DETECTION OF REFLECTED CROSS-SITE SCRIPTING ATTACKS USING NETWORK FORENSICS

A DISSERTATION

Submitted in partial fulfillment of the requirements for the award of the degree of

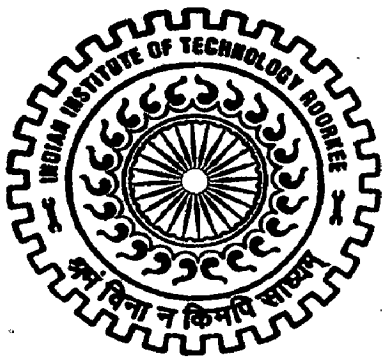
MASTER OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

PANKAJ KUMAR SHAHU



DEPARTMENT OF ELECTRONICS AND COMPUTER ENGINEERING
INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE-247 667 (INDIA)

JUNE, 2011

CANDIDATE'S DECLARATION

I hereby declare that the work, which is being presented in the dissertation entitled **"DETECTION OF REFLECTED CROSS-SITE SCRIPTING ATTACKS USING NETWORK FORENSICS"** towards the partial fulfillment of the requirement for the award of the degree of **Master of Technology** in **Computer Science and Engineering** submitted in the Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, Roorkee, Uttarakhand (India) is an authentic record of my own work carried out during the period from July 2010 to June 2011, under the guidance of **Dr. A.K.SARJE, Professor,** Department of Electronics and Computer Engineering, IIT Roorkee.

The matter presented in this dissertation has not been submitted by me for the award of any other degree of this or any other Institute.

Date: 27-6-2011
Place: Roorkee

Pankaj
(PANKAJ KUMAR SHAHU)

CERTIFICATE

This is to certify that the above statement made by the candidate is correct to the best of my knowledge and belief.

Date: 27-6-2011
Place: Roorkee

27/6/2011
(Dr. A.K.SARJE)
Professor
Department of Electronics and Computer Engineering
IIT Roorkee.

ACKNOWLEDGEMENTS

First and foremost, I would like to extend my heartfelt gratitude to my guide and mentor **Dr. A.K.Sarje**, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, for his invaluable advices, guidance, and encouragement and for sharing his broad knowledge. His wisdom, knowledge and commitment to the highest standards inspired and motivated me. He has been very generous in providing the necessary resources to carry out my research. He is an inspiring teacher, a great advisor, and most importantly a nice person.

I also wish to thank Emmanuel S. Pilli for his valuable suggestions and timely help regarding the domain knowledge. I am greatly indebted to all my friends, who have graciously applied themselves to the task of helping me with ample moral supports and valuable suggestions.

On a personal note, I owe everything to the Almighty and my parents. The support which I enjoyed from my father, mother and other family members provided me the mental support I needed.

PANKAJ KUMAR SHAHU

Abstract

Internet is facilitating numerous services while being the most commonly attacked environment. Hackers attack the vulnerabilities in the protocols used. Network forensics involves monitoring network traffic and determining if anomalies in the traffic indicate an attack. The network forensic techniques enable investigators to trace and prosecute the attackers. The network traffic is stored, examined and analyzed to detect attacks and discover their source. Cross-Site Scripting (also known as XSS) is one of the most common application layer hacking techniques. So there is a serious need to detect, prevent and identify the sources of attacks.

In this dissertation entitled "DETECTION OF REFLECTED CROSS-SITE SCRIPTING ATTACKS USING NETWORK FORENSICS", a framework is proposed which detects the source of Reflected Cross-site Scripting attack. We have run many malicious scripts on our website out of which our sytem detected almost all the attacks. The proposed strategy can be extended to increasing number of security attacks such as Stored XSS and DOM based XSS. The results validate the correctness of the framework and give extra information about the source of attacks.

Table of Contents

Candidate's Declaration & Certificate..... i

Acknowledgements..... ii

Abstract..... iii

Table of Contents..... iv

List of Figures.....vi

List of Tables..... vii

1. Introduction and Statement of the Problem **1**

1.1 Introduction..... 1

1.2 Motivation..... 2

1.3 Statement of the Problem..... 5

1.4 Organization of the Report..... 5

2. Background and Literature Review **7**

2.1 Network Forensics.....7

2.1.1 Classification of Network Forensics.....7

2.1.1 Network Forensics Process Model..... 8

2.2 Application Layer Related Attacks..... 11

2.2.1 Cross-site Scripting Attack.....12

2.2.2 Buffer Overflow Attack..... 15

2.2.3 SQL Injection Attack..... 16

2.2.4 Distributed Denial-of Service attack..... 16

2.2.5 Other Attacks.....17

2.3 Literature Review.....17

2.5 Research Gaps..... 20

3. Proposed Framework for Reflected XSS Attack Detection 21

3.1 Capture module..... 23

3.2 Extraction module..... 24

3.3 Storing module..... 25

3.4 Analysis module..... 25

4. Implementation of the Proposed Framework 27

4.1 Implementation..... 27

4.2 Testing..... 30

5. Results and Discussions 34

6. Conclusion and Future Work 38

6.1 Conclusion..... 38

6.2 Future Work..... 39

REFERENCES.....40

LIST OF PUBLICATIONS..... 44

LIST OF FIGURES

Figure 2.1 Network Forensics Process Model.....8

Figure 2.2 Forensic Data Sources in a Star Network.....10

Figure 2.3 Attacks on Application layer..... 11

Figure 2.4 Pie chart of different attacks..... 12

Figure 2.5 Scenario of Cross-site scripting Attack..... 13

Figure 3.1 Proposed Network Forensics System Architecture..... 22

Figure 4.1 Flow chart of making directory 28

Figure 4.2 Flow Chart of Finding Suspicious IP 29

Figure 4.3 Test Site login 31

Figure 4.4 Feedback form..... 32

Figure 4.5 XSS vulnerable page..... 32

Figure 4.6 Experimental setup..... 33

Figure 5.1 Request message from IP address 192.168.110.1..... 34

Figure 5.2 Response message from IP address 192.168.110.69..... 35

Figure 5.3 Request message from IP address 192.168.110.2..... 36

Figure 5.4 Source of Reflected XSS attack.....36

Figure 5.5 Pie chart of total detected attacks.....37

LIST OF FIGURES

Figure 2.1 Network Forensics Process Model.....8

Figure 2.2 Forensic Data Sources in a Star Network.....10

Figure 2.3 Attacks on Application layer..... 11

Figure 2.4 Pie chart of different attacks..... 12

Figure 2.5 Scenario of Cross-site scripting Attack..... 13

Figure 3.1 Proposed Network Forensics System Architecture..... 22

Figure 4.1 Flow chart of making directory 28

Figure 4.2 Flow Chart of Finding Suspicious IP 29

Figure 4.3 Test Site login 31

Figure 4.4 Feedback form..... 32

Figure 4.5 XSS vulnerable page..... 32

Figure 4.6 Experimental setup..... 33

Figure 5.1 Request message from IP address 192.168.110.1..... 34

Figure 5.2 Response message from IP address 192.168.110.69..... 35

Figure 5.3 Request message from IP address 192.168.110.2..... 36

Figure 5.4 Source of Reflected XSS attack.....36

Figure 5.5 Pie chart of total detected attacks.....37

LIST OF TABLES

Table 5.1 No. of Request message and Response message.....34

Table 5.2 No. of Request message and Response message..... 35

Table 5.3 Result of the attacks 37

Chapter 1

Introduction and Statement of the Problem

1.1 Introduction

The web has become a necessity in today's era. It is a great source of earning and a home to about 100 million websites. Global economies are highly dependent on the web. Every transaction of banking, auctions, shopping etc. is now web based. With an increase in dependence on the web, there arises a need to safeguard all such activities from different kinds of malicious anomalies in the network that are being introduced by various kinds of attackers. As a result the field of network security has become popular because it provides different mechanism, to protect the network from various kinds of attack. Thus security is a very crucial aspect because any disruption in the operation of the Internet can be very inconvenient for most of us. Even the most secure systems are plagued by new security threats such as Web Application Security, the term used to describe the methods of securing web-based software.

Firewalls and intrusion detection systems were brought in use for aiding in the task of network security in which they deal with preventing, detecting, mitigating and responding to such attacks. They lack any investigative features as they were not designed with forensics in mind. As it is a very difficult task to trace back the source of attack, the analysis, examination and reconstruction of the attack cannot be based on the firewall logs and intrusion detection alerts.

Thus, the new area named as "Network forensics" came into existence. Network forensics helps to enhance the network security from a different point of view. It deals with tracking the source of attack on a computer network by capturing, recording and analyzing network events. With increasing cases of cyber crime network security and network forensics have become a very important aspect of Information Technology. When attacks are successful, forensic techniques enable investigators to track the attackers. The main objective of network forensics is to provide sufficient evidence to allow the perpetrator to be prosecuted [1]. The network forensic analysis process involves

preparation, collection, preservation, examination, analysis, investigation and presentation phases [2]. The collection, examination, analysis and investigation phases are most challenging and difficult tasks of all.

The advantages of Network forensic are as follows [3]: It can be used to detect anomalies in network traffic behavior and determine if the anomaly is due to an actual attack or not. It can also be used to track the source of the attack by determining the IP address of the attacker, find out when the attack has been done, find out if the IP is a spoofed one or not.

Websites make use of complex web applications to provide their services to the users according to their preferences and specific needs. This helps them to provide better value to their customers. However, dynamic websites suffer from serious vulnerabilities rendering organizations helpless and prone to cross site scripting attacks on their data.

A web page contains both text and HTML markup that is generated by the server and interpreted by the client browser. Web sites that generate only static pages are able to have full control over how the browser interprets these pages[4]. Web sites that generate dynamic pages do not have complete control over how their outputs are interpreted by the client. The heart of the issue is that if mistrusted content can be introduced into a dynamic page, neither the web site nor the client has enough information to recognize that this has happened and take protective actions.

1.2 Motivation

The Internet is being used by increasing number of users day by day. A survey [5] shows that there are around having 100 million Web sites and 800 million users in World Wide Web are currently using the Internet. With the increasing number of Internet users, the cyber crimes also have increased worldwide to a great value. Many organizations and individuals are suffering due to increasing number of security breaches. More and more sophisticated internet crimes are being committed these days and it is very challenging task to investigate the source of attacks.

Google has revealed that the Gmail accounts of Chinese human rights activists were targeted in a hacking attempt committed on December 2009 [6]. Google says no actual emails were recovered; however they have evidence to suggest that a primary goal of the attackers was accessing the Gmail accounts of Chinese human rights activists. Only two Gmail accounts appear to have been accessed, and that activity was limited to account information (such as the date the account was created) and subject line, rather than the content of emails themselves. As part of their investigation they have discovered that at least twenty other large companies from a wide range of businesses–including the finance, technology, media and chemical sectors–have been similarly targeted. Also as part of its investigation, Google says that the accounts of dozens of US, China and Europe-based Gmail users who are advocates of human rights in China appear to have been routinely accessed by third parties. These accounts have been accessed but not because of a Google security breach but instead due to activists being victims of malware or phishing attempts.

On August 6, 2009, social networking sites like Twitter, Facebook and Google blogger suffered bouts of downtime, with each blaming a denial-of-service (DDoS) attack for its woes [7]. Facebook and Google could recover within a day while Twitter went offline for several hours .Twitter said on its status page that the site was "back up" , although many users reported seeing much of the site remaining inaccessible for the next few hours. Roger Thompson, chief research officer at computer security firm AVG Technologies, said in an e-mail to *InternetNews.com* that "Popular social networking sites, such as Facebook and Twitter, will always be targets to hackers or spammers and prone to attack, and as such, consumers become more vulnerable and run greater risks of becoming victims of online fraud".

