A FRAMEWORK FOR NETWORK FORENSIC ANALYSIS

A THESIS

Submitted in partial fulfilment of the
requirements for the award of the degree
of
DOCTOR OF PHILOSOPHY
in
ELECTRONICS AND COMPUTER ENGINEERING

by

PILLI EMMANUEL SHUBHAKAR

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

JUNE, 2011

INDIAN INSTITUTE OF TECHNOLOGY ROORKEE
ROORKEE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"A FRAMEWORK FOR NETWORK FORENSIC ANALYSIS"** in partial fulfilment of the requirements for the award of the Degree of Doctor of Philosophy and submitted in the Department of Electronics and Computer Engineering of the Indian Institute of Technology Roorkee, Roorkee is an authentic record of my own work carried out during a period from January 2008 to June 2011 under the supervision of **Dr. R. C. Joshi**, Professor and **Dr. Rajdeep Niyogi,** Assistant Professor, Department of Electronics and Computer Engineering of Indian Institute of Technology Roorkee, Roorkee.

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

(PILLI EMMANUEL SHUBHAKAR)

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

(Rajdeep Niyogi)
Supervisor

(R. C. Joshi)
Supervisor

Date: 20.06.2011

The Ph.D. Viva-Voce examination of **Mr. PILLI EMMANUEL SHUBHAKAR**, Research Scholar, has been held on 17.01.2012

Signature of Supervisors

Signature of External Examiner

Abstract

Network forensics is a nascent science that deals with the capture and analysis of the network traffic and logs of intrusions. Network forensics characterizes intrusion or misbehavior features in order to discover the source of security attacks. Network forensics uses scientifically proven techniques to collect, fuse, identify, examine, correlate, analyze, and document digital evidence. This information is collected from multiple, actively processing digital sources and security sensors. The analysis results in detecting and characterizing unauthorized network events meant to disrupt, corrupt, and/or compromise system components. It also provides information to assist in incident response and recover from system compromise or disruption of services.

Network forensics goes beyond network security as it not only detects the attack, but records the evidence as well. There are certain attacks which do not breach network security policies but may be legally prosecutable. These crimes can be handled only by network forensics. Forensic systems act as a deterrent, as attackers become cautious. They spend more time and energy to cover the tracks in order to avoid prosecution. This makes the attack costly, reduces the rate of network crime, thereby enhancing security.

A generic process model for network forensic analysis was proposed based on various existing digital forensics models. A methodology was formalized, specifically for investigation based on network traffic. The proposed model is generic as it handles both the real-time and post attack scenarios. The term 'process model' is used to refer to our and many other theoretical representations of phases involved in network forensics. The model has nine phases – preparation, detection, incident response, collection preservation, examination, analysis, investigation and presentation. The first five phases handle real-time network traffic. The next four phases are common for real-time and post attack scenarios.

Many phases like preparation, detection, collection, preservation, and presentation in our proposed generic process model have been extensively studied and researched. Techniques have been developed for these phases and are standardized. Research is now focused on examination, analysis, investigation and incidence response phases. Few frameworks have been proposed involving these phases. The term 'framework' is used to mean prototype implementation.

A framework is proposed for network forensic analysis, which will capture network traffic data, correlate and analyze this data, perform fusion of alerts and attack information and investigate the source of attack. The three phases of examination, analysis and investigation are handled in three objectives: Identification and Correlation, Data Fusion and Source Traceback.

Network events provide information about the attempts made in compromising the system and help in attack reconstruction. Identifying important sessions of suspicious activity will reduce the data to be analyzed. The correlation of events will validate the occurrence of the malicious incident and guide the decision to proceed with the investigation. An approach to examine the packet captures and identify network events at the application, transport, and network layer was presented. The events specific to distributed denial of service (DDoS) attacks, port scan attacks and cross-site scripting (XSS) are identified and correlated. This approach is validated using an attack dataset.

Attack occurrence is ascertained and validated before proceeding with investigation. Attack information and alerts from multiple security sensors with complementary and contradictory functionality are analyzed. Intrusion detection systems (*Snort* and *Bro*), packet capture and analysis tools (*tcpdump*) or sniffers (*wireshark*), traffic statistic tools (*tcpstat*) and security analysis console (*BASE*) are used. Data fusion is performed on the alert and attack information generated by these sensors using Dempster-Shafer theory of evidence. The suspicious addresses and alert information is used in investigation phase.

IP traceback identifies the actual source of any packet sent across the Internet. The source of the attack can be traced using packet marking mechanisms and attributed with the attack. Two novel approaches are proposed as part of investigation phase – Autonomous System based Deterministic Packet Marking (ASDPM) and Deterministic Router and Interface Marking (DRIM). They involve deterministic marking of each packet with the first internal routers information and either the AS Number (ASN) or the number of the interface through which the packet reached the router.

A single packet is sufficient to detect the attack source. In ASDPM, we trace the first internal router within the source AS of the attacker's network. In DRIM, we move one step closer to the attacker and identify the interface on which the packet reached the router. Simulations were performed to examine the feasibility of the approaches and validate them using discrete event network simulator ns-2.

Acknowledgments

Firstly I would like to thank the Lord Almighty, for all the blessings he has showered on me right throughout my life and also enabling me to reach the milestone of writing this last note in my research work. God has surrounded me with some extraordinary people who have spun a web of support around me. I am grateful to all of them and I would like to thank them for making my stay in this prestigious institution, a time to cherish. I wish to express my heartfelt gratitude for the encouragement and able guidance that my supervisors, Prof. R. C. Joshi and Dr. Rajdeep Niyogi, have given me throughout my work at the Indian Institute of Technology Roorkee, Roorkee, India.

I consider myself very fortunate to be associated with Prof. R. C. Joshi. It is his foresight and vision that inspired me to undertake research in this nascent field of network forensics. His understanding and widespread knowledge in multiple disciplines helped me to identify the correct objectives and put me back on the right track when I was beginning to go tangential. He also ensured that I never lacked any of the necessary resources to carry out my research. When there were difficulties, Prof. Joshi's insistence and optimism to 'keep on going till there is a breakthrough' helped me to gain more confidence, and often come up with solutions.

I benefitted from the fresh inputs which Dr. Rajdeep Niyogi had from his own research experience. He has spent voluminous amount of time with me and helped me in understanding the research issues and improving my presentation skills. He has helped me set milestones and guided me to make steady progress. He has a rare quality of being very accommodative, when I was hitting the wall and at the same time being very firm, when I was getting lenient.

The co-operation and help extended by the Head of the department, Chairman of my research committee, and faculty members, Department of Electronics and Computer Engineering, Indian Institute of Technology Roorkee, is gratefully acknowledged. I thank my research committee members, Prof. R. C. Mittal, Prof. Manoj Misra, and Prof. Kumkum Garg, for their patience during discussions and evaluations. I would like to specially thank Dr. Anjali Sardana, who had been very helpful in providing me with various tools, which made my research work simple and effective. Her constructive criticism and critical inputs have really influenced much of my work.

The department provided an excellent environment for my research in information security. It was a really fruitful time for me as I have learnt many important lessons in life apart from the receiving the rigorous training in research. Special thanks to Mr. Bankat M. Patil and Mrs. Poonam Choudhary, who were partners with me in tea and research discussions. I record here a note of thanks of Mrs. Manju, Mr. P. Sudhir, and Mr. Anil Singh who took the arduous step of proof reading the drafts of the thesis. Colleagues and friends helped in many ways: Dr. T. P. Sharma, Mr. Gaikwad, Mr. Dereje, Mr. Wairiya, Mr. Shiv Kumar, Dr. Sood, Anu, Brij, Manoj, Shree, Nagamma, Keshav & others.

Many students joined me in an informal research group of the Information Security laboratory. Special thanks to Radhika, Atul, Ishan, Shobhit, Mohit, Naresh, Santosh, Manikanta, Thirupathi, and Dipankar. I thank Mr. Raj Narayan Singh (Khati) for providing assistance with the use of Information Security laboratory facilities. I also thank Mr. Anoop Yadav and Mr. Dhanpat for their support in other laboratories.

I would also like to thank my relative Dr. Sundar Daniel and his family who encouraged me to pursue research in IIT Roorkee. I thank all my friends in the married hostel community, specially the families and children of Dr. Solomon Raju & Prasuna, Dr. Edwin & Mabel, Dr. Shashi & Sunitha, Dr. Benny & Angel, Mr. Satish & Nisha, Mr. Sunil & Suchita, Mr. Manoj & Veena, Mr. Aditya & Parul, Mr. Chetan & Aparna, Mr. Vipin & Vandana, Mr. Allen & Allen and others. Mrs. & Mr. Sushil Joshi were parental figures and held us all together. I thank Allwyn, Prithvi, Karthik, Ramesh, Sudhakar, Anu Venkat, Thiru, Anil, Rajesh, Nagaraju, Uday, Chaitu, John and Praveen for their support.

I am grateful to my father, who passed away 28 years earlier, but left an indelible impression in my mind to become an academician like him. I fondly remember my mother, who prayed that this desire might be fulfilled, but could not see me reach this stage of my research. I thank my brothers, sisters-in-law and their children for all the love, care and support. I thank my in-laws, who now play the role of my parents and the families of my wife's siblings for their affection. The most important people in my life, my loving wife Phoebe Vanmathy and our wonderful daughter Pramiti, deserve a special appreciation for their patience and support. They never complained about anything even though they missed my presence a lot. I promise to make up for the lost time.

Pilli Emmanuel Shubhakar

Contents

List of Abbreviations

AAST	Authenticated Autonomous System Traceback
ACID	Analysis Console for Intrusion Databases
AD	Attack Diagnosis
AIDF	Analytical Intrusion Detection Framework
ANN–PCA	Artificial Neural Network and Principal Component Analysis
AS	Autonomous System
ASBR	Autonomous System Boundary Router
ASCII	American Standard Code for Information Interchange
ASDPM	Autonomous System based Deterministic Packet Marking
ASEM	AS based Edge Marking
ASN	Autonomous System Number
ASSPT	AS level Single Packet Traceback
BA	Barabasi-Albert
BASE	Basic Analysis Security Engine
bpa	Basic Probability Assignment
CGI	Common Gateway Interface
CTMC	Continuous Time Markov Chain
CWE	Common Weakness Enumeration
DDoS	Distributed Denial of Service
DEC	Digital Evidence Custodian
DERM	Deterministic Edge Router Marking
DFIF	Digital Forensic Investigation Framework
DFRWS	Digital Forensic Research WorkShop
DGA	Data Generation Agents
DIDSDFM	Distributed Intrusion Detection System based on Data Fusion Method
DigForNet	Digital Forensic in Networking
DNF	Dynamical Network Forensics
DNS	Domain Name System
DoS	Denial of Service

DPM	Deterministic Packet Marking
DPMLS	Deterministic Packet Marking with Link Signatures
DRDC	Defense Research and Development Canada
DRIM	Deterministic Router and Interface Marking
DSS	Data Security Standard
ECS	Evidence Capture Subsystem
ERA	Eliminate Redundancy Algorithm
FISMA	Federal Information Security Management Act
FOD	Frame Of Discernment
FTP	File Transfer Protocol
GA	Genetic Algorithm
GBF	Generalized Bloom Filter
GLBA	Gramm–Leach–Bliley Act
GMT	Greenwich Mean Time
GNF	General Network Forensics
HENPA	Highly Extensible Network Packet Analysis
HIPAA	Health Insurance Portability and Accountability Act
HTTP	Hyper Text Transfer Protocol
ICMP	Internet Control Message Protocol
IDENT	Identification Protocol
IDS	Intrusion Detection System
IEASS	Intrusion Evidence Automated Analysis System
IEFAF	Integrated E-Mail Forensic Analysis Framework
IFDTNFS	Incremental Fuzzy Decision Tree-Based Network Forensic System
IN	Interface Number
INFERD	Information Fusion Engine For Real-Time Decision-Making
IP	Internet Protocol
IPS	Intrusion Prevention System
IRPCMS	Incident Response Probabilistic Cognitive Maps
ISMS	Information Security Management System
ISP	Internet Service Provider
I-TLA	Investigation-Based Temporal Logic Of Actions

I-TLC	Investigation Based Temporal Logic Model Checker
KDD	Knowledge Discovery in Databases
LAN	Local Area Network
LCA	Log Collection Agent
M3L	Multi – Metric – Multi – Link
MANETS	Mobile Ad-hoc NETworks
NAM	Network AniMator
NFAT	Network Forensic Analysis Tool
NFDLC	Network Forensics Development Life Cycle
NFR	Network Forensic Readiness
NFS-FLES	Network Forensic System Based Fuzzy Logic And Expert System
NIDS	Network based Intrusion Detection System
ns-2	Network Simulator 2
NSSA	Network Security Situation Awareness
NTE	Network Traffic Exploration
OWASP	Open Web Application Security Project
PCA	Principal Component Analysis
PCI	Payment Card Industry
PLC	Programmable Logic Controllers
PNFEC	Portable Network Forensic Evidence Collector
PNM	Path Number Marking
PRIVDAM	PRIvacy Violation Detection And Monitoring
QA	Quality Adaptive
RDBMS	Relational Database Management System
RIM	Router Interface Marking
SBL	Session Based Packet Logging
SCADA	Supervisory Control And Data Acquisition
SNF	Strict Network Forensics
SOX	Sarbanes–Oxley Act
SPIE	Source Path Isolation Engine
SRDM	Smart Recording and Data Mining
SS	Storage Subsystem

SYNPM	SYN Based Packet Marking
TANDI	Threat Assessment of Network Data and Information
tcl	Tool Command Language
TCP	Transmission Control Protocol
TOE	Target of Evaluations
UDP	User Datagram Protocol
URL	Uniform Resource Locator
VoIPOW	VoIP Over Wireless
WWW	World Wide Web
XSS (CSS)	Cross-Site Scripting

List of Figures

List of Tables

Chapter 1

Introduction and Problem Statement

1.1 Introduction

On April 25, 2011, Iran has been targeted by a new computer worm named *'Stars'*, which is compatible with the targeted system, causes minimal damage in the initial stage, and the worm is likely to be mistaken for executable files of the government [52]. *'Stars'* is the second computer worm to target Iran after the *'Stuxnet'* worm, which was capable of taking over power plants and had infected many industrial sites. W32.Stuxnet worm has been in the focus of media and researchers in the last one year. *Stuxnet* was discovered in June / July 2010 and is one of the complex threats in recent times. It targets industrial control systems and modifies code on programmable logic controllers (PLCs) to make them work in a manner the attacker intends to [59].

Stuxnet utilized antivirus evasion techniques, complex process injection code, four separate zero-day vulnerabilities and the first ever rootkit designed specifically for PLC systems [53]. It spread via unpatched holes in Windows and USB devices, dropped the rootkit to hide the compromise from administrators, and used fraudulent digital certificates to pose as trusted software [139]. *Stuxnet* targetted PLCs on sites using

Siemens SIMATIC WinCC or STEP 7 SCADA (Supervisory Control And Data Acquisition) systems. The *Stuxnet* computer worm might have been designed specifically to attack Iran's nuclear program as it infiltrated industrial systems mostly in Iran and potentially crippled centrifuges used to enrich uranium [107]. Figure 1.1 shows the geographic distribution of *Stuxnet* infections and Figure 1.2 shows number of users infected with the Rootkit.Win32.Stuxnet.

Stuxnet exploited several Windows vulnerabilities and at least four of them are zero-day vulnerabilities (MS08-067 RPC Exploit, MS10-046 LNK Exploit, MS10-061 Spool Server Exploit, MS10-073 Win32k.sys Exploit, MS10-092 Task Scheduler Exploit) [134]. Iranian security officials who dealt with *Stuxnet* indicate that the threat has not been completely eliminated since worms can have specific life cycles and continue their activities in other forms. They also highlighted that Iran should prepare itself to tackle future worms, which may infect the country's infrastructure. *Stuxnet* has brought before the security community, a glaring possibility of a serious threat to any country's sovereignty. *Network forensics* is definitely one way to be prepared for such eventualities.

Infosecurity [92] reported that there is an increasing demand for network forensics as enterprises want to be sure about who, what, when, why, where and how their services were being accessed and used. Networks forensics cannot stop attacks like *Stuxnet* from happening, but it can provide a way to reduce the impact by providing analysis that enables a more rapid response to the infection.

Solera Networks [148] explains that network forensics prepares organizations to respond swiftly to zero-day, negative day, and unknown threats. It enhances the value and

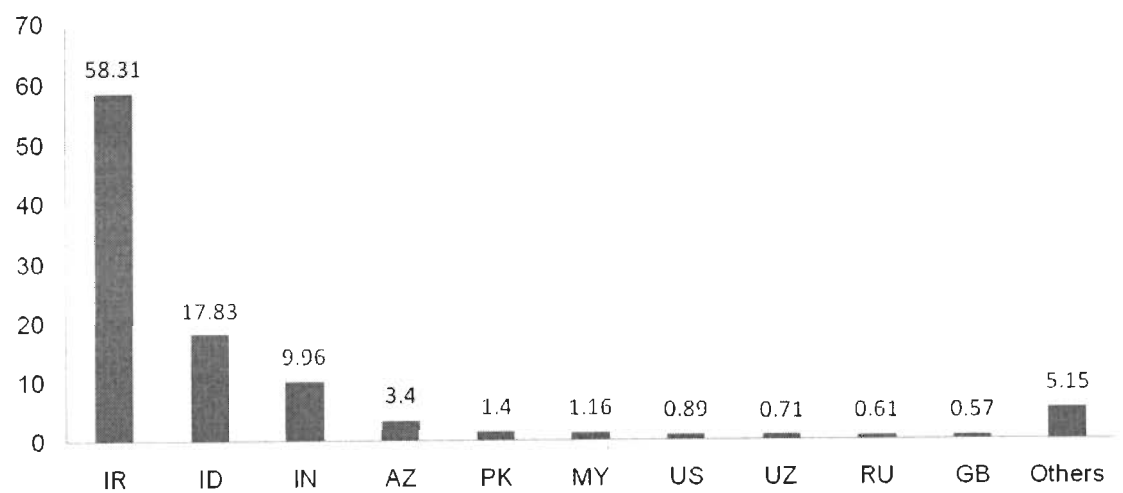

Figure 1.1. Geographic distribution of the Stuxnet worm in %

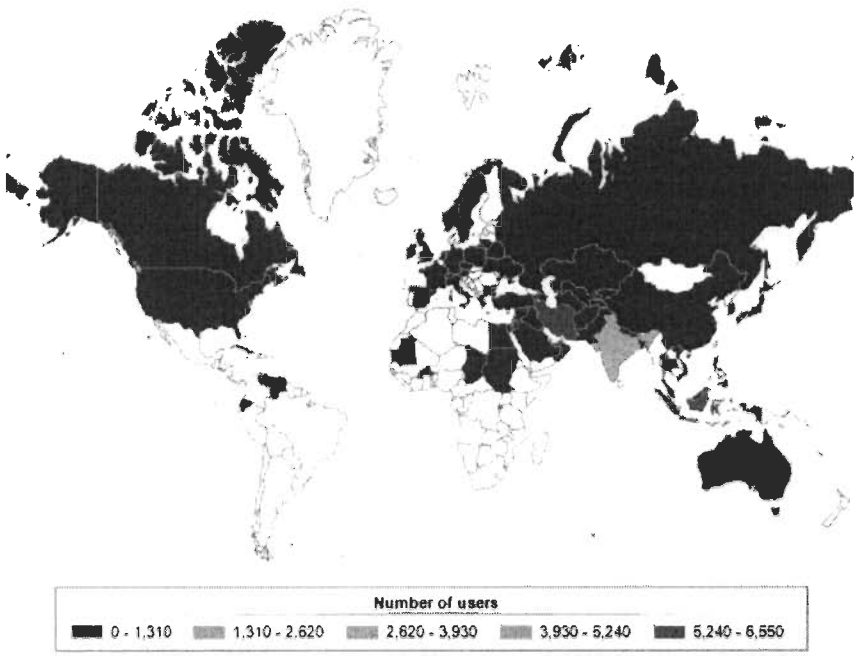

Figure 1.2. Geography of users infected with the Rootkit.Win32.Stuxnet worm

effectiveness of other security investments. It reduces and simplifies the monitoring, reporting, analysis, and remediation time required to defend against attacks. It facilitates prosecution through forensically complete evidence and provides an understanding of the root causes for the breach of security to enable swift, intelligent and effective response to prevent catastrophic events and ongoing risk. It allows for validation of fixes installed after a breach occurs through the ability to replay a network attack.

Network forensics appears to be similar to network security. However the objectives of the two are very much different. Network forensics is a nascent science that deals with capture, recording, and analysis of network traffic. The network traffic data is captured using packet sniffers, alerts and logs are collected from existing network security tools. This data is analyzed for attack characterization and investigated to traceback the perpetuators. This process can bring out deficiencies in security products which can be utilized to guide deployment and improvement of these tools.

The network security approach uses defensive mechanisms like Firewalls and Intrusion Detection System (IDS). The former is used for prevention and the latter for detection. These approaches typically find out network vulnerabilities and block all malicious communications from outside. Firewalls control traffic that enters a network and leaves a network, based on source and destination addresses and port numbers. It

filters malicious network traffic according to the firewall rules. It is difficult to update the signatures of all vulnerabilities as new vulnerabilities will always keep occurring.

Intrusion detection system (IDS) [10] are primarily for learning, detecting and reporting attacks as they happen in real time and have no evidence gathering feature. IDSs are of two types – signature based (misuse) detection and statistical based (anomaly) detection. Pattern matching is done in signature based IDS to detect intrusion signatures. It cannot detect new attacks but has a low false positive rate. Anomaly based IDS does activity monitoring and is able to detect new attacks but has higher false positive rate.

