end
  dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Goo
gle Services\Google\');
  mkdir(dir_path);
  save (strcat(dir_path,'Google HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
  if (existsornot==1)     %Both HTTP and HTTPS traffic exist
    http_https_interactions_combined = [interaction_final;interaction_temp];
    tcp_interactions_final = sortrows (http_https_interactions_combined,1);
    save (strcat(dir_path,'Google TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
    google_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%       statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%       finalstat = statload1.finalstat;
%       finalstat(user_num+1,2)={'1'};
%       save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
  else              %Only HTTPS traffic exists; no HTTP
    http_https_interactions_combined = interaction_temp;
    tcp_interactions_final = sortrows (http_https_interactions_combined,1);
    save (strcat(dir_path,'Google TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
    google_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%       statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%       finalstat = statload1.finalstat;
%       finalstat(user_num+1,2)={'1'};
%       save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
  end
else                % No HTTPS traffic exists
%       http_https_interactions_combined = interaction_final;
%       tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%       tcp_interactions_final = interaction_final;
    if (existsornot==1)     % No HTTPS but HTTP exists
      dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Goo
gle Services\Google\');
      mkdir(dir_path);
      http_https_interactions_combined = interaction_final;
      tcp_interactions_final = sortrows (http_https_interactions_combined,1);
      save (strcat(dir_path,'Google TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
```

```
            google_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%           statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%           finalstat = statload1.finalstat;
%           finalstat(user_num+1,2)={'1'};
%           save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ google_user_activity_final ] =
google_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED5 Summary of this function goes here
%   Detailed explanation goes here



        %   CODE FOR FINDING OUT THE GOOGLE USER ACTIVITIES BASED ON THE
        %   EXTRACTED USER INTERACTIONS (GOOGLE SEARCH, GOOGLE DOCS AND YOUTUBE
ACTIVITIES
        %   WILL BE IDENTIFIED HERE AS THEY ALL SHARE THE SAME RANGE OF GOOGLE IP
ADDRESSES)
        %   ------------------------------------------------------------------------------

j=1;
k=1;
count=0;
google_user_activity_temp = cell(1,10);
google_user_activity_temp_1 = cell(1,18);
google_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user20\Online
Encyclopedia\Wikipedia\Wikipedia TCP Interactions - Final Table.mat', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Google Services\Google\Google TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
    if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 && cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
        google_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
        j=j+1;
    end
end

for i=1:size(google_user_activity_temp,1)

        usr_to_srvr_avg_pkt_size =
ceil(cell2mat(google_user_activity_temp(i,8))/cell2mat(google_user_activity_temp(i,7)));
        google_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user to
Google server)

        srvr_to_usr_avg_pkt_size =
ceil(cell2mat(google_user_activity_temp(i,10))/cell2mat(google_user_activity_temp(i,9)));
        google_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size
(Google server to user)
```

```
        t1 = datevec(cell2mat(google_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating
the time duration of each interaction
        t2 = datevec(cell2mat(google_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
        duration = etime(t2,t1);
        google_user_activity_temp(i,13) = {duration};     %Column 13 --> Duration

    if (cell2mat(google_user_activity_temp(i,7))>=1 && cell2mat(google_user_activity_temp(i,9))>=1)
            google_user_activity_temp_1(k,1:13) = google_user_activity_temp(i,1:13);
            k=k+1;
    end
end


k=1;
j=1;
if (size(google_user_activity_temp_1,1)>=1)
    for k=1:size(google_user_activity_temp_1,1)
        %   CHECK FOR YOUTUBE - WATCHING VIDEO
        if (strcmp(google_user_activity_temp_1{k,6},'443')==1)
            if (cell2mat(google_user_activity_temp_1 (k,13))>=10 )    %Checking if duration is at least 10s (to
assume it's a video stream)
            if (cell2mat(google_user_activity_temp_1 (k,12))>=1000 )
%           if (cell2mat(google_user_activity_temp_1 (k,11))>=45 && cell2mat(google_user_activity_temp_1
(k,11))<=65)
             if (cell2mat(google_user_activity_temp_1(k,7))>=30 &&
cell2mat(google_user_activity_temp_1(k,9))>=30)
%                   if (cell2mat(google_user_activity_temp_1(k,7))/cell2mat(google_user_activity_temp_1(k,9))>=0.7
&& cell2mat(google_user_activity_temp_1(k,7))/cell2mat(google_user_activity_temp_1(k,9))<=1.3)
                  google_user_activity_temp_1(k,14) = {'YouTube-Watching Video'};

                    google_user_activity_temp_1{k,15} = 'YouTube';
                    google_user_activity_temp_1{k,16} = google_user_activity_temp_1{k,14};
                    stime = google_user_activity_temp_1{k,1};
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = google_user_activity_temp_1{k,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    google_user_activity_temp_1{k,17} = stime_formatted;
                    google_user_activity_temp_1{k,18} = etime_formatted;

                    google_user_activity_final(j,1:18) = google_user_activity_temp_1(k,1:18);
                    j=j+1;
%                 end
             end
%           end
            end
            end
        end



        %   CHECK FOR YOUTUBE - VIDEO UPLOAD
        if (strcmp(google_user_activity_temp_1{k,6},'443')==1)
            if (cell2mat(google_user_activity_temp_1 (k,13))>=10 )  %Checking if duration is at least 10s (to
assume it's a video upload)
            if (cell2mat(google_user_activity_temp_1 (k,11))>=1000 )
%           if (cell2mat(google_user_activity_temp_1 (k,12))>=45 && cell2mat(google_user_activity_temp_1
(k,12))<=65)
             if (cell2mat(google_user_activity_temp_1(k,7))>=30 &&
cell2mat(google_user_activity_temp_1(k,9))>=30)
```

```
%               if (cell2mat(google_user_activity_temp_1(k,7))/cell2mat(google_user_activity_temp_1(k,9)))>=0.7
&& cell2mat(google_user_activity_temp_1(k,7))/cell2mat(google_user_activity_temp_1(k,9))<=1.3)
              google_user_activity_temp_1(k,14) = {'YouTube-Video Uploading'};

                 google_user_activity_temp_1{k,15} = 'YouTube';
                 google_user_activity_temp_1{k,16} = google_user_activity_temp_1{k,14};
                 stime = google_user_activity_temp_1{k,1};
                 stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                 endtime = google_user_activity_temp_1{k,2};
                 etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                 google_user_activity_temp_1{k,17} = stime_formatted;
                 google_user_activity_temp_1{k,18} = etime_formatted;

                 google_user_activity_final(j,1:18) = google_user_activity_temp_1(k,1:18);
                 j=j+1;
%               end
         end
%        end
        end
        end
     end


     %   CHECK FOR GOOGLE SEARCH - NAVIGATION THROUGH SEARCH RESULT PAGES
        if (strcmp(google_user_activity_temp_1{k,6},'443')==1)
         if (cell2mat(google_user_activity_temp_1(k,9))==3)
            if (cell2mat(google_user_activity_temp_1 (k,10))==268 )
              google_user_activity_temp_1(k,14) = {'Google Search-Page Navigation'};

                 google_user_activity_temp_1{k,15} = 'Google Search';
                 google_user_activity_temp_1{k,16} = google_user_activity_temp_1{k,14};
                 stime = google_user_activity_temp_1{k,1};
                 stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                 endtime = google_user_activity_temp_1{k,2};
                 etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                 google_user_activity_temp_1{k,17} = stime_formatted;
                 google_user_activity_temp_1{k,18} = etime_formatted;

                 google_user_activity_final(j,1:18) = google_user_activity_temp_1(k,1:18);
                 j=j+1;
            end
         end
        end


     %   CHECK FOR GOOGLE DOCS - EDITING (MODIFYING) A DOCUMENT
        if (strcmp(google_user_activity_temp_1{k,6},'443')==1)
         if (cell2mat(google_user_activity_temp_1(k,9))==2)
            if (cell2mat(google_user_activity_temp_1 (k,10))==270 )
              google_user_activity_temp_1(k,14) = {'Google Docs-Editing'};

                 google_user_activity_temp_1{k,15} = 'Google Docs';
                 google_user_activity_temp_1{k,16} = google_user_activity_temp_1{k,14};
                 stime = google_user_activity_temp_1{k,1};
                 stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
```

```
                endtime = google_user_activity_temp_1{k,2};
                etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                google_user_activity_temp_1{k,17} = stime_formatted;
                google_user_activity_temp_1{k,18} = etime_formatted;

                google_user_activity_final(j,1:18) = google_user_activity_temp_1(k,1:18);
                j=j+1;
            end
         end
        end


   end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Google Services\Google\');
   mkdir(dir_path);
   save (strcat(dir_path,'Google_User_Activities.mat'), 'google_user_activity_final');


       %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
   matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','User_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
   matlab_table1 = cell2table(google_user_activity_final,'VariableNames',matlab_table_col_names);

      %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
   conn_db_1 = database(primary_dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
   sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
   exec(conn_db_1,sql_query_1);
   insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Skype

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_skype_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here


i=1;
j=0;
```

```
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
skype=cell(1,7);
skypeudp=cell(1,7);
c=1;

%user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
%user_num_base1 = dbpath;
%user_num_base2 = strcat('jdbc:sqlite:',dbpath);
%conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);

   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=',"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);

start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

%            query_base = 'select count (*) from IP_logs where S_IP like "157.56.193.%" or D_IP like
"157.56.193.%" or S_IP like "157.56.126.%" or D_IP like "157.56.126.%" or S_IP like "157.56.192.%" or D_IP
like "157.56.192.%"';
          query_base = 'select count (*) from IP_logs where (S_IP like "157.56.193.%" or D_IP like
"157.56.193.%" or S_IP like "157.56.126.%" or D_IP like "157.56.126.%" or S_IP like "157.56.192.%" or D_IP
like "157.56.192.%") or (Tags="1" and S_port between "1024" and "65535" and D_port  between "1024" and
"65535")';
          limit_base = exec(conn, query_base);
          limit_base = fetch(limit_base);
          limit = cell2mat(limit_base.Data);
if (limit>0)
   while (i<=row)

%           if (start_a>count)
%              break;
%           end
           query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and ((S_IP like "157.56.193.%" or D_IP like "157.56.193.%" or S_IP like
"157.56.126.%" or D_IP like "157.56.126.%" or S_IP like "157.56.192.%" or D_IP like "157.56.192.%") or
(Tags="1" and S_port between "1024" and "65535" and D_port  between "1024" and "65535"))');
```

```
%             query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and S_IP like "157.56.193.%" or D_IP like "157.56.193.%" or S_IP like
"157.56.126.%" or D_IP like "157.56.126.%" or S_IP like "157.56.192.%" or D_IP like "157.56.192.%" or
Tags="1" and S_port between "1024" and "65535" and D_port  between "1024" and "65535"');
            limit1_base = exec(conn, query_base);
            limit1_base = fetch(limit1_base);
            limit1 = cell2mat(limit1_base.Data);
            limit2= limit2+limit1;

          query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and ((S_IP like "157.56.193.%" or D_IP like "157.56.193.%" or S_IP like
"157.56.126.%" or D_IP like "157.56.126.%" or S_IP like "157.56.192.%" or D_IP like "157.56.192.%") or
(Tags="1" and S_port between "1024" and "65535" and D_port  between "1024" and "65535"))');
          skype_rows = exec(conn, query_base1);
          skype_rows = fetch(skype_rows);

         if (limit1 ~= 0)
          for j=k+1:limit2
             if (limit2 > limit)
                break;
             end
             skype(x,:) = skype_rows.Data(p,:);
             p=p+1;
             if (x==q)
                user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - VoIP\Skype\');
                if exist(user_service_folder_loc,'dir')
                   user_service_matfile_loc = strcat(user_service_folder_loc, 'Skype',
num2str(matfile_count),'_tcp_udp');
                   save(user_service_matfile_loc,'skype');
                else
                   mkdir(user_service_folder_loc);
                   user_service_matfile_loc = strcat(user_service_folder_loc, 'Skype',
num2str(matfile_count),'_tcp_udp');
                   save(user_service_matfile_loc,'skype');
                end
%                  skypeudpinteraction(x,skype,skypeudp,c,user_service_folder_loc,matfile_count);
                matfile_count=matfile_count+1;
                clear skype;
                x=0;

             elseif (x==limit-((matfile_count-1)*q))
                   user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - VoIP\Skype\');
                if exist(user_service_folder_loc,'dir')
                   user_service_matfile_loc = strcat(user_service_folder_loc, 'Skype',
num2str(matfile_count),'_tcp_udp');
                   save(user_service_matfile_loc,'skype');
                else
                   mkdir(user_service_folder_loc);
                   user_service_matfile_loc = strcat(user_service_folder_loc, 'Skype',
num2str(matfile_count),'_tcp_udp');
                   save(user_service_matfile_loc,'skype');
                end
%                  skypeudpinteraction(x,skype,skypeudp,c,user_service_folder_loc,matfile_count);
                clear skype;
             end
%               if (p>limit1)
%                   break;
%                end
             x=x+1;
          end
        end
          k=limit2;
          p=1;
          start_a = start_b+1;
          start_b = start_b+10000;
          i=i+1;

  end
  %stat(user_num+1,5)={'1'};
  %save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');

  skype_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);
%else
  %stat(user_num+1,5)={'0'};
  %save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');

end
%fprintf('%d',i);

end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
skype_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath )
%UNTITLED22 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       ------------------------------------------------------------------------

iii=1;

skypehttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype',num2str(i),'_tcp_udp.mat'), 'skype');
        for ii=1:500000
            if (strcmp(cellstruct.skype(ii,3),'80')==1 || strcmp(cellstruct.skype(ii,5),'80')==1)
                skypehttpinteractions(iii,:)=cellstruct.skype(ii,:);
                iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype',num2str(i),'_tcp_udp.mat'), 'skype');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.skype(ii,3),'80')==1 || strcmp(cellstruct.skype(ii,5),'80')==1)
                    skypehttpinteractions(iii,:)=cellstruct.skype(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.skype(ii,3),'80')==1 || strcmp(cellstruct.skype(ii,5),'80')==1)
                    skypehttpinteractions(iii,:)=cellstruct.skype(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

for ii=1:(iii-1)
    skypehttpinteractions(ii,8)={'0'};
end




%               FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(skypehttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
for n=1:size(skypehttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(skypehttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%       d_port = bbchttpinteractions{n,5};
    tag = skypehttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = skypehttpinteractions{n,1};
        interaction_temp{counter1, 3} = skypehttpinteractions{n,2};
        interaction_temp{counter1, 4} = skypehttpinteractions{n,3};
        interaction_temp{counter1, 5} = skypehttpinteractions{n,4};
        interaction_temp{counter1, 6} = skypehttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
%           interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
        interaction_temp{counter1, 8} = 0; %data volume