Here is a review [8] of the major attacks seen in 2010-11:

On **31 may 2010** New critical XSS vulnerabilities reported for Skype and Vodafone websites by security researcher "Xylitol". Some days later two more Skype.com XSS vulnerabilities were reported by "Mystick". On **June 21, 2010** According to security

researcher "d3v1l" from Security-Shell, the Norton Update Center was vulnerable to cross-site scripting, redirects and html injections. Malicious people could exploit this vulnerability to redirect Norton product users to drive-by download pages and infect them with malware, adware and spyware. If this information would fall in the wrong hands of a phisher/carder, the financial details of millions of customers could be exposed.

On **June 23, 2010** Security researcher "Zeitjak", has notified that the NSA (National Security Agency) website is vulnerable to a new critical cross-site scripting vulnerability. Malicious people could exploit the XSS by launching a client-side attack against NSA's computers or browsers, with the purpose of obtaining classified information. On **July 5, 2010,** Researchers from a Romanian security team (InSecurityRomania) have revealed a critical persistent cross-site scripting (XSS) vulnerability which affects YouTube's comment field. Now the issue appears to be corrected (Google's Official Statement) but it is possible that malicious users have already exploited it to redirect unwitting YouTube users watching videos to drive-by download pages in order to infect them with malware, adware and spyware. On **July 28, 2010** XSS vunrelibilty discovered on Facebook that is rated as the second most popular website on the internet with 400 million active users can lead to account hijack.

On **September 6, 2010** Stefan Tanase from Kaspersky Labs wrote in the news article "Twitter XSS in the wild", cybercriminals maybe of Brazilian origin maliciously leveraged and exploited this Twitter XSS to steal user cookies and transfer them to two specific servers. On **October 1, 2010** XSS bugs on the websites of the world's largest payment/credit-card processors was found. Most of the world's financial institutions issue a Visa or a MasterCard to consumers. Even if their vulnerable sites do not hold real personal or financial information about consumers, malicious people can still leverage the XSS bugs with phishing techniques to trick millions of unwitting people into sharing sensitive information.

On **October 13, 2010** another critical cross-site scripting vulnerability has been reported by "See Me" for Amazon Seller Central, a secure website where sellers who signed up for the "Checkout by Amazon" service can view and manage their orders. On **March 30,**

2011 famous antivirus-security vendor McAfee has been all over the news, regarding cross-site scripting and information disclosure vulnerabilities that affected several of its websites. It all started when the Burmese-based YGN Ethical Hacker Group published the related details to the Full Disclosure mailing list Domain registrar *Register.com* is hit with a DDoS that causes several days of disruptions for its customers. Register.com is the eighth-largest registrar, managing 2.7 million domains.

The large number of security incidents affecting many organizations and increasing sophistication of these cyber attacks is the main driving force behind network forensics. Now-a-days most of the organizations are interested not only to detect and prevent these attacks but *to* reach the attacker back after an attack happens in the organization.

1.3 Statement of the Problem

To develop a system capable of detecting the source of Reflected Cross-site Scripting Attack by Network Forensics analysis.

In this dissertation, we have made an attempt to design and implement a framework to solve the mentioned problem using the following steps:

(i) Create a website on which the reflected Cross site Scripting attack would be launched by the attacker.

(ii) Using network forensics Analysis finding the source of Reflected Cross-site Scripting Attack.

1.4 Organization of the Report

This dissertation report comprises of six chapters including this chapter that introduces the topic and states the problem. The rest of the report is organized as follows.

Chapter 2 gives the background of network forensics, description of well known attacks on application layer, details of HTTP protocol and brief literature review of related work including research gaps.

Chapter 3 describes the framework designed for network forensic analysis for detection of Reflected Cross-site Scripting attack.

Chapter 4 gives the implementation details of the proposed framework, details of experiments performed.

Chapter 5 discusses the result obtained and the validation of the framework.

Chapter 6 concludes the dissertation work and gives suggestions for future work.

Chapter 2

Background and Literature Review

2.1 Network Forensics

Network forensics is the field of applying forensic science to the computer networks in order to discover the source of network crimes. The main aim of network forensics is to identify malicious activities from the traffic logs, search for their details, and to estimate the damage [3]. Network forensics is defined 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" [9].

Security policy violations are also being detected and investigated by network forensics. Forensic specialist, keep a track of the network continuously and makes a note of the relevant packets in the standard format. The stored packets information is further analyzed, either manually or by using different approaches, to find if there is an anomaly in the network and whether that anomaly is an attack. If any attack is found, the type of attack is determined and the source of the attack is investigated. Forensic experts can find the information about the attacker by proper monitoring, capturing, analyzing the network traffic and investigating.

2.1.1 Classification of Network Forensics

Network forensic systems are divided into two types each based on various characteristics that are as follows: [10]

- Purpose: 'General Network Forensics' to enhance network security & 'Strict Network Forensics' to get evidence satisfying the legal principles [2].

- Collection of Traffic: 'Catch-it-as-you-can' systems where all the packets passing through a particular traffic point are captured and analysis is subsequently done requiring large amounts of storage and 'Stop-look-and-listen' systems where each packet is analyzed in memory and certain information is saved for future analysis requiring a faster processor [11].

2.1.2 Network Forensics Process Model

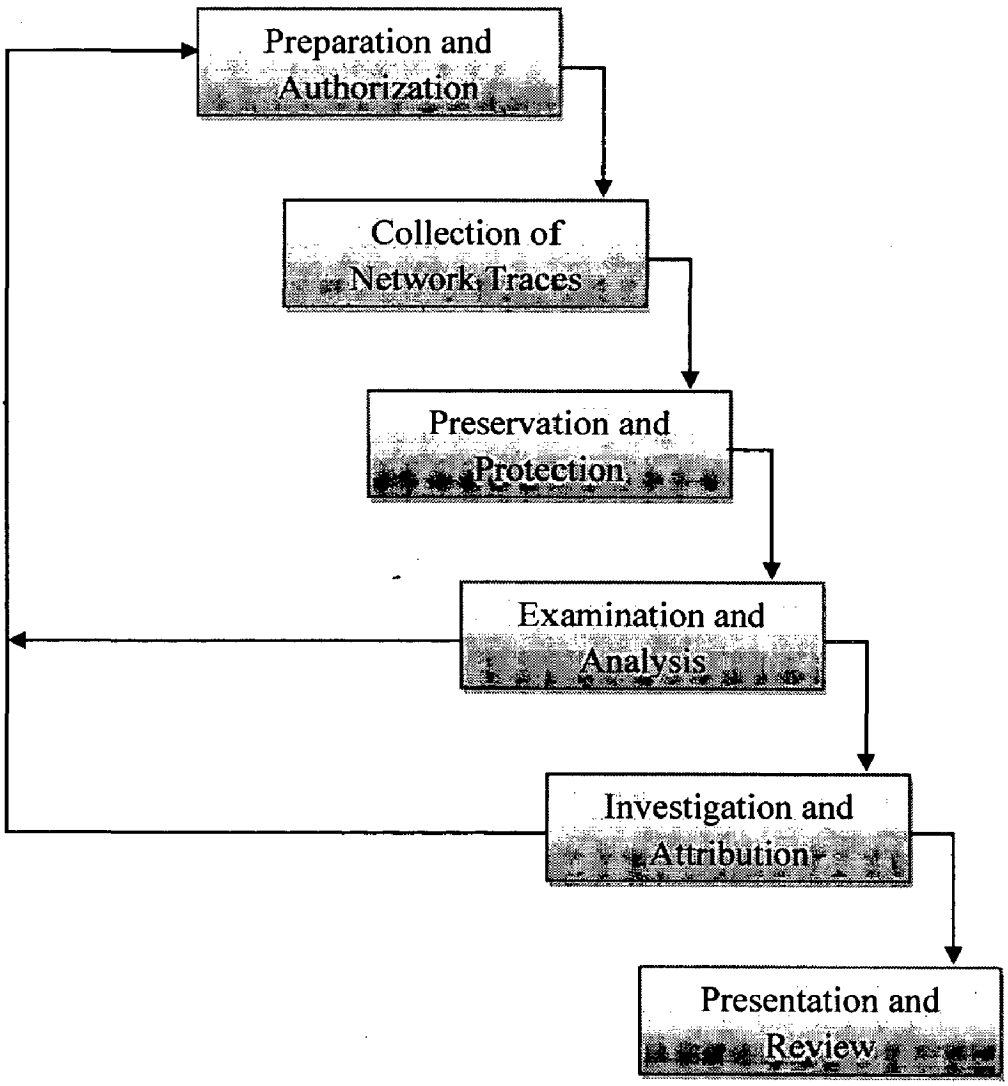


Figure 2.1 Network Forensics Process Model

A process model is formalized for network forensics based on the digital forensic models [2, 8, 12, 13, and 14]. It consists of the six steps: (i) Preparation and authorization (ii) Collection of network traces and logs (iii) Preservation and protection (iv) Examination and analysis (v) Investigation and attribution and (vi) Presentation and review. The network forensic investigators go through these processes in order to find the network attacks and their source. The generic framework for network forensics is shown in Figure 2.1.

Network forensics is applicable only to environments where network security tools (like intrusion detection systems, packet analyzers, and firewalls are deployed at various strategic points on the network. Data sources for forensic analysis are shown in Figure 2.2. Preparation is needed to deploy these tools and configure them. The required authorizations to monitor the network traffic are obtained so that privacy of individuals and the organization is not violated. Any unauthorized events and anomalies noticed will be analyzed. The presence and nature of the attack is determined from various parameters.

Data is acquired from the security tools are used to collect traffic data. The tools used must be secure, fault tolerant, have limited access and must be able to avoid compromise. The integrity of data logged and network events recorded must be ensured. Collection is the most difficult part as traffic data changes at a rapid pace and it is not possible to generate the same trace at a later time. The amount of data logged will be enormous requiring huge memory space and system must be able to handle different formats appropriately. The original data obtained in the form of traces and logs is stored on a back up device. A hash of all the trace data is taken and the data is protected. Another copy of the data will be used for analysis and the original collected network traffic is preserved. The traces obtained from various security tools are integrated and fused to form one large dataset on which analysis can be performed. Redundant information is removed and minimum representative attributes are identified so that the least information with the highest probable evidence is analyzed.