The network forensic approach collects the required evidence for incident response and investigation of the crime. Network security protects system against attack. Network security tools are generalized and continuously monitor the network for possible harmful behaviors. Network forensics involves postmortem investigation of the attack and is initiated *notitia criminis* (after crime notification). It is case specific as each crime scenario is different in many aspects and the process is time bound. There may be certain crimes which do not breach network security policies but may be legally prosecutable. These crimes can be handled only by network forensics [25]. The major differences between network security and network forensics are given in Table 1.1.

Network forensics can be generally defined as a science of discovering and retrieving evidential information in a networked environment about a crime in such a way as to make it admissible in court [75]. The investigation of a cyber crime often involves cases related to homeland security, corporate espionage, child pornography, traditional crime

Table 1.1. Comparison of Network Security and Network Forensics

Network Security	**Network Forensics**
Protects the system against attack	Does not protect the system against attack
Usually in real time	Post mortem
Generalized – looking for any possible harmful behaviors	Case restricted – want to reconstruct the criminal scenario
Keep alert 24 hours every day	After crime notification – notitia criminis
Continuous process	Time bound process
Established field of computer science	Very immature and young science

assisted by computer and network technology, employee monitoring, or medical records, where privacy plays an important role.

Network forensics is a natural extension of computer forensics. Computer forensics [21] was introduced by law enforcement and has many guiding principles from the investigative methodology of judicial system. Computer forensics involves preservation, identification, extraction, documentation, and interpretation of computer data. Network forensics evolved as a response to the hacker community and involves capture, recording, and analysis of network events in order to discover the source of attacks.

In computer forensics, the investigator and the hacker being investigated are at two different levels with investigator at an advantage. In network forensics, the network investigator and the attacker are at the same skill level. The hacker uses a set of tools to launch the attack and the network forensic specialist uses similar tools to investigate the attack. Network forensic investigator is more at a disadvantage, as investigation is one of the many jobs he is involved. The hacker has all the time at his disposal and will regularly enhance his skills, motivated by million dollars at stake. The seriousness of what is involved makes network forensics an important research field. The major differences between computer forensics and network forensics are given in Table 1.2.

Digital Forensic Research Workshop (DFWRS 2001) [158] proposed a network forensics framework that includes the following steps: "identification, preservation, collection, examination, analysis, presentation, and decision." The components of this

Table 1.2. Comparison of Computer Forensics and Network Forensics

Computer Forensics	**Network Forensics**
Introduced by law enforcement to handle computer data	Evolved as a response to the hacker community
The investigator and attacker are on two different levels	The investigator and the attacker are at the same skill level
The investigator and attacker use different tools, investigator has upper hand	The investigator and attacker use same tools and practices
Computer forensics involves preservation, identification, extraction, documentation, and interpretation of computer data	Network forensics involves the capture, recording and analysis of network events in order to discover the source of attacks
It is about acquiring, providing chain-of-custody, authenticating, and interpretation	It is about investigation of packet filters, firewalls logs and IDS logs

model are illustrated in Figure 1.3 and explained below:

1. Identification – recognizing an incident from indicators and determining its type
2. Preservation – isolate, secure, and preserve the state of physical and digital evidence
3. Collection – record the physical scene and duplicate digital evidence using standardized and accepted procedures.
4. Examination – in-depth systematic search of evidence relating to the suspected crime
5. Analysis – determine significance, reconstruct fragments of data and draw conclusions based on evidence found
6. Presentation – summarize and provide explanation of conclusions
7. Decision – attribution of attack or crime to a particular host or network with valid proofs

The preservation, collection and presentation phases have been extensively studied and researched. The established computer forensic field lays a strong foundation for network forensics as standard procedures and tools are in place for preserving and collecting digital evidence. The presentation and decision phases work more closely with the legal system and follow admissibility procedures in a court of law.

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

Figure 1.3. DFRWS Framework for Network Forensics

1.2 Motivation

The real motivation for our present study comes directly from the limitations in the defensive approaches of network security like firewalls and intrusion detection systems. They can address attacks only from prevention, detection, and reaction perspectives. The alternative approach of network forensics becomes important as it involves the investigative component as well [7]. Network forensics ensures that an attacker spends more time and energy to cover his / her tracks, thus making the effort of an attack costly. Network criminals will be more cautious to avoid prosecution for their illegal actions. This acts as a deterrent and may reduce network crime rate, thereby improving security.

The large number of security incidents affecting many organizations and increasing sophistication of the cyber attacks is the main driving force behind network forensics. Successful attackers often ensure that they cover their trails. Unsuccessful attacks often go unnoticed, and little information is available to assist with diagnosis even when they are noticed [115]. Internet Service Providers (ISPs) are also being made responsible for what passes over their network [166]. Companies doing business on Internet cannot hide a security breach and are now expected to prove the state of their security as a compliance measure for regulatory purposes.

The ISO 27001/27002 standard (Information technology – security techniques – information security management) [93] specifies the requirements for establishing, implementing, operating, monitoring, reviewing, maintaining and improving a documented Information Security Management System (ISMS) within the context of the organization's overall business risks. Comprehensive audit data are to be maintained to meet the compliance requirements of many regulations.

Sarbanes–Oxley (SOX) Act controls over the release of information to individuals or organizations. Gramm–Leach–Bliley Act (GLBA) ensures the privacy and integrity of customer records. Health Insurance Portability and Accountability Act (HIPAA) was established to protect the health-related data. Federal Information Security Management Act (FISMA) monitors security programs for federal agencies. By adhering to the Payment Card Industry (PCI) Data Security Standard (DSS), retailers, service providers and allied organizations can dramatically reduce the vulnerabilities that are easily exploited for the purpose of compromising corporate data. An integrated network forensic

process will facilitate meeting compliance requirements [82] for organizations and ISPs by adhering to strict security measures and maintaining comprehensive audit data [145].

Network forensics also facilitates recording evidence for investigation and helps in understanding the attacker's methodology. It provides insight about the tools used by the attacker and new ways in which perimeter defenses were circumvented. This information can also bring to light the deficiencies in existing network security tools. These tools can be hardened to become robust enough to stand the onslaught of many zero-day and hybrid attacks.

1.3 Recent Trends in Network Forensics

Network Forensics was traditionally applied to wired environments and was focused on the Version 4 of the Internet Protocol and related protocols at the network layer of the TCP/IP protocol suite. Following are some of the recent works in network forensics:

- **Steganography**: Many attackers use somewhat "light" forms of cryptography to render the recognition of rootkits or attack patterns tobe more difficult, which otherwise would have been easily spotted by any IDS [63].
- **Honeypot Forensics**: Honeypots are placed to be compromised and provide information on the blackhat's techniques and tools, before and after the intrusion on the honeypot. New forms of rootkits, trojans, and potential zero-day exploits can be discovered. A better understanding of the areas of interest and hidden links between blackhat teams can be obtained [179, 180].
- **IP Version 6 Forensics**: IPv6 Internet provides malicious users a temporary safe haven, as events are poorly logged and monitored. Many free tunnel brokers provide simple and relatively anonymous connectivity [152]. The transition from IPv4 to IPv6 will take time and both protocols co-exist for quite some time requiring inter-operation mechanism. This dual stack arrangement will bring new security vulnerabilities and exploits which will need forensic analysis [74].
- **Botnet Forensics**: Compromised machines can be linked up to form "Bot-nets" under external control, which are used to send spam e-mails or disable websites with a flood of bogus requests. It is very difficult to trace the identity of spammers by just analyzing the electronic trail [226, 227].

- **Wireless Network Forensics**: Companies are embracing wireless technology at a rapid pace and the frequency of data leakage and theft is constantly increasing. There is a great need for profiling user activities emphasizing the need for 802.11 network monitoring and content inspection [174]. There is a clear lack of tools and procedures for forensic computing investigations to effectively handle wireless devices. Hence, there are many forms of misuse that escape detection [219].

  VoIP over wireless (VoIPoW) networks is becoming the most popular system for mobile communication in the world. However, studies of attacks on wireless VoIP networks are still in their infancy [164]. Challenges exist in Mobile Ad-hoc Networks (MANETs) where number of the evidence packets is controlled by the level of reliability [156, 241].

- **Application Layer Forensics**: Attacks have moved from network and transport layer to the application layer of the TCP / IP protocol suite. Attacks on Web security include Cross-Site Scripting (XSS), SQL Injection, Buffer overflows, etc. Reliable digital evidence can be provided form the payload of the network data traffic being transmitted to and from the web service [77]. Domain Name Service Forensics is also an important challenge [150] .

- **SCADA Network Forensics**: SCADA systems are widely used in industrial control and automation. Modern SCADA protocols often employ TCP / IP to transport sensor data and control signals. The use of TCP / IP as a carrier protocol and the interconnection of IT and SCADA networks raise serious security issues. Successful attacks on an IT network and its gateway devices could tunnel into a SCADA network, wreaking havoc on the industrial process [108, 109].

- **Grid Forensics**: Grid computing aggregates all kinds of heterogeneous resources that are geographically distributed and requires in-depth security services to protect its resources and data. It also entails suitable forensics techniques that can be employed to assess the responsibility of the wrongdoers. Security teams lack the experience of grid forensics as grid computing is itself, a comparatively newer technology [144].

- **Forensic Data Representation**: Garfinkel [69] predicts an impending crisis in digital forensics as many observes have identified the continuation of current trends. There is a serious need to make digital forensics research more efficient through the creation of new abstractions for data representation forensic processing.

- **Cloud Forensics**: Cloud computing will require a change to corporate and security policies concerning remote access and the use of the data over a browser, privacy and audit mechanisms, reporting systems, and management systems that incorporate how data is secured on a rented computer system that can be anywhere in the world. The complex series of inter-linkages between the cloud provider and the cloud consumer provides a fertile ground for hackers and criminals. Network forensics in cloud computing requires a new investigative mindset, where some data will not be available, some data will be suspect, and only some data will be court ready [121].
- **Intelligent Network Forensics:** An intelligent network forensics system reconstructs intrusion scenarios and makes attack attributions require knowledge about intrusions signatures, evidences, impacts, and objectives. Problem solving knowledge that describes how the system can use domain knowledge to analyze malicious activities is essential. Saad and Traore [190] adapt recent researches in semantic-web, information architecture, and ontology engineering to design method of ontology for network forensics analysis.

1.4 Challenges in Network Forensic Analysis

The challenges in network forensic analysis [6, 120, 183] are elaborated and classified based on the phases in the DFRWS model (2001):

- **Identification**: Attacks must be identified instantaneously and trigger the forensics process. The network events which are malicious must be identified. Future and zero-day attacks must be predicted based on the common attack features. The hacker groups have common types of attacking tools and the frequently utilized techniques.
- **Preservation**: Network traffic is very volatile (dynamic) and must be captured and preserved immediately, otherwise it is lost forever. Most network security tools do not produce hash values for captured data or utilize the same hash algorithms resulting in inconsistencies. Integrity of collected data has to be preserved so that the captured data will pass stringent legal procedures and qualify as evidence in a court of law.
- **Collection**: Capturing real-time network traffic transmitted throughout high-speed networks, without network traffic packets being dropped or lost, is an important challenge. Full packet captures will result in a very large amount of data. The process

can be made efficient by collecting useful data only. Data collected may be reduced by filtering the data according to rules customized for a specific purpose. Network security devices must be able to handle unique input formats and produce different output formats. They must also facilitate universal time synchronization and display time in different formats and time zones between the devices.

- **Examination**: Packet captures are to be examined to identify protocol features which are manipulated. This information is correlated with attack events and the compromise is validated. Validation of attack takes the process to the investigation phase. Packets are reorganized into individual transport-layer connections between machines and the attack behavior is analyzed by replaying the attack.
- **Analysis**: End-to-end and link encryption technology prevents captured network traffic from being analyzed. Logging data from different locations can give reconnaissance of the attacking behavior. The analysis of the aggregation of the data sets, which are from multiple sources, such as firewalls, IDSes and sniffers, can build the chain of the clues and display the full scene of the crime.
- **Presentation**: Many network security devices do not have the ability for visually analyzing the network traffic and log data. Documentation needs to be done for every step in order to ensure that all precautions have been taken and that no privacy violations have taken place.
- **Decision (Investigation)**: IP trace back methods can trace a steady stream of anonymous Internet packets back towards their source to the attack origin. These methods do not rely on knowledge or cooperation from intervening ISPs along the path. The attacker may launch the attack in a very short time and use only a few packets making the traceback process difficult.

1.5 Statement of the Problem

The objective of the present research work is *"to develop a network forensic framework which captures network traffic data, identifies and correlates attack attributes, analyzes and validates the attack by performing data fusion of alerts and attack information, in order to investigate the source of attack and attribute it to an attacker."*

The availability of a framework with thoroughly researched and practical solutions for the examination, analysis and investigation phases strengthens the forensic mechanism. The framework should aggregate existing open source network security tools and facilitate the objective of identifying the source of attacks. Novel, blended and zero-day attacks must also be handled and investigated.

We focus on the following objectives while developing this framework to address these important issues:

1. To propose a generic process model, building on existing digital forensic models, considering phases which are specific to network forensics analysis.
2. To propose a framework for network forensic analysis after identifying phases which have been well researched and focusing only on those phases which do not have a mature prototype implementation.
3. To identify network events at the network, transport and application layers and are correlated with attacks manipulating various protocol header fields.
4. To perform data fusion of information and alerts from multiple security sensors and multiple hosts. Security sensors with complementary and contradictory functions are used.
5. To propose novel approaches for network forensic traceback based on deterministic marking of IP packets with information which can be used to obtain information about the attacker.

1.6 Thesis Organization

Rest of the thesis is organized as follows:

Chapter 2 gives the definition, motivation and classification of network forensics. The literature review of various general process models and frameworks proposed for network forensic analysis is presented.

Chapter 3 proposes a generic process model for network forensics with nine phases. This model is built over the existing models for digital forensics by considering phases specific to network forensics. It also proposes a network forensic framework comprising of challenging phases which do not have mature implementation.

Chapter 4 presents a practical approach to identify network events at the transport and network layer of the TCP/IP protocol stack and correlate them with attacks. The events specific to distributed denial of service (DDoS) attacks and port scan attacks are identified and correlated. Events specific to Cross-Site Scripting attack are also identified and correlated. Analysis is validated using datasets generated in our research lab.

Chapter 5 presents a technique for performing data fusion of information from multiple security sensors. Tools with complementary and contradictory functions are identified and fusion is performed on the alert and attack information generated by them. The proposed technique is validated by applying it on an attack dataset generated in the lab.

Chapter 6 proposes two novel approaches for network forensic traceback: Autonomous System based Deterministic Packet Marking (ASDPM) and Deterministic Router and Interface Marking (DRIM). As part of investigation phase, the packets are deterministically marked by the first router to facilitate traceback to the source of attack packets. Simulations are performed using network simulator ns-2 to validate both the approaches.

Chapter 7 concludes the thesis with summary of contributions towards network forensic analysis and also gives directions for future work.

Chapter 2

Background and Literature Survey

2.1 Introduction

Network forensics deals with the capture and analysis of the trace and log data of network intrusions from the current network security products and provides information to characterize intrusion or misbehavior features. The power of various network forensic analysis tools available as open source can be integrated so that the investigator can have an edge over the attacker. The storage to handle large volumes of data and computing power to analyze the same is now available at cheaper rate. An effective network forensic system will increase the cost of the network crimes and reduce network crime rates.

2.1.1 Network Forensics

The concept of network forensics deals with data found across a network connection mostly ingress and egress traffic from one host to another. Network forensics tries to analyze traffic data logged through firewalls or IDS or at network devices like routers and switches.

Network forensics is defined in [158] as "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."

Ranum [177] is credited with defining network forensics as "capture, recording, and analysis of network events in order to discover the source of security attacks or other problem incidents."

Network forensics involves monitoring network traffic and determining if there is an anomaly in the traffic and ascertaining whether it indicates an attack. If an attack is detected, then the nature of the attack is also determined. Network forensic techniques enable investigators to track back the attackers. The ultimate goal is to provide sufficient evidence to allow the perpetrator to be prosecuted [237].

2.1.2 Classification of Network Forensics Systems

Network forensic systems are classified into different types, based on various characteristics:

- **Purpose:** *General Network Forensics (GNF)* focuses on enhancing security. The network traffic data is analyzed and attack patterns are discovered. *Strict Network Forensics (SNF)* involves rigid legal requirements as the results obtained will be used as evidence for prosecution of the network crimes [186].
- **Packet Capture:** *Catch-it-as-you-can* systems capture all packets passing through a particular traffic point and subsequently analyze them, requiring large amounts of storage. *Stop-look-and-listen* systems analyze each packet in memory and only certain information is saved for future analysis, requiring a faster processor [68].
- **Platform:** The network forensic system can be a *hardware appliance with pre-installed software*. It can capture data, analyze and present the results on a computer interface. It can be exclusively *software* which is installed on a host, analyzing stored packet captures or netflow records.

- **Time of Analysis:** Commercial network forensic analysis appliances involve *real time* network surveillance, signature-based anomaly detection, data analysis and forensic investigation. Many open source software tools are designed for *post-mortem* investigation of packet captures. Full packet data is captured by sniffer tools, stored in a host and analyzed off line at a later time.
- **Data Source:** *Flow based systems* collect statistical information based on some criteria within the network traffic as it passes through the network. The network equipment collects this data and sends it to a flow collector which stores and analyzes the data. *Packet based systems* involve full packet captures at various points in the network. The packets are collected and stored for deep packet inspection.

2.2 Network Forensic Analysis Tools

Network Forensic Analysis Tools (NFATs) [205] allow administrators to monitor networks, gather all information about anomalous traffic, assist in network crime investigation and help in generating a suitable incident response. NFATs also help in analyzing the insider theft and misuse of resources, predict attack targets in near future, perform risk assessment, evaluate network performance, determine hardware and network protocols in use, aggregate data from multiple security tools, and help to protect intellectual propriety. The requirements of analysis tools are also illustrated in [28]. NFATs can paint a general picture of all the events happening on the network. They can decrease the time sent on evidence gathering and data analysis.

NFATs capture the entire network traffic, allow users to analyze network traffic according to their needs and discover significant features about the traffic. NFATs synergize with IDS and firewalls and make long term preservation of network traffic records for quick analysis. The attack traffic can be replayed and attackers' moves can be analyzed for malicious intent [46]. NFATs facilitate organization of captured network traffic packets to be viewed as individual transport layer connections between machines, which enable the user to analyze protocol layers, packet content, retransmitted data, and extract traffic patterns between various machines. There are some NFATs available that provide reliable data acquisition and powerful analysis capabilities. A partial list of NFATs [31] with a brief description is given below:

- **NetDetector** [153]: Captures intrusions, integrates signature-based anomaly detection, reconstructs application sessions and performs multi time-scale analysis on diverse applications and protocols. It has an intuitive management console and full standards based reporting tools. It imports and exports data in a variety of formats. NetIntercept (now part of Niksun) was also a leading NFAT which captured network traffic, reassembled individual data streams, analyzed them by parsing the protocol, and generated a variety of reports from the results.

- **NetWitness** [55]: Captures all network traffic and reconstructs the network sessions to the application layer for automated alerting, monitoring, interactive analysis and review.

- **Iris** [195]: Collects network traffic and reassembles it in its native session based format, reconstructs actual text of the session, replays traffic for audit trial of suspicious activity, provides a variety of statistical measurements and has advanced search and filtering mechanism for quick identification of data.

- **Infinistream** [147]: Utilizes intelligent Deep Packet Capture (iDPC) technology and performs real-time or back-in-time analysis. It does high-speed capture of rich packet details, statistical analysis of packet or flow based data and recognizes hundreds of applications. It uses sophisticated indexing and Smart Recording and Data Mining (SRDM) for optimization.

- **Solera DS 5200** [149]: DS 5200 is an appliance for high-speed data capture, complete indexed record of network traffic, filtering, regeneration and playback. DeepSee forensic suite has three softwares – Reports, Sonar and Search – to index, search and reconstruct all network traffic.

- **OmniPeek** [232]: Provides real-time visibility into every part of the network. It has high capture capabilities, centralized console, distributed engines, and expert analysis. Omnipliance is a network recording appliance with a multi-terabyte disk farm and high-speed capture interfaces. OmniEngine software captures and stores network traffic. OmniPeek interface searches and mines captured data for specific information.

- **SilentRunner** [1]: Captures, analyzes and visualizes network activity by uncovering break-in attempts, abnormal usage, misuse and anomalies. It generates an interactive graphical representation of the series of events and correlates actual network traffic. It also plays back and reconstructs security incidents in their exact sequence.

- **NetworkMiner** [146]: Captures network traffic by live sniffing, performs host discovery, reassembles transferred files, identifies rogue hosts and assesses how much data leakage was affected by an attacker. Xplico Captures Internet traffic, dissects data at the protocol level, reconstructs and normalizes it for use in manipulators. The manipulators transcode, correlate and aggregate data for analysis and present the results in a visualized form.

- **PyFlag** [173]: An advanced forensic tool to analyze network captures in libpcap format while supporting a number of network protocols. It has the ability to recursively examine data at multiple levels and is ideally suited for network protocols which are typically layered. PyFlag parses the pcap files, extracts the packets and dissects them at low level protocols (IP, TCP or UDP). Related packets are collected into streams using reassembler. These streams are then dissected with higher level protocol dissectors (HTTP, IRC, etc).