%           interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = skypehttpinteractions{n,4};
        interaction_temp{counter2, 4} = skypehttpinteractions{n,5};
        interaction_temp{counter2, 5} = skypehttpinteractions{n,2};
        interaction_temp{counter2, 6} = skypehttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume

        flag1=1;
        counter1=counter1+2;
        counter2=counter2+2;

    end

    if (flag1>0)
     for j=1:size(skypehttpinteractions,1)
            s_ip = skypehttpinteractions{j,2};
            s_port = skypehttpinteractions{j,3};
            d_ip = skypehttpinteractions{j,4};
            d_port = skypehttpinteractions{j,5};
            tag = skypehttpinteractions{j,7};
      if (strcmp(skypehttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


        interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
        interaction_temp{counter1-2, 2} = skypehttpinteractions{j,1}; %end time
        interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(skypehttpinteractions{j,6}); %total data volume
        skypehttpinteractions(j,8) = {'1'};
                       sum_col_eight = sum_col_eight+1;
%           new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%                loopcounter1=loopcounter1+1;

       end
%      end
%
%      if (strcmp(bbchttpinteractions(j,8),'0')==1)
        if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
            if (counter3==0)
            interaction_temp{counter2-2, 1} = skypehttpinteractions{j,1};
            counter3=1;
          end
        interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
        interaction_temp{counter2-2, 2} = skypehttpinteractions{j,1}; %end time
        interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(skypehttpinteractions{j,6}); %total data volume
        skypehttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
      end
     end
     counter3=0;

   end





end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   %dir_path = strcat('C:\Users\djoy\Desktop\DataSet_for_testing\phptest\',num2str(user_num),'\Microsoft
Services - VoIP\Skype\');
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - VoIP\Skype\');
   mkdir(dir_path);
   save (strcat(dir_path,'Skype HTTP Interactions - Intial Table.mat'), 'interaction_temp');
   clear interaction_temp;

   existsornot = 1;    % If HTTP traffic does exist
   skype_http_interactions_final(x, matfile_count,user_num,existsornot,dbpath,case_name_fldr,primary_dbpath);
else          %If HTTP traffic doesn't exist
   existsornot = 0;
   skype_http_interactions_final(x, matfile_count,user_num,existsornot,dbpath,case_name_fldr,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
skype_http_interactions_final( x,matfile_count,user_num,existsornot,dbpath,case_name_fldr,primary_dbpath )
%UNTITLED23 Summary of this function goes here
%   Detailed explanation goes here
```

```
if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
   interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
   if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
      interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
   else
   interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
   end
   interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
   interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
   interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
   interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
   interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
   interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
   interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
   interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
   ii=ii+1;
%     i=i+2;
end
   save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mi
crosoft Services - VoIP\Skype\Skype HTTP Interactions - Final Table.mat'), 'interaction_final');

skype_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,dbpath,case_name_fldr,p
rimary_dbpath);
else          %No HTTP traffic exists
   interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

skype_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,dbpath,case_name_fldr,p
rimary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
skype_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,dbpath,case_name_fldr,
primary_dbpath )
%UNTITLED24 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%        ------------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
```

```
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
   matfile_count_val_for_udp_interactions = matfile_count-1;
   for i=1:(matfile_count-1)
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype',num2str(i),'_tcp_udp.mat'), 'skype');
       for ii=1:500000
          if (strcmp(cellstruct.skype(ii,3),'443')==1 || strcmp(cellstruct.skype(ii,5),'443')==1)
                httpsinteractions(iii,:)=cellstruct.skype(ii,:);
                iii=iii+1;
          end
       end
   end
else
   matfile_count_val_for_udp_interactions = matfile_count;
   for i=1:matfile_count
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype',num2str(i),'_tcp_udp.mat'), 'skype');
       if (i<matfile_count)
          for ii=1:500000
             if (strcmp(cellstruct.skype(ii,3),'443')==1 || strcmp(cellstruct.skype(ii,5),'443')==1)
                httpsinteractions(iii,:)=cellstruct.skype(ii,:);
                iii=iii+1;
             end
          end
       else
          for ii=1:(x-1)
             if (strcmp(cellstruct.skype(ii,3),'443')==1 || strcmp(cellstruct.skype(ii,5),'443')==1)
                httpsinteractions(iii,:)=cellstruct.skype(ii,:);
                iii=iii+1;
             end
          end
       end
   end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%           for ii=1:500000
%              if (bbc(ii,7)== 'S')
% %                  facebookinteractions()
%              end
%           end
%     else
%           for ii=1:(x-1)
%               if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                   httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                   iii=iii+1;
%               end
%           end
% %           for ii=1:(iii-1)
% %                 httpsinteractions(ii,8)={'0'};
% %           end
%
%     end
% end




%                           FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                           ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%               interaction_temp{counter1, 7} = 0;
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%               interaction_temp{counter1, 8} = 0;
%               interaction_temp{counter1, 9} = 0;
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%               interaction_temp{counter1, 10} = 0;


    if (timestamp2-timestamp1 < 6.9444e-06)

          s_ip = httpsinteractions{n,2};
          s_port = httpsinteractions{n,3};
          d_ip = httpsinteractions{n,4};
          d_port = httpsinteractions{n,5};
          tag = httpsinteractions{n,7};

%           if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
              fprintf('yes');
%               if isempty (interaction_temp{counter1, 1}==1)
%               interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%               end
              interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

              if (strcmp(counter_increment_status,'yes')==1)
                  if (strcmp(s_port,'443')==1)
                      interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                      interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                      interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                      interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                      interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                      counter_increment_status='no';
                  else
                      interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                      interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                      interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                      interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                      interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                      counter_increment_status='no';
                  end
              end

              if (isempty (interaction_temp{counter1, 7})==1)
              interaction_temp{counter1, 7} = 0;
              end
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
              if (isempty (interaction_temp{counter1, 8})==1)
              interaction_temp{counter1, 8} = 0;
              end
              if (isempty (interaction_temp{counter1, 9})==1)
              interaction_temp{counter1, 9} = 0;
              end
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
              if (isempty (interaction_temp{counter1, 10})==1)
              interaction_temp{counter1, 10} = 0;
              end

              if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%              interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%              interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%              interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%              interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                  if(strcmp(s_port,'443')==1)

                  interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (User to Service)
                  interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                      if (n == (size(httpsinteractions,1)-1))
                         interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                         interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                      end
                  else
                  interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
                  interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                      if (n == (size(httpsinteractions,1)-1))
                         interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                         interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                      end
                  end
              end
            if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%              interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%              interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)
                               fprintf('\na');
               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                     interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               else
                               fprintf('\nb');
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                     interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               end
            end
%              flag1=1;

          else
               if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                               fprintf('\na');
               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
               counter1=counter1+1;
               counter_increment_status='yes';
               elseif (strcmp(counter_increment_status,'no')==1)
                               fprintf('\nb');
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
               counter1=counter1+1;
               counter_increment_status='yes';
               end

          end

   else
      if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                        fprintf('\na');
      interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of Packets
(Service to User)
      interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
```

```
        counter1=counter1+1;
        counter_increment_status='yes';
        elseif (strcmp(counter_increment_status,'no')==1)
                          fprintf('\nb');
        interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                   % Number of Packets
(User to Service)
        interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
        counter1=counter1+1;
        counter_increment_status='yes';
        end

    end
%  end
end
    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - VoIP\Skype\');
    mkdir(dir_path);
    save (strcat(dir_path,'Skype HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
    if (existsornot==1)     %Both HTTP and HTTPS traffic exist
        http_https_interactions_combined = [interaction_final;interaction_temp];
        tcp_interactions_final = sortrows (http_https_interactions_combined,1);
        save (strcat(dir_path,'Skype TCP Interactions - Final Table.mat'), 'tcp_interactions_final');

udp_interactions_skype_preliminary(user_num,matfile_count_val_for_udp_interactions,tcp_interactions_final,d
bpath,case_name_fldr,primary_dbpath);  %Calling funtion for extracting UDP interactions

        %statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        %finalstat = statload1.finalstat;
        %finalstat(user_num+1,5)={'1'};
        %save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    else                   %Only HTTPS traffic exists; no HTTP
%         http_https_interactions_combined = interaction_temp;
        tcp_interactions_final = interaction_temp;
        save (strcat(dir_path,'Skype TCP Interactions - Final Table.mat'), 'tcp_interactions_final');

udp_interactions_skype_preliminary(user_num,matfile_count_val_for_udp_interactions,tcp_interactions_final,d
bpath,case_name_fldr,primary_dbpath);  %Calling funtion for extracting UDP interactions

        %statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        %finalstat = statload1.finalstat;
        %finalstat(user_num+1,5)={'1'};
        %save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    end
else                      % No HTTPS traffic exists
%         http_https_interactions_combined = interaction_final;
%         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%         tcp_interactions_final = interaction_final;
        if (existsornot==1)     % No HTTPS but HTTP exists
            dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
rosoft Services - VoIP\Skype\');
            mkdir(dir_path);

            tcp_interactions_final = interaction_final;
            save (strcat(dir_path,'Skype TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
```

```
udp_interactions_skype_preliminary(user_num,matfile_count_val_for_udp_interactions,tcp_interactions_final,d
bpath,case_name_fldr,primary_dbpath);  %Calling funtion for extracting UDP interactions

            %statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
            %finalstat = statload1.finalstat;
            %finalstat(user_num+1,5)={'1'};
            %save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        end
end
end
```

Stage 5: Analysis for initial level UDP traffic extraction

```
function [ skype_udp_valid_temp ] =
udp_interactions_skype_preliminary( user_num,matfile_count,tcp_interactions_final,dbpath,case_name_fldr,pri
mary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here

j=1;
% k=1;
l=1;
n=1;

%if (user_num >= 19)
%    user_num_copy=user_num+12;  %Only for users 19 to 27 (cause of the changes in the database files)
%else
%    user_num_copy=user_num;
%end