. The collected data is classified and clustered into groups so that the volume of data to be

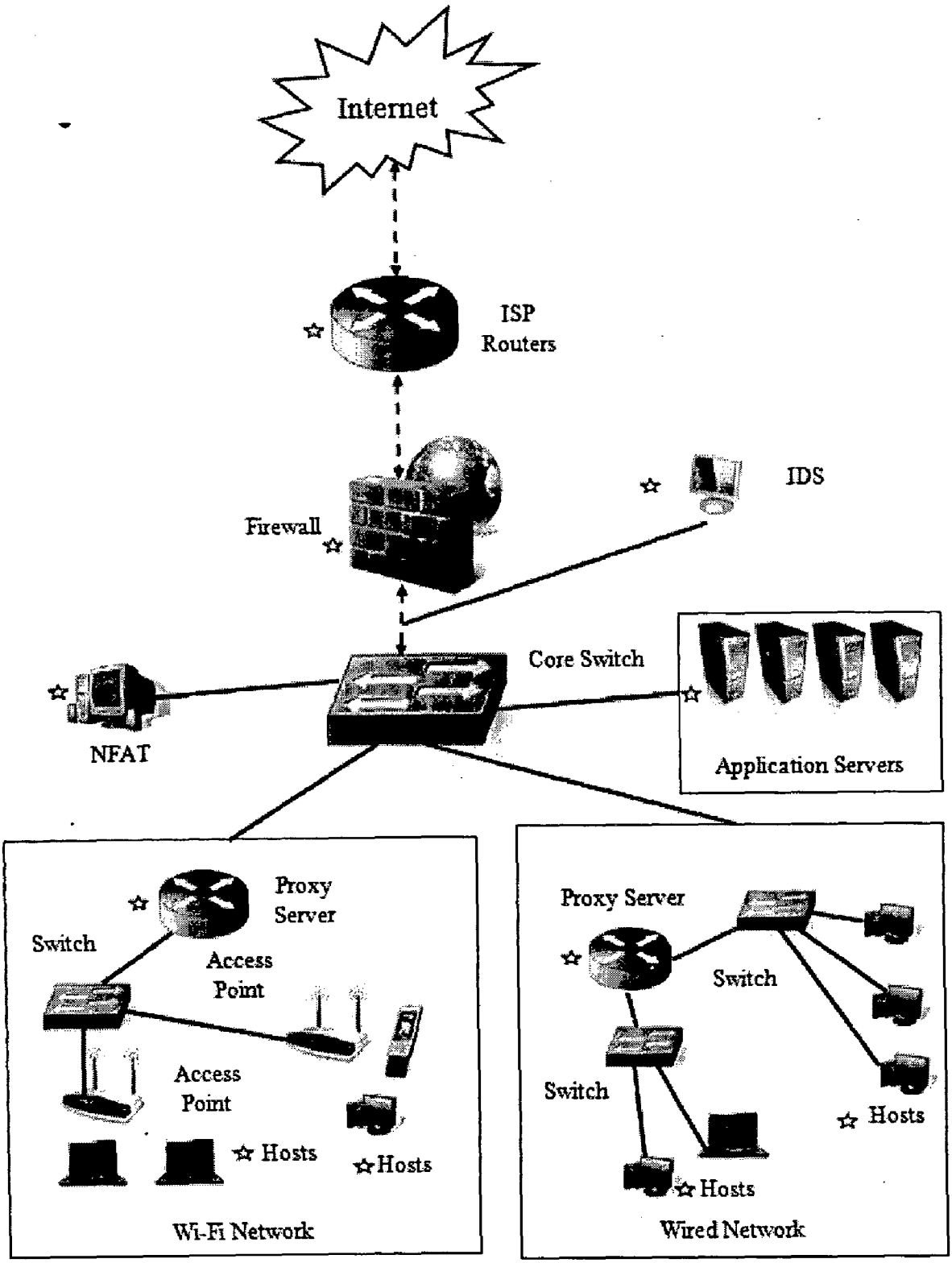


Figure 2.2 Forensic Data Sources in a star network

stored may be reduced to manageable chunks. The evidence collected is searched methodically to extract specific indicators of the crime. The indicators are classified and correlated to deduce important observations using the existing attack patterns. Statistical and data mining approaches are used to search the data and match attack patterns.

The information obtained from the evidence traces is used to identify who, what, where, when, how and why of the incident. The goal is to determine the path from a victim network or system through any intermediate systems and communication pathways, back to the point of attack origination. The packet captures and statistics obtained are used for attribution of the attack. This phase may require some additional features from the analysis phase and hence these two phases are iteratively performed to arrive at the conclusion.

The observations are presented in an understandable language to the organizations management and legal personnel while providing explanation of the various standard procedures used to arrive at the conclusion. The results are documented to influence future investigations and in improvement of security products.

2.2 Application Layer Related Attacks

A summary of major attacks on application layer is given below.

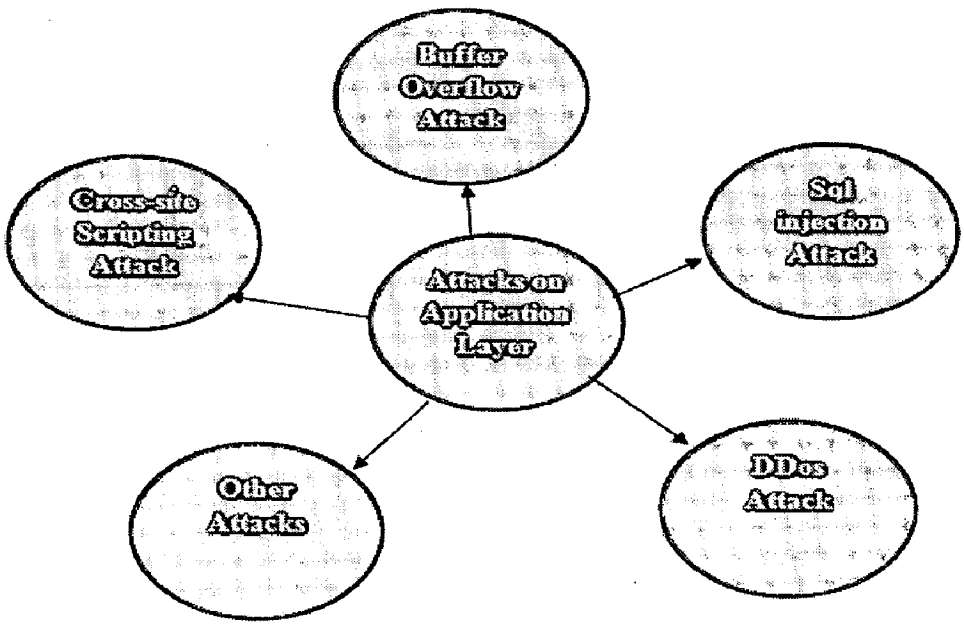


Fig 2.3 Various Application layer Attacks

2.2.1 Cross-site Scripting Attack

Cross-Site Scripting (also known as XSS) is one of the most common application layer hacking techniques. Cross-Site Scripting attacks are a type of injection problem, in which malicious scripts are injected into the trusted web sites. Cross-site scripting (XSS) attacks occur when an attacker uses a web application to send malicious code, to a different end user [15].

The pie-chart below, created by the Web Hacking Incident Database for 2011 (WHID) [16] clearly shows that whilst many different attack methods exist, SQL injection and XSS are the most popular. To add to this, many other attack methods, such as Information Disclosures, Content Spoofing and Stolen Credentials could all be side-effects of an XSS attack.

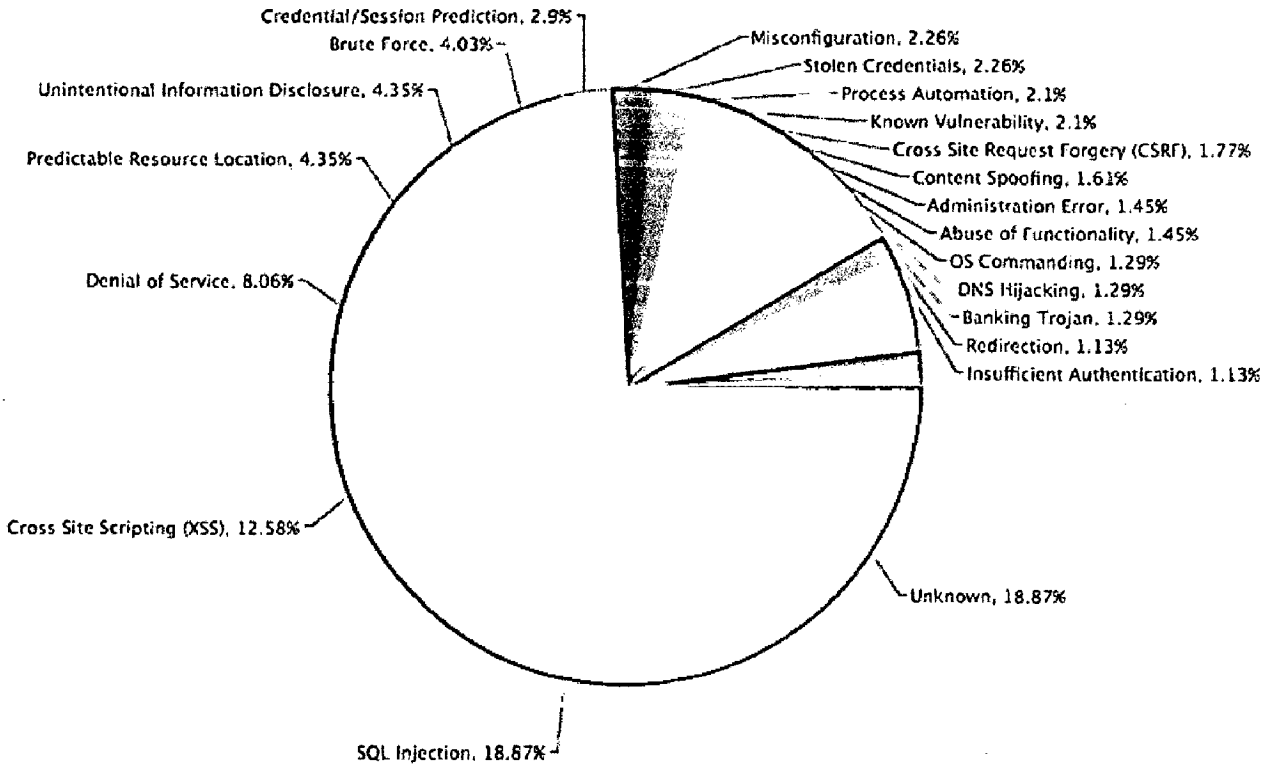


Fig 2.4 Pie chart of different attacks

In a typical XSS attack the hacker infects a legitimate web page with his malicious client-side script. When a user visits this web page, the script is downloaded to his browser and executed. There may be slight variations to this scenario, however all XSS attacks follow this pattern, which is shown in the figure2.6.

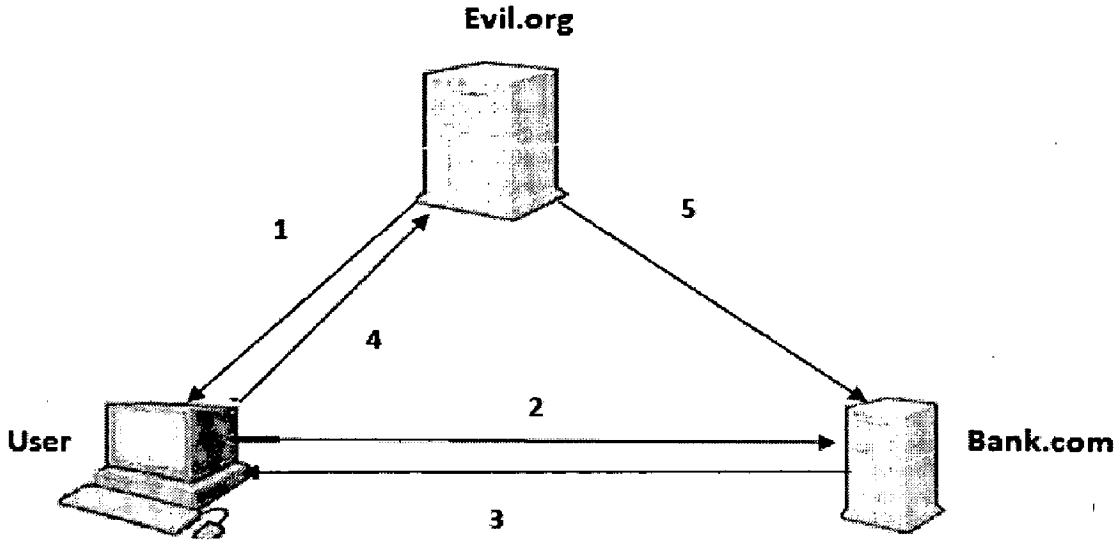


Fig 2.5 scenario of Cross-site scripting Attack

(1) Link of bank.com sent to user via HTTP by evil.org. Link can be send via email and can be post in any website feedback form, discussion forum.

(2) User sends the script embedded as data to bank.com.

(3) Script/data is returned and executed by the browser.

(4) Script sends user's cookie and session information without the user's consent or knowledge.

(5) Evil.org uses stolen session information to impersonate the user.

Various types of Cross-site Scripting attacks are given below [17].