2.3 Network Forensic Process Models

Proven investigative techniques and methods exist for the traditional computer forensic discipline. However as we become more and more networked and mobile in home and business, there is a need to expand our forensic view from disk level to the network level. There is a need to factor this transition into concepts, designs and prototypes. Various digital forensic models were proposed to handle the networked environments since 2001.We use the term 'model' to imply a theoretical representation of phases involved in network forensics. The model may or may not have been implemented.

The first attempt to apply digital forensic science to networked environments was taken up as one of the objectives in the first Digital Forensic Research Workshop (DFRWS), 2001 and a framework [158] was proposed. The framework included the following steps: identification, preservation, collection, examination, analysis, presentation, and decision. Reith et al [181] improvised the above model and produced an abstract digital forensic model that is not dependant on a particular technology or crime. Preparation and approach strategy phases have been added and returning the evidence in place of decision phase has been included.

Mandia and Prosise [131] developed an incident response methodology which is simple and accurate. An initial response phase to ascertain the incident and formulation of a response strategy was added. The investigation phase included collection and analysis phases as in the earlier models. Presentation was called reporting and resolution phase. It suggested improvements, changes, and long term fixes.

Casey and Palmer [34] proposed an investigative process model to encourage a complete rigorous investigation, ensure proper evidence handling and reduce chance of mistakes. Apart from the common phases, assessment phase validated the incident and a decision was taken whether to continue with the investigation. Harvesting, reduction, organization & search phases arranged the data so that it is the smallest set with high potential evidence. Persuasion and testimony phases presented the case in layman terms.

Carrier and Spafford [30] proposed an integrated digital investigation process based on the techniques used for physical investigations. Readiness phase ensured operations infrastructure is geared up. Survey, search and collection phases gathered and processed the data. Reconstruction is similar to analysis phase. Documentation phase recorded all the evidence.

Ó Ciardhuáin [41] combined existing models and proposed an extended model of cybercrime investigations which represents the information flows and captures the full investigation. Awareness is the first step which announces investigation. Authorization is taken from internal and external entities. Planning involves strategies and policies. Dissemination is also done for guiding future investigations and procedures.

Baryamureeba and Tushabe [13] proposed an enhanced digital investigation process model reorganizing the phases in [30]. The model was built based on the physical crime investigation process. Two new phases traceback and dynamite are included. They have sub-phases like investigation, authorization, reconstruction and communication giving clarity and granularity to the major phases.

Beebe and Clarke [15] proposed a multi-tier, hierarchical, objectives based framework for digital investigative process in contrast to the single tier higher order process models. Their model consists of the common phases in first tier, providing simplified view and a conceptual understanding. These common phases consist of sub-phases, placed in lower tiers, to provide specificity and granularity, guided by principles and objectives. The sub-phase structure for the Data Analysis Phase was presented and analyzed.

Ieong [90] proposed a digital forensics investigation framework FORZA, which incorporates legal issues. The author identifies 8 roles to fulfill the fundamental principles for digital forensic investigation namely, reconnaissance, reliability and relevancy. He lists 6 questions for each role - what, why, how, who, where and when. These roles and questions are incorporated into the Zachman's framework for enterprise architecture and FORZA is composed. This model is being automated by developing a data acquisition scripts generator which will collect relevant information from the network logs.

Selamat et al. [198] produced the mapping process between the processes / activities and output for each phase in Digital Forensic Investigation Framework (DFIF). A study of the existing digital forensic frameworks is done and then the mapping is constructed. The authors group and merge the same activities or processes that provide the same output into an appropriate phase. The mapping process is designed in order to balance the process on achieving the overriding goal that can produce concrete evidence for presentation in a court of law. The phases are preparation, collection and preservation, examination and analysis, presentation and reporting, and disseminating the case.

Yong-Dal [242] presented a new digital forensics investigation procedure model which comprises the following phases: investigation preparation, classifying cyber crime, deciding investigation priority, investigating damaged (victim) digital crime scene, criminal profiling consultant and analysis, tracking suspects, investigating injurer digital crime scene, summoning suspect, additional investigation, writing criminal profiling, and writing report.

Himanshu et. al. [86] proposed a process model comprising of four phases: Preparation, Detection and strategy development, Local investigation, and recovery and Incident closure. The authors have developed a prototype system called 'Palantir' to provide a software environment that supports the collaborative response and investigation process.

All the models mentioned above are applicable to digital investigation and include network forensics in a generalized form. Ren and Jin [186] were the first to discuss network forensics taxonomy, conceptual model, legal principles, key techniques, canonical processes and its accessory facilities, system architecture and deployment. The general process model for network forensics had the following steps: capture, copy, transfer, analysis, investigation and presentation.

2.4 Network Forensic Frameworks

Many variant network forensic process models were proposed with different phases as discussed in the previous section. Researchers developed many frameworks which implement the some or all of the phases in these models. We use the term 'framework' to mean prototype implementation. They are classified under various categories based on distributed systems, soft computing, honeypots, aggregation and heterogeneous systems.

2.4.1 Distributed Systems Based Frameworks

Internet and LANs are distributed in nature and networks attack events are logged in clients at various locations. There is a need to collect these logs, fuse them and analyze on a central server. A general scheme for the frameworks is shown in Figure 2.1.

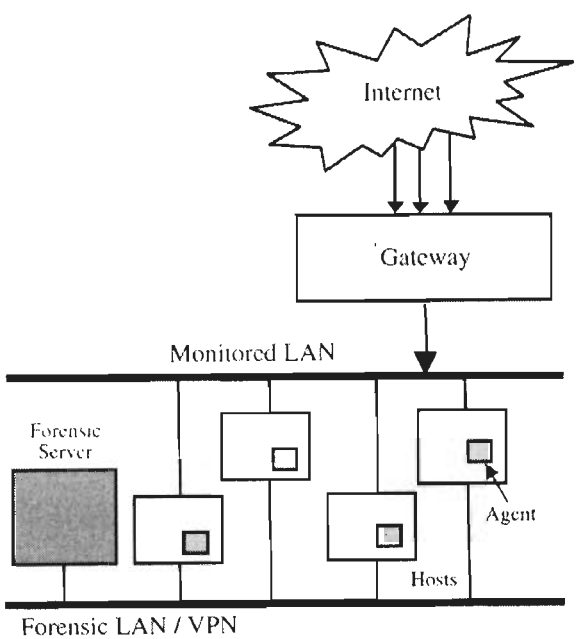

Figure 2.1 – A general scheme for distributed frameworks

Shanmugasundaram et al. [201, 203] proposed ForNet, a distributed network logging mechanism to aid digital forensics over wide area networks. It has two functional components – Syn-App, designed to summarize and remember network events for a period of time and a Forensic Server, which is a centralized authority for a domain that manages a set of SynApps in that domain. A forensic server receives queries from outside its domain, processes them in co-operation with SynApps and returns query results back

to the senders after authentication and certification. The overall architecture involves a network filter, synopsis engine, synopsis controller, configuration manager, security manager, storage management and query processor. Evidence of crimes can be found in packet headers or application dependent payloads. ForNet can identify network events like TCP connection establishment, port scanning, connection record details and uses bloom filters to track other events.

Ren [184] 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 misbehavior packets based on adaptive filter rules, analyzing the overall cooperative database to discover the potential misbehavior, and replaying the misbehavior for forensic analysis. It can discover the profile of the attacker and obtain clues for further investigation.

Ren and Jin [185] further extended their previous model [184] 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 forensic 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 an incident and improve the ability of emergence response.

Tang and Daniels [213] proposed a simple framework for distributed forensics. It is based on distributed techniques providing an integrated platform for automatic evidence collection and efficient data storage, easy integration of known attribution methods and an attack attribution graph generation mechanism. The model is based on proxy and agent architecture. Agents collect, store, reduce, process and analyze data. Proxies generate the

attack attribution graph and perform stepping stone analysis. This model aims at providing a method to collect, store and analyze forensic information. It also provides automatic evidence and quick response to attacks.

Nagesh [143] implemented a distributed network forensics framework using JADE mobile agent architecture. A node acting as a server, hosting the network forensics-agent, dispatches mobile agents to monitored heterogeneous locations. They gather network traffic logs, examine them and return the results which will be displayed on a user interface. The interface enables the analyst to specify data to be collected and analyze the resultant network events displayed. The solution automates collection of network data from distributed heterogeneous systems using mobile agents. The implementation is scalable, reduces network traffic, addresses single point of failure and provides real-time monitoring.

Wang et al. [228] developed a dynamical network forensics (DNF) model based on the artificial immune theory and the multi-agent theory. The system provides a real-time method to collect and store data logs simultaneously, provide automatic evidence collection and quick response to network criminals. The system includes a Forensic Server and three agents namely Detector-Agent, Forensics-Agent and Response-Agent. The Detector-Agent captures real-time network data, matches it with intrusion behavior and sends a forensics request message to the Forensics-Agent. The Forensics-Agent collects the digital evidence, creates a digital signature using a hash function and transmits the evidence to the Forensic Server. The Forensic Server analyzes evidence and replays the attack procedure. The Response-Agent is being developed.

The implementations discussed above use client–server / agent–proxy architectures to collect the attack features and analyze them. The advantage with them is that they represent the distributed structure of Internet. They are limited in identifying the network features correctly and some components are still being developed.

2.4.2 Soft Computing Based Frameworks

The soft computing implementations are used to analyze captured data and classify the attack data. Neural network and Fuzzy tools are used for validation of attack occurrence. A general scheme for the fuzzy logic based frameworks is shown in Figure 2.2.

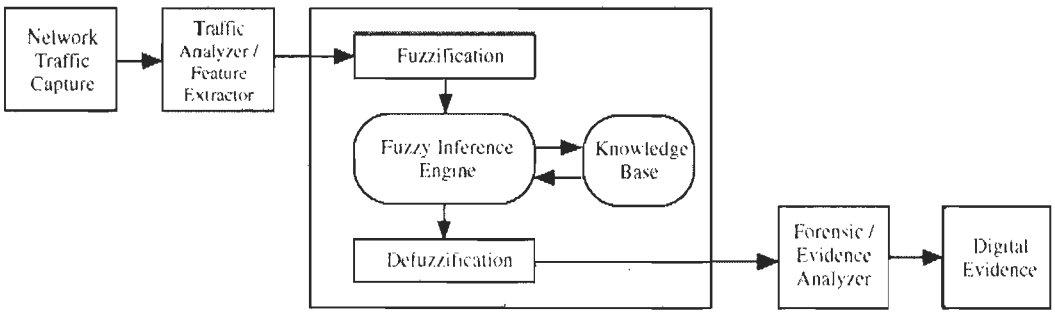

Figure 2.2. A general scheme for fuzzy based frameworks

Kim et al. [110] developed a fuzzy logic based expert system for network forensics to aid the decision making processes involving sources of imprecision that are non-statistical in nature. It can provide analyzed information for a forensic expert and reduce the time and cost of forensic analysis. The framework consists of six components. Traffic analyzer captures network traffic and analyses it using sessionizing. Knowledge base stores rules which are used by the fuzzy inference engine. The rules are written for various attacks using linguistic variables and terms. Membership functions are defined for each fuzzy set and a crisp value of degree of membership is determined. Each input variables crisp value is first fuzzified into linguistic values. Fuzzy inference engine derives output linguistic values using aggregation and composition. Defuzzification defuzzifies the output values into crisp values and the forensic analyzer validates the occurrence of an attack.

Liu and Feng [125] proposed the Incremental Fuzzy Decision Tree-Based Network Forensic System (IFDTNFS). This is an efficient way to create a classification model by extracting key features from network traffic by providing the resulting fuzzy decision tree with better noise immunity and increasing applicability in uncertain or inexact contexts. IFDTNFS consists of three components: network traffic separator, traffic detector, and forensic analyzer. The network traffic separator component is responsible for capturing network traffic and separating it according to the service type, and directing the separated traffic to corresponding traffic detector. The traffic detector consists of four components: feature extractor extracts features from network traffic, fuzzy rule base is the knowledge base using which fuzzy decision trees are built, rule base updater adds new samples to the fuzzy decision tree that has been constructed and also adjusts the optimal tree size, and fuzzy inferencer fuzzifies input values and processes them with the rule base. Forensic analyzer includes collecting relative event data, analyzing correlated information relating with the event, and establishing digital evidences.

Zhang et al. [244] proposed network forensic computing based on Artificial Neural Network and Principal Component Analysis (ANN–PCA). The major challenge faced in network forensics is massive information to be stored and analyzed. Extraction of key features reduces storage by correlating the features with attacks. ANN–PCA techniques are used to identify all possible violations, extract features and build signatures for new attacks. Classification is done using FAAR algorithm to mine association rules and calculate the PCA values. Classification accuracy increases and information storage size decreases after feature extraction is performed using ANN–PCA.

Neurofuzzy Techniques [8] were used by Anaya et al., to address the challenges of enormous data to be logged and analyzed for network forensic computing. The Neurofuzzy solution is based on ANN and fuzzy logic and is used for evidence differentiation into normal and abnormal flows. ANNs are used in information processing to learn from the data and generalize a solution. Fuzzy logic is used to generate a grade of membership to different behaviors so that attacks are determined. The model consists of four modules. The Monitor control module stores all the network information. Information preprocessing module is made up of syntax sub module and correlation sub module. Syntax module is responsible for normalizing the inputs and correlation sub module aggregates the different flow formats and groups the PDUs into a flow. Dependencies module relates all network element logs and takes decision about the flows. The decider module distinguishes between normal and abnormal flows.

Liao et al. [119] proposed a Network Forensic System based Fuzzy Logic and Expert System (NFS-FLES), an effective and automated analysis system, which guarantees evidence reliability by collecting information from different sensors. It also analyzes computer crimes and makes automatic digital evidence using the approach of fuzzy logic and expert systems. The NFS-FLES consists of the following components – traffic capture, feature extractor, fuzzification, fuzzy inference engine, knowledge base, defuzzification and forensic analyzer. The whole operation is done in four parts – real-time forensic data acquisition and preprocessing, knowledge base construction and dynamic rule generation, fuzzy linguistic operation of input attack data and computing aggregation fuzzy value and total fuzzy score of every kind of attack. The forensic result is then output in time.

The soft computing tools give desirable results, provided the rules to determine attack traffic can be generated. Detecting zero-day attacks is also a challenge.

2.4.3 Honeypot Based Frameworks

Honeypot frameworks are used to attract the attackers so that their process methodology can be observed and analyzed to improve defense mechanisms. Honeytraps [238] were proposed as a deception tool to collect information about blackhat activities and learn their techniques so that protection and defense mechanisms can be formulated. Honeytraps are Honeypot or Honeynet systems which attract intruders to enter a host by emulating a known vulnerability. Once an attacker penetrates a honeytrap, data are captured to detect and record his actions. This data can be used to profile the tools and tactics used by the attackers putting the investigators in an offensive mode.

Two architectures, serial and parallel, facilitate the forensic investigation. The serial architecture places honeytrap between the Internet and the production system. Recognized users are filtered to the production systems and blackhats are contained in the honeytrap. The parallel architecture allows honeytrap to be independent of the production system. Once the system detects the presence of blackhat, the forensic alert system is activated. If the attack is detected, forensic processes are activated on the honeytrap and production systems. Once the attack is contained, the investigation process is begun to determine the identity of the intruder on the production system.

Thonnard and Dacier [216] proposed a framework for attack patterns' discovery in Honeynet data. Their work aims at finding groups of network traces sharing various kinds of highly similar patterns within an attack data set. They design a flexible clustering tool and analyze one specific aspect of the Honeynet data, time series of attacks. Malicious network traffic is obtained from the distributed set of Honeynet responders. Time signature is used as a primary clustering feature and attack patterns are discovered using attack trace similarity. Attacks are detected as a series of connections. Zero-day and polymorphic attacks are detected based on similarity to other attacks. Knowledge from the Honeynet data is used in intrusion detection efforts. The clustering method does feature selection and extraction, defines a pattern proximity measure and groups similar patterns. The result of clustering applied to time series analysis enables detection of worms and botnets in the traffic collected by Honeypots.

Merkle [137] investigated automated analysis of network based evidence in response to cyberspace attacks. The two major challenges of network forensics, namely 'complexity' problem of analyzing raw traffic data and 'quantity' problem of amount of

data to analyze are addressed in his solution. The model integrates results of data logged by various tools into a single system that exploits computational intelligence to reduce human intervention. This integrated tool is referred as 'automated network forensic' tool. An isolated network of virtual machines is built into a Honeynet. Open source forensic tools are used for collecting data. The information produced by various tools in one stage is characterized and transformed for use by other tools in the succeeding stages. Time consuming and error prone processes are identified and automated. The data sets are partitioned, system is trained and then tested.

Honeynets meet the expectations of forensic analyzers but they cannot be used for investigative purposes as evidence generated is not valid in legal system.

2.4.4 Aggregation Framework

Network forensic analysis involves many phases and various security tools can be used for specific phases [167]. The aggregation frameworks harness the strength of these tools to facilitate forensic investigation rather than building a new tool from scratch.

Almulhem and Traore [7] proposed a Network Forensics System (NFS) that records data at the host level and network level. The system consists of three main modules – marking, capture and logging. Marking module decides whether a passing packet is malicious. One or more sensors (like IDS) report suspicious IP addresses. Capture module is a collection of lightweight capture modules which wait for the marked packets. They are arranged in an order for reliable transportation to the logging module for archival. Logging module is a system repository where the attack data are being stored. It uses three types of loggers – host logger stores data sent by capture module, sensor logger stores sensors' alerts, where raw logger is optional and is used when other loggers fail. The capture module was implemented using Sebek, marking module used Snort IDS, and logging module used server-side Sebek, Snort's barnyard tool, ACID Lab and TCPDump.

Nikkel [151] proposed a Portable Network Forensic Evidence Collector (PNFEC) which was built using inexpensive embedded hardware and open source software. The compact and portable device has been designed for traffic collection between a network and a single node, having specific modes of operation, rapid deployment and stealthy inline operation. The traffic on the Ethernet Bridge is promiscuously captured using various pcap based capture tools and stored on a hard disk. The operating system,

additional software, configuration files and investigator activity logs are stored on a compact flash. Administrative access controls various aspects of the device like startup, scheduling, configuration of capturing filters, forensic functions such as preserving and transferring the evidence. The PNFEC is easy to deploy and operate (plug and play). The network traffic collected can be stored in encrypted form. PNFEC also controls filtering of captured data using TCPDump to ensure there are no privacy violations. A script is used to create a cryptographic hash of the packet capture files and preserved. OpenBSD is the operating system used, as many of the functionalities like secure access, packet capture, encrypted file system, evidence preservation, disk wiping and formatting tools, are included by default. Tools for trouble shooting (TCPFlow) and pcap management (tcpslice) are also added. PNFEC operates in three modes – investigator, server and user.

Vandenberghe [223] proposed a Network Traffic Exploration (NTE) Application being developed by Defense Research and Development Canada (DRDC) for security event and packet analysis. This tool combines six key functional areas into a single package. They are intrusion detection (signature and anomaly based), traffic analysis, scripting tools, packet playback, visualization features and impact assessment. NTE has three layers with MATLAB as development environment, low level packet analysis library and unified application front end. It provides an environment where statistical analysis, session analysis and protocol analysis can exchange data.

2.4.5 Heterogeneous Frameworks

Sekar et al. [197] pointed out that one fundamental invariant across all network attacks (present and future) is that there must be communication among attacker, the associated set of compromised hosts and the victim(s). This communication enables the attack to progress and fortunately this communication is visible to the network. The authors outline a high-level vision of an investigative capability for the Internet that permits identification and fine-grained analysis of the communication patterns leading to an attack. They build the framework on two fundamental components - *Attacker Identification* and *Attack Reconstruction.* Attacker Identification is the ability to accurately pinpoint the source(s) of the attack or infection. Attack Reconstruction is the process of inferring which communications carry the attack forward.

Wang and Daniels [230] developed a novel graph-based approach toward network forensics analysis. A hierarchical reasoning framework consisting of two levels – local reasoning aims to infer the functional states of network entities from local observations and global reasoning aims to identify important entities from the graph structure and extract groups of densely correlated participants in the attack scenario. The basic architecture has *Evidence collection module, Evidence preprocessing module, Attack knowledge base, Assets knowledge base, Evidence graph manipulation module, and Attack reasoning module.*

Rekhis et al., [182] developed a system for Digital Forensic in Networking (DigForNet) which is useful to analyze security incidents and explain the steps taken by the attackers. DigForNet uses intrusion response team knowledge and formal tools to reconstruct potential attack scenarios. They integrate the analysis performed by the IRT on a compromised system through the use of the Incident Response Probabilistic Cognitive Maps (IRPCMs). They also provide a formal approach to identify potential attack scenarios using I-TLA (Investigation-based Temporal Logic of Actions). They generate executable potential attack scenarios using a model checker tool called I-TLC (Investigation based Temporal Logic Model Checker).

Locasto et al., [126] proposed automatic repair validation for online network forensics, a concept which helps in self healing with verification. They designed and implemented a model, *Bloodhound*, which tags and tracks information between the kernel and the application and correlates symptoms of exploits with high-level data. The tasks include preferentially recording network traffic, searching through these flows after the application has been healed, and replaying the relevant flows to test this repair. The work can be improved by dealing with taint-tracking through the application itself.

Cohen [43] introduced PyFlag as an innovative and advanced network forensic platform. It integrates the following aspects into a single package. It can efficiently process very large capture files, extract high level information and is able to substantiate each deduction. The design architecture includes I/O source, virtual file system, and scanners. It also has the following network forensics modules like the stream reassembler, packet handlers, and stream dissectors. The model needs to be improved to accommodate large number of protocols present on the Internet and there is a dire need to flexibly and quickly adapt to these new protocols.