% matfile_count= cell2mat(cellstructc.matfile_count_cell(user_num,1));
skype_tcp_udp=cell(1,7);
% misc_udp=cell(1,7);
% skype_tcp=cell(1,8);
skype_udp_valid=cell(1,8);
skype_udp_valid_temp=cell(1,8);

for matfile_count1=1:matfile_count

cellstructa =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microsoft Services - VoIP\Skype\Skype',num2str(matfile_count1),'_tcp_udp.mat'), 'skype');
for i=1:size(cellstructa.skype,1)
    if (strcmp(cellstructa.skype(i,7),'1')==1)
%         if (strcmp(cellstructa.skype(i,2),'user36')==1 || strcmp(cellstructa.skype(i,4),'user36')==1)
        if (strcmp(cellstructa.skype(i,2),num2str(user_num))==1 ||
strcmp(cellstructa.skype(i,4),num2str(user_num))==1)
            if (str2double(cellstructa.skype(i,3))>=1024 && str2double(cellstructa.skype(i,5))>=1024)
                skype_tcp_udp(j,:)=cellstructa.skype(i,:);
                j=j+1;
%             else
%                 misc_udp(k,:)=cellstructa.skype(i,:);
%                 k=k+1;
            end
%         else
%             misc_udp(k,:)=cellstructa.skype(i,:);
%           k=k+1;
      end
   else
      skype_tcp_udp(j,:)=cellstructa.skype(i,:);
      j=j+1;
   end
end
clear cellstructa;
end

condition1=j;
skype_tcp_udp(:,8)={'0'};
j=1;
i=1;

if (condition1>1)
   for i=j:size(skype_tcp_udp,1)   %8955   %6333   %size(skype_tcp_udp,1)  %1606
   % while (i<=4781)
   % if (l>=i)
      if (strcmp(skype_tcp_udp(i,7),'S')==1)
        if (i>=j)

         s_ip = skype_tcp_udp(i,2);
         s_port = skype_tcp_udp(i,3);
         d_ip = skype_tcp_udp(i,4);
         d_port = skype_tcp_udp(i,5);

         skype_tcp_udp(i,8) = {'1'};
         skype_udp_valid(l,:)=skype_tcp_udp(i,:);
         l=l+1;


            for j=i+1:size(skype_tcp_udp,1)  %8955  %size(skype_tcp_udp,1) %1606
               if (strcmp(skype_tcp_udp(j,7),'F.')==1)
                  if (strcmp(s_ip,skype_tcp_udp(j,2))==1 && strcmp(s_port,skype_tcp_udp(j,3))==1 &&
strcmp(d_ip,skype_tcp_udp(j,4))==1 && strcmp(d_port,skype_tcp_udp(j,5))==1)
                     skype_tcp_udp(j,8) = {'1'};
                     skype_udp_valid(l,:)=skype_tcp_udp(j,:);
                     l=l+1;
                     i=j+1;
                     break;
                  elseif (strcmp(s_ip,skype_tcp_udp(j,4))==1 && strcmp(s_port,skype_tcp_udp(j,5))==1 &&
strcmp(d_ip,skype_tcp_udp(j,2))==1 && strcmp(d_port,skype_tcp_udp(j,3))==1)
                     skype_tcp_udp(j,8) = {'1'};
                     skype_udp_valid(l,:)=skype_tcp_udp(j,:);
                     l=l+1;
                     i=j+1;
                     break;
                  end

               else
                  if (strcmp(skype_tcp_udp(j,7),'1')==1)
   %                    if (str2double(skype_tcp_udp(j,3))>=1024 && str2double(skype_tcp_udp(j,5))>=1024)
                        skype_tcp_udp(j,8) = {'1'};
                        skype_udp_valid(l,:)=skype_tcp_udp(j,:);
                        l=l+1;
   %                    end
                  else
                     if (strcmp(skype_tcp_udp(j,7),'S')==1)
   %                       skype_tcp_udp(:,8)={'0'};
```

```
                        skype_udp_valid(l,:)={'0'};
                        l=l+1;
   %                       l=1;
   %                       i=j;
                        break;
                     end
                     skype_tcp_udp(j,8) = {'1'};
                     skype_udp_valid(l,:)=skype_tcp_udp(j,:);
                     l=l+1;
                  end
               end
            end

        end
      end
   end

   if (l>1)
      for m=1:size(skype_udp_valid,1)
         if (strcmp(skype_udp_valid(m,7),'1')==1)% || strcmp(skype_udp_valid(m,7),'0')==1)
            skype_udp_valid_temp(n,:) = skype_udp_valid(m,:);
            n=n+1;
         end
      end

         if (n>1)    %Calling function to extract the UDP interactions

udp_interactions_skype(skype_udp_valid_temp,user_num,tcp_interactions_final,dbpath,case_name_fldr,primar
y_dbpath);
         end
   end
else    % NO UDP OR TCP FOUND (NOT SURE IF THIS ELSE PART WILL EVER BE NEEDED!)
%     udp_interactions_skype(skype_udp_valid_temp,user_num,tcp_interactions_final,dbpath,case_name_fldr);
   user_num_fn_var = user_num;
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_fn_va
r),'\Microsoft Services - VoIP\Skype\');
   mkdir(dir_path);
   skype_tcp_udp_all_interactions_temp = tcp_interactions_final;
   skype_tcp_udp_all_interactions = sortrows(skype_tcp_udp_all_interactions_temp,1);
   save (strcat(dir_path,'Skype_tcp_udp_interactions_final'),'skype_tcp_udp_all_interactions');

   clear skype_tcp_udp_all_interactions;
%     user_num_fn_var = user_num;
   skype_user_activities_extraction_2_fn(user_num_fn_var,dbpath,case_name_fldr,primary_dbpath);
end


%     if (n>1)
%         udp_interactions_skype(skype_udp_valid_temp,user_num,user_num_copy);
%     end
end
```

Stage6: Analysis for final level UDP traffic extraction

```
function [ skype_udp_all_interactions,skype_tcp_udp_all_interactions ] =
udp_interactions_skype( skype_udp_valid_temp,user_num_fn_var,tcp_interactions_final,dbpath,case_name_fld
r,primary_dbpath )
```

```
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here

% httpsinteractions=cell(1,8);
interaction_temp=cell(1,10);
counter_increment_status='yes';
counter1=1;
b=1;
skype_udp_all_interactions=cell(1,10);

% cellstruct =
load('C:\Users\djoy\Documents\MATLAB\User_services5\Skype\user24\Valid_Skype_udp_temp','skype_udp_
valid_temp');
% skype_udp_valid_temp = cellstruct.skype_udp_valid_temp;
if (size(skype_udp_valid_temp)>1)
for n=1:size(skype_udp_valid_temp)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(skype_udp_valid_temp(n,1));
 timestamp2 = datenum(skype_udp_valid_temp(n+1,1));

%                interaction_temp{counter1, 7} = 0;
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%               interaction_temp{counter1, 8} = 0;
%               interaction_temp{counter1, 9} = 0;
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%               interaction_temp{counter1, 10} = 0;


    if (timestamp2-timestamp1 < 6.9444e-04) %Timestamp difference should be less than 1s

            s_ip = skype_udp_valid_temp{n,2};
            s_port = skype_udp_valid_temp{n,3};
            d_ip = skype_udp_valid_temp{n,4};
            d_port = skype_udp_valid_temp{n,5};
            tag = skype_udp_valid_temp{n,7};

%             if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,skype_udp_valid_temp{n+1, 2}) && strcmp(s_port,skype_udp_valid_temp{n+1, 3})
&& strcmp(d_ip, skype_udp_valid_temp{n+1, 4}) && strcmp(d_port, skype_udp_valid_temp{n+1, 5})) ||
(strcmp(s_ip,skype_udp_valid_temp{n+1, 4}) && strcmp(s_port,skype_udp_valid_temp{n+1, 5}) &&
strcmp(d_ip, skype_udp_valid_temp{n+1, 2}) && strcmp(d_port, skype_udp_valid_temp{n+1, 3})))
               fprintf('yes');
%               if isempty (interaction_temp{counter1, 1}==1)
%               interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%               end
               interaction_temp{counter1, 2} = skype_udp_valid_temp{n+1,1}; %End Time

               if (strcmp(counter_increment_status,'yes')==1)
                  if (strcmp(s_ip,num2str(user_num_fn_var))==1)
                     interaction_temp{counter1, 1} = skype_udp_valid_temp{n,1}; %Start Time
                     interaction_temp{counter1, 3} = skype_udp_valid_temp{n,2}; %User
                     interaction_temp{counter1, 4} = skype_udp_valid_temp{n,3}; %User port
                     interaction_temp{counter1, 5} = skype_udp_valid_temp{n,4}; %Service IP
                     interaction_temp{counter1, 6} = skype_udp_valid_temp{n,5}; %Service Port
                     counter_increment_status='no';
                  else
                     interaction_temp{counter1, 1} = skype_udp_valid_temp{n,1}; %Start Time
```

```
                    interaction_temp{counter1, 5} = skype_udp_valid_temp{n,2}; %User
                    interaction_temp{counter1, 6} = skype_udp_valid_temp{n,3}; %User port
                    interaction_temp{counter1, 3} = skype_udp_valid_temp{n,4}; %Service IP
                    interaction_temp{counter1, 4} = skype_udp_valid_temp{n,5}; %Service Port
                    counter_increment_status='no';
                end
            end

            if (isempty (interaction_temp{counter1, 7})==1)
            interaction_temp{counter1, 7} = 0;
            end
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
            if (isempty (interaction_temp{counter1, 8})==1)
            interaction_temp{counter1, 8} = 0;
            end
            if (isempty (interaction_temp{counter1, 9})==1)
            interaction_temp{counter1, 9} = 0;
            end
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
            if (isempty (interaction_temp{counter1, 10})==1)
            interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,skype_udp_valid_temp{n+1, 2}) && strcmp(s_port,skype_udp_valid_temp{n+1, 3})
&& strcmp(d_ip, skype_udp_valid_temp{n+1, 4}) && strcmp(d_port, skype_udp_valid_temp{n+1, 5})))
%               interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%               interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                if(strcmp(s_ip,num2str(user_num_fn_var))==1)

                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (User to Service)
                    if (n == (size(skype_udp_valid_temp,1)-1))
                        interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
                        interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n+1,6}); %Data Volume (User to Service)
                    end
                else
                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (Service to User)
                    if (n == (size(skype_udp_valid_temp,1)-1))
                        interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                        interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n+1,6}); %Data Volume (Service to User)
                    end
                end
            end

            if ((strcmp(s_ip,skype_udp_valid_temp{n+1, 4}) && strcmp(s_port,skype_udp_valid_temp{n+1, 5})
&& strcmp(d_ip, skype_udp_valid_temp{n+1, 2}) && strcmp(d_port, skype_udp_valid_temp{n+1, 3})))
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                if(strcmp(s_ip,num2str(user_num_fn_var))==1)
```

```
                              fprintf('\na');
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (User to Service)
                  if (n == (size(skype_udp_valid_temp,1)-1))
                    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (Service to User)
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n+1,6}); %Data Volume (Service to User)
                  end
                else
                              fprintf('\nb');
                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (Service to User)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (Service to User)
                  if (n == (size(skype_udp_valid_temp,1)-1))
                    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of
Packets (User to Service)
                    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n+1,6}); %Data Volume (User to Service)
                  end
                end
              end
%               flag1=1;

             else
                if(strcmp(skype_udp_valid_temp{n,2},num2str(user_num_fn_var))==1) &&
(strcmp(counter_increment_status,'no')==1)
                              fprintf('\na');
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (User to Service)
                counter1=counter1+1;
                counter_increment_status='yes';
                elseif (strcmp(counter_increment_status,'no')==1)
                              fprintf('\nb');
                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                 % Number of
Packets (Service to User)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (Service to User)
                counter1=counter1+1;
                counter_increment_status='yes';
                end

             end

         else
           if(strcmp(skype_udp_valid_temp{n,2},num2str(user_num_fn_var))==1) &&
(strcmp(counter_increment_status,'no')==1)
                         fprintf('\na');
           interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                 % Number of Packets
(User to Service)
           interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (User to Service)
           counter1=counter1+1;
           counter_increment_status='yes';
           elseif (strcmp(counter_increment_status,'no')==1)
```

```
                    fprintf('\nb');
        interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                    % Number of Packets
(Service to User)
        interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(skype_udp_valid_temp{n,6}); %Data Volume (Service to User)
        counter1=counter1+1;
        counter_increment_status='yes';
        end

    end
%  end
end
end

if (counter1>1) %Only if entries exist in the 'interaction_temp' cell

%     dir_path =
strcat('C:\Users\djoy\Documents\MATLAB\User_services5\Skype\user',num2str(user_num_fn_var),'\');
%     mkdir(dir_path);
%     save (strcat(dir_path,'udp_interactions1'),'interaction_temp');
    fprintf('%d',counter1);


% cellstruct = load
('C:\Users\djoy\Documents\MATLAB\User_services5\Skype\user24\udp_interactions','udp_interactions_fin');

    for a=1:size(interaction_temp,1)

        if (cell2mat(interaction_temp(a,7))>=40)
            temp2 = cell2mat(interaction_temp(a,7));
            temp3 = (temp2*20)/100;
            if (cell2mat(interaction_temp(a,9))>=(temp2-temp3) &&
cell2mat(interaction_temp(a,9))<=(temp2+temp3))
                skype_udp_all_interactions(b,:) = interaction_temp(a,:);
                b=b+1;
            end

        elseif (cell2mat(interaction_temp(a,9))>=40)
            temp2 = cell2mat(interaction_temp(a,9));
            temp3 = (temp2*20)/100;
            if (cell2mat(interaction_temp(a,7))>=(temp2-temp3) &&
cell2mat(interaction_temp(a,7))<=(temp2+temp3))
                skype_udp_all_interactions(b,:) = interaction_temp(a,:);
                b=b+1;
            end
        end
    end

        if (b>1)    %Saves only if there is any skype-udp-interactions
            dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_fn_va
r),'\Microsoft Services - VoIP\Skype\');
            mkdir(dir_path);
            save (strcat(dir_path,'udp_interactions1'),'interaction_temp');