(i)Reflected XSS

Reflected Cross-site scripting arises when the data Reflected XSS exploits occur when an attacker causes a user to supply dangerous content which includes malicious script (a

script which steals the cookie information), is submitted to a vulnerable web application, which is then reflected back to the user and executed by the web browser. The most common mechanism for delivering malicious content is to include it as a parameter in a URL that is posted publicly or e-mailed directly to victims. URLs constructed in this manner constitute the core of many phishing schemes, whereby an attacker convinces victims to visit a URL that refers to a vulnerable site. After the site reflects the attacker's content back to the user, the content is executed and proceeds to transfer private information, such as cookies that may include session information, from the user's machine to the attacker or perform other nefarious activities.

This type of attack is also known as First order XSS or non persistent XSS. The term First Order XSS is used for the type of attacks that introduces a payload that is delivered and executed by means of a single request and response. Approximately 75% of all Cross Site Scripting vulnerabilities encountered on the internet are the First Order XSS attacks.

(ii) Stored XSS

This vulnerability arises when the data having malicious script is submitted to a web application by a user, then the script is stored permanently on the server (typically in a back-end database or file system), and later displayed to other users on a web page without being properly filtered or sanitized. This type of attack is also known as Second order XSS.

Because second order XSS attacks involve specifically crafted malicious code that is first stored on the server, and then displayed to other users, the attack script is rendered more than once, enabling an attacker to affect multiple users with very little effort.

(iii) DOM-based XSS

DOM(document object model) based Cross-Site scripting attack is unique form of XSS, used very similarly to non-persistent XSS, but here the JavaScript malware payload does not need to be sent or echoed by the Web site to exploit a user.

In Dom-based XSS, attacker sends the URL of a maliciously constructed web page to the

user, using email or another mechanism. If user clicks on the link then the malicious web page's java script opens a vulnerable HTML page installed locally on the user's computer. The vulnerable HTML page contains JavaScript, which executes in the network of the user.

2.2.2 Buffer Overflow Attack

A buffer overflow is the condition wherein the data transferred to a buffer exceeds the storage capacity of the buffer and some of the data "overflows" into another buffer, one that the data was not intended to go into[18]. Since buffers can only hold a specific amount of data, when that capacity has been reached the data has to flow somewhere else, typically into another buffer, which can corrupt the data already in that buffer. Exploiting buffer overflow can lead to a serious system security breach (buffer-overflow attack) when necessary conditions are met. The seriousness of buffer-overflow attacks ranges from writing into another variable, another process memory (segmentation fault), or redirecting the program flow to execute malicious or unexpected code. The following c code is an example of this.

```
void authenticate (void)
        {
                char buffer[8];
                   int i ;
                for(i=0; i<16; i++)
            {
                buffer[i] = 'x';
             }
        }
```

In this function we have a buffer capable of holding eight ASCII characters .Assuming we are on a 32-bit system, this means 16 bytes of memory have been allocated to the buffer. We then place the buffer in an initialization loop and force-feed 15 "x" characters into it through programming error. Obviously they are not all going to fit, and nine of them must spill over into some other memory area.

2.2.3 SQL Injection Attack

SQL injection is an attack in which malicious code is inserted into strings that are later passed to an instance of SQL Server for parsing and execution. Any procedure that constructs SQL statements should be reviewed for injection vulnerabilities because SQL Server will execute all syntactically valid queries that it receives [19]. Even parameterized data can be manipulated by a skilled and determined attacker.

The primary form of SQL injection consists of direct insertion of code into user-input variables that are concatenated with SQL commands and executed. A less direct attack injects malicious code into strings that are destined for storage in a table or as metadata. When the stored strings are subsequently concatenated into a dynamic SQL command, the malicious code is executed

2.2.4 Distributed Denial-of-Service Attack

A distributed denial-of-service (DDoS) attack is one in which a multitude of compromised systems attack a single target, thereby causing denial of service for users of the targeted system [20]. The flood of incoming messages to the target system essentially forces it to shut down, thereby denying service to the system to legitimate users.

In a typical DDoS attack, a hacker begins by exploiting a vulnerability in one computer system and making it the DDoS master. It is from the master system that the intruder identifies and communicates with other systems that can be compromised. The intruder loads cracking tools available on the Internet on multiple sometimes thousands of compromised systems. With a single command, the intruder instructs the controlled machines to launch one of many flood attacks against a specified target.

2.2.5 Other attacks

Apart from above attacks some other attacks are given below:

(i) Insecure Storage

In this attack weak encryption techniques may lead to broken encryption that makes it possible for confidential information (SSN, Credit Cards) to decrypted by malicious users.

(ii) Insecure Configuration Management

Here, web application or database servers are configured insecurely, that causes attackers to exploit known vulnerabilities in the web, application or database servers.

2.3 Literature Review

In 2004 Wei [21] proposed a reference model of distributed cooperative network forensic system. It is based on client server architecture. The server captures network traffic, builds mapping topology database, filters, dumps and transforms the network traffic stream into database values, mines forensic database, and replays network behavior. It also does network surveying, attack statistic analysis and visualization. The distributed agent clients integrate data from firewall, IDS, Honeynet and remote traffic. The goal of this model is dumping the misbehavior packets traffic on the basis of adaptive filter rules, analyzing the overall cooperative database to discover the potential misbehavior, and replaying the misbehavior for the analysis of forensics. It can discover the profile of the attacker and obtain clues for further investigation. The limitations of this model are lack of privacy protection while traffic monitoring, inability to handle encrypted traffic and large traffic volumes etc.

In 2005, Wei and Jin [22] further extended the above model as distributed agent-based real time network intrusion forensics system. The goals of this framework include log system information gathering, adaptive capture of network traffic, active response for investigational forensics, integration of forensics data and storing the historical network

misuse pattern. The four elements in the system are network forensics server, network forensics agents, network monitor, and network investigator. Network forensics agents are engines of the data gathering, data extraction and data secure transportation. Network monitor is a packet capture machine which adaptively captures the network traffic. Network investigator is the network survey machine. Network forensics server integrates the forensics data, analyzes it and launches an investigation program on the network investigator. The model can expedite the investigation of the incident and improve the ability of emergence response. There is need to improve the co-operation between the agent and the server, to provide security to the agent and make it efficient to handle large traffic volumes.

Omar et al. [23] proposed a client-side system that automatically detects XSS vulnerability by manipulating either request or server response. Their system also shares the indication of vulnerability via a central repository.

Natarajan et al. [24] discussesed the different tools and techniques available to conduct network forensics. Some of the tools discussed include: eMailTrackerPro – to identify the physical location of an email sender; Web Historian – to find the duration of each visit and the files uploaded and downloaded from the visited website; packet sniffers like Ethereal – to capture and analyze the data exchanged among the different computers in the network.

Mohammad et al. [25] suggested a general model of propagation of Cross Site Scripting worms in virtual social network. They examined the effect of the friend visiting probability in such networks on the propagation of the worms. Then they analyzed the simulation results to see that the general model conforms to the simulation results.

Zhang et al. [26] presented an execution-flow analysis for JavaScript programs running in a web browser to prevent Cross-site Scripting (XSS) attacks. They constructed finite-state automata (FSA) to model the client-side behavior of Ajax applications under normal execution and the system is deployed in proxy mode. The proxy analyzes the execution flow of client-side JavaScript before the requested web pages arrive at the browser to prevent potentially malicious scripts, which do not conform to the FSA.

Emmanuel et al. [10] proposed a generic process model for network forensics is proposed which is built on various existing models of digital forensics. They made an exhaustive survey of various network forensic frameworks proposed and proposed a generic process model for network forensics which is built on various existing models
of digital forensics.

Galan et al. [27] introduced a novel multi-agent system for the automated scanning of web sites to detect the presence of XSS vulnerabilities exploitable by an stored-XSS Attack .

Atul Kaushik et al. [28] proposed a simple architecture for network forensics to overcome the problem of handling large volumes of network data and the resource intensive processing required for analysis. Their technique shows the improvement in the storage space required to store the log file of captured packets by capturing only the packets those are relevant for the analysis of port scanning attack. The system uses open source network security tools to collect and store the data.

Atul Kaushik et al.[29] discussed a model for collecting network data, identifying suspicious packets, examining protocol features misused and validating the attack. This model has been built with specific reference to security attacks on ICMP protocol. In this model packet capture file is analyzed for significant ICMP protocol features to mark suspicious packets.

2.4 Research Gaps

1. Most of the XSS detection techniques use web proxy. It is not guaranteed that the collected data from detecting /collecting proxies are 100% correct; false alert, human interference, and transmission interception may result in severe consequences.

2. There is no any generic model for the detection of Cross-site Scripting Attack. So there should be a feature in XSS detection model that can add new vulnerabilities that are related to XSS attack.

3. Thus a need for such a technique arises which is more efficient in terms of IP traceback. It can happen that the source of the attack is spoofed one.

4. Most of the techniques use client side detection mechanism for detection of XSS attack. Although there are some server side detection techniques, but they are not sufficient to prevent the attack.

Based on the above discussion we propose a scheme to detect Reflected Cross-Site scripting attack s with following area.

1. It should be able to detect a Cross-Site scripting Attack at sever side without using any kind of proxy.

2. It should able to capture the historical data and if the website is vulnerable to XSS attack and attack is launched in that website then it should be able to detect the source of attack.

Aiming to fulfill the above mentioned requirements, we proposed a framework for network forensic system for Cross-site Scripting attack, which is elaborated below.

Chapter 3

Proposed Framework for Reflected XSS Attack Detection

We propose an extended model for detection of Cross-Site scripting attack that was given by Kaushik et al. [28]. The model of Kaushik et al. capable of detecting only transport layer attacks such as port scanning attack and ICMP attacks. We extend the model for application layer attacks. We made some changes from the earlier model such as we are not including the marking module in this architecture because while capturing. The marking module is necessary in Port scanning attack and ICMP attack detection because in these attack we are concerned with only TCP-SYN, TCP-ACK packets etc. Here we have not any need for marking module because we run Wireshark with only http packet option that option automatically filters other packets.

There are three main modules in our proposal, namely Capture module, Extraction module and Analysis module. Capture module is further divided into two sub modules called XSS attack module, Capturing Network Traffic. A website is created on local host on which attack is done by the attacker . We run Wireshark on the background of the server so that we capture the raw data. Since we need only relevant information, we extract the raw data to find the relevant information. That's why we run Wireshark with only http packet option that option automatically filters other packets.

After extracting the relevant data, it is saved in hard disk drive for further analysis. The information obtained from the hard disk is used to identify who, what, where, when, how and why of the incident. This will help in source traceback, reconstruction of the attack scenario and find the source of the attack. Further we divided Analysis module in two sub module for clear understanding of the approach. Firstly we detect the possibility of reflected Cross-site scripting Attack, if it would be present then we try to find the source of Reflected XSS attack. We check if the request packet contains any malicious script, if this is the case and the response packet is also found to contain the same script we declare that an Reflected XSS attack occurred from this IP address. Two sub modules are given as Detection of attack and Tracing source of attack.

The model is for detection of Reflected Cross Site-site Scripting Attack shown in Figure 3.1 and explained below.