Rachid Hadjidj et al., [78] developed an Integrated E-mail Forensic Analysis Framework (IEFAF) using statistical and machine learning techniques complemented with social networking techniques. Major functionality of IEFAF include the ability to investigate e-mail archives and compute the required statistical distributions, results plotted using different visualization techniques, capability of keyword searching, development of data mining models to help classify e-mails in different categories or cluster them according to some undiscovered relationships, detection of anomalous behaviors by matching the observed e-mail communication with the pre-recorded normal communication model of users, performance of e-mail authorship analysis on the basis of stylometric features, and capability to map selected IP addresses by applying geographical localization technique to determine the physical location of that IP.

2.5 Identification and Correlation of Network Events

Bhattacharya et al. [22] proposed PRIVDAM, a data mining based intelligent architecture of a privacy violation detection and monitoring system whose purpose is to detect possible privacy violations and to prevent them in the future. It elaborates on the use of network characteristics for differentiating between normal network traffic and potential malicious attacks. These attacks are usually hidden in common network services like HTTP, FTP, UDP etc.

Lee et al. [116]proposed a method for the description of the threats for developing protection profiles (PP) or countermeasures by introducing the concept of the assets protected by Target of Evaluations (TOE). The security environments consist of assumptions, threats, and organizational security policies and an editor of the PP describes the threats. Kim and Kwon [111] proposed a method for applying security engineering to build security countermeasures. It identifies the threats, undesirable event characterized in terms of a threat agent, a presumed attack method, a motivation of attack, and an identification of the information or systems under attack.

DDoS threats deplete the network resources rapidly particularly link parameters. Modeling these attacks provides a strong base for analyzing the attack characteristics. Jayashree and Easwarakumar [98] proposed a solution which uses active networks for implementation. They present a model based on packet attributes to characterize the attack traffic and a detection and response framework based on the model.

Law enforcement agencies need to monitor, detect, and analyze undesirable network traffic. This may be against the goal of maintaining privacy of individuals whose network communications are being monitored. Pande et al. [159] proposed PickPacket, network monitoring tool that handles the conflicting issues of network monitoring and privacy through its judicious use.

Capability based alert correlation uses notion of capability to correlate IDS alerts where capability is the abstract view of attack extracted from IDS alerts/alert. To make correlation process semantically correct and systematic, there is a strong need to identify the algebraic and set properties of capability. Pandey et al. [160] identify the potential algebraic properties of capability in terms of operations, relations and inferences. Time-based notion was added which avoiding temporal ambiguity between capability instances.

Raghavan and Raghavan [175] present an evidence composition model based on time of occurrence of such events. The time interval between events promise to reveal many key associations across events, especially on multiple sources. The time interval is then used as a parameter to a correlation function which determines quantitatively the extent of correlation between events.

2.6 Multisensor Data Fusion

Network forensics deals with data found across a network connection mostly ingress and egress traffic from one host to another. Various network security tools can be applied to this traffic and data fusion performed on various output values generated. Many models have been proposed for data fusion of intrusion detection data. They are mostly based on Dempster-Shafer (DS) theory of evidence. The models are surveyed to obtain the direction for data fusion in the context of network forensic analysis.

Onwubiko [155] proposed a model for data fusion in security evidence analysis. Pieces of evidence from heterogeneous defense systems are fused / combined to detect the attacks. Data fusion techniques effectively combine evidence from multiple sensors or a single sensor used in multiple places. A multi-source fusion system which uses the DS technique to combine beliefs from multiple security tools is investigated.

Intrusion Detection System based on Data Fusion Method (IDSDFM) [217] was proposed by Tian et al., which correlates and merges alerts of different types of IDS. The set of alerts can be partitioned into different alert tracks. IDSDFM consists of alert correlation module, security estimation module, and management & control module. Two types of alert aggregation is done – alerts that make up an attack and alerts that represent the behavior of a single attacker. Distributed IDSDFM (DIDSDFM) [231] was proposed earlier by Wang et al. consisting of two layers. The lower layer consists of host and network based sensors, which collect local features and differentiate easy-to-detect attacks. The upper layer is a fusion control center, which makes global decision on these events by adopting Dempster's combination rule.

Network Security Situation Awareness (NSSA) [124] is a new notion where situation security analysis is made to understand the intrusion alerts and take appropriate actions. The network security situation elements are analyzed, data is fused, and correlation identified using colored petri-net and the D-S theory of evidence. NSSA fuses data from tools of IDS, virus detection system, firewall, netflow records to monitor the network for intrusions and predict course of action. The security events are preprocessed and situation assessment is done through correlation to gather information about the attacks.

A pattern recognition approach [71] is applied to network intrusion detection based on the fusion of multiple classifiers. Each member of the classifier ensemble is trained on a distinct feature representation of patterns, then the individual results are combined using a number of fusion rules. Expert knowledge about the characteristics that distinguish attacks from normal traffic can be used to extract features based on content (payload), intrinsic (network connection information) and traffic features (statistics). This evidence is combined to produce a final decision.

Intrusion Evidence Automated Analysis System (IEASS) framework [122] collects the evidences from multiple network sensors. Alerts of IDS are treated as primary evidences and logs from vulnerability scanner, network monitors, firewalls and others can be used as secondary evidences. The Log Collection Agent (LCA) collects intrusion logs from various sensors, pre-aggregates and adds a signature. LCA has security event parsers which use regular expressions to automate log aggregation. An Eliminate Redundancy Algorithm (ERA) is proposed for retaining effective information after removing redundant information.

Analytical Intrusion Detection Framework (AIDF) [211] was proposed for information integration and realization of a distributive IDS environment with multiple sensors and a mechanism for selecting and integrating the probabilistic inference results to aid most probable forensic explanation. The probabilistic approach is also used for integrating information from different sensor sources in a distributive NIDS environment.

A novel cyber fusion system [236] is proposed to specifically address the tracking and projection of multistage attacks. It uses information fusion to provide situation awareness and threat prediction from massive volumes of sensed data. The system is based on Information Fusion Engine for Real-time Decision-making (INFERD) and Threat Assessment of Network Data and Information (TANDI). INFERD efficiently correlates IDS alerts and identifies individual multistage attacks and provides situational measures of the identified attacks. TANDI fuses information extracted from each attack track estimates, to determine threatened entities and differentiates them by threat scores.

Multi – Metric – Multi – Link (M3L) PCA [36] based method provides a technique of fusing and combining data of heterogeneous monitors spread throughout the network. PCA aims to reduce the dimensionality of dataset in which there are large number of interrelated variables while retaining as much as possible the variation present in the dataset. These components are called PCs. Data fusion are also discussed in [161] [233].

2.7 Internet Packet Traceback

IP Traceback [4, 17, 65, 79, 192] is an important strategy to contain the ongoing attacks or to investigate and attribute the attacks in the post mortem stage. The traceback mechanism is shown in Figure 2.3. IP traceback problem is defined as "identifying the actual source of any packet sent across the Internet". IP traceback techniques are not capable of preventing and mitigating the attack. They can only identify the source of attack packets. However, this information can be used to conduct post mortem investigation of the attack.

The traceback measures are classified as reactive or proactive. A traceback technique is considered reactive when the process is initiated on the fly in response to an attack. Link testing is a reactive technique, for which input debugging [210] and controlled flooding [26] are examples. These methods make use of the large amount of traffic in a DDoS attack and make attack detection decisions while the attack is in progress.

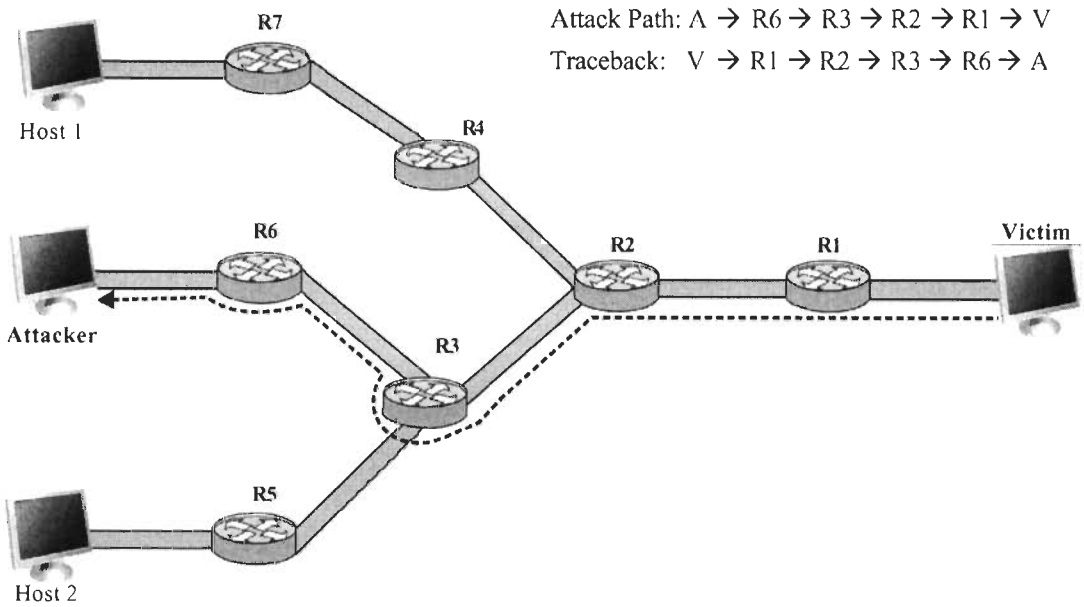

Figure 2.3. IP Traceback Mechanism

The techniques fail when the attack traffic subsides and hence are not suitable for post-mortem analysis. A mechanism is proactive when the traceback information is concurrently generated or stored, as the packets are routed through the network. Proactive measures include packet logging, packet marking (probabilistic and deterministic), hybrid approaches (logging and marking) and AS-level traceback techniques.

Packet logging at key routers facilitates identification of the true origin of attack traffic throughout the Internet. The major problem is the processing and storage resources required at the routers. Snoeren et al. [206] proposed source path isolation engine (SPIE) capable of tracing a single IP packet using packet logging. SPIE system has a centralized traceback manager (STM) to control the data generation agents (DGA) and collection and reduction agents (SCAR). A hash of multiple fields in the IP packet header is computed and logged in the digest tables using space-efficient Bloom filters. When a traceback request is made, STM dispatches the information to appropriate SCARs, which query the SPIE enabled router. STM reconstructs the attack path using the results.

Baba and Matsuda [11] proposed an autonomous management network (AMN), which has a monitoring manager which receives requests from sensors and queries the tracers. Sensors detect the attacks and send the tracing requests. Tracers, implemented in forwarding nodes, maintain log information about incoming packets and their datalink-level identifiers. The tracer compares the log data with information about the tracing packet and finds a trace path.

Zhang and Guan [243] proposed a Bloom filter-based topology-aware single packet IP traceback system, TOPO, which utilizes router's local topology information for traceback. When a packet travels through the TOPO enabled router, it records the packet signature and predecessor information. If an attack packet is identified by the victim, the victim's address, packet signature, and packet arrival time, are reported to TOPO as a traceback request. All responses from queried TOPO-equipped routers are gathered by TOPO to generate the attack graph. The attack graph is used for further analysis and traceback.

Packet-marking involves placing the routers' part or complete address into the IP packet along the attack path. Packets are marked either probabilistically or deterministically. Packets are marked by selecting them randomly with a fixed probability (PPM) or packet may be marked only once by the ingress edge router (DPM).

Probabilistic packet marking (PPM) techniques require many packets for convergence of attacker information. Savage et al. [193] proposed PPM where each router receives a stream of packets and probabilistically marks them with partial address information. Packets are marked with a probability p = 0.04 (one in 25 packets). The victim can construct the attack path comprising of all PPM enabled routers after it has received enough packets. The IP Identification field (IP ID) within the IP header is used to store the traceback information. Many variants of PPM have been proposed.

Song and Perrig [208] proposed advanced and authenticated packet marking (AAPM) to further reduce the storage space requirements by encoding the IP address into an 8 bit hash value. It is also assumed that the victim has a complete network map of all upstream routers. When an attack is detected, the marks are extracted and the attack path is reconstructed by comparing router IP address hashes derived from the network map. Authentication marking scheme based on message authentication codes (MAC) is used to prevent tampering.

Dean et al. [50] proposed algebraic packet marking (APM) that employs algebraic techniques from the field of coding theory to calculate the values of 15-bit marks as points on polynomials. Several schemes like full path encoding, randomized path encoding and edge encoding are used. Many attack path reconstruction methods are presented. Encoded path information can be stored in the IP Fragment ID (16-bit) field of the IP header. Decoding is done by Vandermonde matrix.

Aljifri et al. [5] proposed a Simple, Novel IP Traceback using Compressed Headers (SNITCH) that is based upon PPM. This technique employs header compression to increase the number of bits available for insertion of traceback information. If an initial frame is sent with a full header, subsequent frames can be sent without the static content (the context) being included in the header.

Yaar et al. [235] proposed fast internet traceback (FIT) that has a packet marking scheme deployed at routers, and path reconstruction algorithms used by end hosts. FIT packet markings contain three elements: a fragment of the hash of the marking router's IP address, the number of the hash fragment marked in the packet, and a distance field. Victim uses the hash fragments and distance calculations from the markings in conjunction with its router map.

Deterministic packet marking (DPM) mechanisms are well suited for network forensics as a stream of few packets can sufficiently determine the source of the attacker. They are discussed in detail in the next section. *Hybrid mechanisms* combine logging and marking of packets. Duwairi and Govindarasu [2] proposed distributed link list traceback (DLLT) based on "store, mark and forward" approach. A single marking field is allocated in each packet. Any router that decides to mark the packet, stores the current IP address found in the marking field along with the packet ID in a special data structure called *Marking Table* maintained at the router, and then marks the packet by overwriting the marking field by its own IP address, and then forwards the packet as usual. The marking field serves as a pointer to the last router that did the marking for the given packet and the marking table of that router contains a pointer of the previous marking router.

Jing et al. [101] proposed hierarchical IP traceback system (HITS) with three components for marking, evidence collection and traceback processing. Each traceback enabled router has a Marking Agent (MA) for logging the marking information into its cache or local log database. Traceback Service Provider (TSP) manages the MAs and collects logs from them into a centralized log database. Evidence Collection Agent (ECA) is responsible for collecting marking information as evidence for attacks. The 16 bit ID field and 13 bit Offset field are used to encode the marking information, which consists of 8 bit Old TTL value and 21 bit hash value of the MAs IP address.

Gong and Sarac [73] developed hybrid single packet IP traceback (HIT) based on marking (append router ID into the marking field) and logging (compute and record

packet digest). Traceback enabled routers audit traffic and a traceback server having the network topology information constructs attack graph by querying routers. Each router has an ID of 15 bits. The mark is stamped overloading the ID field. The left most bit is used as logging flag bit, set to 1 if router commits logging.

Jing and Lin [100] proposed logging & deterministic packet marking (LDPM) built on a distributed hierarchical IP traceback system. Autonomous System (AS) is considered an independent unit of the Internet. Two kinds of ASes (source and destination) and two kinds of routers (ingress and border) are considered. The goal of LDPM is to trace the special edge connecting ingress and border routers. The 16 bit Id field is used to store the AS ID and 13 bit fragment offset field stores the router ID.

Messaging techniques are also proactive. Bellovin [20] proposed that each router probabilistically selects a packet and generates an ICMP traceback message (iTrace) that is sent to the same destination as the packet. One iTrace message, generated for every 20000 packets, includes the router id, timestamp, previous and next IP addresses, MAC addresses and some HMAC authentication data. Intention driven iTrace [132] is an enhancement to enable the receiver to request for on demand traceback. This request is received by the upstream routers which set an intention bit in the packet forwarding table. The AS based traceback and router interface marking mechanisms are discussed laters.

2.7.1 Deterministic Packet Marking

Belenky and Ansari [16, 18] first proposed the idea of deterministic packet marking (DPM) where only the ingress edge routers mark the packets and all other routers are exempt from marking. Each border router marks every packet with its identity before the packet enters the network. DPM uses the 16-bit ID field and the 1-bit reserved field for marking. The IP address of edge routers is split into two segments with 16 bits each. Victim can recover the address once it receives both the segments from the same router. One bit is used as a flag to indicate which portion of the IP address is carried.

Rayanchu and Barua [178] proposed a deterministic edge router marking (DERM) where the entire marking information fits into a single packet. 16-bit packet ID field is marked with the 16-bit hash of the 32-bit IP address of edge router. The victim has a record table consisting of HashMark and IngressAddressList so that the IP address for a corresponding hash is identified.

Lin and Lee [123] proposed a robust and scalable DPM scheme where multiple hash functions are used to reduce the probability of address digest collisions. 3 bits are used to distinguish the 8 different kinds of marks and the remaining 14 bits carry partial address information comprising of these marks. The scheme has been designed to send every bit of the IP address at least twice and allows tradeoff between false positive rate and false negative rate while considering packet loss due to congestion.

Jin and Yang [99] proposed a DPM based redundant decomposition (DPM-RD) for IP traceback where the marking field consists of only two sections, information and index. Every ingress edge router decomposes its corresponding IP address into several fragments with neighboring fragments having some redundant bits with each other. The IP ID field is marked with one of the fragments. Redundant decomposition makes the address decomposition more flexible, while decreasing false positives.

Xiang et al. [234] proposed a flexible DPM (FDPM) to find the real source of attacking packets. It adopts a flexible mark length strategy for compatibility to different network environments and it changes the marking rate according to the load of the participating router by a flexible flow-based marking scheme.

2.7.2 Autonomous System Based Traceback

Autonomous System (AS) is a connected group of one or more IP prefixes run by one or more network operators which has a single and clearly defined routing policy [83]. Each AS is identified with a globally unique AS Number (ASN) which is used in the exchange of exterior routing information. ASN is a 16-bit integer, assigned and managed by IANA [89].

Paruchuri et al. [162] proposed authenticated autonomous system traceback (AAST) to probabilistically mark packets with AS numbers. Two schemes, AS marking and Authenticated AS marking are presented. The mechanism needs 25 bits for marking and the TOS (8 bits), ID (16 bits) and unused fragment flag bit are used. Marking is done at AS Border Routers (ASBR), when a packet is forwarded to a router belonging to another AS. It uses 19 bits (16 bits for ASN and 3 bits for the AS_distance field). Authenticated marking assumes a symmetric key infrastructure in each AS. The 25 bit AS marking field is assigned a cipher text generated as E (ASN || RP, K_{AS}). RP is the 9 bit redundancy predicate set to a hash of source destination address pair.

Gao and Ansari [66] propose autonomous system based edge marking (ASEM) in which only the ingress edge routers of each AS mark packets with AS number according to certain probability. Packets are not remarked by all other routers. The 32-bit marking information consists of four parts, 16-bit AS_PATH storing the transformed ASPATH information, 1-bit FLAG indicating whether the packet has been marked, 3 bits recording the length of ASPATH, and 12-bit hash function of the IP address. Victim needs to receive only a few packets to reconstruct the attack path. ASPATH attribute provides an ordered list of ASes to be traversed, verifying the path.

Tupakula et al. [222] proposed DoSTRACK, that can efficiently deal with the TCP SYN and reflection Distributed Denial of Service (DDoS) attacks. The main aim of the scheme is to prevent the attack traffic at the ingress edge router that is nearest to the source of attack. The egress edge router that is connected to the victim network updates the victim's details (such as 16-bit hash value of the 32-bit IP address) to all other ingress routers within the AS/ISP domain. The egress router validates the traffic that is destined to the victim's network and marks the packet with the unique ID of the ingress router. The ingress edge routers apply ingress filtering on the traffic destined to the victim. Prevention of the attack traffic will be done until the ingress edge routers receive a reset signal from the egress edge router.

Korkmaz et al. [113] consider AS level deployment of log based IP traceback and propose AS-level single packet traceback (AS-SPT). It logs packet digests at the border routers of participating ASes and traces a given attack packet toward its origin at the AS-level. Each AS-SPT enabled AS maintains an AS traceback (AST) server that monitors the operation of border nodes logging the packets. When the traceback query arrives from victims, AST queries the border routers and sends the response back with collected information or forwards the query recursively to its preceding ASes in case the border routers have not logged the packets.

Castelucio et al. [35] propose an AS-level overlay network that operates on the border routers of an AS and builds an overlay network after exchanging information with BGP. The marking system inserts the routers data into the generalized bloom filter (GBF) of an IP packet. The `community attribute` in the `update messages` of BGP is used to group destinations that share the same common characteristics. Marking is done by an exclusive OR operation of 16-bit AS number with 8-bit TTL and 8 MSB of 0's.

2.7.3 Router and Interface Marking

Interface marking mechanisms consider a router interface as an atomic unit for traceback instead of the router itself. Chen et al. [38] proposed the router interface marking for IP traceback where a RIM enabled router probabilistically marks each packet with the identifier of one of the hardware interfaces that processed the packet. RIM uses a string composed of locally-unique router input IDs as a globally unique identifier of a path. RIM uses 5 bits for distance, 6 bits for XOR, and 6 bits for IID. A router probabilistically marks a packet by resetting the distance field to zero and copying the IID of the packet's incoming interface to both the IID and the XOR fields.

In [39] an improvement of the above technique for attack diagnosis (AD) which is a novel attack mitigation scheme, adopting a divide-and-conquer strategy, activated by the victim after the attack is detected. The victim instructs the upstream routers to mark the packets deterministically and can traceback one attack source. AD combines the concepts of Pushback and packet marking, Attack detection is performed near the victim host and packet filtering is executed close to the attack sources. By instructing its upstream routers to mark packets deterministically, the victim can trace back one attack source and command an AD-enabled router close to the source to filter the attack packets.