%             save
(strcat('C:\Users\djoy\Documents\MATLAB\User_services5\Skype\user',num2str(user_num_fn_var),'\Skype_u
dp_interactions_final'),'skype_udp_all_interactions');
            save (strcat(dir_path,'Skype_udp_interactions_final'),'skype_udp_all_interactions');
            skype_tcp_udp_all_interactions_temp =
[skype_udp_all_interactions;tcp_interactions_final];  %Combining the TCP (https & http) and UDP interactions
            skype_tcp_udp_all_interactions = sortrows(skype_tcp_udp_all_interactions_temp,1);   %Sorting the
TCP/UDP interactions based on the start time
            save (strcat(dir_path,'Skype_tcp_udp_interactions_final'),'skype_tcp_udp_all_interactions');    %Saving
the all-final Skype interactions

            clear interaction_temp;
            clear skype_udp_all_interactions;
            clear skype_tcp_udp_all_interactions;
            %skype_user_activities_extraction_1_fn(user_num_fn_var,dbpath,case_name_fldr);     %CALLING
FUNCTION FOR FINDING USER ACTIVITIES (SKYPE)
            skype_user_activities_extraction_2_fn(user_num_fn_var,dbpath,case_name_fldr,primary_dbpath);
        else    % No udp interactions; so saving the tcp-interactions
            dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_fn_va
r),'\Microsoft Services - VoIP\Skype\');

            skype_tcp_udp_all_interactions_temp = tcp_interactions_final;
            skype_tcp_udp_all_interactions = sortrows(skype_tcp_udp_all_interactions_temp,1);
            save (strcat(dir_path,'Skype_tcp_udp_interactions_final'),'skype_tcp_udp_all_interactions');

            clear skype_tcp_udp_all_interactions;
            skype_user_activities_extraction_2_fn(user_num_fn_var,dbpath,case_name_fldr,primary_dbpath);
        end
else       %MIGHT NEED AN ELSE PART (WHICH MEANS NO UDP TRAFFIC FOR SKYPE EXISTS) TO
CALL THE USER-ACTIVITY-EXTRACTION-FUNCTION TO IDENTIFY ANY SKYPE ACTIVITY THAT
COMES FROM TCP TRAFFIC ALONE (e.g: text chat?). THIS MAY REQUIRE ADDING APPROPRIATE
PART TO THAT FUNCTION AS WELL.
    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_fn_va
r),'\Microsoft Services - VoIP\Skype\');

    skype_tcp_udp_all_interactions_temp = tcp_interactions_final;
    skype_tcp_udp_all_interactions = sortrows(skype_tcp_udp_all_interactions_temp,1);
    save (strcat(dir_path,'Skype_tcp_udp_interactions_final'),'skype_tcp_udp_all_interactions');

    clear skype_tcp_udp_all_interactions;
    skype_user_activities_extraction_2_fn(user_num_fn_var,dbpath,case_name_fldr,primary_dbpath);

end
end
```

Stage 7: Application of interaction-based approach to extract online user activities

```
function [ user_activity_array_temp_1 ] =
skype_user_activities_extraction_2_fn( user_num_fn_var,dbpath,case_name_fldr,primary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here

j=1;
k=1;
user_activity_array_temp = cell(1,10);
user_activity_array_temp_1 = cell(1,10);
user_activity_array = cell(1,12);
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_
fn_var),'\Microsoft Services -
VoIP\Skype\Skype_tcp_udp_interactions_final.mat'),'skype_tcp_udp_all_interactions');


for i=1:size(cellstructb.skype_tcp_udp_all_interactions,1)
   %CHECKING FOR TCP BASED ACTIVITIES (IDLE, CLICKING ON CONTACTS, TEXT CHATTING)
   if (strcmp((cellstructb.skype_tcp_udp_all_interactions(i,6)),'443')==1 &&
cellstructb.skype_tcp_udp_all_interactions{i,7}==1)
       if (cellstructb.skype_tcp_udp_all_interactions{i,8}==572 ||
cellstructb.skype_tcp_udp_all_interactions{i,8}==587)   %CHECKING FOR IDLE
           user_activity_array_temp_1(j,:) = cellstructb.skype_tcp_udp_all_interactions(i,:);
           j=j+1;
       end
       if (cellstructb.skype_tcp_udp_all_interactions{i,8}==731 ||
cellstructb.skype_tcp_udp_all_interactions{i,8}==747)   %CHECKING FOR CLICKING ON CONTACTS
           user_activity_array_temp_1(j,:) = cellstructb.skype_tcp_udp_all_interactions(i,:);
           j=j+1;
       end
       if (cellstructb.skype_tcp_udp_all_interactions{i,8}>794 &&
cellstructb.skype_tcp_udp_all_interactions{i,8}<=900)    %CHECKING FOR TEXT CHAT (base = 794, upto
106 characters)
           user_activity_array_temp_1(j,:) = cellstructb.skype_tcp_udp_all_interactions(i,:);
           j=j+1;
       end

   %CHECKING FOR UDP STREAM BASED ACTIVITIES (AUDIO CALL, VIDEO CALL, FILE
TRANSFER)
   elseif (strcmp(cellstructb.skype_tcp_udp_all_interactions(i,6),'443')==0 &&
strcmp(cellstructb.skype_tcp_udp_all_interactions(i,6),'80')==0)
       if (cell2mat(cellstructb.skype_tcp_udp_all_interactions(i,8))>0 ||
cell2mat(cellstructb.skype_tcp_udp_all_interactions(i,10))>0)
           user_activity_array_temp_1(j,:) = cellstructb.skype_tcp_udp_all_interactions(i,:);
           j=j+1;
       end
   end
end


for i=1:size(user_activity_array_temp_1,1)
   u1_to_u2_avg_pkt_size =
ceil(cell2mat(user_activity_array_temp_1(i,8))/cell2mat(user_activity_array_temp_1(i,7)));
   user_activity_array_temp_1(i,11) = {u1_to_u2_avg_pkt_size};   % Column 11 --> user1 to user2 average
packet size

   u2_to_u1_avg_pkt_size =
ceil(cell2mat(user_activity_array_temp_1(i,10))/cell2mat(user_activity_array_temp_1(i,9)));
   user_activity_array_temp_1(i,12) = {u2_to_u1_avg_pkt_size};   % Column 12 --> user2 to user1 average
packet size

   t1 = datevec(cell2mat(user_activity_array_temp_1(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating the
time duration of each activity
   t2 = datevec(cell2mat(user_activity_array_temp_1(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
   duration = etime(t2,t1);
   user_activity_array_temp_1(i,13) = {duration};  % Column 13 --> Duration of interaction

   % Column 14 --> Determining the activity
   if (str2double(user_activity_array_temp_1{i,6})==443)   % Checking if port number is 443 (TCP based
activities)
       if (user_activity_array_temp_1{i,11}==572 || user_activity_array_temp_1{i,11}==587)
           user_activity_array_temp_1{i,14} = 'Skype-Idle';
```

```
    end
    if (user_activity_array_temp_1{i,11}==731 || user_activity_array_temp_1{i,11}==747)
      user_activity_array_temp_1{i,14} = 'Skype-Click on contacts';
    end
    if (user_activity_array_temp_1{i,11}>794 && user_activity_array_temp_1{i,11}<=900)
      user_activity_array_temp_1{i,14} = 'Skype-Text message';
    end
  end
  if (str2double(user_activity_array_temp_1{i,6})>=1024)  % Checking if port number is equal to or greater
than 1024 (UDP based activities)
    if (u1_to_u2_avg_pkt_size>=61 && u1_to_u2_avg_pkt_size<=150)
      if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
        user_activity_array_temp_1{i,14} = 'Skype-Audio call';
      end
    end
    if(u1_to_u2_avg_pkt_size>=750 && u1_to_u2_avg_pkt_size<=1400)
      if (u2_to_u1_avg_pkt_size>=750 && u2_to_u1_avg_pkt_size<=1400)
        if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
          user_activity_array_temp_1{i,14} = 'Skype-Video call';
        end
      end
    end
    if(u1_to_u2_avg_pkt_size>=1000 && u1_to_u2_avg_pkt_size<=1400)
      if (u2_to_u1_avg_pkt_size>=27 && u2_to_u1_avg_pkt_size<=80)
        if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
          user_activity_array_temp_1{i,14} = 'Skype-File transfer (sending)'; % Large file
        end
      end
    end
    if(u1_to_u2_avg_pkt_size>=27 && u1_to_u2_avg_pkt_size<=80)
      if (u2_to_u1_avg_pkt_size>=1000 && u2_to_u1_avg_pkt_size<=1400)
        if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
          user_activity_array_temp_1{i,14} = 'Skype-File transfer (receiving)';   % Large file
        end
      end
    end
    if(u1_to_u2_avg_pkt_size>=480 && u1_to_u2_avg_pkt_size<=580)
      if (u2_to_u1_avg_pkt_size>=27 && u2_to_u1_avg_pkt_size<=80)
        if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
          user_activity_array_temp_1{i,14} = 'Skype-File transfer (sending)';  % Medium file
        end
      end
    end
    if(u1_to_u2_avg_pkt_size>=27 && u1_to_u2_avg_pkt_size<=80)
      if (u2_to_u1_avg_pkt_size>=480 && u2_to_u1_avg_pkt_size<=580)
        if (cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))>=0.8 &&
cell2mat(user_activity_array_temp_1(i,7))/cell2mat(user_activity_array_temp_1(i,9))<=1.3)
          user_activity_array_temp_1{i,14} = 'Skype-File transfer (receiving)';    % Medium file
        end
      end
    end
  end
%     else
%         user_activity_array_temp_1{i,14} = 'Skype-Unidentified';
%     end
```

```
    user_activity_array_temp_1{i,15} = 'Skype';
    user_activity_array_temp_1{i,16} = user_activity_array_temp_1{i,14};
    stime = user_activity_array_temp_1{i,1};
    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
    endtime = user_activity_array_temp_1{i,2};
    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
    user_activity_array_temp_1{i,17} = stime_formatted;
    user_activity_array_temp_1{i,18} = etime_formatted;
end

save(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num_
fn_var),'\Microsoft Services - VoIP\Skype\Identified_User_Activities'),'user_activity_array_temp_1');

    %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
matlab_table1 = cell2table(user_activity_array_temp_1,'VariableNames',matlab_table_col_names);