Proposed Network Forensics System Architecture

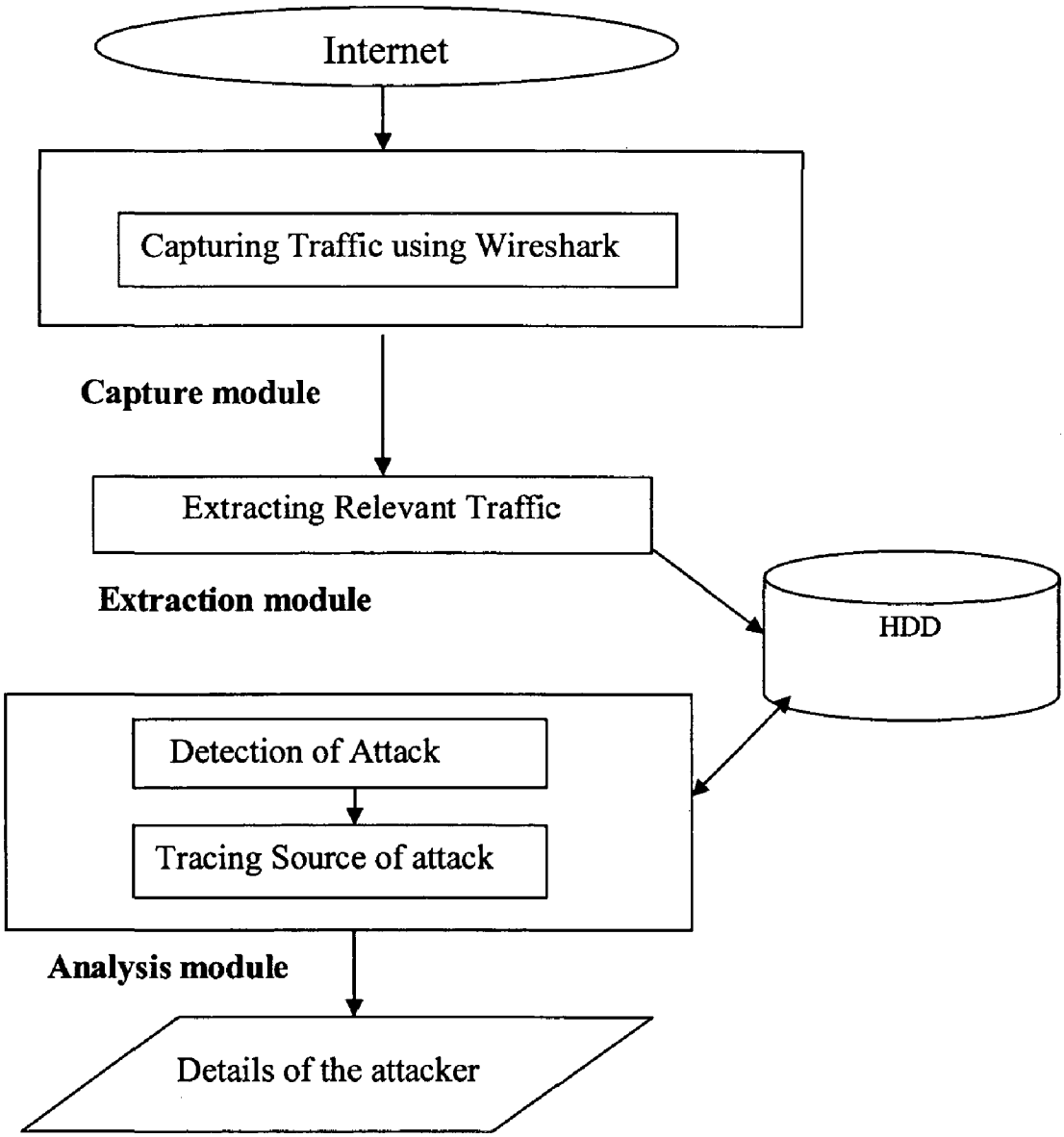


Figure 3.1 The overall architecture of the system

3.1 Capture Module

Basically there are two types of systems for collecting the data. Catch-it-as-you-can'systems and 'Stop-look-and-listen' systems [11]. In Catch-it-as-you-can'systems, all packets passing through a particular traffic point are captured and analysis is subsequently done requiring large amounts of storage and in 'Stop-look-and-listen' systems where each packet is analyzed in memory and certain information is saved for future analysis requiring a faster processor . Data is acquired from the tools used to collect traffic data. The tools used must be secure, fault tolerant, have limited access and must be able to avoid compromise. Capturing is the phase of storing all the raw traffic data flowing into the network of host system so a well defined procedure using reliable tools, hardware and software, must be in place to gather maximum evidence causing minimum impact to the victim. This phase is significant because the traffic stored during this phase, are the raw material for analysis. The increasing number of packet loss during this phase may lead to an incorrect analysis. Collection is the most difficult part as traffic data changes at a rapid pace and it is not possible to generate the same trace at a later time. The amount of data logged will be enormous requiring huge memory space. The network traces are collected from the victim system using packet capture tools like wireshark [30], snort [31], tcpdump [32] etc.

The brief description of packet capture tools is given below:

Wireshark: We use Wireshark for capturing the raw data over the network. Wireshark is a free and open-source packet analyzer. It has a graphical front-end and some information sorting and filtering options. Wireshark allows the user to put the network interfaces that support promiscuous mode into that mode, in order to attempt to see all traffic being passed over the network . Wireshark uses pcap to capture packets, so it can only capture the packets on the types of networks that pcap supports. Libpcap [33] is the library developed by the developers of tcpdump to perform low-level packet(network traffic) capture; read and write capture files and analyze them. Its file extension is *.pcap*. Wireshark can filter packets on many criteria. We collect traffic with "only HTTP" option.

Snort: Snort is an open source intrusion detection system which is capable of capturing the network traffics and optionally logs those captured traffic into some storage device. Snort provides the raw input for network forensic analysis by sniffing the traffic flowing through network devices such as switches and hubs. Snort sniffs the traffic from the Ethernet interface and displays the headers of the traffic it has captured when it is run in the sniffer mode.

Tcpdump: Tcpdump is a common packet analyzer that runs under the command line. It allows the user to intercept and display TCP/IP and other packets being transmitted or received over a network to which the computer is attached . It is a very useful tool for network forensics point of view; it provide the ability to analyze network behavior, performance and applications that generate or receive network traffic.

The original data obtained in the form of traces and logs is stored on a back up device. A hash of all the trace data is taken and the data is protected. Standard procedures are used to ensure accuracy and reliability of the preserved data. Another copy of the data will be used for analysis and the original collected network traffic is preserved.

We made a website on the local host to launch Cross-site Scripting attack . Attacker injects some malicious scripts embedded in the data and submitted to a web application. The network must be monitored to identify future attacks. The integrity of data logged and network events recorded must be insured.

For this we have installed Wireshark on the background of the server to capture the network traffic. The traffic that is captured by Wireshark is raw and we need only relevant traffic.

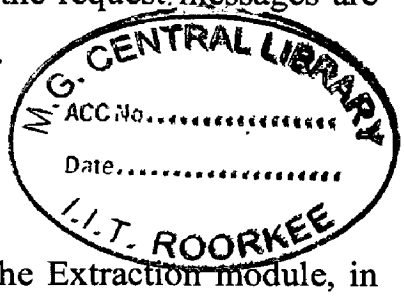
3.2 Extraction Module

The task of Extraction module is to extract the traffic that is relevant for further analysis; here the strategy to extract the traffic as relevant may vary while investigating the source of different kind of attacks. This strategy is limited to the detection of XSS Attack and will be able to trace the attacker only if attacker has used Reflected method for XSS attack.

Initially we divide the captured packets based on their IP addresses. A directory is created for each IP address that is visiting the website. The filtered packets are then written in request and response files depending on their type (request or response). This is done so that crucial information is not lost or mixed up. The collected data is classified and clustered into groups so that the volume of data to be stored may be reduced to manageable chunks. It is easy to analyze large groups of organized data. Redundant information and unrelated data is removed and minimum representative attributes are identified so that the least information with the highest probable evidence needs analysis.

3.3 Storing Module

The task of this module is to store the data according to IP address into hard disk of the host system for further analysis. This module ensures that only the relevant data used for Cross –Site Scripting attack are stored. These results in less amount of storage required as compared to the normal process which captures all the packets passing through network interface and stores them on the host system. . Here a chain of custody is strictly needed so that there is no unauthorized use or tampering . Data is stored in the form of directory according to IP addresses of the user that is visiting the website. In that directory there is one request file contains all the requests and response file containing all the response from the server. In request file all the request messages are stored and in the response file all the response message is stored.



3.4 Analysis Module

This module analyzes the data stored in the host system by the Extraction module, in order to discover the source of network attacks. In this module we try to find the source of the attack. The traces obtained from various security tools are integrated and fused to form one large data set on which analysis can be performed. However this process needs to be done so that crucial information from important sources is not lost. The evidence collected is searched methodically to extract specific information of the crime. Minimum attack attributes are identified so that the least information recorded contains the highest probable evidence.

The analysis module will provide the reduced database having qualitative information that can be directly used for the investigation process. The attacker information reported by our model also helps in the investigation of the actual attacker. Pattern matching approaches are used to search the data and match attack patterns. The attack patterns are put together and the attack is reconstructed and replayed to understand the intention and methodology of the attacker. The result of this phase is the validation of the suspicious activity.

The information obtained from the analysis module is used to identify who, what, where, when, how and why of the incident. This will help in source traceback, reconstruction of the attack scenario and attribution to a source. The most difficult part of the network forensic analysis is establishing the identity of the attacker. Two simple strategies of the attacker to hide himself are IP spoofing and stepping stone attack[34]. Researchers have proposed many IP traceback schemes to address the first attack and is still an open problem. Stepping stones are created by attackers to use compromised systems to launch their attacks. They can be detected using similarity and anomaly based approaches applied to packet statistics.

Chapter 4

Implementation of the Proposed Framework

We implemented the proposed network forensic system using JAVA language. We created. The proposed network forensics system architecture is designed and implemented to trace Reflected Cross-site scripting attack. It is implemented in Ubuntu 9.04, a flavor of UNIX using C. The implementation handles the process of capturing the packets, extracting their content, analyzing the database, creating the statistics and reporting the information about the attacker. We created a website on local host so that we can launch attack on it. Website is created in php language. It has basic feature like login and feedback form that is basic need to show the Reflected Cross-site scripting attack.

4.1 Implementation

The data captured by Wireshark was analyzed by the system. Only HTTP packets were considered. We made directory according to IP address of the user that is visiting the website. In that directory a file was created for storing requests and another file for storing requests.

After collecting the raw data from Wireshark, it is analyzed line by line. If the line contains Internet Protocol field it signifies the beginning of an IP packet and the data that follows contains Source IP address and Destination IP address, which are then extracted. If the line contains Hyper text transfer protocol field then it signifies the beginning of HTTP packet and the data that follows is http data containing both HTTP headers and the packet. We read the HTTP header beginning from the next line of the file to determine the packet type, i.e. request and response. If the packet is a request we consider the source IP address and if the packet contains a response packet then it is considered as destination IP address. This is done because the source IP address in request packet and destination IP address in response packet gives the client IP address and we want to segment the data for different clients. Now depending on the type of the packet we create a file in the directory given by the IP selected.