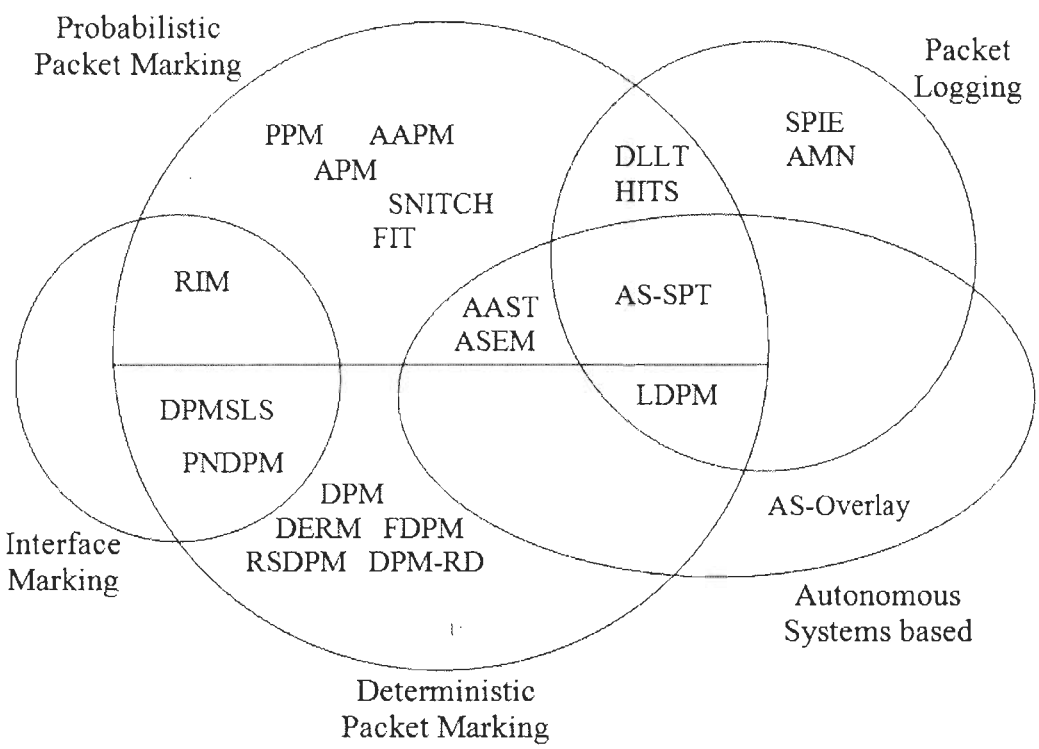

Fig. 2.4 Relation between various Traceback Mechanisms

Yi et al. [240] proposed DPM with link signature, which marks every packet passing through a router with link signature, which is the digest of the address information of the two adjacent nodes or a random 16-bit value. Each router will participate in marking deterministically and the mark will change. The entire path information is available in each packet and single packet IP traceback is possible.

Peng et al. [165] proposed an enhanced and authenticated DPM where path numbering is used for traceback. There are two types of routers DPM enabled and PNM enabled routers. DPM enabled routers are deployed at the edge of a subnet to mark each packet traversing them by the incoming interface. PNM enabled routers are closest to the source of the packet and mark each packet with the path identifiers representing the path linking them to the DPM enabled routers. The victim can not only detect and filter attacks, but can also obtain accurate information by the authenticated marks.

The relation between various traceback mechanisms can be seen in Figure 2.4.

2.7.4 Network Forensic Traceback

Mitropoulos et al. [140] surveyed various approaches for IP traceback and classified them so that the power of digital forensics may be enhanced and the limitations of classic incident handling and response capabilities may be countered. The focus is on their nature (host based, network based or both), behavior (proactive or reactive), architecture (centralized or distributed), applicability (local network, autonomous systems or the Internet) and complexity (number of reengineering functions to be performed).

Carrier and Shields [29] propose the Session TOken Protocol (STOP) to assist in the forensic investigation and traceback of a malicious host. The protocol is based on the Identification protocol (IDENT) and is aimed to automatically trace attackers logging through a series of stepping stones. STOP saves the user and application level data associated with a particular TCP connection and returns a random token. It also allows hosts that are not present in the connection chain to make requests on behalf of another host. STOP modifies the request message to provide more options and the response message to protect privacy. The request types allow tokens to be generated along the entire path of hosts. ID request type saves the user name and returns a random token. SV type saves the user name and also the data associated with the process. ID_REC and SV_REC are the recursive daemons which require a random session identifier.

Daniels [48] proposed a functional reference model using passive approach for tracing network traffic. The general reference model for passive origin identification defines the components in terms of their general behavior and goals. Passive approaches do not modify traffic, but they store observations for later analysis. The model has network monitors that communicate with analysis program through a reporting unit. The reporting unit provides the observation information and a control unit interprets commands from the analysis program. The analysis program can query the monitors for observations and correlate observations to determine the origin of network data elements (packets).

Demir et al. [51] propose two lightweight novel approaches, session based packet logging (SBL) and SYN based packet marking (SYNPM), for traceback by providing simple and effective logging. These techniques store log information for longer periods and respect the privacy of communications as well. SBL uses the SYN and FIN packets to record only the critical information (IP addresses and the duration of communication) over the logging period. The header information and the first four bytes of data payload of each logged packets are recorded. SYNPM enables the router to insert distinguishable identifiers in the first SYN packet whenever it routes it. The identifiers are special signatures of routers and are appended to the packets to record the router along the path.

Cohen [44] explores the problem of determining the real source behind the network address translation (NAT) gateway. The author presents a model for disentangling observed traffic into discrete sources and relies on correlation of a number of artifacts which allow the identification of sources. Author based the attribution model on *streams* defined as a set of packets with the same source and destination addresses and *sources* which are set of streams attributed to the same host. Assignment of streams to a particular source is handled in an optimized way using energy function. The energy function reduces when a correct assignment is made and increases otherwise. Energy function can be constructed based on attributable artifacts like IP IDs, HTTP referrers, cookies, etc.

A payload attribution system (PAS) is one of the core components in a network forensics system enabling investigation of cybercrimes on the Internet. Ponec et al. [168] proposed several new methods for payload attribution, which utilize Rabin fingerprinting, shingling, and winnowing. The accuracy of attribution increases with the length of the excerpt and the specificity of the query. The collected payload digests can be stored and queries performed by an untrusted party without disclosing any payload information. Guan and Zhang [76] explained the open problems in attack traceback and attribution.

2.8 Shortcomings and Research Gaps

The frameworks for network forensic analysis which have been proposed till date have been discussed in detail in the previous sections. The limitations associated with different phases in each framework have also been analyzed. The well researched phases have been recognized and the following phases with research gaps were identified:

Examination Phase:

- The useful network events are to be identified for detecting the attacks.
- The various protocol features being manipulated by the attackers need to be listed.
- Correlation of these attack features with possible attack scenarios must be performed.
- The attack must be identified and validated before making a decision to proceed with the investigation analysis.
- Effective mechanism is to be in place to identify attack features from packet captures.

Analysis Phase:

- Attack information and alerts must be taken from various security sensors as no single security tool can give comprehensive alert information.
- Information must be considered from various hosts from a compromised network for reconnaissance.
- Data fusion of these alerts and statistics must be performed to validate the attack.

Investigation Phase:

- The analysis of alerts, logs and network traffic must lead to the source of attacks.
- Suspicious source addresses can be determined in the analysis phase. However IP spoofing will hide the true information about the attacker.
- Traceback to the source of the attack using IP address is a major challenge.
- The investigation must enable the attribution of the attack to a host or a network.

2.9 Summary

Network forensics was defined formally and various types of network forensic systems were classified. A detailed survey on the process models proposed for digital forensics in general and for network forensics in particular were presented. An exhaustive survey of the various frameworks proposed for network forensic analysis was made and were categorized into various classes – distributed, soft computing, honeypot based, aggregation nd heterogenous. A detailed study of various data fusion techniques for validating attacks in the field of security and specifically intrusion detection. An extensive study of IP traceback mechanisms is presented, while identifying the most suitable techniques for network forensics. The research gaps existing in the frameworks have also been identified phase wise.

Chapter 3

Proposed Framework for Network Forensic Analysis

3.1 Introduction

Many models for digital forensics were proposed after the first Digital Forensics Research Workshop in 2001. The first model for digital forensics in networked environments (network forensics) was also proposed. Many of these models were described in section 2.3. We use the term 'model' to imply a theoretical representation of phases involved in network forensics and the term 'framework' to mean prototype implementation. There have been many frameworks which have been built based on the phases in the process models. The frameworks were exhaustively surveyed in section 2.4.

We have proposed a comprehensive process model for network forensics after considering all the models proposed till date and we have accommodated all phases which are essential for network forensics. We have also proposed a framework for network forensic analysis after identifying the well researched phases and phases which need research emphasis. Our framework handles the crucial phases of the process model, namely: examination, analysis and investigation.

3.2 Proposed Generic Process Model

We propose a generic process model for network forensic analysis based on various existing digital forensics models discussed in section 2.3. We formalize a methodology specifically for network traffic based investigation as shown in Figure 3.1. The proposed model is generic as it handles both the real-time and post attack scenarios. We use the term 'process model' to refer to the theoretical representations of phases. The model has 9 phases – preparation, detection, incident response, collection preservation, examination, analysis, investigation and presentation. The first five phases handle real-time network traffic. The next four phases are common for real-time and post attack scenarios.

The NFATs – NetIntercept, NetWitness, NetDetector, Iris, Infinistream, Solera DS 5150, OmniPeek, SilentRunner, NetworkMiner, and Xplico work in all the other phases, except a few which are not applicable to preservation and investigation phases. PyFlag does not involve packet capture and starts with the examination phase. The various phases in the model are explained below:

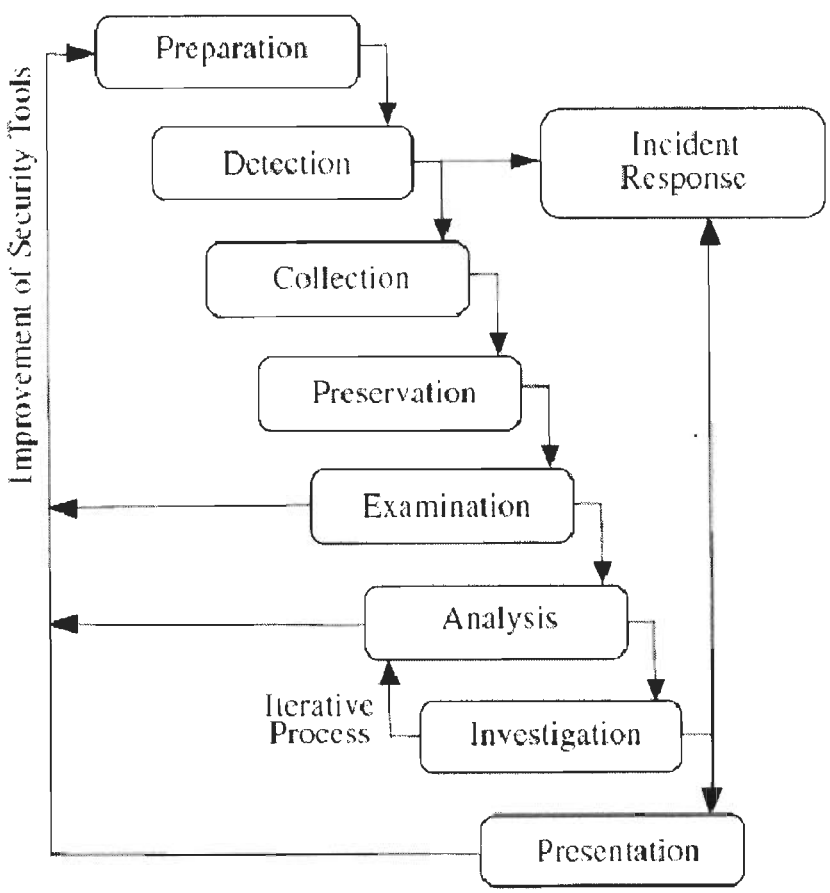

Figure 3.1. Proposed Generic Process Model for Network Forensics

3.2.1 Preparation

Network forensics is applicable only to environments where network security tools (sensors) like packet capture tools, intrusion detection systems, packet analyzers, sniffers, firewalls, traffic flow measurement software are deployed at various strategic points on the network. The staff handling these tools must be trained to ensure that maximum and quality evidence may be collected in order to facilitate attribution of the crime. The required authorizations to monitor the network traffic are obtained and a well defined security policy is in place so that privacy of individuals and the organization is not violated. Readiness of an organization strengthens the security policy and reduces the overall cost for every incident.

3.2.2 Detection

The alerts generated by various security tools, indicating a security breach or policy violation, are observed. Unauthorized events and anomalies noticed will be analyzed. The presence and nature of the attack are determined from various parameters. A quick validation is done to assess and confirm the suspected attack. This will facilitate the important decision whether to continue investigation or ignore the alert as a false alarm. Precaution should be taken in order that the evidence is not altered in the process. This phase has many practices common with network security. It branches in two directions – incident response and collection.

3.2.3 Incident Response

The response to crime or intrusion detected is initiated based on the information gathered to validate and assess the incident. The response initiated depends on the type of attack identified and is guided by organization policy, legal and business constraints. An action plan on how to defend future attacks and recover from the existing damage is initiated. At the same time, the decision whether to continue the investigation and gather more information is also taken. A similar response is to be initiated after the investigation phase (discussed below) where the information obtained may require certain actions to control and mitigate the attack. An important criteria to respond to an incident while performing network forensic analysis is to ensure that data being collected as evidence is neither tampered nor obstructed.

3.2.4 Collection

Data are acquired from the sensors used to collect the traffic data. The sensors used must be secure, fault tolerant, have limited access and must be able to avoid compromise. A well defined procedure using reliable hardware and software tools, must be in place to gather maximum evidence causing minimum impact to the victim. The network must be monitored to identify future attacks. The integrity of data logged and network events recorded must be ensured. This phase is very significant as the traffic data change 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 the system must be able to handle different log data formats appropriately.

3.2.5 Preservation

The original data obtained in the form of traces and logs are stored on a backup device like read only media. A hash of all the trace data is preserved. Standard procedures are used to ensure accuracy and reliability of the preserved data. Chain of custody is strictly enforced so that there is no unauthorized use or tampering. A copy of the data will be analyzed and the original network traffic data are untouched. This is done to facilitate legal requirements which may expect that the results obtained by the investigation are proved same when the process is repeated on the original data.

3.2.6 Examination

The traces obtained from various security sensors are integrated and fused to form one large data set on which analysis can be performed. There are some issues like redundant information and overlapping time zones which need appropriation. There may be cases where alerts from various sources are contradictory. However this process needs to be done so that crucial information from important sources is not lost. Data hidden or camouflaged by the attacker needs to be recovered. 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. The evidence collected is searched methodically to extract specific indicators of the crime. Minimum attack attributes are identified so that the least information recorded contains the highest probable evidence. A feedback is given to improve the security tools.

3.2.7 Analysis

The indicators are classified and correlated to deduce important observations using the existing attack patterns. Statistical, soft computing and data mining approaches are used to search the data and match attack patterns. Statistical techniques are used to validate the occurrence of an attack if the threshold value is exceeded. The combination rule of Dempster Shafer (DS) theory of evidence is used to fuse the information collected from various tools. Soft computing techniques like ANN, Fuzzy and Genetic Algorithm (GA) for classifying the attack. Data mining techniques help in clustering and classification of traffic data.

Some of the important parameters are related to network connection establishment, DNS queries, packet fragmentation, protocol and operating system fingerprinting. The attack patterns are put together, reconstructed and replayed to understand the intention and methodology of the attacker. A feedback is given to improve the security tools. The result of this phase is the validation of the suspicious activity.

3.2.8 Investigation

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 two simple strategies of the attacker to hide himself, IP spoofing and stepping stone attack, are still open problems. 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. The approach of the investigation depends on the type of attack. The investigation phase provides data for incident response and prosecution of the attacker. Attribution is establishing the identity of the attacker and is the most difficult part of the network forensic process. Many of the NFATs do not have an inbuilt traceback mechanism.

3.2.9 Presentation

The observations are presented in an understandable language for legal personnel while providing explanation of the various procedures used to arrive at the conclusion. The systematic documentation is also included to meet the legal requirements. The conclusions are also presented using visualization so that they can be easily grasped. This process concludes the network forensic analysis as the information presented results in the prosecution of the attacker. The statistical data is interpreted in support of the conclusions arrived. A thorough review of the incident is done and counter measures are recommended to prevent similar incidents in future. The entire case is documented to influence future investigations and to provide feedback to guide the deployment and improvement of security products.

The proposed model is generic as it handles network forensics both in real-time and post attack scenarios. The first five phases (including incident response) handle real-time network traffic. The preparation phase ensures the monitoring tools are in place. Detection phase helps in attack discovery and collection phase captures network packets ensuring integrity of data. A suitable incident response is generated based on the nature of attacks. A hash of the data is created and a copy is made in the preservation phase. The next four phases are common for real-time and post attack scenarios.

The post attack investigation begins at the examination phase, where a copy of the packet capture (libpcap) file is given for investigation. The examination phase identifies the attack using features of the protocols used. The attack indicators are correlated. The analysis phase fuses inputs from various security sensors and validates the attack. It also classifies attack patterns using data mining, soft computing or statistical approaches. The investigation phase involves traceback and attribution. The final presentation phase results in the prosecution of the attacker.

The proposed model is the first comprehensive model on network forensics covering most of the phases needed for analyzing network traffic. It is built on the well researched phases of computer forensics. The generic process model brings out the clear distinction of network forensics and other forms of digital forensics.

The proposed model is compared with the related process models as discussed in section 2.3. The observations are illustrated in Table 3.1 given below:

Table 3.1. Comparison of Proposed Generic Process Model with Related models

Pilli, Joshi & Niyogi 2009	DFRWS 2001	Reith, Carr & Gunsch 2002	Prosise & Mandia 2003	Casey & Palmer 2003	Carrier & Spafford 2003	Ciardh-uáin 2004	Baryamu-reeba & Tushabe 2004	Beebe & Clarke 2006	Ren & Jin 2006
Prepara-tion & Authoriz-ation	---	Prepara-tion	Pre-incident Prepara-tion	---	Readiness, Authoriz-ation	Aware-ness, Authoriza-tion, Planning	Readiness, Authoriza-tion, Confirma-tion	Prepara-tion	---
Detection	Identifica-tion	Identifica-tion	Detection of incident	Incident Alerts, Assess-ment	Detection, Notifica-tion	Notifica-tion	Detection,	Incident Response	---
Incident Response	---	Approach Strategy	Initial Response, Response Strategy	---	---	---	---	Incident Response	---
Collection	Collection	Collection	Investiga-tion (Data Collection)	Crime Scene Protocol, Identifica-tion & Seizure	Survey	Search & Identifica-tion, Collection	Submiss-ion	Data Collection	Capture
Preserva-tion	Preserva-tion	Preserva-tion	---	Preserva-tion	Preserva-tion	Transport, Storage	Preserva-tion	---	Copy, Transfer

Table 3.1. Comparison of Proposed Generic Process Model with Related models (continued)

Pilli, Joshi & Niyogi 2009	DFRWS 2001	Reith, Carr & Gunsch 2002	Prosise & Mandia 2003	Casey & Palmer 2003	Carrier & Spafford 2003	Ciardh-uáin 2004	Baryamu-reeba & Tushabe 2004	Beebe & Clarke 2006	Ren & Jin 2006
Examina-tion	Examina-tion	Examina-tion	---	Recovery, Harvesting Reduction, Organiza-tion & Search	Search & Collection	Examina-tion	Survey	---	---
Analysis	Analysis	Analysis	Investiga-tion (Forensic Analysis)	Analysis	Reconst-ruction	Hypothesis	Search & Collection	Data Analysis	Analysis
Investiga-tion	---	---	---	---	---	---	Traceback (Investiga-tion) Reconst-ruction	---	Investiga-tion
Presenta-tion & Review	Presenta-tion, Decision	Presenta-tion, Returning Evidence	Reporting, Resolution	Reporting, Persuasion & Testimony	Presenta-tion, Review	Presenta-tion, Proof of Defense, Dissemina-tion	Communi-cation Review	Presenta-tion of Findings, Incident Closure	Presenta-tion

3.3 Researched Phases in the Process Model

Our proposed generic process model has nine phases and many of the phases are well researched and studied. Many frameworks have been proposed for these phases and many of the prototype implementations are accepted as standards. We present some of the latest works which were proposed and case studies by many researchers on these phases:

3.3.1 Preparation

Eoghan Casey [32] explained that a moderate amount of forensic preparation in an organization can mitigate the impact of a major incident and can enable the organization to obtain restitution. Forensic readiness reduces the per incident costs and improves an organization's overall security posture when used in conjunction with an information security program. It integrates proper evidence handling mechanisms into an organization's incident handling capabilities.

Johnston and Reust [106] suggested that the increasing costs and penalties associated with exposure of sensitive data can be mitigated through forensic preparation and the ability to employ digital forensics. The proper training of personnel, who handle the intrusions, is critical to ensuring that anomalies indicative of compromises are detected even when standard incident response toolkits are undermined by modern rootkits. A proper understanding of an investigation's scope allows for the construction of targeted questions that need to be addressed. The correct tools need to be used to obtain the most complete and accurate data. Proper procedures for documenting the integrity and authenticity of the evidence must be followed.

Endicott-Popovsky et al. [56] proposed a theoretical framework for organizational network forensic readiness. Network forensic readiness (NFR) is defined as "maximizing the ability of an environment to collect credible digital evidence while minimizing the cost of an incident response". The authors proposed the Network Forensics Development Life Cycle (NFDLC) which includes the following phases:

- Initiation Phase which includes steps to determine what assets on the network would warrant digital forensic protection.