    %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
conn_db_1 = database(primary_dbpath,'','','org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`      TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
exec(conn_db_1,sql_query_1);
insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
```

Twitter

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_twitter_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here



%               CODE FOR EXTRACTING TRAFFIC - SERVICE --> TWITTER
%               ------------------------------------------------


i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
```

```
limit1=0;
limit2=0;
matfile_count=1;
twitter=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=','"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);


start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

            query_base = 'select count (*) from IP_logs where S_IP like "185.45.5.%" or D_IP like "185.45.5.%" or
S_IP like "104.244.42.%" or D_IP like "104.244.42.%" or S_IP like "199.96.57.%" or D_IP like "199.96.57.%"';
            limit_base = exec(conn, query_base);
            limit_base = fetch(limit_base);
            limit = cell2mat(limit_base.Data);
if (limit>0)
   while (i<=row)

%            if (start_a>count)
%               break;
%            end
            query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "185.45.5.%" or D_IP like "185.45.5.%" or S_IP like
"104.244.42.%" or D_IP like "104.244.42.%" or S_IP like "199.96.57.%" or D_IP like "199.96.57.%")');
            limit1_base = exec(conn, query_base);
            limit1_base = fetch(limit1_base);
            limit1 = cell2mat(limit1_base.Data);
            limit2= limit2+limit1;

           query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "185.45.5.%" or D_IP like "185.45.5.%" or S_IP like
"104.244.42.%" or D_IP like "104.244.42.%" or S_IP like "199.96.57.%" or D_IP like "199.96.57.%")');
           twitter_rows = exec(conn, query_base1);
           twitter_rows = fetch(twitter_rows);
```

```
        if (limit1 ~= 0)
         for j=k+1:limit2
            if (limit2 > limit)
               break;
            end
            twitter(x,:) = twitter_rows.Data(p,:);
            p=p+1;
            if (x==q)
               user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
               if exist(user_service_folder_loc,'dir')
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Twitter', num2str(matfile_count));
                  save(user_service_matfile_loc,'twitter');
               else
                  mkdir(user_service_folder_loc);
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Twitter', num2str(matfile_count));
                  save(user_service_matfile_loc,'twitter');
               end
               matfile_count=matfile_count+1;
               clear twitter;
               x=0;

            elseif (x==limit-((matfile_count-1)*q))
                  user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
               if exist(user_service_folder_loc,'dir')
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Twitter', num2str(matfile_count));
                  save(user_service_matfile_loc,'twitter');
               else
                  mkdir(user_service_folder_loc);
                  user_service_matfile_loc = strcat(user_service_folder_loc, 'Twitter', num2str(matfile_count));
                  save(user_service_matfile_loc,'twitter');
               end
               clear twitter;
            end
%              if (p>limit1)
%                 break;
%              end
            x=x+1;
         end
        end
         k=limit2;
         p=1;
         start_a = start_b+1;
         start_b = start_b+10000;
         i=i+1;

   end
%     stat(user_num+1,6)={'1'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
   twitter_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);

% else
%     stat(user_num+1,6)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);
```

```
end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
twitter_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED19 Summary of this function goes here
%   Detailed explanation goes here

%       TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%       ------------------------------------------------------------------------

iii=1;

twitterhttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
   for i=1:(matfile_count-1)
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter',num2str(i),'.mat'), 'twitter');
       for ii=1:500000
          if (strcmp(cellstruct.twitter(ii,3),'80')==1 || strcmp(cellstruct.twitter(ii,5),'80')==1)
                twitterhttpinteractions(iii,:)=cellstruct.twitter(ii,:);
                iii=iii+1;
          end
       end
   end
else
   for i=1:matfile_count
       cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter',num2str(i),'.mat'), 'twitter');
       if (i<matfile_count)
          for ii=1:500000
             if (strcmp(cellstruct.twitter(ii,3),'80')==1 || strcmp(cellstruct.twitter(ii,5),'80')==1)
                twitterhttpinteractions(iii,:)=cellstruct.twitter(ii,:);
                iii=iii+1;
             end
          end
       else
          for ii=1:(x-1)
             if (strcmp(cellstruct.twitter(ii,3),'80')==1 || strcmp(cellstruct.twitter(ii,5),'80')==1)
                twitterhttpinteractions(iii,:)=cellstruct.twitter(ii,:);
                iii=iii+1;
             end
          end
       end
```

```
    end
end

for ii=1:(iii-1)
    twitterhttpinteractions(ii,8)={'0'};
end




%               FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(twitterhttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
for n=1:size(twitterhttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(twitterhttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
    tag = twitterhttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = twitterhttpinteractions{n,1};
        interaction_temp{counter1, 3} = twitterhttpinteractions{n,2};
        interaction_temp{counter1, 4} = twitterhttpinteractions{n,3};
        interaction_temp{counter1, 5} = twitterhttpinteractions{n,4};
        interaction_temp{counter1, 6} = twitterhttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
%         interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
        interaction_temp{counter1, 8} = 0; %data volume

%         interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = twitterhttpinteractions{n,4};
        interaction_temp{counter2, 4} = twitterhttpinteractions{n,5};
        interaction_temp{counter2, 5} = twitterhttpinteractions{n,2};
        interaction_temp{counter2, 6} = twitterhttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume
```

```
      flag1=1;
      counter1=counter1+2;
      counter2=counter2+2;

   end

   if (flag1>0)
    for j=1:size(twitterhttpinteractions,1)
          s_ip = twitterhttpinteractions{j,2};
          s_port = twitterhttpinteractions{j,3};
          d_ip = twitterhttpinteractions{j,4};
          d_port = twitterhttpinteractions{j,5};
          tag = twitterhttpinteractions{j,7};
     if (strcmp(twitterhttpinteractions(j,8),'0')==1)
      if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


      interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
      interaction_temp{counter1-2, 2} = twitterhttpinteractions{j,1}; %end time
      interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(twitterhttpinteractions{j,6}); %total data volume
      twitterhttpinteractions(j,8) = {'1'};
                  sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
      m=m+1;
      a=n;
      loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

     end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
      if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
         if (counter3==0)
            interaction_temp{counter2-2, 1} = twitterhttpinteractions{j,1};
            counter3=1;
         end
      interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
      interaction_temp{counter2-2, 2} = twitterhttpinteractions{j,1}; %end time
      interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(twitterhttpinteractions{j,6}); %total data volume
      twitterhttpinteractions(j,8) = {'1'};
                  sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
      m=m+1;
      a=n;
      loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

     end
    end
   end
   counter3=0;

   end
```

```
end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
   mkdir(dir_path);
   save (strcat(dir_path,'Twitter HTTP Interactions - Intial Table.mat'), 'interaction_temp');

   existsornot = 1;    % If HTTP traffic does exist
   twitter_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else           %If HTTP traffic doesn't exist
   existsornot = 0;
   twitter_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
twitter_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath)
%UNTITLED20 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
   interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
   if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
      interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
   else
   interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
   end
   interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
   interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
   interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
   interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
   interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
   interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
   interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
   interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
```

```
    ii=ii+1;
%     i=i+2;
end
    save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mi
croblogging\Twitter\Twitter HTTP Interactions - Final Table.mat'), 'interaction_final');

twitter_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath);
else            %No HTTP traffic exists
    interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

twitter_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
twitter_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpath,
primary_dbpath)
%UNTITLED21 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%        ----------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter',num2str(i),'.mat'), 'twitter');
        for ii=1:500000
            if (strcmp(cellstruct.twitter(ii,3),'443')==1 || strcmp(cellstruct.twitter(ii,5),'443')==1)
                httpsinteractions(iii,:)=cellstruct.twitter(ii,:);
                iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter',num2str(i),'.mat'), 'twitter');
```

```
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.twitter(ii,3),'443')==1 || strcmp(cellstruct.twitter(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.twitter(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.twitter(ii,3),'443')==1 || strcmp(cellstruct.twitter(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.twitter(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%           for ii=1:500000
%               if (bbc(ii,7)== 'S')
% %                   facebookinteractions()
%               end
%           end
%     else
%           for ii=1:(x-1)
%               if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                   httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                   iii=iii+1;
%               end
%           end
% %           for ii=1:(iii-1)
% %                 httpsinteractions(ii,8)={'0'};
% %           end
%
%     end
% end





%                           FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                           ---------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%                 interaction_temp{counter1, 7} = 0;
% %                 interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%                interaction_temp{counter1, 8} = 0;
%                interaction_temp{counter1, 9} = 0;
% %                 interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%                interaction_temp{counter1, 10} = 0;


   if (timestamp2-timestamp1 < 6.9444e-06)

          s_ip = httpsinteractions{n,2};
          s_port = httpsinteractions{n,3};
          d_ip = httpsinteractions{n,4};
          d_port = httpsinteractions{n,5};
          tag = httpsinteractions{n,7};

%             if (flag1==0 && strcmp(tag, 'S'))
            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
             fprintf('yes');
%                if isempty (interaction_temp{counter1, 1}==1)
%                interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%                end
             interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

             if (strcmp(counter_increment_status,'yes')==1)
                if (strcmp(s_port,'443')==1)
                   interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                   interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                   interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                   interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                   interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                   counter_increment_status='no';
                else
                   interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                   interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                   interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                   interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                   interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                   counter_increment_status='no';
                end
             end

             if (isempty (interaction_temp{counter1, 7})==1)
             interaction_temp{counter1, 7} = 0;
             end
% %                 interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
             if (isempty (interaction_temp{counter1, 8})==1)
             interaction_temp{counter1, 8} = 0;
             end
             if (isempty (interaction_temp{counter1, 9})==1)
             interaction_temp{counter1, 9} = 0;
             end
% %                 interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
             if (isempty (interaction_temp{counter1, 10})==1)
             interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%              interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%              interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%              interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%              interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)

               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (User to Service)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (Service to User)
                     interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               else
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (Service to User)
                     interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               end
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%              interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%              interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)
                                    fprintf('\na');
               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (Service to User)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (Service to User)
                     interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               else
                                    fprintf('\nb');
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (Service to User)
```

```
                    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
                end
              end
%               flag1=1;

            else
                if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                                fprintf('\na');
                interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of
Packets (Service to User)
                interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                counter1=counter1+1;
                counter_increment_status='yes';
                elseif (strcmp(counter_increment_status,'no')==1)
                                fprintf('\nb');
                interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of
Packets (User to Service)
                interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                counter1=counter1+1;
                counter_increment_status='yes';
                end

            end

    else
        if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                        fprintf('\na');
        interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                % Number of Packets
(Service to User)
        interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
        counter1=counter1+1;
        counter_increment_status='yes';
        elseif (strcmp(counter_increment_status,'no')==1)
                        fprintf('\nb');
        interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                % Number of Packets
(User to Service)
        interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
        counter1=counter1+1;
        counter_increment_status='yes';
        end

    end
%  end
end
    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
    mkdir(dir_path);
    save (strcat(dir_path,'Twitter HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
    if (existsornot==1)     %Both HTTP and HTTPS traffic exist
        http_https_interactions_combined = [interaction_final;interaction_temp];
        tcp_interactions_final = sortrows (http_https_interactions_combined,1);
        save (strcat(dir_path,'Twitter TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
        twitter_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%       statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%       finalstat = statload1.finalstat;
%       finalstat(user_num+1,6)={'1'};
%       save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    else                %Only HTTPS traffic exists; no HTTP
        http_https_interactions_combined = interaction_temp;
        tcp_interactions_final = sortrows (http_https_interactions_combined,1);
        save (strcat(dir_path,'Twitter TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
        twitter_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%       statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%       finalstat = statload1.finalstat;
%       finalstat(user_num+1,6)={'1'};
%       save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
    end
else                % No HTTPS traffic exists
%       http_https_interactions_combined = interaction_final;
%       tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%       tcp_interactions_final = interaction_final;
        if (existsornot==1)     % No HTTPS but HTTP exists
            dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
            mkdir(dir_path);
            http_https_interactions_combined = interaction_final;
            tcp_interactions_final = sortrows (http_https_interactions_combined,1);
            save (strcat(dir_path,'Twitter TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
            twitter_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%               save (strcat(dir_path,'Twitter TCP Interactions - Final Table.mat'), 'interaction_final');

%               statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%               finalstat = statload1.finalstat;
%               finalstat(user_num+1,6)={'1'};
%               save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
        end
end
end
```

Stage 5: Application of interaction-based approach to extract online user activities

```
function [ twitter_user_activity_final ] =
twitter_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here



%   CODE FOR FINDING OUT THE TWITTER USER ACTIVITIES BASED ON THE EXTRACTED USER
INTERACTIONS
        %   ------------------------------------------------------------------------------------

j=1;
k=1;
m=1;
count=0;
count1=0;
```

```
twitter_user_activity_temp = cell(1,10);
twitter_user_activity_temp_1 = cell(1,18);
twitter_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user19\News\BBC\BBC TCP Interactions - Final
Table.mat', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Microblogging\Twitter\Twitter TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
   if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 || cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
      twitter_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
      j=j+1;
   end
end

for i=1:size(twitter_user_activity_temp,1)
   usr_to_srvr_avg_pkt_size =
ceil(cell2mat(twitter_user_activity_temp(i,8))/cell2mat(twitter_user_activity_temp(i,7)));
   twitter_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user to
BBC server)

   srvr_to_usr_avg_pkt_size =
ceil(cell2mat(twitter_user_activity_temp(i,10))/cell2mat(twitter_user_activity_temp(i,9)));
   twitter_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size (BBC
server to user)

   t1 = datevec(cell2mat(twitter_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating the
time duration of each interaction
   t2 = datevec(cell2mat(twitter_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
   duration = etime(t2,t1);
   twitter_user_activity_temp(i,13) = {duration};      %Column 13 --> Duration
   if (cell2mat(twitter_user_activity_temp(i,7))>=1 && cell2mat(twitter_user_activity_temp(i,9))>=1)
         twitter_user_activity_temp_1(k,1:13) = twitter_user_activity_temp(i,1:13);
         k=k+1;
   end
end



j=1;
if (size(twitter_user_activity_temp_1,1)>=1)
   for i=1:size(twitter_user_activity_temp_1,1)
      % CHECKING FOR TWEET    (192 base value, plus up to 144 characters which is the tweet limit)
      if (cell2mat(twitter_user_activity_temp_1 (i,7))==1 )
         if (cell2mat(twitter_user_activity_temp_1 (i,11))>=192 && cell2mat(twitter_user_activity_temp_1
(i,11))<=332)
            twitter_user_activity_temp_1(i,14) = {'Twitter-Tweeting'};
                  twitter_user_activity_temp_1{i,15} = 'Twitter';
                  twitter_user_activity_temp_1{i,16} = twitter_user_activity_temp_1{i,14};
                  stime = twitter_user_activity_temp_1{i,1};
                  stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                  endtime = twitter_user_activity_temp_1{i,2};
                  etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                  twitter_user_activity_temp_1{i,17} = stime_formatted;
                  twitter_user_activity_temp_1{i,18} = etime_formatted;

                  twitter_user_activity_final(j,1:18) = twitter_user_activity_temp_1(i,1:18);
```

```
                j=j+1;
            end
        end

        % CHECKING FOR UPLOAD
%           if (cell2mat(twitter_user_activity_temp_1 (i,7))==1 )
            if (cell2mat(twitter_user_activity_temp_1 (i,11))>=1000) %&& cell2mat(twitter_user_activity_temp_1
(i,11))<=332)
              if (cell2mat(twitter_user_activity_temp_1(i,7))>=30 &&
cell2mat(twitter_user_activity_temp_1(i,9))>=30)
                twitter_user_activity_temp_1(i,14) = {'Twitter-Upload'};
                  twitter_user_activity_temp_1{i,15} = 'Twitter';
                  twitter_user_activity_temp_1{i,16} = twitter_user_activity_temp_1{i,14};
                  stime = twitter_user_activity_temp_1{i,1};
                  stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                  endtime = twitter_user_activity_temp_1{i,2};
                  etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                  twitter_user_activity_temp_1{i,17} = stime_formatted;
                  twitter_user_activity_temp_1{i,18} = etime_formatted;

                  twitter_user_activity_final(j,1:18) = twitter_user_activity_temp_1(i,1:18);
                  j=j+1;
              end
            end
%           end


        % CHECKING FOR CLICKING ON CONTACT
        if (cell2mat(twitter_user_activity_temp_1 (i,7))==1 )
            if (cell2mat(twitter_user_activity_temp_1 (i,11))==747)
              twitter_user_activity_temp_1(i,14) = {'Twitter-Click on contacts'};
                  twitter_user_activity_temp_1{i,15} = 'Twitter';
                  twitter_user_activity_temp_1{i,16} = twitter_user_activity_temp_1{i,14};
                  stime = twitter_user_activity_temp_1{i,1};
                  stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                  endtime = twitter_user_activity_temp_1{i,2};
                  etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                  twitter_user_activity_temp_1{i,17} = stime_formatted;
                  twitter_user_activity_temp_1{i,18} = etime_formatted;

                  twitter_user_activity_final(j,1:18) = twitter_user_activity_temp_1(i,1:18);
                  j=j+1;
            end
        end



        % CHECKING FOR IDLE
        if (cell2mat(twitter_user_activity_temp_1 (i,9))>=2 && cell2mat(twitter_user_activity_temp_1 (i,9))<=10)
            if (cell2mat(twitter_user_activity_temp_1 (i,12))>=625 && cell2mat(twitter_user_activity_temp_1
(i,12))<=675)
              twitter_user_activity_temp_1(i,14) = {'Twitter-Idle'};
                  twitter_user_activity_temp_1{i,15} = 'Twitter';
                  twitter_user_activity_temp_1{i,16} = twitter_user_activity_temp_1{i,14};
                  stime = twitter_user_activity_temp_1{i,1};
```

```
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = twitter_user_activity_temp_1{i,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    twitter_user_activity_temp_1{i,17} = stime_formatted;
                    twitter_user_activity_temp_1{i,18} = etime_formatted;

                    twitter_user_activity_final(j,1:18) = twitter_user_activity_temp_1(i,1:18);
                    j=j+1;
          end
      end

   end

   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Mic
roblogging\Twitter\');
   mkdir(dir_path);
   save (strcat(dir_path,'Twitter_User_Activities.mat'), 'twitter_user_activity_final');


       %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
   matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
   matlab_table1 = cell2table(twitter_user_activity_final,'VariableNames',matlab_table_col_names);

       %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
   conn_db_1 = database(primary_dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
   sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
   exec(conn_db_1,sql_query_1);
   insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Wikipedia

Stage 1: Traffic metadata analysis for service traffic extraction