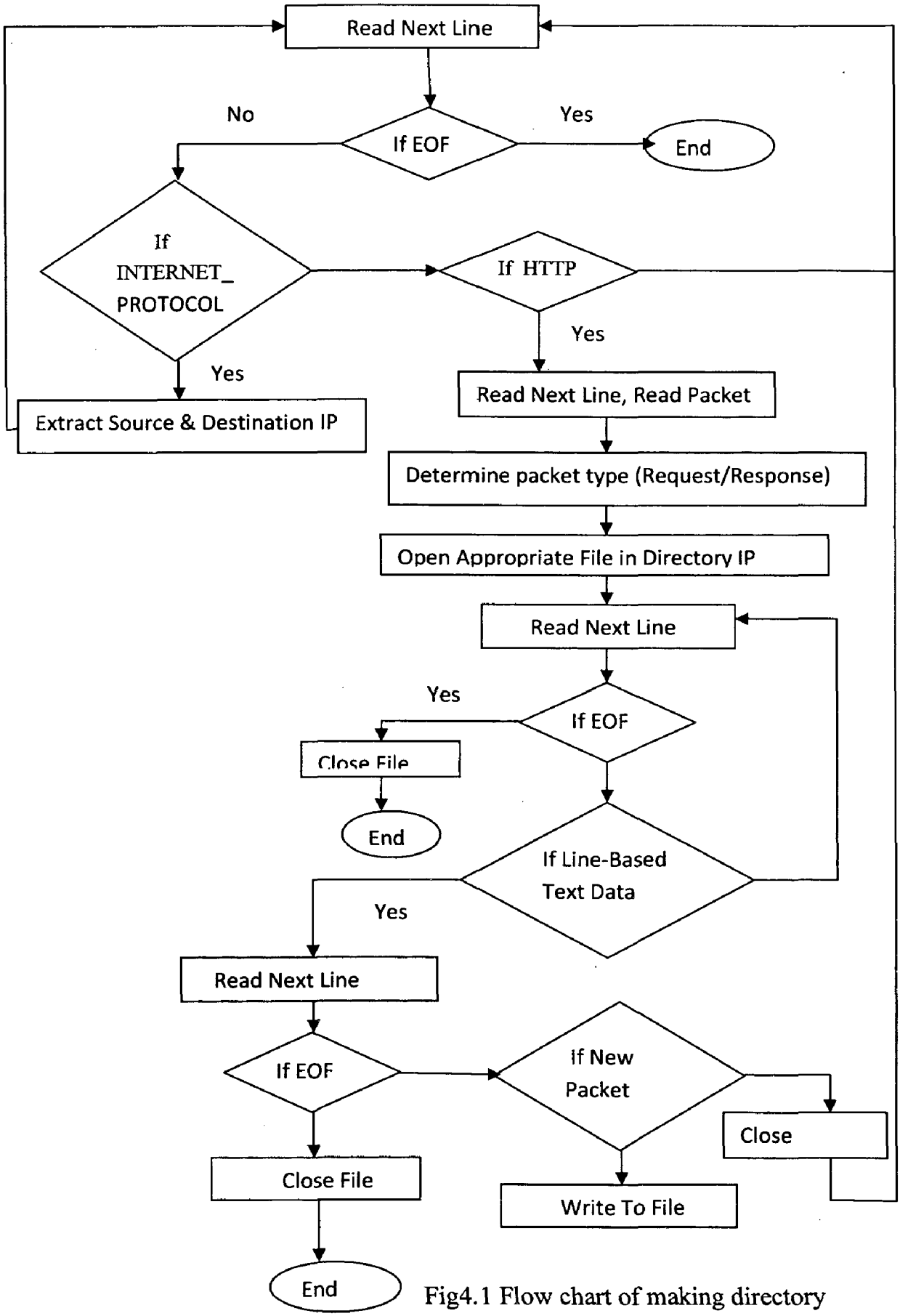


Fig4.1 Flow chart of making directory

Then we read the http packet till we reach the data part of the packet which is signified by the presence of "line based text data" in the line read. Now the data that follows is written to the appropriate file as it is. The end of this packet is either signified by the end of file or the beginning of a new packet upon which the file is closed. If there are more packets these are processed in a similar way. The flow chart of making directory according to IP addresses is given in figure 4.1.

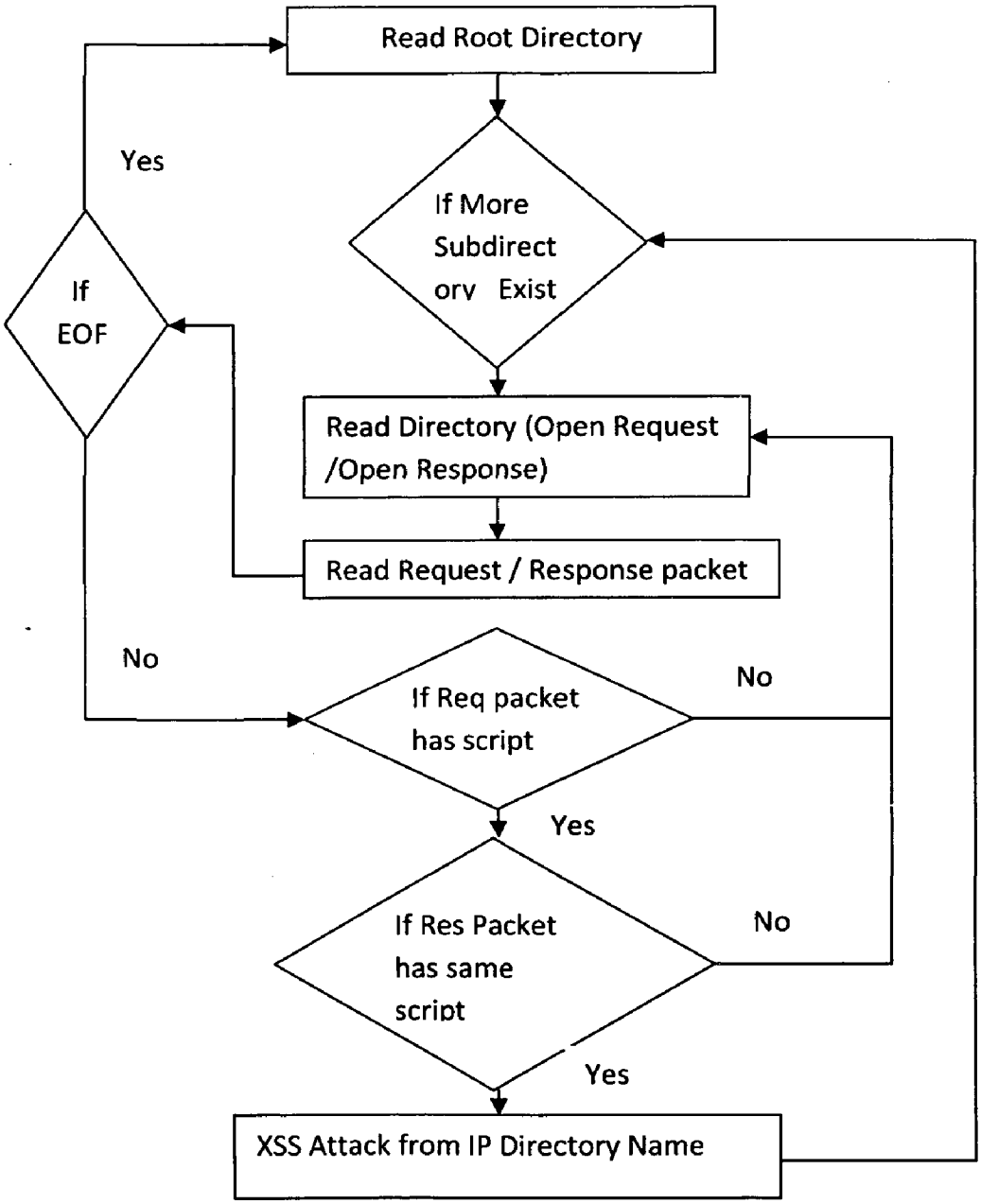


Fig.4.2 Flow Chart of Finding Suspicious IP

After storing the request and response data in directories, in this module we try to find the suspicious IP addresses that are launching XSS attack. The flow chart for finding suspicious IP is given in figure 4.2.

Initially we filter the captured packets based on their IP address. A directory is created for each IP address. The filtered packets are then written in request and response files depending on their type.

Once the files are ready next step is to determine if an XSS attack is done from any IP address. For this we processed the packets from each IP address. The two files are read in sequential manner in such a way that when the ith request packet from the request file is read, the ith response packet is also read from the response file. Each REQUEST and the corresponding RESPONSE is analyzed for XSS attack. A characteristics property of XSS attack is that the request and the response contain the same script. The request is scanned for a script. If any script exists in the REQUEST then it is searched in the RESPONSE. If both the REQUEST and RESPONSE contain the same script then it is considered to be reflected XSS attack. TO trace the source of the attack the source IP address is extracted from the REQUEST packet and is included in the "alert" to the administrator.

4.2 Testing:

To test the forensics system it was deployed on a system hosting "testsite" website. Various reflected XSS attacks were conducted on the website from other systems. Some of following scripts were used to test the forensics system.

1. <script>alert(document.cookie)</script>
2. <script>alert("xss attack")</script>
3. <script>alert(document.cookie)</script>
4. <script>alert(1)</script>
5. <script>alert(document.xss)</script>
6. <SCRIPT>SRC=http://ha.ckers.org/org/xss.js></SCRIPT>

The script **< script>alert(document.cookie)</script>** prints the cookie value in meassge box of the website. Here "document" is a browser object.

The script **<SCRIPT>SRC=http://ha.ckers.org/org/xss.js></SCRIPT>** sends the links of the website ha.ckers.org and from that site the malicious script is download on the victims website's server.

We use **Apache** web server for launching the website. Apache is generally recognized as the world's most popular Web server (HTTP server). We are using Apache because is reliable, free, and relatively easy to configure. All the webpages are store in www directory. All the data is store in test site folder. We create a webpage name login.php for login details of the user. Feedback form and feedback result are stored as page name index.php.

We use **MySQL** for storing the user name and password. MySQl is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. Basically, a MySQL database allows you to create a relational database structure on a web-server somewhere in order to store data or automate procedures. We create a table name users table for details of username and password.

We have created a website on local host to launch XSS attack on it. The snap-shot of login page is given in figure 4.3.

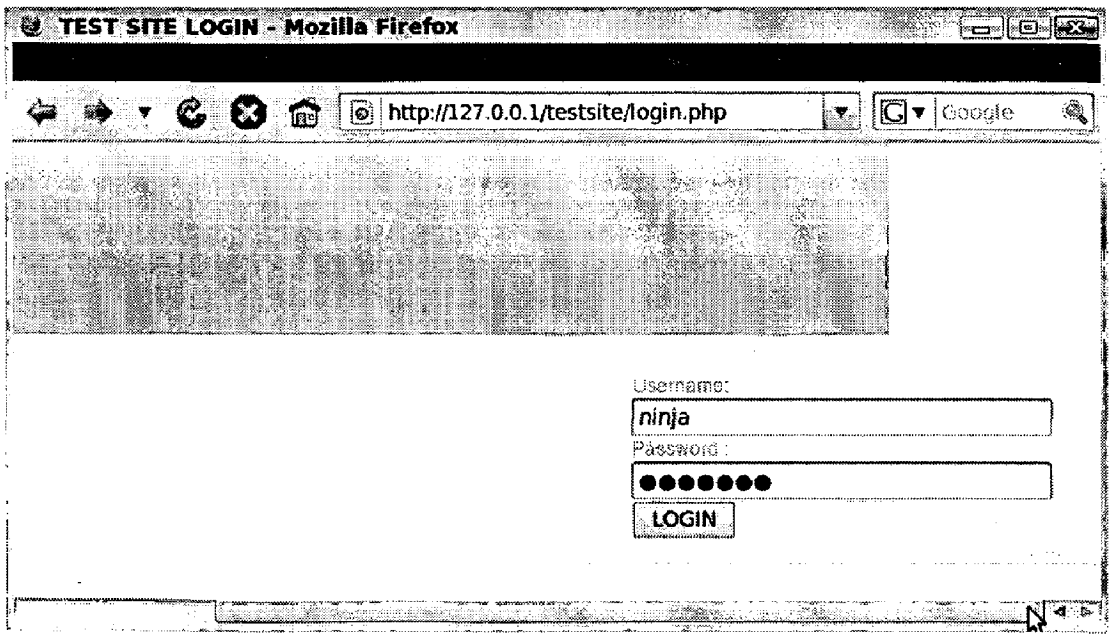


Fig.4.3 Test Site login

The website has the facility of inputting the data in the feedback form. After login in the page feedback form pages appears where script is injected by the attacker.