- Acquisition / Development Phase which ensures that any device or procedure collecting forensic data on the system will do so in a manner compliant with courtroom standards. Analysts find previously published checklists useful to determine what existing forensic procedures, tools and technologies could be embedded, building on prior research.
- Implementation Phase which involves calibration tests are recommended to verify the performance of devices used to collect evidence and to document the performance of the network itself. This would be accomplished by base lining the network by systematic analysis of a network, point to point, for dataflow, communication sequencing, performance statistics, etc.
- Operation / Maintenance Phase which ensures that audits would be performed at regular intervals, and as the network grows and changes, to confirm results of previous baseline and verification / calibration tests.
- Disposition Phase handles the chain of custody procedures to be incorporated to ensure preservation of the value of potential evidence residing in a retired system.

Rowlingson [189] proposed a ten step process for an organization to implement forensic readiness:

1. Define the business scenarios that require digital evidence.
2. Identify available sources and different types of potential evidence.
3. Determine the evidence collection requirement.
4. Establish a capability for securely gathering legally admissible evidence.
5. Establish a policy for secure storage and handling of potential evidence.
6. Ensure monitoring is targeted to detect and deter major incidents.
7. Specify circumstances when escalation to a full formal investigation (which may use the digital evidence) should be launched.
8. Train staff in incident awareness, so that all those involved understand their role in the digital evidence process and the legal sensitivities of evidence.
9. Document an evidence-based case describing the incident and its impact.
10. Ensure legal review to facilitate action in response to the incident.

3.3.2 Detection

Detecting the occurrence of an attack is the goal of the established field of network security. Countermeasures exist for many of the attacks which take place and regularly keep re-occurring. Many security tools like IDS, intrusion prevention systems, firewalls, network monitoring systems (IPS), network statistic tools exist in commercial and open source varieties. These tools can detect most of the current attacks.

Attackers are increasing sophistication in their attacks by using blended attacks. They are combining many attack vectors in a single attack. The existing security tools may not be able to detect the new attack but raise alerts for the independent attack vectors. Attackers also use zero day vulnerabilities to launch novel attacks which circumvent the security tools. However the attackers follow certain procedures which are common to many attackers. The security tools may not detect the zero day attacks but can definitely detect the anomalous behavior of the attacker by examining the common features with previous attacks.

Jayashree and Easwarakumar [98] proposed a complete solution to anomaly detection of intrusions in the network. A two stage detection process is assumed where the IDS performs the common pattern detection using signatures in the first stage followed by an anomaly detector in the second stage. It comprises of protocol decoder, misuse detector and anomaly detector, based on traffic pattern analysis. They build the solution taking into account the parameters of detection rate, false alarm rate, ease of deployment and infrastructure adaptation.

Jain et al. [97] proposed an intelligent real-time reactive network management system to harness the strengths of existing tools. As no one tool generally detects all the attacks in a network, the administrator launches various tools manually and takes manual decisions based on the alert information / attack data collected. Automating the process makes the real-time reaction to different anomalies occurring concurrently in the network possible. The proposed system aims to minimize the resource consumption by using a distributed system and a sophisticated layered analysis of the events generated. The decision making at each step of the protocol is governed by the set of predefined 'rules' by taking the collected data into consideration. The set of predefined rules can be compared with intrusion signatures. The tools are launched at the relevant nodes only when required.

3.3.3 Collection

Capturing and storing data packets from networks consume a lot of CPU power and storage capacity resources. Cheng and Chen [40] emphasized the development of a network forensic control mechanism which can dynamically adjust the amount of data to be collected on an evidence flow according to the storage capacity level on the storage subsystem. Their solution is able to select an appropriate full collection (FC) and selective collection (SC) margins to minimize data loss associated with storage subsystem saturation while preserving reasonable acceptance ratio of new forensic collection requests.

They proposed a network forensic evidence collection architecture which consists of the *Storage Subsystem* (SS) and the *Evidence Capture Subsystem* (ECS). There are three keys components inside the ECS: Packet Capturer, Filter/Classifier and Evidence Collection Agent (ECA). They also proposed two evidence collection models, *Non-Quality Adaptive* (Non-QA) model and *Quality Adaptive* (QA) model, with preferential treatments for network forensics. They are modeled as the *Continuous Time Markov Chain* (CTMC). The sophisticated commercial optimization tool, LINGO, is then applied to solve the model.

Tae-Kyou and Ilkyeun [212] proposed a forensic logging system that collects fine-grained evidence from target servers and networks. They developed a TCSEC-B1 level secure operating system and a dedicated network processor that collects network traffic. The forensic logging system has three principal technical goals:

- Monitor server activities at the kernel of the target server OS as well as at the network packet level of the target network
- The detail users' activities information gathered from the target server and the network processor should be transmitted immediately to a separate forensic server (log machine) on which a secure OS has been installed to ensure robust access control.
- Third, a database should be available to respond to high-level queries of the log files stored on the forensic server.

The logging system is also capable of protecting servers from malicious attacks as well as allowing security managers to obtain forensic evidences when the target server is assaulted by violations.

Meyler and Sutherland [138] proposed flexible and open source software architecture for real-time analysis of the Web and local area networks in order to identify and track images and other forms of illicit files or malware. The architecture seeks to achieve three main goals: open source method of identifying and tracking files across local area networks and across the World Wide Web, for the purpose of gathering intelligence and forensic evidence and standard means of extending the functionality of the tool architecture.

Broadway et al. [24] outlined a framework, Highly Extensible Network Packet Analysis (HENPA), which takes the output of a packet sniffer and processes the data to extract potential forensic evidence. It aims to reduce the amount of manual analysis required on the part of the investigator in criminal investigations by facilitating the automation of the packet analysis process and putting a large emphasis on producing high-level output that can easily be understood by the end user. HENPA has a forensic focus, is extensible, shares information and requires limited technical skills. It is an object-oriented framework and has four main subsystems: Core, GUI, Storage and Parsers.

An efficient storage infrastructure is needed for providing both high insertion rates and fast data access as there is a need for network security monitoring systems to store and examine very large amounts of historical network flow data. Giura and Memon [72] proposed a new column oriented storage infrastructure for network flow records, called *NetStore*. NetStore is aware of network data semantics and access patterns, and benefits from the simple column oriented layout without the need to meet general purpose RDBMS requirements. Experiments show that NetStore can provide more than ten times faster query response compared to other storage systems while maintaining much smaller storage size.

3.3.4 Preservation

Davis et al. [49] described a network based storage architecture that helps address the issues of maintaining the integrity of the evidence and storing digital evidence for extended periods of time. The Digital Evidence Custodian (DEC) is a network-based solution for storing and handling large quantities of digital evidence. Its architecture supports collaborative efforts by examiners and investigators located at geographically dispersed sites.

The design is intended to streamline digital forensic investigations and support the collaborative analysis of digital evidence at multiple locations.

Hunt and Slay [88] designed the real-time forensically sound, secure traffic monitoring, logging and alert system to achieve the goals of providing network status and generation of alerts, honeypot and traceback systems to understand the attacker's dynamics, adjustment of granularity and logging data in a forensically sound manner, appropriate reporting to the system administrator, real-time feedback where counter actions can be performed against attacks, forensically sound safe keeping of traffic and log data over the period of interest and a comprehensive tool set for real-time and after-the-event analysis.

3.3.5 Presentation

Digital investigators are responsible for ensuring that evidence is reliable enough to be admissible in court or to be useful in corporate disciplinary or termination proceedings. Solon

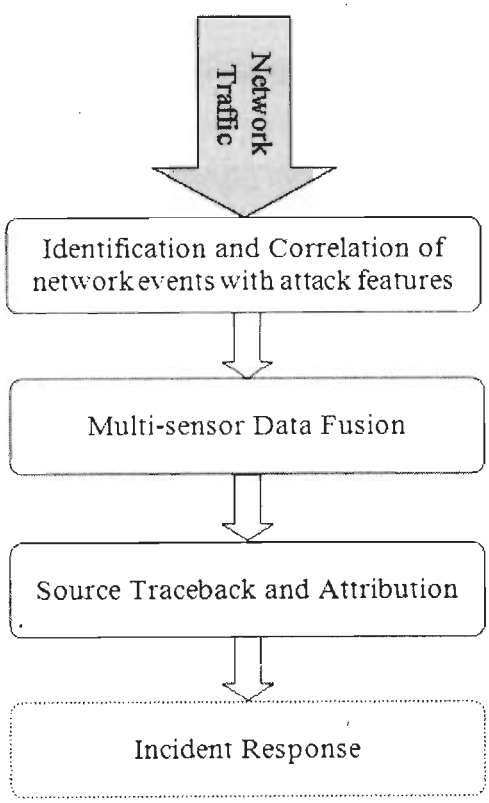

Figure 3.2. Proposed Framework for Network Forensic Analysis

and Harper [207] gave some basic guidelines to make sure the evidence is protected and that notes made at the time are professional. The evidence must also be presented in a comprehensible fashion to be useful. They also suggest guidelines for report writing and give a sample report structure.

3.4 Framework for Network Forensic Analysis

We propose a novel framework for network forensic analysis which will capture network traffic data, perform fusion of alerts and attack information, classify, correlate, and analyze this data in order to investigate the source of attack. We have discussed the well researched phases like preparation, collection, detection, preservation, and presentation in the previous section. Standard techniques have been developed which are well tested by time. The remaining phases in our proposed generic process model, namely examination, analysis, investigation and incidence response, need to be addressed. Our proposed framework addresses three objectives as shown in Figure 3.2. The mapping between the objectives of our framework for network forensic analysis and phases in the generic process model is illustrated in Figure 3.3.

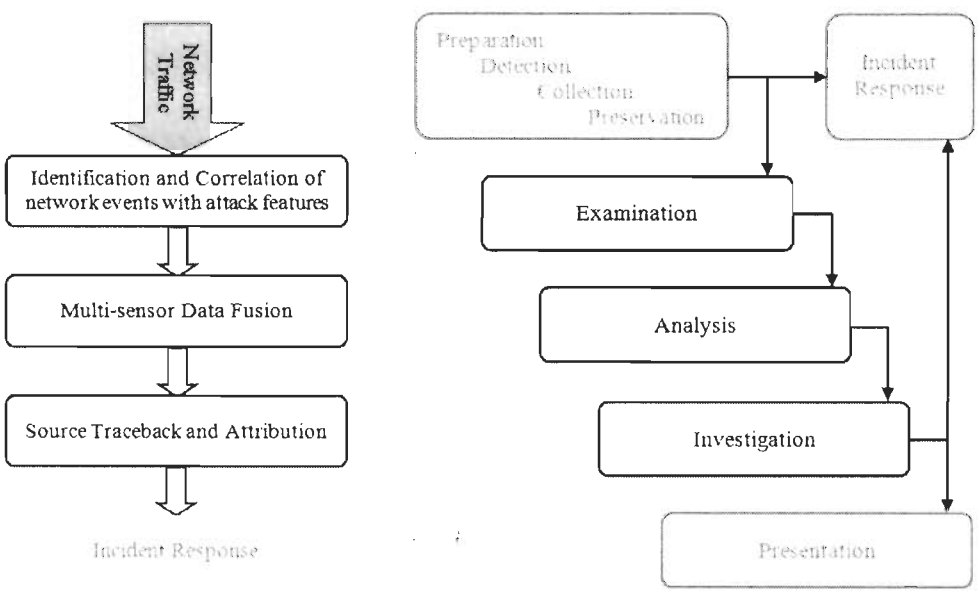

Figure 3.3. Mapping between objectives in the Framework and phases in the Generic Process Model

The objective of identifying and correlating network events with attack features in the framework corresponds to the examination phase in the generic process model. The multi sensor data fusion objective corresponds to analysis phase and the source traceback and attribution relates to the investigation phase.

Incident response naturally works on determining the vulnerability exploited to compromise the system and enforcing protection against exploitation of the same on other systems. Incident response also includes a strategy regarding containment and eradication of, and recovery from the attack. Apparently the main focus of incident response would be to arrest the attack traffic from reaching the victim system and to take the compromised system offline, to avoid further infection. This objective will work against network forensics as it will hinder the collection of attack traffic as evidence.

The phase of incident response in network forensics, involves an important criteria to respond to an incident while ensuring that data being collected as evidence is neither tampered nor obstructed. The various parameters to be controlled and the tradeoffs are to be determined so that evidence is collected and collateral damage minimized. An action plan on how to defend future attacks can be prepared after the above process is completed. Our future work will involve a prototype model for incident response.

The prototype implementation of our proposed framework involves specific independent models designed for each objective. Our framework handles post-mortem network forensics and starts with the packet capture file [95] comprising of network packets as shown in Figure 3.4. We use Net::pcap [218] module of the Perl language to open the file and read the contents of the file. Net::Pcap can encode and extract various protocol features indicating each field with a self explanatory attribute name. The contents of the packet capture file are copied into a database. The protocol attributes usually manipulated in the header and payloads are examined in each packet and the type and nature of the attacks are detected.

If an attack is detected, we validate it by data fusion using Dempster-Shafer (DS) theory of evidence. We fuse the alert information from various security tools and confirm the attack. We proceed with the investigation with the alert information and suspicious IP addresses. We use two deterministic packet marking approaches for traceback and attack attribution. The various objectives of the framework are explained in detail:

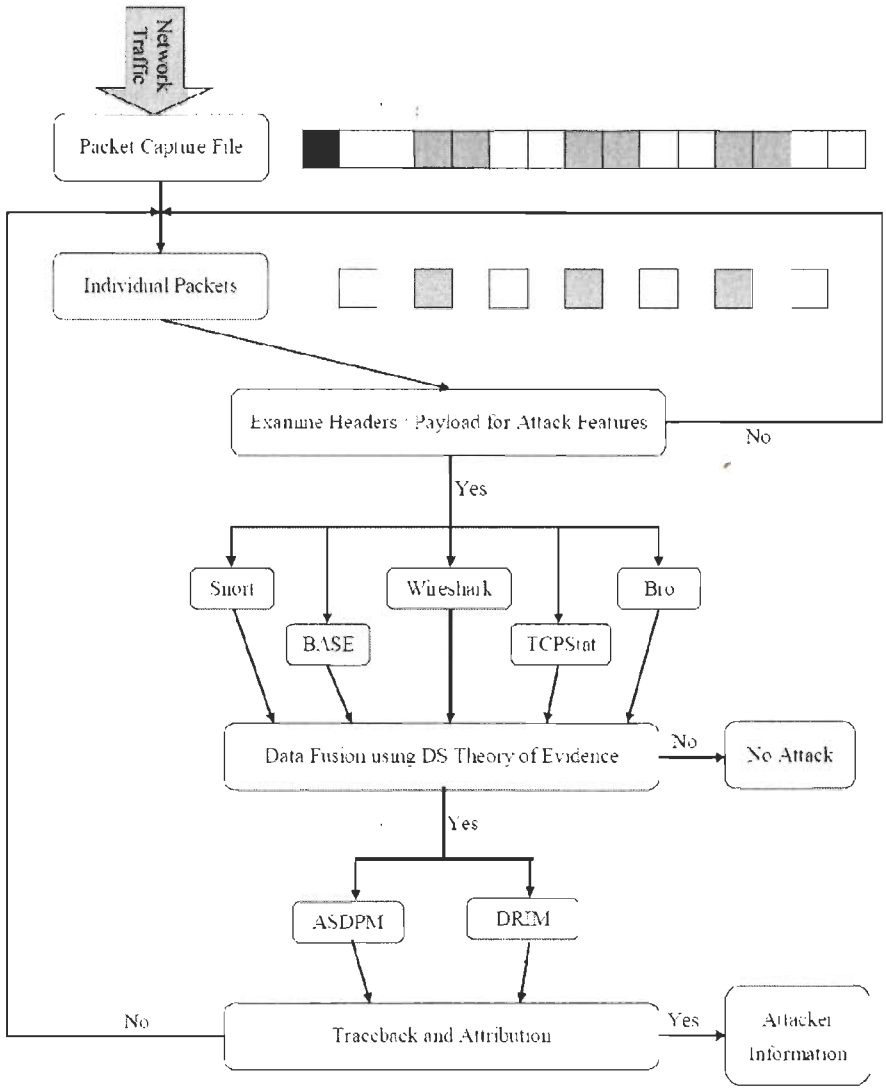

Figure 3.4. Flow Diagram for the Network Forensic Analysis Framework

3.4.1 Identification and Correlation

Important network events are identified at the application, transport and network layer of the TCP/IP protocol stack and are correlated with attack features manipulating the HTTP, TCP, UDP, ICMP and IP protocol header fields. The packet capture files are parsed using Perl language module pcap::net. The header information of each packet in the capture file is read and various fields which are manipulated for attacks are examined. The events specific to distributed denial of service (DDoS) attacks, port scan attacks and Cross-site Scripting (XSS) attacks are identified and correlated as a case study in our work.

The strength of this phase is that the network events provide information about the attempts made in compromising the system and helps in attack reconstruction. Information of these events from various hosts will help in reconnaissance of the attack. Identifying important sessions of suspicious activity will reduce the data to be analyzed. The correlation of events will validate the occurrence of the malicious incident and guide the decision to proceed with the investigation.

3.4.2 Multi-sensor Data Fusion

The attacks are identified in the previous phase and are analyzed by performing data fusion of information from multiple security sensors. Security sensors with complementary and contradictory functions are used. Security tools with similarity build the redundancy and reliability of attack information. Diversity among the tools will ensure versatility. Data fusion is performed on the alert and attack information generated by these sensors so that the attack evidence is more accurate. D-S Theory of evidence [217] is used to perform fusion of the alert information to ascertain the validity of the attack occurrence.

Open source tools which are commonly available on the network to gather attack information. IDS's (*Snort* [188] and *Bro [163]*), packet capture and analysis tools (*tcpdump [96]*) or sniffers (*wireshark [45]*), traffic statistic tools (*tcpstat [84]*) and security analysis console (*BASE [105]*) are used to generate alerts, attack statistics and information. The alert information from these five tools is fused, using the combination rule of the D-S Theory of evidence to validate the attacks. The suspicious addresses and alert information is passed to the investigation phase.

3.4.3 Traceback and Attribution

The results of the data fusion of attack and alert information will lead to identifying suspicious network by its IP address. IP Traceback is a method for reliably determining the origin of a packet on the Internet. Techniques based on packet marking, packet logging or hybrid approaches can be used. The attack attribution can be done by analyzing the data packets transmitted, applications being run, traffic patterns observed and protocols violated.

Deterministic packet marking is more suitable for network forensics as many attack packet streams may consist of only few packets and investigation needs to be performed using the limited evidence. The basic idea is to record the access point as close as possible to the attacker. This information is obtained from sources which the attacker cannot manipulate. No other router modifies this marked information. A single packet is sufficient to detect the attack source as each packet carries the mark to traceback to the attack source. Routers in the attacker's network participate and all other routers may not participate in traceback.

Two novel approaches for network forensic traceback as part of investigation phase are proposed – Autonomous System based Deterministic Packet Marking (ASDPM) and Deterministic Router and Interface Marking (DRIM). ASDPM involves deterministic marking of each packet with the hash of the IP address of the first internal router and AS Number (ASN) of the AS boundary router when it is leaving the source AS. The DRIM approach involves deterministic marking of each packet with the hash value of the address of the first ingress router and the number of the interface through which the packet reached it.

3.5 Summary

A generic process model for network forensics, which handles both real time and post-mortem network forensics, has been proposed. The phases which have been well researched and studied have been identified, namely Preparation, Detection, Collection, Preservation and Presentation. Prototype implementations have been proposed and the techniques have been standardized for these phases. A framework for network forensic analysis, which handles the research challenges in the remaining phases of Examination, Analysis and Investigation, was proposed. Future work will involve a prototype model for Incident Response phase.

Chapter 4

Identification and Correlation of Network Events

4.1. Introduction

Network forensics deals with the analysis of the trace and log data of network intrusions captured by the existing network security products and provides useful information to characterize intrusion or misbehavior features. The collected data acts as evidence for incident response and investigation of the crime. Network forensics does not block the network crimes but collects enough evidence of the crimes. The monitoring and analysis of data from live systems and networks will become essential to law enforcement as caseloads increase and judicial boundaries blur. Network criminals will be punished for their illegal actions thereby providing a deterrent to online crime [213]. The power of various network security and forensic analysis tools [167] available as open source can be integrated so that the investigator can have an edge over the attacker.

The challenge of network forensics system is to identify useful network events and choose a minimum representative set that would potentially be evidence in a variety of cybercrimes [142, 202] . We have identified the various attributes which are being misused at the network and transport layer of the protocol. The network features being manipulated by the attackers are correlated with a particular type of attack. These key attributes of the protocol are extracted and copied into a database. The statistical thresholds are calculated from these attributes for various attacks [27].

We propose that this least amount of information comprises of the highest probable evidence. It contains attack attributes indicating suspected attacks. This information is used to extract suspicious and evidence packets from the captured trace and log files. The attack packets are converted back into the same format enabling analysis using existing open source tools, harnessing their strengths. The idea is not to create another tool but to effect data reduction of packet capture files for efficient and faster analysis.

4.2. Packet Capture Format

Network security and monitoring tools are not designed to handle forensic investigations. In order to achieve this, it is required to capture the entire data packets and analyze them in detail. Packets can be captured in libpcap (.pcap) files by running a packet sniffer like tcpdump. These captures can help in order to understand who, what, when, where and how network traffic is flowing.