```
function [ x,matfile_count,user_num ] =
service_wikipedia_version_1_fn( dbpath,user_num,case_name_fldr,primary_dbpath )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here



%            CODE FOR EXTRACTING TRAFFIC - SERVICE --> WIKIPEDIA
%            --------------------------------------------------
```

```
i=1;
j=0;
x=1;
k=0;
p=1;
q=500000;
limit1=0;
limit2=0;
matfile_count=1;
wikipedia=cell(1,7);

% user_num_base1 = strcat('C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% user_num_base2 = strcat('jdbc:sqlite:C:\Users\djoy\Documents\MATLAB\user',num2str(user_num),'.db');
% conn = database(user_num_base1,",",'org.sqlite.JDBC',user_num_base2);
conn = database(dbpath,",",'org.sqlite.JDBC',strcat('jdbc:sqlite:',dbpath));

count_sql_query = 'select count (*) from IP_logs';
tot_num_rows = exec(conn, count_sql_query);
tot_num_rows = fetch(tot_num_rows);
count = cell2mat(tot_num_rows.Data);


   %   FOLLOWING 5 LINES OF CODE ARE TO STORE THE TOTAL NUM. OF PACKETS FOR THIS
USER INTO THE
   %   USERS_SERVICES_STAT DATABASE SO THAT IT COULD BE USED LATER TO PLOT
   %   THE CHARTS
count1=num2str(count);
conn_1 =
database(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\users_services_st
at.db'),",",'org.sqlite.JDBC',strcat('jdbc:sqlite:','C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case
_name_fldr,'\users_services_stat.db'));
count_sql_query_1 = strcat('update Stat set User_tot_num_pkts=',count1,' where Users=',"",user_num,"" );
User_tot_num_pkts = exec(conn_1, count_sql_query_1);
User_tot_num_pkts = fetch(User_tot_num_pkts);



start_a=1;
start_b=10000;
row=count/start_b;
row=ceil(row);

            query_base = 'select count (*) from IP_logs where S_IP like "91.198.174.%" or D_IP like
"91.198.174.%"';
            limit_base = exec(conn, query_base);
            limit_base = fetch(limit_base);
            limit = cell2mat(limit_base.Data);
if (limit>0)
   while (i<=row)

%           if (start_a>count)
%              break;
%           end
            query_base = strcat('SELECT count (*) from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "91.198.174.%" or D_IP like "91.198.174.%")');
            limit1_base = exec(conn, query_base);
            limit1_base = fetch(limit1_base);
            limit1 = cell2mat(limit1_base.Data);
            limit2= limit2+limit1;
```

```
        query_base1 = strcat('SELECT * from IP_logs where (rowid>=',num2str(start_a),' and
rowid<=',num2str(start_b),')',' and (S_IP like "91.198.174.%" or D_IP like "91.198.174.%")');
        wikipedia_rows = exec(conn, query_base1);
        wikipedia_rows = fetch(wikipedia_rows);

       if (limit1 ~= 0)
        for j=k+1:limit2
          if (limit2 > limit)
            break;
          end
          wikipedia(x,:) = wikipedia_rows.Data(p,:);
          p=p+1;
          if (x==q)
            user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
            if exist(user_service_folder_loc,'dir')
              user_service_matfile_loc = strcat(user_service_folder_loc, 'Wikipedia', num2str(matfile_count));
              save(user_service_matfile_loc,'wikipedia');
            else
              mkdir(user_service_folder_loc);
              user_service_matfile_loc = strcat(user_service_folder_loc, 'Wikipedia', num2str(matfile_count));
              save(user_service_matfile_loc,'wikipedia');
            end
            matfile_count=matfile_count+1;
            clear wikipedia;
            x=0;

          elseif (x==limit-((matfile_count-1)*q))
              user_service_folder_loc =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
            if exist(user_service_folder_loc,'dir')
              user_service_matfile_loc = strcat(user_service_folder_loc, 'Wikipedia', num2str(matfile_count));
              save(user_service_matfile_loc,'wikipedia');
            else
              mkdir(user_service_folder_loc);
              user_service_matfile_loc = strcat(user_service_folder_loc, 'Wikipedia', num2str(matfile_count));
              save(user_service_matfile_loc,'wikipedia');
            end
            clear wikipedia;
          end
%           if (p>limit1)
%             break;
%           end
          x=x+1;
        end
       end
        k=limit2;
        p=1;
        start_a = start_b+1;
        start_b = start_b+10000;
        i=i+1;

   end
%    stat(user_num+1,8)={'1'};
%    save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');

   wikipedia_http_interactions_initial(x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath);
%    wikipedia_https_interactions_final(x,matfile_count,user_num);
% else
%     stat(user_num+1,8)={'0'};
%     save('C:\Users\djoy\Documents\MATLAB\User_services3\Stat.mat', 'stat');
end
% fprintf('%d',i);


end
```

Stage 2: Analysis for initial level HTTP traffic extraction

```
function [ interaction_temp ] =
wikipedia_http_interactions_initial( x,matfile_count,user_num,dbpath,case_name_fldr,primary_dbpath)
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here


%         TO CREATE THE INITIAL-HTTP-INTERACTIONS TABLE FROM THE MAT FILES (FOR
WIKIPEDIA)
%         -------------------------------------------------------------------------------

iii=1;

wikipediahttpinteractions=cell(1,7);  %All http rows (input for the main part of the code)

flag1=0;
% flag2=0;
counter1=1;
counter2=2;
counter3=0;
sum_col_eight=0;
interaction_temp=cell(1,8); %All http interactions (including C->S and S->C side; before creating the final http
interaction table)


if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia',num2str(i),'.mat'), 'wikipedia');
        for ii=1:500000
            if (strcmp(cellstruct.wikipedia(ii,3),'80')==1 || strcmp(cellstruct.wikipedia(ii,5),'80')==1)
                  wikipediahttpinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                  iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia',num2str(i),'.mat'), 'wikipedia');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.wikipedia(ii,3),'80')==1 || strcmp(cellstruct.wikipedia(ii,5),'80')==1)
                    wikipediahttpinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.wikipedia(ii,3),'80')==1 || strcmp(cellstruct.wikipedia(ii,5),'80')==1)
                    wikipediahttpinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

for ii=1:(iii-1)
    wikipediahttpinteractions(ii,8)={'0'};
end




%                FOR CREATING THE INITIAL HTTP INTERACTIONS TABLE


m=1;
new=cell(1,8);
loopcounter=size(wikipediahttpinteractions,1);
% temp1=0;
% temp2=0;
% loopcounter1=0;
% unwanted=0;

% while (loopcounter>0)
% for y=1:size(bbchttpinteractions,1)
% while (strcmp(bbchttpinteractions(:,8),'1')==0)
% for loopcounter1=0: size(bbchttpinteractions,1)
% while (loopcounter1<size(bbchttpinteractions,1))
%     if (loopcounter + sum_col_eight == size(bbchttpinteractions,1))
%         break;
%     end
if (iii>1)
for n=1:size(wikipediahttpinteractions,1)
% for j=1:size(bbchttpinteractions,1)

if (strcmp(wikipediahttpinteractions(n,8),'0')==1)
%     s_ip = bbchttpinteractions{n,2};
%     s_port = bbchttpinteractions{n,3};
%     d_ip = bbchttpinteractions{n,4};
%     d_port = bbchttpinteractions{n,5};
    tag = wikipediahttpinteractions{n,7};

    if (flag1==0 && strcmp(tag, 'S'))  %&& strcmp(tag, 'S')
        interaction_temp{counter1, 1} = wikipediahttpinteractions{n,1};
        interaction_temp{counter1, 3} = wikipediahttpinteractions{n,2};
        interaction_temp{counter1, 4} = wikipediahttpinteractions{n,3};
        interaction_temp{counter1, 5} = wikipediahttpinteractions{n,4};
        interaction_temp{counter1, 6} = wikipediahttpinteractions{n,5};
        interaction_temp{counter1, 7} = 0;% number of packet
%           interaction_temp{counter1, 8} = str2double(bbchttpinteractions{n,6}); %data volume
        interaction_temp{counter1, 8} = 0; %data volume

%           interaction_temp{counter1+1, 1} = bbchttpinteractions{n,1};
```

```
        interaction_temp{counter2, 1} = '0';
        interaction_temp{counter2, 2} = '0';
        interaction_temp{counter2, 3} = wikipediahttpinteractions{n,4};
        interaction_temp{counter2, 4} = wikipediahttpinteractions{n,5};
        interaction_temp{counter2, 5} = wikipediahttpinteractions{n,2};
        interaction_temp{counter2, 6} = wikipediahttpinteractions{n,3};
        interaction_temp{counter2, 7} = 0;% number of packet
        interaction_temp{counter2, 8} = 0; %data volume

        flag1=1;
        counter1=counter1+2;
        counter2=counter2+2;

    end

    if (flag1>0)
     for j=1:size(wikipediahttpinteractions,1)
            s_ip = wikipediahttpinteractions{j,2};
            s_port = wikipediahttpinteractions{j,3};
            d_ip = wikipediahttpinteractions{j,4};
            d_port = wikipediahttpinteractions{j,5};
            tag = wikipediahttpinteractions{j,7};
      if (strcmp(wikipediahttpinteractions(j,8),'0')==1)
       if (strcmp(s_ip,interaction_temp{counter1-2, 3}) && strcmp(s_port,interaction_temp{counter1-2, 4}) &&
strcmp(d_ip, interaction_temp{counter1-2, 5}) && strcmp(d_port, interaction_temp{counter1-2, 6}))


        interaction_temp{counter1-2, 7} = interaction_temp{counter1-2, 7}+1;% number of packet
        interaction_temp{counter1-2, 2} = wikipediahttpinteractions{j,1}; %end time
        interaction_temp{counter1-2, 8} = interaction_temp{counter1-2,
8}+str2double(wikipediahttpinteractions{j,6}); %total data volume
        wikipediahttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;

       end
%       end
%
%       if (strcmp(bbchttpinteractions(j,8),'0')==1)
        if (strcmp(s_ip,interaction_temp{counter2-2, 3}) && strcmp(s_port,interaction_temp{counter2-2, 4}) &&
strcmp(d_ip, interaction_temp{counter2-2, 5}) && strcmp(d_port, interaction_temp{counter2-2, 6}))
            if (counter3==0)
                interaction_temp{counter2-2, 1} = wikipediahttpinteractions{j,1};
                counter3=1;
            end
        interaction_temp{counter2-2, 7} = interaction_temp{counter2-2, 7}+1;% number of packet
        interaction_temp{counter2-2, 2} = wikipediahttpinteractions{j,1}; %end time
        interaction_temp{counter2-2, 8} = interaction_temp{counter2-2,
8}+str2double(wikipediahttpinteractions{j,6}); %total data volume
        wikipediahttpinteractions(j,8) = {'1'};
                    sum_col_eight = sum_col_eight+1;
%         new(m,:)=bbchttpinteractions(n,:);
        m=m+1;
        a=n;
        loopcounter=loopcounter-1;
%               loopcounter1=loopcounter1+1;
```