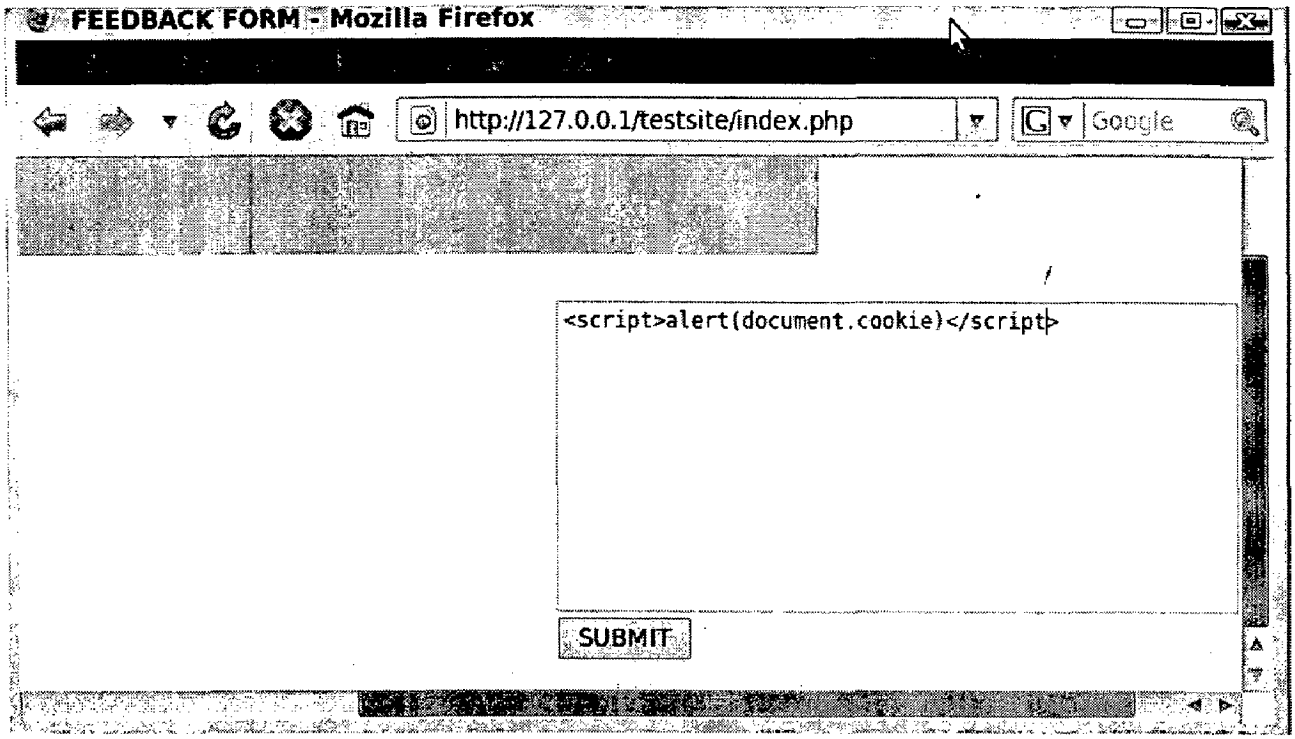


Fig 4.4 Feedback form

After injecting the script in the feedback form the vulnerable page that comes is given in figure 4.5. Auth=1 is the cookie information. Auth is cookie name and 1 is cookie value.

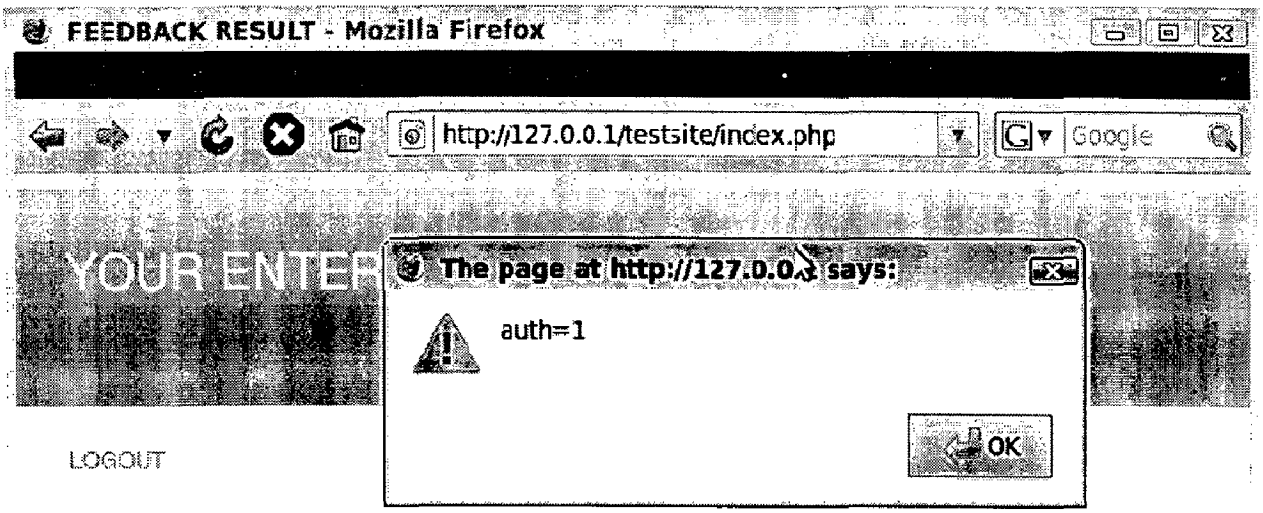


Fig. 4.5 XSS vulnerable page

We have launched attack from two IP addresses name as 192.168.110.1 and 192.168.110.2 to the victims IP name as 192.168.110.69. Wireshark tool is installed in victim's computer.

The experimental setup is given in figure 4.6.

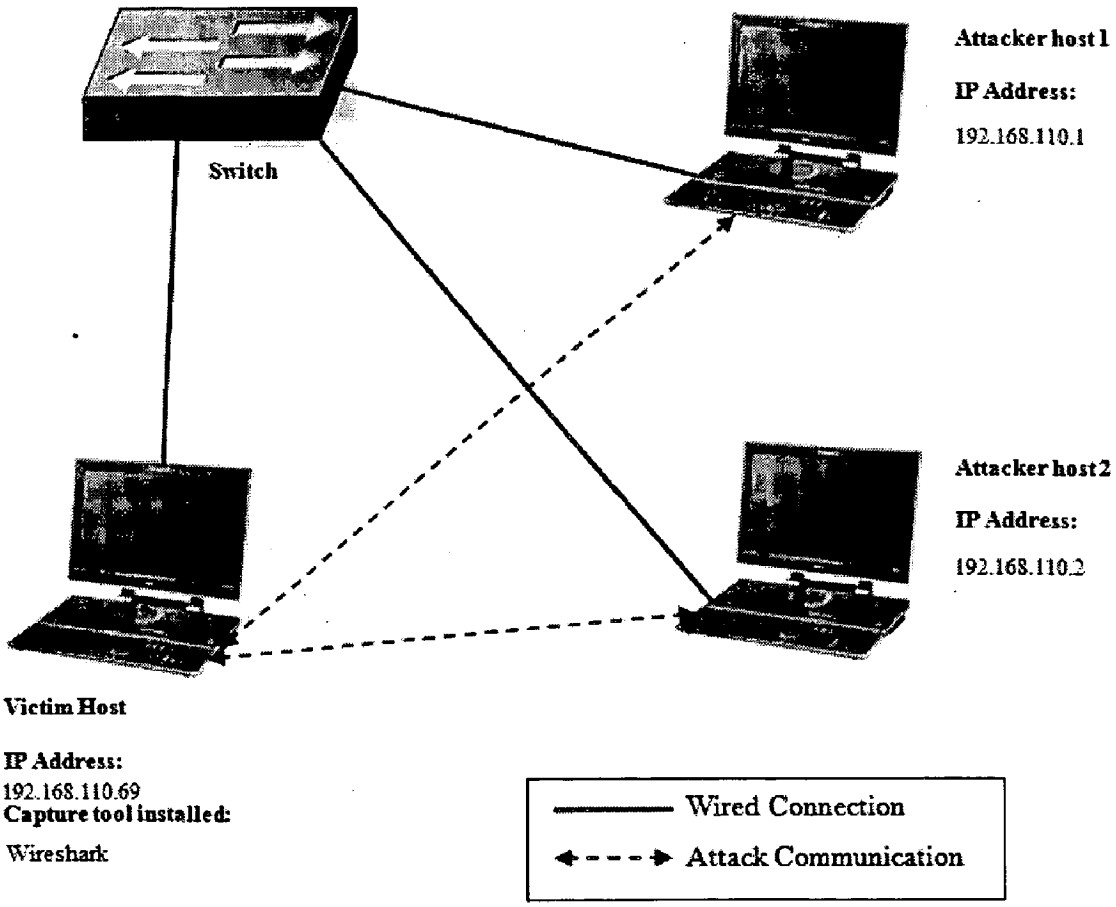


Fig 4.6 multiple host communicating XSS attack on the server

Chapter 5

Results and Discussions

Initially we filtered the captured packets based on their IP addresses. A directory is created for each IP. The filtered packets are then written in request and response files depending on their type. A table of number of request messages and response messages from 192.168.110.1 and 192.168.110.69 respectively is given below.

No. of Request messages from IP address 192.168.110.1	10
No. of Response messages from IP address 192.168.110.69	10

Table 5.1 No. of Request message and Response message

Snap shot of request message from IP address 192.168.110.1 is given in figure 5.1.
In this figure user enters the username and password and then enters the script **<script>alert(document.cookie);</script>** . Then the browser encodes the script and send to the server.

```
Request
    uname=ninja&pass=phoenix
RequestEnd

Request

RequestEnd

Request
    value=%3Cscript%3Ealert%281%29%3C%2Fscript%3E
RequestEnd
```

Fig. 5.1 Request message from IP address 192.168.110.1

The browser accepts the request send by the IP address 192.168.110.69 and replies with response message to the request message. The snapshot of response is given in figure 5.2. In response message the script **<script>alert(document.cookie);</script>** is contained .

```
Response
    \t<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">\r\n    <html
xmlns="http://www.w3.org/1999/xhtml">\r\n    <head>\r\n    <meta
http-equiv="Content-Type" content="text/html; charset=iso-8859-
1" />\r\n    <link href="style.css" rel="stylesheet"
type="text/css" />\r\n    <title>FEEDBACK RESULT</title>\r\n
</head>\r\n    \t<body>\r\n    \t<div class="header"
style="height:76px;"><a href="#">YOUR ENTERED FEEDBACK IS</a>
</div>\r\n    \t<a href="logout.php">LOGOUT</a>\r\n    \t\t\t
<div id="navcontainer">\r\n    \t<script>alert
(document.cookie);</script>\r\n    \t<br>
ResponseEnd
```

Fig 5.2 Response message from IP address 192.168.110.69

A table of number of request messages and response messages from 192.168.110.2 and 192.168.110.69 respectively is given below.

No. of Request messages from IP address 192.168.110.2	11
No. of Response messages from IP address 192.168.110.69	11

Table 5.2 No. of Request message and Response message

Snap shot of request message from IP address 192.168.110.1 is given in figure 5.4.
In this figure user enters the username and password and then enters the script **<script>alert(document.cookie)</script>** . Then the browser encodes the script and send to the server.

```
Request
    uname=ninja&pass=phoenix
RequestEnd

Request

RequestEnd

Request
    value=%3Cscript%3Ealert%28document.cookie%29%3B%3C%
2Fscript%3E
RequestEnd
```

Fig. 5.1 Request message from IP address 192.168.110.2

Finally when we run the program and find the attacks and when response packet has same script as request then an find the suspicious IP address. We are calling it suspicious because the source IP can be spoofed. The result of finding IP address is given in fig 5.3

Finding Attacks

Response packet has same script as Request.

Response packet has same script as Request.

XSS ATTACK from IP : 192.168.110.1

XSS ATTACK from IP : 192.168.110.2

Figure 5.3 Source of Reflected XSS attack.