Libpcap [94] is the very basic file format used to save captured network data. The file extension is .pcap. The file has a global header containing some global information followed by zero or more records for each captured packet as shown in Figure 4.1. The captured packet in a libpcap file does not contain all the data in the packet as it appeared on the network. It contains at most the first N bytes of each packet. The value of N is called the 'snapshot length'. N will be a value larger than the largest possible packet to ensure that no packet in the capture is sliced, with a typical value of 65535.

The global header is placed first in the file with fields indicating the file format, byte ordering and the *version* number. It specifies the correction time in seconds between GMT and the local time zone and the accuracy of time stamps in the capture. The packet capture length N is specified by the field *snaplen*. The type of data link layer is also mentioned.

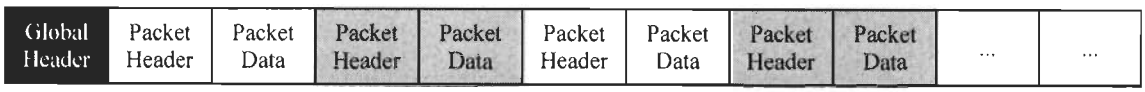

Figure 4.1. Libpcap File Format

The global header is followed by a sequence of packet headers and packet data. The packet header has information fields, *ts_sec* which gives the date and time when this packet was captured, *ts_usec*, the microseconds offset to *ts_sec* when the packet was captured, *incl_len* the number of bytes of packet data actually captured and saved in the file and *orig_len* field gives the length of the packet as it appeared on the network. The actual packet data will immediately follow the packet header as a data blob of *incl_len* bytes without a specific byte alignment.

4.3. TCP/IP Protocol Suite

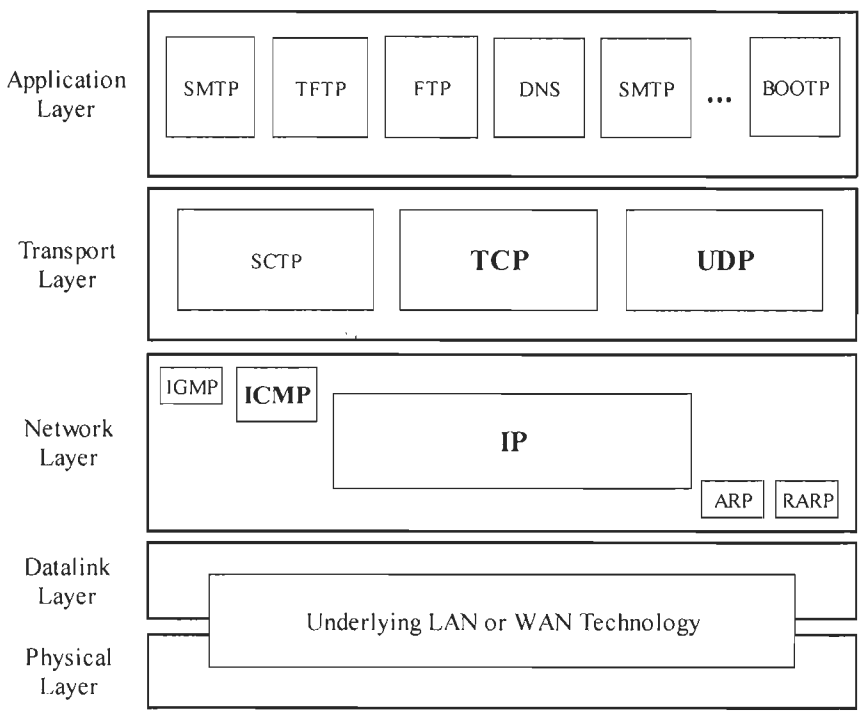

Figure 4.2. TCP/IP Protocol Suite

The TCP/IP protocol suite [118] was designed to provide a simple, efficient, open communication infrastructure in an academic and collaborative environment. The five layers of the protocol suite are shown in Figure 4.2 taken from [61]. Attackers use the

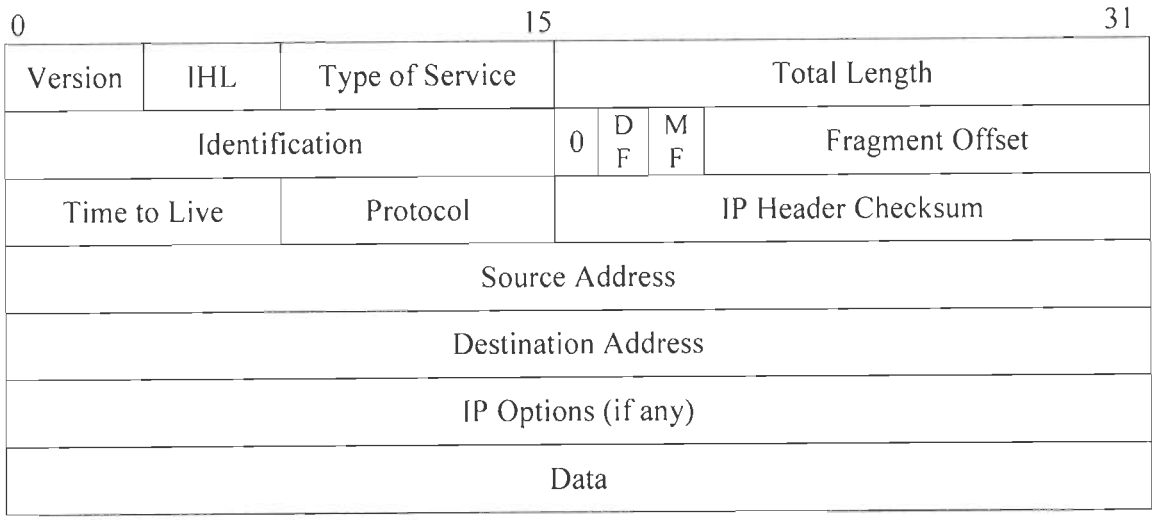

Figure 4.3. Internet Protocol Packet Structure

vulnerabilities in the implementation of the TCP/IP protocol stack and exploit them to launch attacks. Important protocols in each layer are discussed briefly in this section and the attacks corresponding to the protocols are discussed in the subsequent sections.

The Internet Protocol (IP) [170] operates at the network layer of the Internet and routes a packet to its destination. The packets go through a series of routers and at each router, the next hop for the packet is determined. It is possible that two packets from the same source going towards the same destination may take two different paths. The structure of Internet Protocol is shown in Figure 4.3.

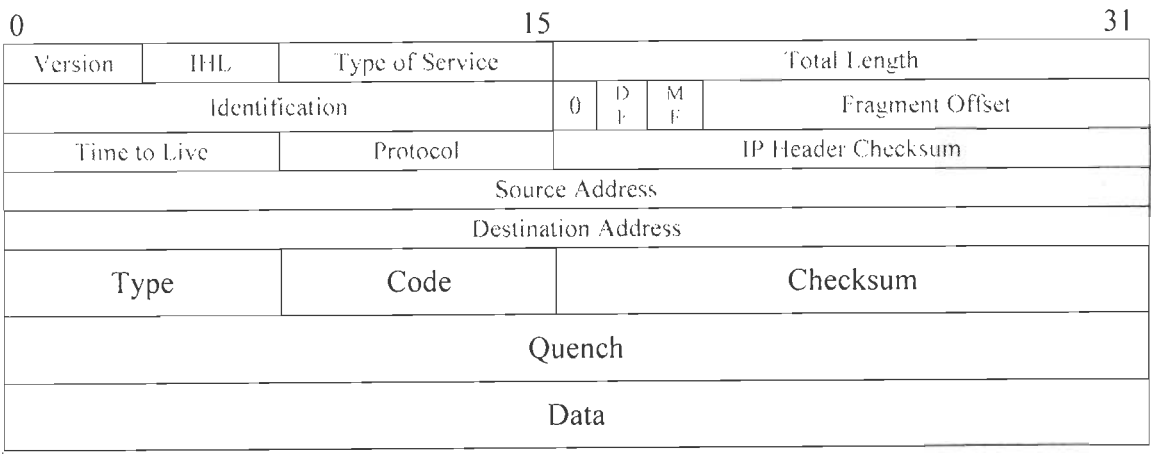

Figure 4.4. Internet Message Control Protocol Header

The Internet Control Message Protocol (ICMP) [171] facilitates sending one-way informational message to a host. The protocol header is graphically illustrated in Figure 4.4. ICMP is transported in the payload of the IP packet and has several data structures of its own. ICMP is used by a router or a destination host to inform the source host about errors in datagram processing.

ICMP allows routers to send error or control messages to other routers or hosts. It also provides communication between the two machines communicating at the network layer. The ICMP protocol is used for two types of operations - reporting non-transient error conditions and probing the network with request & reply messages. ICMP messages are therefore classified into two categories: ICMP Error Messages and ICMP Query Messages. Each ICMP message is assigned a number, known as the *message type* which specifies the type of message. Another number represents a *code* for the specified ICMP type.

Transmission Control Protocol (TCP) [172] runs on top of IP, and provides a connection oriented service between the source and the destination. TCP provides guaranteed delivery and ensures that the packets are delivered in sequence. It uses various mechanisms, such as sequence numbers, acknowledgments, 3-way handshakes and timers. The TCP structure is shown in Figure 4.5.

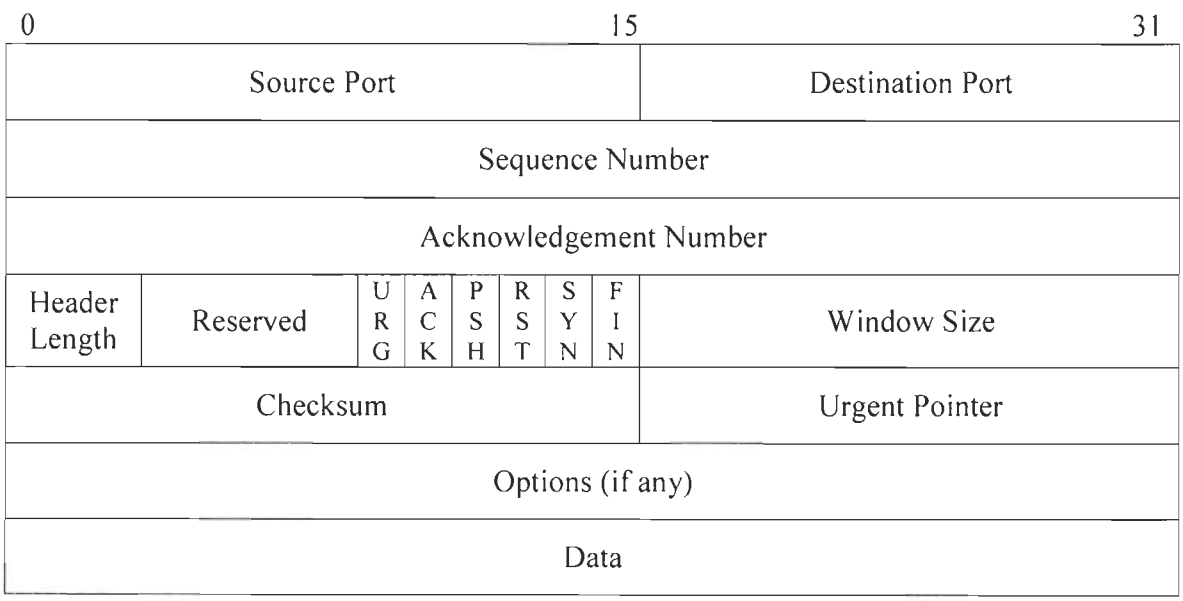

Figure 4.5. Transmission Control Protocol Packet Structure

Source Port	Destination Port
Length	Checksum
Data (optional)	

Figure 4.6. User Datagram Protocol Structure

User Datagram Protocol (UDP) [169] is basically an application interface to IP. It provides a mechanism for one application to send a datagram to another. The UDP layer is extremely thin and has low overheads, but it requires that the application takes responsibility for error recovery. The UDP structure is shown in Figure 4.6.

The Hypertext Transfer Protocol (HTTP) [60] is an application-level protocol for distributed, collaborative, hypermedia information systems. It is a generic, stateless, protocol which can be used for many tasks beyond its use for hypertext, such as name servers and distributed object management systems, through extension of its request methods, error codes and headers. It is mainly used for accessing data on the World Wide Web (WWW).

Data is transferred between Clients and Servers using HTTP messages. HTTP messages are read and interpreted by the HTTP server and HTTP client (browser). The format of request and response messages is similar. A request message consists of a request line, header, and a body. Response message has a status line instead of the request line. Request message has many methods for specific actions. A partial list of methods is given in Figure 4.7, taken from [61].

Attack	**Protocol Fields Examined**
GET	Requests a document from the server
POST	Sends some information from the client to the server
HEAD	Requests information about a document
PUT	Sends a document from the server to the client
TRACE	Echoes the incoming request

Figure 4.7. HTTP Methods

4.4. Identification and Correlation

The architecture for the identification and correlation of network events with attack features is shown in Figure 4.8.

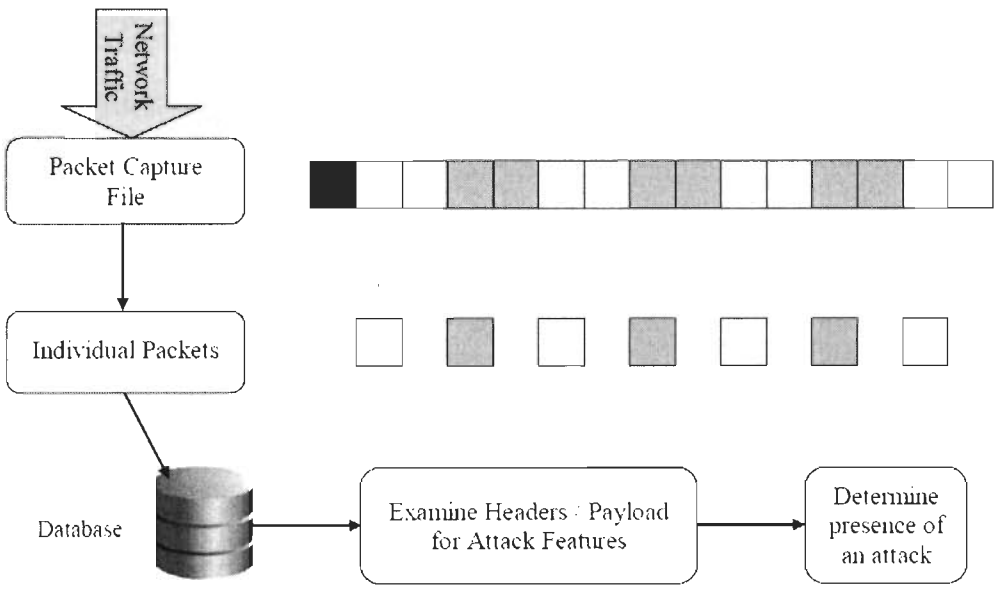

Figure 4.8 Identification and correlation of network events with attack features

The network traffic data is captured from the compromised system and transported securely to a forensic analysis server. The packet headers of the Libpcap contain information about the number of packets stored in the capture file and specify the capture length. The Perl language module Net::Pcap was used to extract the packet headers. The attribute *snaplen* gives the actual size of all packets stored.

The packet data portion of a libpcap file contains the information of various headers like Ethernet, IP, TCP, ICMP and UDP encapsulated in a recursive manner [130]. The protocol features are recursively extracted from the file and inserted in a database. Net::Pcap can encode and extract various protocol features indicating each field with a self explanatory attribute name. The IP packet features have attributes like ver, hlen, tos, len, id, flags, offset, ttl, proto, cksum, src_ip, dest_ip, and options. The TCP header has attributes src_port, dest_port, seqnum, acknum, hlen, reserved, flags, winsize, urg and toptions. The attributes associated with UDP are src_port, dest_port, len and cksum. Attributes like dest_mac,

src_mac, tos, id, flags, ttl, src_ip, dest_ip, and options are collected from the pseudo header attached to the UDP. This is a part of IP packet in which user datagram is encapsulated.

These attributes are ported to a database and the tables are created to hold packets specific to each protocol, i.e. TCP, UDP etc. The time stamp associated with each packet is also recorded and each packet is given an automatically generated frame number. The attacks cannot be classified based on a single packet information and can only be decided upon observing a sufficient number of packets. The tables created above are used to perform statistical analysis and calculating the thresholds for various attacks. These derived attribute values are calculated from basic attributes of a packet and help in attack detection queries as calculated in [154].

The IDS generates a list of suspicious addresses and the Net::Pcap module stores the protocol attributes in a database. The suspicious attack packets can be identified from the database using the alert information and placed in a separate table. Some of these packets may be legitimate traffic arising out of the suspicious IP address. The attack features and protocol attributes were correlated in the previous section. The statistical thresholds for various attacks were also determined. The suspicious packets can be filtered to result in evidence packets by running various queries on the database using the correlations and thresholds.

4.5. Events at the Network and Transport Layer

Security issues were not considered at the design level and though TCP/IP protocol suite is most used, it does not provide authentication, integrity, and privacy mechanisms [19]. Attackers use these vulnerabilities and exploit them to launch the attacks. There are many classifications of attacks which are categorized according to individual protocols [80, 81, 91, 103, 224].

We discuss two major types of attacks occurring at the Network and Transport layer. They are Distributed Denial of Service (DDoS) and Port Scan attacks. A brief description of some of the examples of these attacks are discussed in this section.

4.5.1.Distributed Denial of Service (DDoS) Attacks:

This attack uses the differences in implementation for resolving overlapping fragment offsets [42, 62]. The attacker modifies the fragment offset such that when the firewall assembles it, the malicious content gets hidden. However the packet becomes malicious when the victim reassembles it [102, 239]. The attacks are also evaluated in [87].

- **TearDrop:** The attacker exploits the weakness of IP packet reassembly process by purposely sending packets with overlapping fragment offset field.
- **SynDrop:** The attacker initiates many half connections with the victim by not completing the three way handshake protocol with ACK packet. The kernel maintains a buffer for such half connections, which eventually overflows causing system crash or DoS
- **Jolt:** The attacker sends very large, fragmented ICMP packets to a target machine. The ICMP packets are fragmented in such a way that the target machine is unable to reassemble them for use.
- **Ping of Death [70]:** While a single IP packet cannot exceed 65536-bytes, the attacker can make the fragments add up to more than this value. It is usually associated with ICMP, but can contain any protocol. *IceNewk* is an example.
- **Fraggle:** The attacker sends a large number of UDP echo (ping) traffic at spoofed source IP address of the victim. UDP echo packets are directed at the Unix UDP services echo (port 7), chargen (port 19), daytime (port13) and qotd (port 17).
- **Smurf [127]:** The attacker sends many ICMP echo request packets with spoofed source IP address of the victim. All replies to this broadcast are received by the victim, resulting in denial of service.
- **Bonk:** It is a variant of the teardrop attack and manipulates the fragment offset field in TCP/IP packets. Bonk attack manipulates this number and causes the target machine to reassemble a packet that is much too big to be reassembled and causes the target computer to crash.
- **Boink:** It is a modified version of the bonk attack, which allows UDP port ranges. It also manipulates the fragment offset field and causes the target computer to crash.
- **NewTear:** NewTear attack is simply a modified version of Teardrop which changes padding length and increases the UDP header length field to twice the size of the packet.

4.5.2.Port Scan Attacks

Port scanning [64, 225] is the process of identifying the listening ports of the victim system and is used to discover and map services that are listening on a specified port. Many of these methods use the packet fields in the TCP protocol. The various types of scans are as follows:

- **Connect Scan:** The attacker attempts to make a full connection with each port to determine whether it is open by issuing the connect system call. A failed connection indicates the port is closed. This is a slow scan, and might also turn up in the system logs.
- **SYN Scan:** SYN scan is relatively stealthy and is referred to as half-open scanning because attacker doesn't make a complete TCP connection. The attacker checks for open ports by sending SYN packets in succession to different ports. Open ports respond with a SYN-ACK, and closed ports with RST.
- **FIN Scan:** The attacker sets the FIN flag on to bypass some firewalls that do not block FIN packets to identify whether a port is open. If the port is closed the machine will send a RST and if the port is open, it will ignore the FIN.
- **ACK Scan:** This is used to find out which ports are filtered by a firewall. Open and closed ports respond with RST, whereas filtered ports would not respond.
- **TCP Null Scan:** Attacker sends packet with none of the flags, URG, ACK, PUSH, RST, SYN and FIN, set.
- **TCP XMas Scan:** This is used to scan TCP frames with URG, PUSH and FIN flag set. Closed ports respond with RST and open ports do not respond.
- **UDP Scan:** Attacker sends an UDP packet to a port and the system will respond with an ICMP port unreachable message if the port is closed. The absence of a response is used to infer that a port is open.
- **ICMP IP Address Sweep:** In this the attacker (one IP source) sends at least 10 ICMP echo requests (ping) to different hosts within a defined interval of about 5000 microseconds. *pingscan* or *ipsweep* are good examples. The attackers determine active hosts and then perform more direct targeted attacks specific to those hosts.

Table 4.1. Attack and Protocol Feature Correlation DDoS Attacks

Attack	Protocol fields to be examined
Teardrop	Overlapping Fragment Offsets
Jolt	Protocol = ICMP, Large Fragment Offsets
Ping of Death	Length of all fragments of a packet > 65535
SYN Flood	SYN Flag in Source and ACK Flag in destination addresses
Fraggle	UDP echo on 7, 19, 13, 17
Smurf	Type = 0 without sending type = 8
Boink	Manipulated Fragment offset field in IP Packets
NewTear	Manipulated Fragment offset field in IP Packets

The attacker manipulates the use of certain fields in each of the protocol to cause a particular type of attack [135]. We have made an attempt to analyze the various attacks, identify the parameters and correlate them. We have analyzed attacks at the network and transport layers of the TCP / IP protocol suite. Two specific attacks, Distributed Denial of Service (DDoS) and Port Scan attacks, are analyzed. The protocol attributes are identified and correlated with attack features. The attack feature correlation is given in Table 4.1 and Table 4.2.