```
       end
      end
     end
     counter3=0;

    end




end
% end
flag1=0;
end
%     flag1=0;
%     flag2=0;
%     n=1;
% end
    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
    mkdir(dir_path);
    save (strcat(dir_path,'Wikipedia HTTP Interactions - Intial Table.mat'), 'interaction_temp');

    existsornot = 1;    % If HTTP traffic does exist
    wikipedia_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
else          %If HTTP traffic doesn't exist
    existsornot = 0;
    wikipedia_http_interactions_final(x,
matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpath);
end
end
```

Stage 3: Analysis for final level HTTP traffic extraction

```
function [ interaction_final ] =
wikipedia_http_interactions_final( x,matfile_count,user_num,existsornot,case_name_fldr,dbpath,primary_dbpat
h )
%UNTITLED2 Summary of this function goes here
%   Detailed explanation goes here

if (existsornot==1)     %HTTP traffic exists
ii=1;
% matfile_count=1;
interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
cellstruct1 =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia HTTP Interactions - Intial Table.mat'), 'interaction_temp');
% num_rows = size(cellstruct1.interaction_temp,1);

for i=1:2:((size(cellstruct1.interaction_temp,1))-1)
    interaction_final(ii,1) = cellstruct1.interaction_temp(i,1);    %Start Time
    if (strcmp(cellstruct1.interaction_temp(i+1,2),'0')==1)
        interaction_final(ii,2) = cellstruct1.interaction_temp(i,2);
    else
    interaction_final(ii,2) = cellstruct1.interaction_temp(i+1,2);  %End Time
    end
    interaction_final(ii,3) = cellstruct1.interaction_temp(i,3);    %User
    interaction_final(ii,4) = cellstruct1.interaction_temp(i,4);    %User Port
    interaction_final(ii,5) = cellstruct1.interaction_temp(i,5);    %Service IP
    interaction_final(ii,6) = cellstruct1.interaction_temp(i,6);    %Service Port
    interaction_final(ii,7) = cellstruct1.interaction_temp(i,7);    %Number of frames (User to Service)
    interaction_final(ii,8) = cellstruct1.interaction_temp(i,8);    %Data Volume (User to Service)
    interaction_final(ii,9) = cellstruct1.interaction_temp(i+1,7);  %Number of frames (Service to User)
    interaction_final(ii,10) = cellstruct1.interaction_temp(i+1,8); %Data Volume (Service to User)
    ii=ii+1;
%     i=i+2;
end
    save
(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\On
line Encyclopedia\Wikipedia\Wikipedia HTTP Interactions - Final Table.mat'), 'interaction_final');

wikipedia_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpa
th,primary_dbpath);
else            %No HTTP traffic exists
    interaction_final = cell(1,10); %All http interactions (the final table created from the 'interaction_temp' table)
- in this case it will be empty

wikipedia_https_interactions_final(x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbpa
th,primary_dbpath);
end
end
```

Stage 4: Analysis for HTTPS traffic extraction

```
function [ tcp_interactions_final ] =
wikipedia_https_interactions_final( x,matfile_count,user_num,interaction_final,existsornot,case_name_fldr,dbp
ath,primary_dbpath )
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here

%        TO CREATE THE FINAL-HTTPS-INTERACTIONS TABLE FROM THE MAT FILES (FOR BBC)
%        -----------------------------------------------------------------------

% x=2949;   %for user16
% x=8080; %for user14
% x=187481;   %for user15
% matfile_count=2;
iii=1;
counter1=1;
counter_increment_status='yes';

httpsinteractions=cell(1,7);    %All https rows (input for the main part of code)
interaction_temp=cell(1,10);    %All https interactions (output)
tcp_interactions_final=cell(1,10);   %All tcp interactions (after http and https rows sorted based on timestamps)

if (x==0)
    for i=1:(matfile_count-1)
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia',num2str(i),'.mat'), 'wikipedia');
        for ii=1:500000
            if (strcmp(cellstruct.wikipedia(ii,3),'443')==1 || strcmp(cellstruct.wikipedia(ii,5),'443')==1)
```

```
                httpsinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                iii=iii+1;
            end
        end
    end
else
    for i=1:matfile_count
        cellstruct =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia',num2str(i),'.mat'), 'wikipedia');
        if (i<matfile_count)
            for ii=1:500000
                if (strcmp(cellstruct.wikipedia(ii,3),'443')==1 || strcmp(cellstruct.wikipedia(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                    iii=iii+1;
                end
            end
        else
            for ii=1:(x-1)
                if (strcmp(cellstruct.wikipedia(ii,3),'443')==1 || strcmp(cellstruct.wikipedia(ii,5),'443')==1)
                    httpsinteractions(iii,:)=cellstruct.wikipedia(ii,:);
                    iii=iii+1;
                end
            end
        end
    end
end

% save
('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user14\News\BBC\https','httpsinteraction
s');


% for i=1:matfile_count
% cellstruct = load(strcat('C:\Users\djoy\Documents\MATLAB\User_services4\user14\Social
Networking\Facebook\Facebook',num2str(i),'.mat'), 'facebook');
%     if (i<matfile_count)
%           for ii=1:500000
%               if (bbc(ii,7)== 'S')
% %                   facebookinteractions()
%               end
%           end
%     else
%           for ii=1:(x-1)
%               if (strcmp(cellstruct.facebook(ii,3),'443')==1 || strcmp(cellstruct.facebook(ii,5),'443')==1)
%                   httpsinteractions(iii,:)=cellstruct.facebook(ii,:);
%                   iii=iii+1;
%               end
%           end
% %             for ii=1:(iii-1)
% %                 httpsinteractions(ii,8)={'0'};
% %             end
%
%     end
% end
%                        FOR CREATING THE FINAL HTTPS INTERACTIONS TABLE
%                        ----------------------------------------------

if (iii>1)
for n=1:size(httpsinteractions,1)-1
% for n=1:116

%  if (strcmp(httpsinteractions(n,8),'0')==1)

 timestamp1 = datenum(httpsinteractions(n,1));
 timestamp2 = datenum(httpsinteractions(n+1,1));

%               interaction_temp{counter1, 7} = 0;
% %               interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
%               interaction_temp{counter1, 8} = 0;
%               interaction_temp{counter1, 9} = 0;
% %               interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
%               interaction_temp{counter1, 10} = 0;


   if (timestamp2-timestamp1 < 6.9444e-06)

         s_ip = httpsinteractions{n,2};
         s_port = httpsinteractions{n,3};
         d_ip = httpsinteractions{n,4};
         d_port = httpsinteractions{n,5};
         tag = httpsinteractions{n,7};

%            if (flag1==0 && strcmp(tag, 'S'))
           if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})) ||
(strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) && strcmp(d_ip,
httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
             fprintf('yes');
%               if isempty (interaction_temp{counter1, 1}==1)
%               interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
%               end
             interaction_temp{counter1, 2} = httpsinteractions{n+1,1}; %End Time

             if (strcmp(counter_increment_status,'yes')==1)
                if (strcmp(s_port,'443')==1)
                   interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                   interaction_temp{counter1, 5} = httpsinteractions{n,2}; %User
                   interaction_temp{counter1, 6} = httpsinteractions{n,3}; %User port
                   interaction_temp{counter1, 3} = httpsinteractions{n,4}; %Service IP
                   interaction_temp{counter1, 4} = httpsinteractions{n,5}; %Service Port
                   counter_increment_status='no';
                else
                   interaction_temp{counter1, 1} = httpsinteractions{n,1}; %Start Time
                   interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
                   interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
                   interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
                   interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
                   counter_increment_status='no';
                end
             end

             if (isempty (interaction_temp{counter1, 7})==1)
             interaction_temp{counter1, 7} = 0;
             end
```

```
% %              interaction_temp{counter1, 8} = str2double(httpsinteractions{n,6});
            if (isempty (interaction_temp{counter1, 8})==1)
            interaction_temp{counter1, 8} = 0;
            end
            if (isempty (interaction_temp{counter1, 9})==1)
            interaction_temp{counter1, 9} = 0;
            end
% %              interaction_temp{counter1, 10} = str2double(httpsinteractions{n+1,6});
            if (isempty (interaction_temp{counter1, 10})==1)
            interaction_temp{counter1, 10} = 0;
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 2}) && strcmp(s_port,httpsinteractions{n+1, 3}) &&
strcmp(d_ip, httpsinteractions{n+1, 4}) && strcmp(d_port, httpsinteractions{n+1, 5})))
%               interaction_temp{counter1, 3} = httpsinteractions{n,2}; %User
%               interaction_temp{counter1, 4} = httpsinteractions{n,3}; %User port
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)

               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (User to Service)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
                     interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               else
               interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (User to Service)
               interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                     interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
               end
            end

            if ((strcmp(s_ip,httpsinteractions{n+1, 4}) && strcmp(s_port,httpsinteractions{n+1, 5}) &&
strcmp(d_ip, httpsinteractions{n+1, 2}) && strcmp(d_port, httpsinteractions{n+1, 3})))
%               interaction_temp{counter1, 5} = httpsinteractions{n,4}; %Service IP
%               interaction_temp{counter1, 6} = httpsinteractions{n,5}; %Service Port
               if(strcmp(s_port,'443')==1)
                                    fprintf('\na');
               interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;                  % Number of
Packets (Service to User)
               interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
                  if (n == (size(httpsinteractions,1)-1))
                     interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;                  % Number of
Packets (Service to User)
                     interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
                  end
```

```
          else
                        fprintf('\nb');
          interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;           % Number of
Packets (User to Service)
          interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
            if (n == (size(httpsinteractions,1)-1))
              interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;           % Number of
Packets (Service to User)
              interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n+1,6}); %Data Volume (Service to User)
            end
          end
        end
%         flag1=1;

       else
          if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                        fprintf('\na');
          interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;           % Number of
Packets (Service to User)
          interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
          counter1=counter1+1;
          counter_increment_status='yes';
          elseif (strcmp(counter_increment_status,'no')==1)
                        fprintf('\nb');
          interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;           % Number of
Packets (User to Service)
          interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
          counter1=counter1+1;
          counter_increment_status='yes';
          end

       end

  else
    if(strcmp(httpsinteractions{n,3},'443')==1) && (strcmp(counter_increment_status,'no')==1)
                  fprintf('\na');
    interaction_temp{counter1, 9} = interaction_temp{counter1, 9}+1;           % Number of Packets
(Service to User)
    interaction_temp{counter1, 10} = interaction_temp{counter1, 10} +
str2double(httpsinteractions{n,6}); %Data Volume (Service to User)
    counter1=counter1+1;
    counter_increment_status='yes';
    elseif (strcmp(counter_increment_status,'no')==1)
                  fprintf('\nb');
    interaction_temp{counter1, 7} = interaction_temp{counter1, 7}+1;           % Number of Packets
(User to Service)
    interaction_temp{counter1, 8} = interaction_temp{counter1, 8} +
str2double(httpsinteractions{n,6}); %Data Volume (User to Service)
    counter1=counter1+1;
    counter_increment_status='yes';
    end

  end
%  end
end
```

```
   dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
   mkdir(dir_path);
   save (strcat(dir_path,'Wikipedia HTTPS Interactions - Final Table.mat'), 'interaction_temp');
% save (strcat('C:\Users\djoy\Documents\MATLAB\User_services_All_Interactions\user15\Social
Networking\Facebook\Facebook HTTPS Interactions - Final Table',num2str(i-1),'.mat'), 'interaction_temp');
   if (existsornot==1)     %Both HTTP and HTTPS traffic exist
       http_https_interactions_combined = [interaction_final;interaction_temp];
       tcp_interactions_final = sortrows (http_https_interactions_combined,1);
       save (strcat(dir_path,'Wikipedia TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
       wikipedia_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%         statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%         finalstat = statload1.finalstat;
%         finalstat(user_num+1,8)={'1'};
%         save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   else                  %Only HTTPS traffic exists; no HTTP
       http_https_interactions_combined = interaction_temp;
       tcp_interactions_final = sortrows (http_https_interactions_combined,1);
       save (strcat(dir_path,'Wikipedia TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
       wikipedia_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );
%         statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%         finalstat = statload1.finalstat;
%         finalstat(user_num+1,8)={'1'};
%         save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
   end
else                     % No HTTPS traffic exists
%         http_https_interactions_combined = interaction_final;
%         tcp_interactions_final = sortrows (http_https_interactions_combined,1);
%         tcp_interactions_final = interaction_final;
       if (existsornot==1)     % No HTTPS but HTTP exists
           dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
           mkdir(dir_path);
           http_https_interactions_combined = interaction_final;
           tcp_interactions_final = sortrows (http_https_interactions_combined,1);
           save (strcat(dir_path,'Wikipedia TCP Interactions - Final Table.mat'), 'tcp_interactions_final');
           wikipedia_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath );

%             statload1 = load('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
%             finalstat = statload1.finalstat;
%             finalstat(user_num+1,8)={'1'};
%             save('C:\Users\djoy\Documents\MATLAB\User_services3\Final_Stat.mat', 'finalstat');
       end
end
end
```

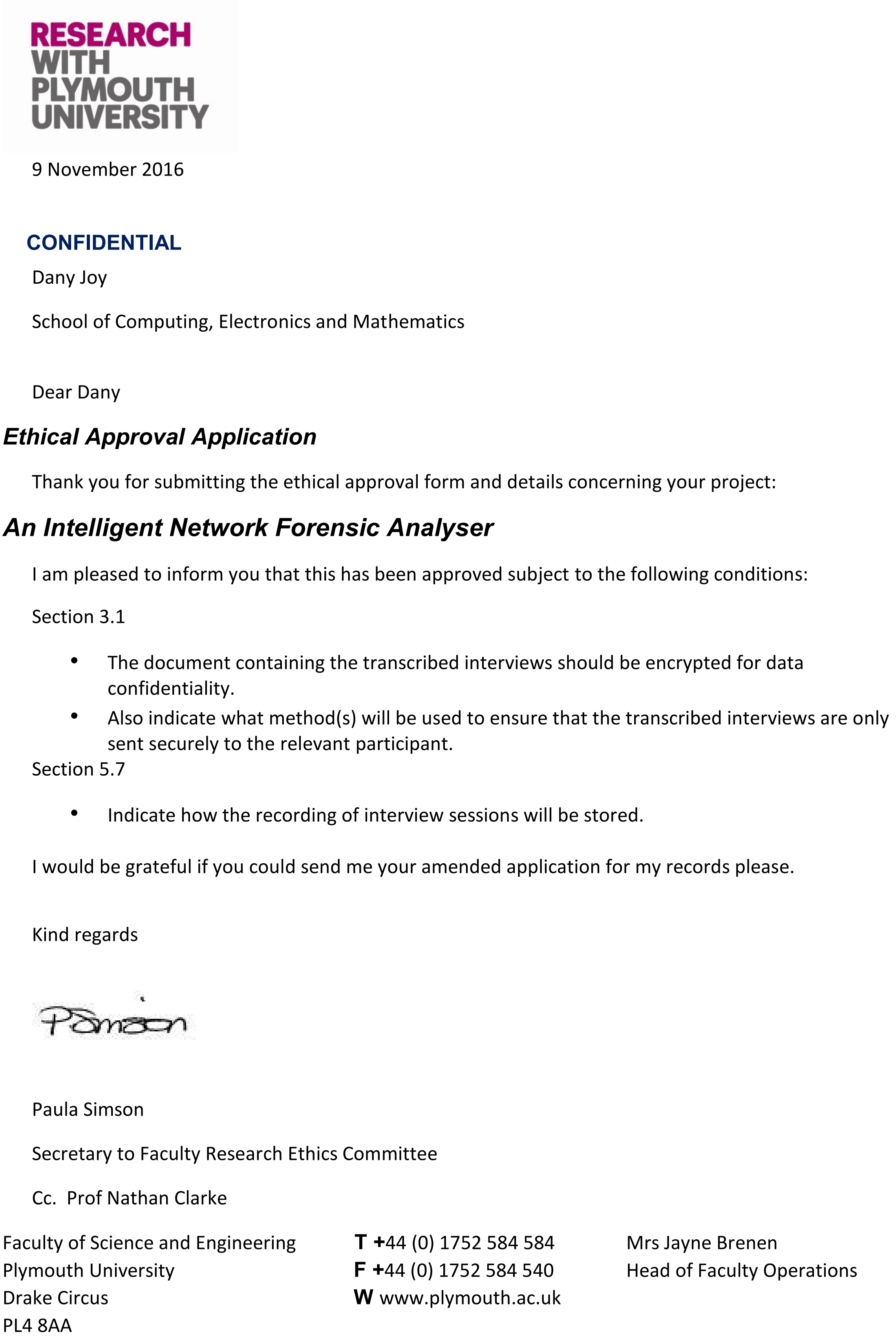
Stage 5: Application of interaction-based approach to extract online user activities

```
function [ wikipedia_user_activity_final ] =
wikipedia_user_activities_extraction_1_fn( user_num,case_name_fldr,dbpath,primary_dbpath )
%UNTITLED3 Summary of this function goes here
%   Detailed explanation goes here


       %   CODE FOR FINDING OUT THE WIKIPEDIA USER ACTIVITIES BASED ON THE EXTRACTED
USER INTERACTIONS
```

```
    %   -----------------------------------------------------------------------------------------

j=1;
k=1;
count=0;
wikipedia_user_activity_temp = cell(1,10);
wikipedia_user_activity_temp_1 = cell(1,18);
wikipedia_user_activity_final = cell(1,18);
%cellstructb = load('C:\Users\djoy\Documents\MATLAB\My results from System 2\Output after running code -
All services - Users 10 to 20\User_services_All_Interactions1\user20\Online
Encyclopedia\Wikipedia\Wikipedia TCP Interactions - Final Table.mat', 'tcp_interactions_final');
cellstructb =
load(strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'
\Online Encyclopedia\Wikipedia\Wikipedia TCP Interactions - Final Table.mat'),'tcp_interactions_final');

for i=1:size(cellstructb.tcp_interactions_final,1)  %Eliminating rows with zero packet size on both sides
  if (cell2mat(cellstructb.tcp_interactions_final(i,8))>0 && cell2mat(cellstructb.tcp_interactions_final(i,10))>0)
    wikipedia_user_activity_temp(j,:) = cellstructb.tcp_interactions_final(i,:);
    j=j+1;
  end
end

for i=1:size(wikipedia_user_activity_temp,1)

    usr_to_srvr_avg_pkt_size =
ceil(cell2mat(wikipedia_user_activity_temp(i,8))/cell2mat(wikipedia_user_activity_temp(i,7)));
    wikipedia_user_activity_temp(i,11) = {usr_to_srvr_avg_pkt_size};  %Column 11 --> Avg packet size (user
to Wikipedia server)

    srvr_to_usr_avg_pkt_size =
ceil(cell2mat(wikipedia_user_activity_temp(i,10))/cell2mat(wikipedia_user_activity_temp(i,9)));
    wikipedia_user_activity_temp(i,12) = {srvr_to_usr_avg_pkt_size};  %Column 12 --> Avg packet size
(Wikipedia server to user)

    t1 = datevec(cell2mat(wikipedia_user_activity_temp(i,1)),'yyyy.mm.dd.HH:MM:SS.FFF');    %Calculating
the time duration of each interaction
    t2 = datevec(cell2mat(wikipedia_user_activity_temp(i,2)),'yyyy.mm.dd.HH:MM:SS.FFF');
    duration = etime(t2,t1);
    wikipedia_user_activity_temp(i,13) = {duration};

  if (cell2mat(wikipedia_user_activity_temp(i,7))>=40 && cell2mat(wikipedia_user_activity_temp(i,9))>=40)
      wikipedia_user_activity_temp_1(k,1:13) = wikipedia_user_activity_temp(i,1:13);
      k=k+1;
  end
end

k=1;
j=1;
if (size(wikipedia_user_activity_temp_1,1)>=1)
  for k=1:size(wikipedia_user_activity_temp_1,1)
    %   CHECK FOR DOWNLOAD
    if (cell2mat(wikipedia_user_activity_temp_1 (k,12))>=1000 )%&&
cell2mat(wikipedia_user_activity_temp_1 (k,12))<=1400)
     if (cell2mat(wikipedia_user_activity_temp_1 (k,11))>=45 && cell2mat(wikipedia_user_activity_temp_1
(k,11))<=65)
        if (cell2mat(wikipedia_user_activity_temp_1(k,7))>=30 &&
cell2mat(wikipedia_user_activity_temp_1(k,9))>=30)
           if
(cell2mat(wikipedia_user_activity_temp_1(k,7))/cell2mat(wikipedia_user_activity_temp_1(k,9))>=0.7 &&
cell2mat(wikipedia_user_activity_temp_1(k,7))/cell2mat(wikipedia_user_activity_temp_1(k,9))<=1.3)
```

```
                wikipedia_user_activity_temp_1(k,14) = {'Wikipedia-Download'};

                    wikipedia_user_activity_temp_1{k,15} = 'Wikipedia';
                    wikipedia_user_activity_temp_1{k,16} = wikipedia_user_activity_temp_1{k,14};
                    stime = wikipedia_user_activity_temp_1{k,1};
                    stime_formatted = strcat('Date(',stime(1:4),',',num2str(str2double(stime(6:7))-
1),',',stime(9:10),',',stime(12:13),',',stime(15:16),',',stime(18:19),')');
                    endtime = wikipedia_user_activity_temp_1{k,2};
                    etime_formatted = strcat('Date(',endtime(1:4),',',num2str(str2double(endtime(6:7))-
1),',',endtime(9:10),',',endtime(12:13),',',endtime(15:16),',',endtime(18:19),')');
                    wikipedia_user_activity_temp_1{k,17} = stime_formatted;
                    wikipedia_user_activity_temp_1{k,18} = etime_formatted;

                    wikipedia_user_activity_final(j,1:18) = wikipedia_user_activity_temp_1(k,1:18);
                    j=j+1;
                end
              end
            end
          end
    end
    dir_path =
strcat('C:\xampp\htdocs\forlearning\PhpProject1\UONFAT_Cases\',case_name_fldr,'\',num2str(user_num),'\Onli
ne Encyclopedia\Wikipedia\');
    mkdir(dir_path);
    save (strcat(dir_path,'Wikipedia_User_Activities.mat'), 'wikipedia_user_activity_final');


      %   CREATING MATLAB TABLE FROM MATLAB CELL ARRAY
    matlab_table_col_names =
{'Start_Time','End_Time','User_1','User_1_Port','User_2','User_2_Port','User_1_Tot_packets','User_1_Tot_size',
'User_2_Tot_packets','User_2_Tot_size','User_1_Avg_pkt_size','User_2_Avg_pkt_size','Activity_Duration','Use
r_1_Activity','Application','Activity','Strt_time','End_time_to_plot'};
    matlab_table1 = cell2table(wikipedia_user_activity_final,'VariableNames',matlab_table_col_names);

      %   WRITING THE MATLAB TABLE (WAS CREATED RIGHT ABOVE) INTO THE NEW TABLE
CREATED (Skype) IN THE ORIGINAL USER DATABASE
    conn_db_1 = database(primary_dbpath,'','','org.sqlite.JDBC',strcat('jdbc:sqlite:',primary_dbpath));
    sql_query_1 = 'CREATE TABLE `All_activities` (`Start_Time` TEXT,`End_Time` TEXT,`User_1`
TEXT,`User_1_Port` TEXT,`User_2` TEXT,`User_2_Port`     TEXT,`User_1_Tot_packets`
INTEGER,`User_1_Tot_size` INTEGER,`User_2_Tot_packets` INTEGER,`User_2_Tot_size`
INTEGER,`User_1_Avg_pkt_size` INTEGER,`User_2_Avg_pkt_size` INTEGER,`Activity_Duration`
REAL,`User_1_Activity` TEXT,`Application` TEXT,`Activity` TEXT,`Strt_time` TEXT,`End_time_to_plot`
TEXT)';
    exec(conn_db_1,sql_query_1);
    insert(conn_db_1,'All_activities',matlab_table_col_names,matlab_table1);

end
end
```

Appendix C: Ethical Approval for Expert Evaluation



RESEARCH WITH PLYMOUTH UNIVERSITY

9 November 2016

CONFIDENTIAL

Dany Joy

School of Computing, Electronics and Mathematics

Dear Dany

Ethical Approval Application

Thank you for submitting the ethical approval form and details concerning your project:

An Intelligent Network Forensic Analyser

I am pleased to inform you that this has been approved subject to the following conditions:

Section 3.1

- The document containing the transcribed interviews should be encrypted for data confidentiality.
- Also indicate what method(s) will be used to ensure that the transcribed interviews are only sent securely to the relevant participant.

Section 5.7

- Indicate how the recording of interview sessions will be stored.

I would be grateful if you could send me your amended application for my records please.

Kind regards

Paula Simson

Secretary to Faculty Research Ethics Committee

Cc. Prof Nathan Clarke

Faculty of Science and Engineering	**T** +44 (0) 1752 584 584	Mrs Jayne Brenen
Plymouth University	**F** +44 (0) 1752 584 540	Head of Faculty Operations
Drake Circus	**W** www.plymouth.ac.uk	
PL4 8AA		