We have run 30 malicious scripts on our website out of which our sytem detected 26 attacks.There were 2 false positive and 4 false negative alerts. The reason behind the false postive alerts is that if there is a script such as <script> </script> (which is not malicious), the system will consider it an attack. There were two such instances, and the system generated false alerts in those cases. The reason behind the false negative alerts is that if the attacker used some novel script such as **";!--"<XSS>=&{()}** etc which does not contain the <script> </script> tags then the system failed to detect such attacks.

The detail is given in table 5.4.

Total attacks	30
Attacks Detected	26
False Positive	2
False Negative	4

Table 5.3 Result of the attacks

The pie chart of the detected attacks is given in figure 5.5.

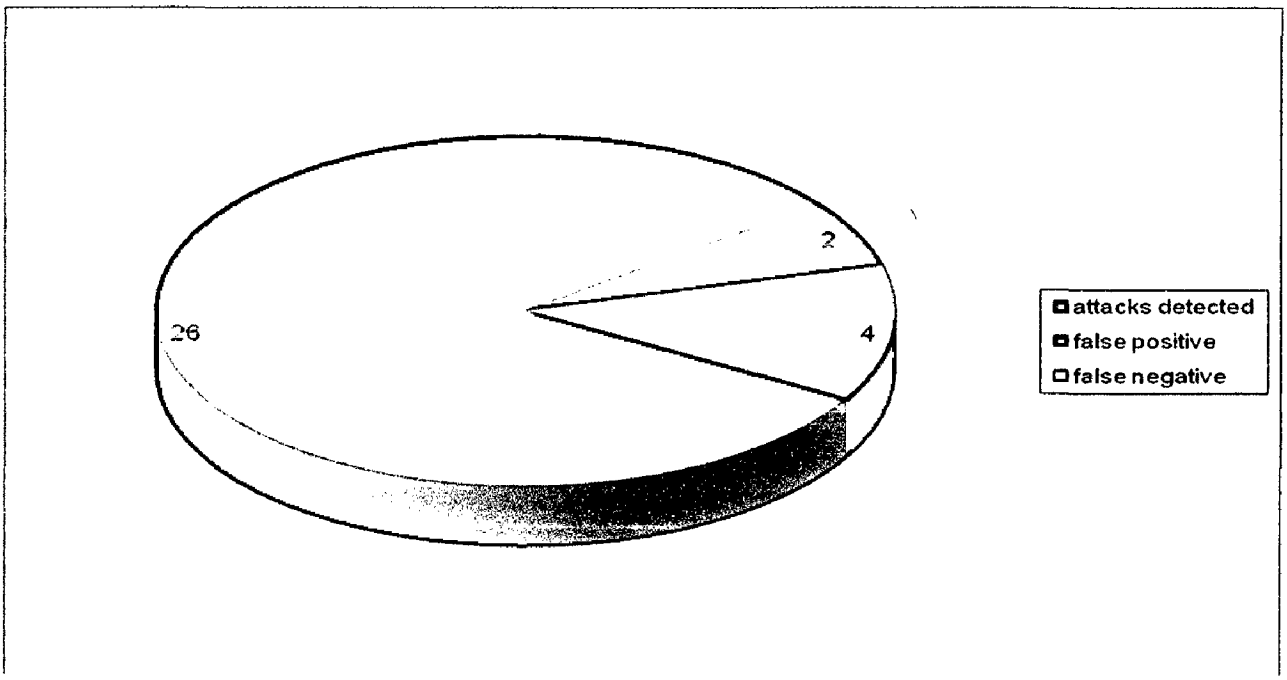


Fig 5.5. Pie chart of total detected attacks

Chapter 6

Conclusion and Future Work

6.1 Conclusion

A network forensics system is definitely a valuable investigation tool to discover the source of network attacks and to provide information about the attacker. Previously there was a network forensics system for detecting only transport layer attacks such as Port Scanning Attack and ICMP attacks. We address this problem by extending the model to detect the application layer attacks. We focused on some specific attack on application layer such as Cross-Site scripting attack and have tested our approach on a packet capture file from a victim system.

The following conclusions can be made from the results obtained using the proposed framework and above mentioned data:

- The proposed framework can find the source of major Reflected Cross-Site scripting attacks. Our result shows a significant detection in the number of attacks. Out of 30 attacks our system detects 26 attacks. There are also 2 false positive and 4 false negative attacks.
- Our results validate the correctness of the framework and in fact giving the good result with respect to Reflected Cross-Site Scripting attack.
- The proposed framework finally provides the reduced database having qualitative information that gives the investigation phase a concrete raw material to start with. Also the information from our result of analysis phase will be useful for investigation process.
- The framework is scalable to the increasing number of Reflected XSS attacks on any kind of network.

6.2 Future Work

There is obviously significant room for improving the methods that we used for the network forensic analysis. The possible improvements in the future are listed as below:

- The future work may be to add more attacks on various protocols at the application layers such as Persistent XSS and DOM based XSS attack etc.
- One major work may be to extend the proposed framework by including the investigation module, which will trace the actual attacker if the IP discovered by the proposed framework is spoofed.
- Since the network traffic may be encrypted, an entire field of reconstructing the network events by extending this framework to handle the encrypted traffic needs attention.
- To build a comprehensive framework where the proposed framework can be integrated with existing open source tools for forensic analysis. The proposed framework will facilitate the source of Reflected Cross-Site scripting attack and the established tools will get the ability to handle application layer attacks.
- Our implementation worked slowly in the heavy traffic so the future work can be to implement the proposed system architecture in multi processing environment using CUDA etc.

REFERENCES

[1] A. Yasinsac and Y. Manzano, "Policies to Enhance Computer and Network Forensics," *In IEEE Workshop on Information Assurance and Security*, United States Military Academy, West Point, New York, June 2001, pp. 289-295.

[2] W. Ren and H. Jin, "Modeling the network forensics behaviors," *in proceedings of 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks* (SecureComm 2005), Athens, Greece, Sept 2005, pp. 1–8.

[3] R. Chnadran, "Network Forensics", in Know Your Enemy: Learning about Security Threats, Ed. L. Spitzner, Second Edition, Addison Wesley Professional, 2004, pp 281 – 325.

[4] Theory of XSS[online] available http://www.acunetix.com/ website security/ cross-site- scripting.htm.

[5] Internet System Consortium, "ISC Domain Survey: Number of Internet Hosts." http://ftp.isc.org/www/survey/reports/2010/04/. Last updated: May 12, 2011.

[6] The official google blog, "A new approach to china." Internet: http://googleblog.blogspot.com/2010/01/new-approach-to-china.html. Last updated: Jan 12, 2010.

[7] J. C. Parez, "DDOS attackers continue hitting Twitter, Facebook, Google." Internet:http://www.computerworld.com/s/article/9136402/DDOS_attackers _continue _hitting_Twitter_ Facebook_Google. Last updated: Aug 8, 2009.

[8] XSS attacks [online] available http://www.xssed.com/newslist.

[9] G. Palmer, "A Road Map for Digital Forensic Research," *in proceedings of 1st Digital Forensic Research Workshop (DFRWS 2001)*, Utica, New York, Aug 2001, pp. 27–30.

[10] E. S. Pilli, R. C. Joshi and R. Niyogi [illegible]: "A Generic Framework for Network Forensics". *International Journal of Computer Applications* 1(11):1–6, February 2010. Published By Foundation of Computer Science.

[11] S. Garfinkel, "Network Forensics: Tapping the Internet." Internet: http://www.oreillynet.com/pub/a/network/2002/04/26/ nettap.html. Last updated: April 26, 2002.

[12] M. Reith, C. Carr and G. Gunsch, "An examination of digital forensic models," *International Journal of Digital Evidence,* vol. 1, Issue 3, pp. 1-12, Fall 2002

[13] B. Carrier and E. H. Spafford, "Getting physical with the digital investigation process," *International Journal of Digital Evidence,* vol. 2, no. 2, pp. 1-20, Fall 2003.

[14] V. Baryamureeba and F. Tushabe, "The enhanced digital investigation process model," in *proceedings of 4th Digital Forensic Research Workshop* (DFRWS 2004). Baltimore, Maryland, Aug 2004, pp. 1-9.

[15] XSS for beginners[online]available: http://rupeshhacktheworld. blogspot.com /2011/05/xss-cross-site-scripting-for-beginners.html.

[16] WebHackingThreats[online]available:http://techtimely.wordpress.com/2011/04/ 22/web-hacking-threats/

[17] Theory of XSS[online] available http://www.acunetix.com/ website security/ crosssite -scripting.htm.

[18] Buffer over flow attack[online] available: http://www.webopedia.com /TERM/ buffer _overflow.html.

[19] SQL injection attack [online] available: http://msdn.microsoft.com/enus/library /ms161953.aspx .

[20] Distributed denial-of-service attack [online] available http://searchsecurity. techtarget.com/definition/distributed-denial-of-service-attack.

[21] W. Ren, "On The Reference Model of Distributed Cooperative Network Forensics System," *in proceedings of 6th International Conference of Information Integration and Web-based Application & Services (iiWAS2004)*, Jakarta, Indonesia, Sept 2004, pp 771-775.

[22] W. Ren and H. Jin, "Distributed Agent-based Real Time Network Intrusion Forensics System Architecture Design," *in proceedings of 19th International Conference of Advanced Information Networking Applications (AINA 2005)*, Taipei, Taiwan, Mar 2005, pp. 177-182.

[23] Ismail, Etoh, M.; Kadobayashi, Y.; Yamaguchi, S.; "A proposal and implementation of automatic detection/collection system for cross-site scripting vulnerability," *Advanced Information Networking and Applications, 2004. AINA 2004. 18th International Conference on* , vol.1, no., pp. 145- 151 Vol.1, 2004.

[24] Natarajan, Sumanth and Loretta A.," Tools and techniques for Network Forensics" *in the preceding of International Journal of Network Security & Its Applications 1.1 (2009) 14-25.*

[25] Faghani, M.R.; Saidi, H.; "Social Networks' XSS Worms," *Computational Science and Engineering, 2009. CSE '09. International Conference on* , vol.4, no., pp.1137-1141, 29-31 Aug. 2009.

[26] Zhang, Qianjie; Chen, Hao; Sun, Jianhua; , "An execution-flow based method for detecting Cross-site Scripting attacks," *Software Engineering and Data Mining (SEDM), 2010 2nd International Conference on* , vol., no., pp.160-165, 23-25 June 2010.

[27] Galán, E.; Alcaide, A.; Orfila, A.; Blasco, J.; , "A multi-agent scanner to detect stored-XSS vulnerabilities," *Internet Technology and Secured Transactions (ICITST), 2010 International Conference for* , vol., no., pp.1-6, 8-11 Nov. 2010.

[28] Kaushik, A.K.; Pilli, E.S.; Joshi, R.C.; , "Network forensic system for port scanning attack," *Advance Computing Conference (IACC), 2010 IEEE 2nd International* , vol., no., pp.310-315, 19-20 Feb.

[29] Kaushik and R. C. Joshi. "Network Forensic System for ICMP Attacks," *International Journal of Computer Applications,* vol. 2, no. 3, pp.14–21, May 2010. Published By Foundation of Computer Science.

[30] Wireshark[online] available www.wireshark.org.

[31] Snort[online] available www.snort.org.

[32] TCPDump[online] available www.tcpdump.org.

[33] V. Jacobson, C. Leres and S. McCanne, "pcap - Packet capture library. "LawrenceBerkeleyNationalLaboratory, Internet: //www.tcpdump.org/pcap3 _man.html. Last updated: Nov 21, 2003.

[34] Stepping stone attack [online] available:[www.cs.berkeley.edu/~dawnsong/ papers/stepstone.pdf

LIST OF PUBLICATIONS

[1] **Pankaj Kumar Shahu,** A.K.Sarje "**Detection of Reflected Cross-site Scripting Attack using Network forensics**" in International Conference on Computer Science and Information Technology (CSIT-2011),Bhubaneswar, July 24 ,2011. Publisher: IJICT Journal (Accepted).