Table 4.2. Attack and Protocol Feature Correlation Port Scan Attacks

Attack	Protocol fields to be examined
Connect Scan	Large number of failed connects and sequential port requests
SYN Scan	SYN Flag and no corresponding ACK
FIN Scan	FIN Flag and Sequence number
ACK Scan	ACK Flag and RST as replies
Null Scan	No Flags set
XMas Scan	URG, PUSH and FIN Flags set
UDP Scan	UDP requests and ICMP port unreachable messages
Ping Sweep	Type = 8 and code = 0

4.6. Events at the Application Layer

Web based attacks are considered by security experts to be the greatest and often times the least understood of all risks related to confidentiality, availability, and integrity. The purpose of a web based attack is significantly different than other attacks. Web based attacks occur at the application layer.

The advent of first generation web applications was severely limited in their ability to provide any more information than a brochure you might receive in the mail. Static HTML was provided as a tool to display pictures and inert information. Consequently, as the internet and web access became more and more ubiquitous so too did the needs of those users who were accessing web applications. As a result web applications evolved to provide user conveniences such as searching, posting, and uploading.

CGI (Common Gateway Interface) protocol provided a means for users to interact with web pages by submitting data into forms. Upon submission back end CGI scripts would process this data presented and represent HTML back to the end user. CGI through the interaction with end users effectively became one of the first web application attack vectors known. Newer more evolved frameworks like PHP, ASP.NET, J2EE, AJAX, Ruby on Rails, etc manifested to incorporate more interactive features, allowing users more flexibility and power.

Securing web applications has become incredibly important as the information processed by web applications has become critical to corporations, customers, organizations, and countries. Web applications manage a wide array of information including financial data, medical records, social security numbers, intellectual property and national security data. There is even a serious need to record digital evidence to be used for investigating the attack or atleast understand the attacker's methodology.

We develop a vulnerable website and illustrate the most commonly occurring attack in the cyberspace, Cross-Site Scripting (XSS) attack. We log the data on the web server and identify the attributes and correlate the network events with the attack. The web application attacks involve payload and are more complex than attacks in the lower layers. Privacy protection of legitimate users is to be ensured while analyzing the payload.

4.6.1.Cross-Site Scripting (XSS) Attacks

The popular social networking site Twitter was hit with a 'mouse over' XSS attack on September 21, 2010 [129]. That attack redirected users to a porn site based in Japan. Wicherski, Kaspersky Lab Expert, analyzed that "the user only needed to hover over a malicious link in order to trigger the flaw and the attack was dangerous as it loaded a secondary JavaScript from an external URL" [194]. Twitter identified the flaw and patched the same by evening but only after thousands of users were affected [220].

Open Web Application Security Project (OWASP) lists XSS as the second most prevalent critical web application security risk in its 'top ten' for 2010 [157]. The tenth edition of the Website Security Statistic Report [196] released in fall 2010 by WhiteHat Security lists XSS as the most prevalent class of vulnerability with 71% likelihood of being found in any given website. The Top 25 Most Dangerous Software Errors list compiled by CWE (Common Weakness Enumeration) and SANS (SysAdmin, Audit, Network, Security) for 2010 places the 'improper neutralization of input during web page generation' error which causes XSS, at the top of the list [47].

Cross-site scripting [23] is yet another form of computerized attack over the Internet and are usually aimed at applications running on web servers, i.e. a website with forms to be filled in. The attack is different from most other forms of attack because it uses a vulnerable website application to allow a malicious person to attack other users. In other words, there are at least three parties in a XSS attack:

1. malicious person or attacker
2. vulnerable website application
3. user of the website – the victim

XSS attacks [67] are those attacks against web applications in which an attacker gets control of a user's browser in order to execute a malicious script (usually an HTML or JavaScript code) within the context of a trusted web application's site. As a result, and if the embedded code is successfully executed, the attacker might then be able to access, passively or actively, to any sensitive browser resource associated to the web application (e.g., cookies, session IDs, etc.).

XSS attacks are of two types: persistent and non-persistent [3]. Persistent attacks are also called HTML Injection attacks. Non persistent attacks are called Stored and Reflected XSS attacks. In persistent attacks, the malicious JavaScript code injected by the attacker into the web application is persistently stored into the application's data repository. In turn, when an application's user loads the malicious code into its browser, and since the code is sent out from the trusted web site's application, the user's browser allows the script to access its repository of cookies. Thus, the script is allowed to steal victim's sensitive information and send the same to the website of the attacker. The script circumvents the basic security policy of JavaScript engine. Any JavaScript engine restricts the access of data to only those scripts that belong to the same origin where the information was set up. Persistent XSS attacks are traditionally associated to message boards web applications with weak input validation mechanisms.

Reflected XSS exploits the vulnerability that appears in a web application when it utilizes information provided by the user in order to generate an outgoing page for that user. The malicious code itself is directly reflected back to the user by means of a third party mechanism, instead of storing the malicious code embedded into a message by the attacker. By using a spoofed email, for instance, the attacker can trick the victim to click a link which contains the malicious code. If so, that code is finally sent back to the user but from the trusted context of the application's web site. The victim's browser executes the code within the application's trust domain, and may allow it to send associated information (e.g., cookies and session IDs) without violating the same origin policy of the browser's interpreter. Non-persistent XSS attacks are the most common type of XSS attacks against current web applications, and are commonly combined together with other techniques, such as phishing and social engineering.

Many frameworks were proposed to handle XSS attacks. xHunter [9] takes as input a web trace and scans it for identifying possible XSS exploits. XSSDS [104] is a server side approach to detect XSS attacks. Similarly a client side solution for mitigating XSS attacks, Noxes [112], was also proposed. Di Lucca et al. [54] proposed a technique to assess the XSS vulnerability, which combined static and dynamic analysis of the web design. Wang et al. [229] provided a scheme to know how to collect evidence after suffering XSS attacks from network systems. They gave strategies to prevent XSS attacks from network intrusions.

4.6.2.Cookie Stealing Attack Dataset

We create an XSS Attack dataset using full packet capture on the Server hosting the web site. A XSS vulnerable website is hosted on a server on the LAN. The website is a message board application of a social community. Members login, post their messages and read other's messages. Attacker creates a cookie stealing script and posts it on the message board. If any other member (victim) visits the message board and clicks on the script, his / her cookies are copied and sent to a website set up by the attacker for storing the cookies.

Attacker gets the details of the victim, and logs in as the victim using the cookie information. Attacker can post any type of messages on the message board. We include the snapshots of the attack and the steps in the attack process are as follows:

1. Attacker 'truddy' logs in to the website and places XSS Script as shown in Figure 4.9.

```
<a href = "javascript:window.location =
          'http://emmshub.0fees.net/something.php?cookie='
          +document.cookie"> Click Me </a>
```

Figure 4.9 Cookie stealing script using XSS

2. The text to be placed in the hyperlink can be made more appealing than 'Click Me' to attract gullible victims. The snapshot with the attack script is shown in Figure 4.10.

Figure 4.10 Attacker placing the cookie stealing script

3. Victim 'alice' logs in and clicks on the malicious link. Her cookie is transferred to attacker's site as shown in Figure 4.11.
4. Attacker 'truddy' logs in as Victim 'alice' using the cookie and places odd messages using victim's identity as shown in Figure 4.12.

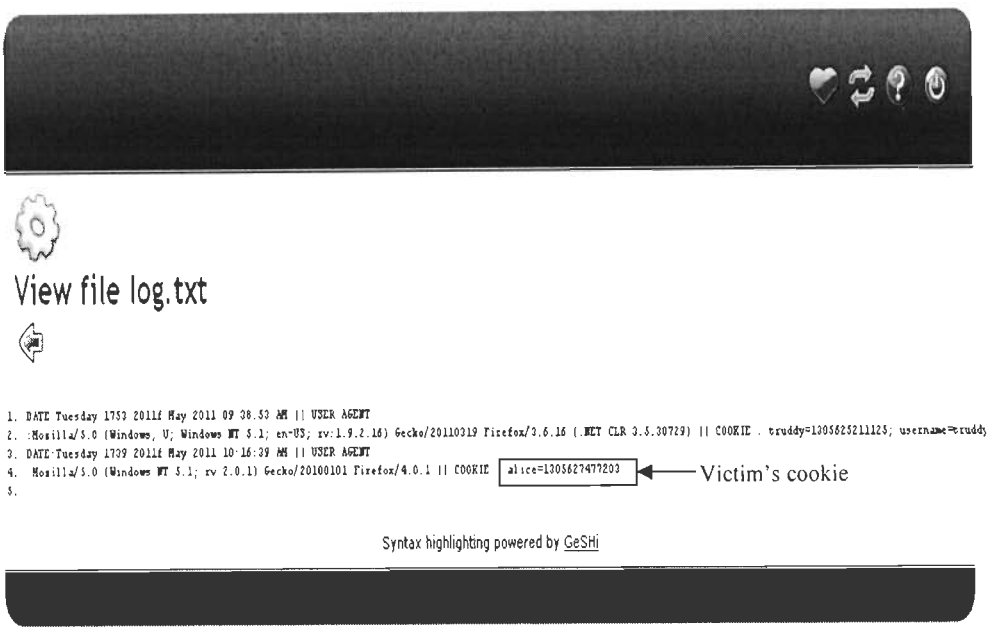

Figure 4.11 Cookie copied to a log file on the Attackers website

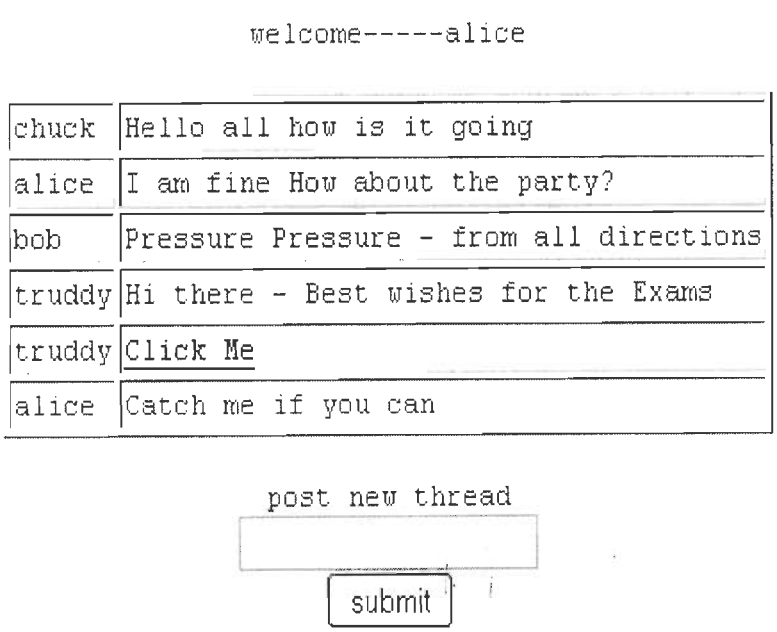

Figure 4.12 Attacker using victims cookie to login and place messages

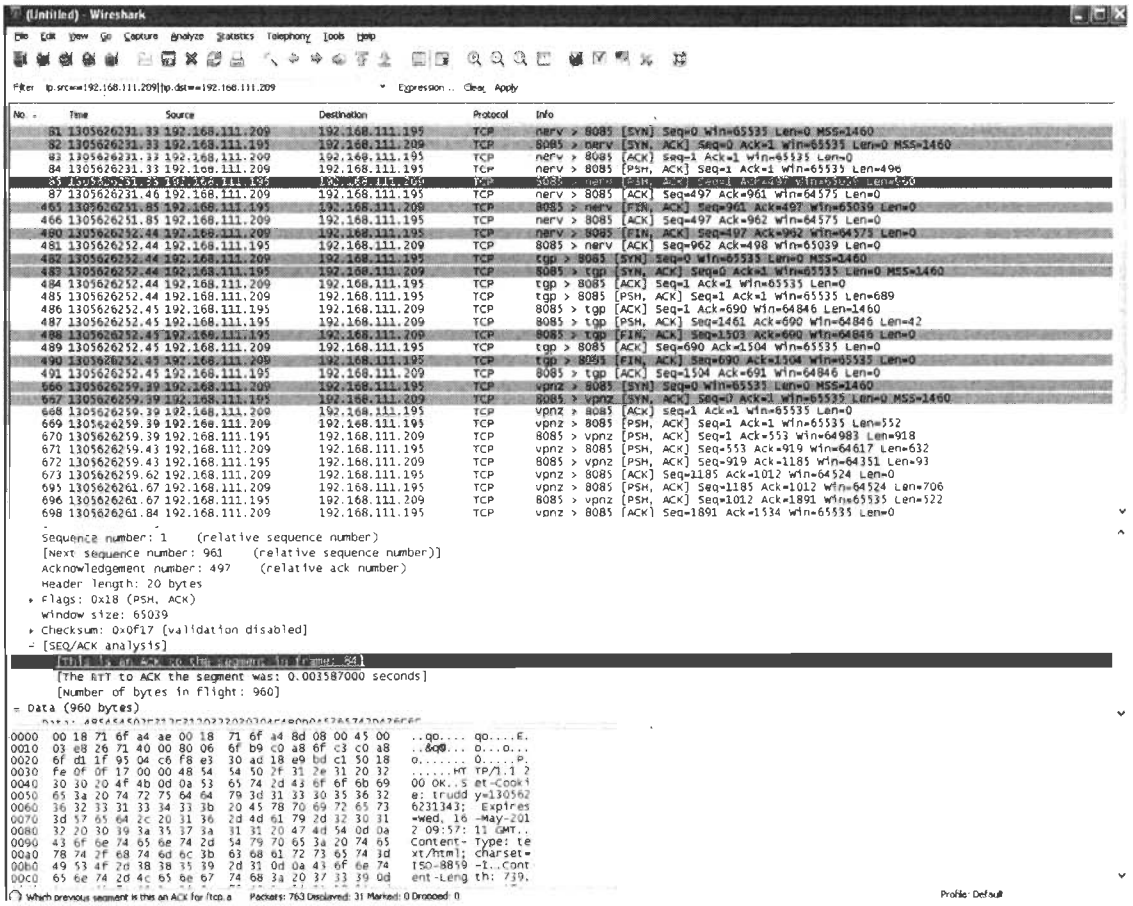

Figure 4.13 Snapshot of wireshark analyzing the packet capture

Full packet capture is performed on the web server and the analyzed using Wireshark [114]. The identification HTTP attributes and correlation of the payload is performed with the attack vectors. The packet capture file is opened in Wireshark and the all the packets who have the server's IP address as source or destination are selected. The snapshot is shown in Figure 4.13.

The snapshots of analysis using wireshark are included below and the examination steps are as follows:

1. Attacker 'truddy' login and cookie information can be read by examining the payload of TCP ACK segments being sent from the server to the client. The snapshot is shown in Figure 4.14.
2. Attacker 'truddy' placing the malicious script as given in Figure 4.8 for stealing victim's cookies can also be seen in the payload as shown in Figure 4.15.

3. Victim 'alice' login and cookie information can be seen in the payload in Figure 4.16.

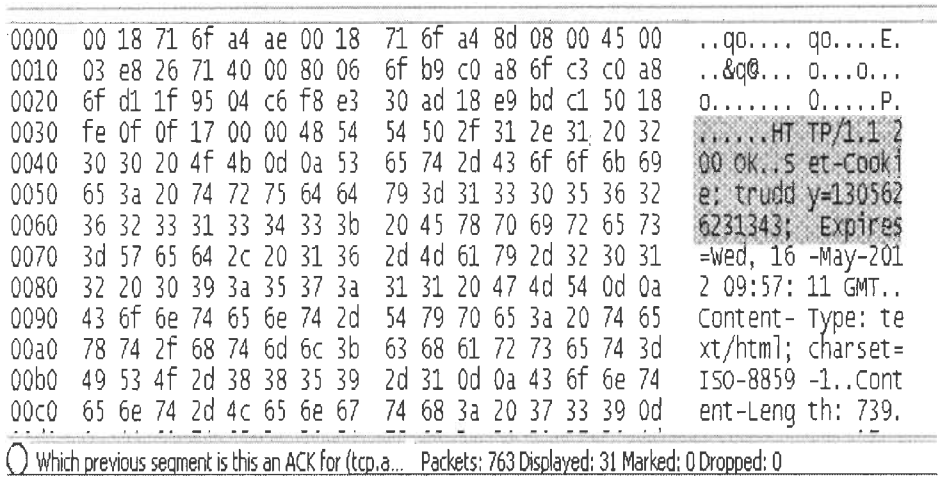

Figure 4.14 Closer look – Attackers login

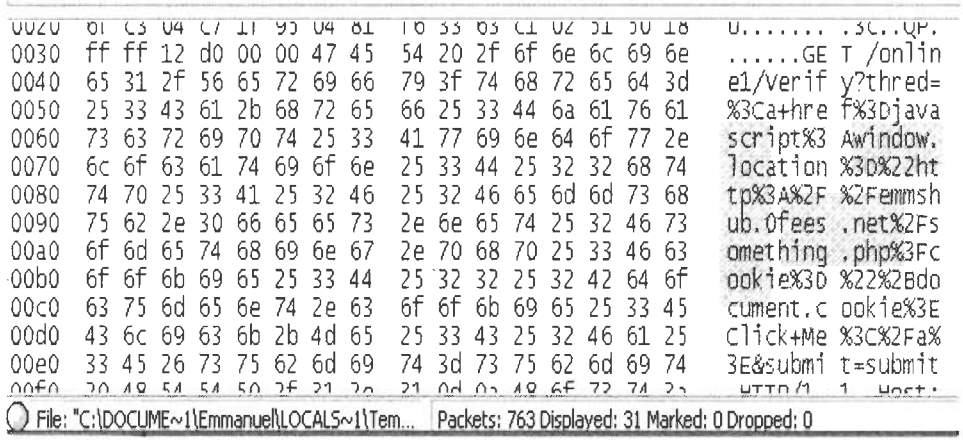

Figure 4.15 Closer look – Attacker placing XSS Script

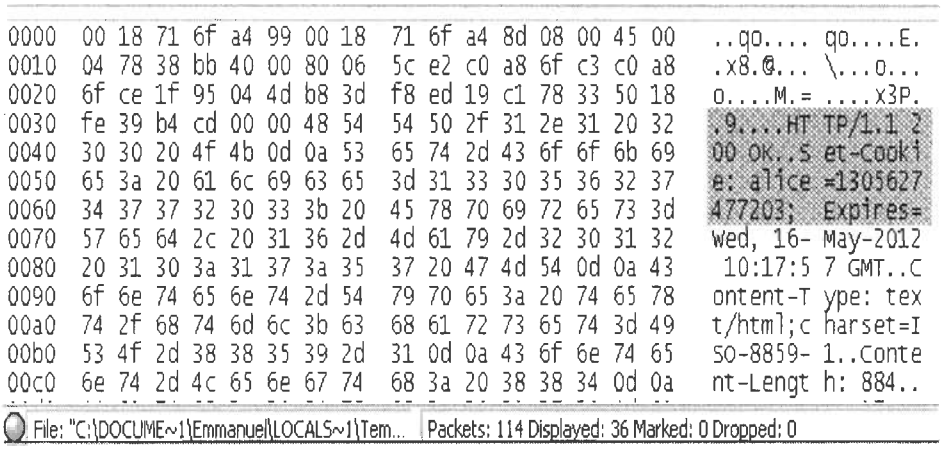

Figure 4.16 Closer look – Victim's login

4. Attacker 'truddy' uses victim 'alice' cookie information and posts odd message on her message board and shown in Figure 4.17 and 4.18.

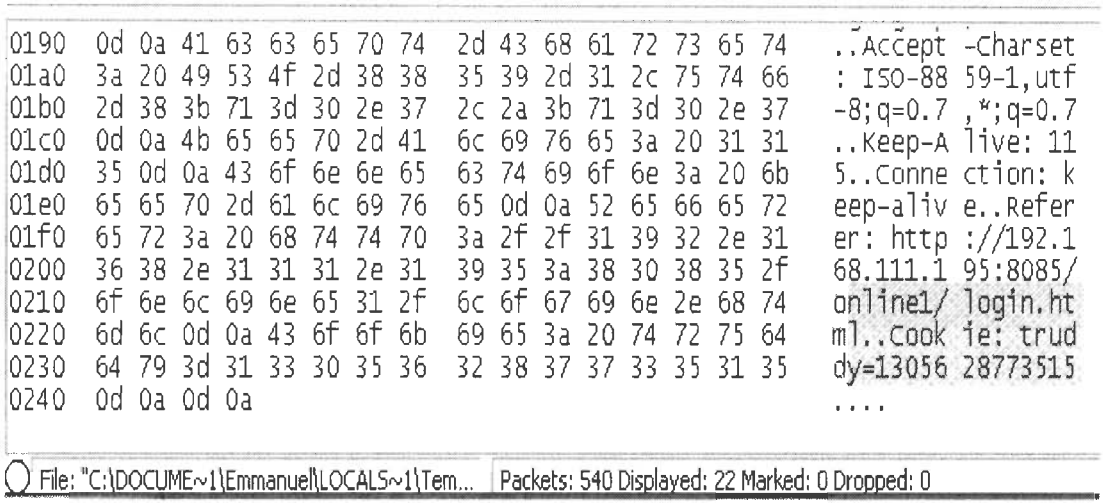

Figure 4.17 Closer look – Attackers logins again

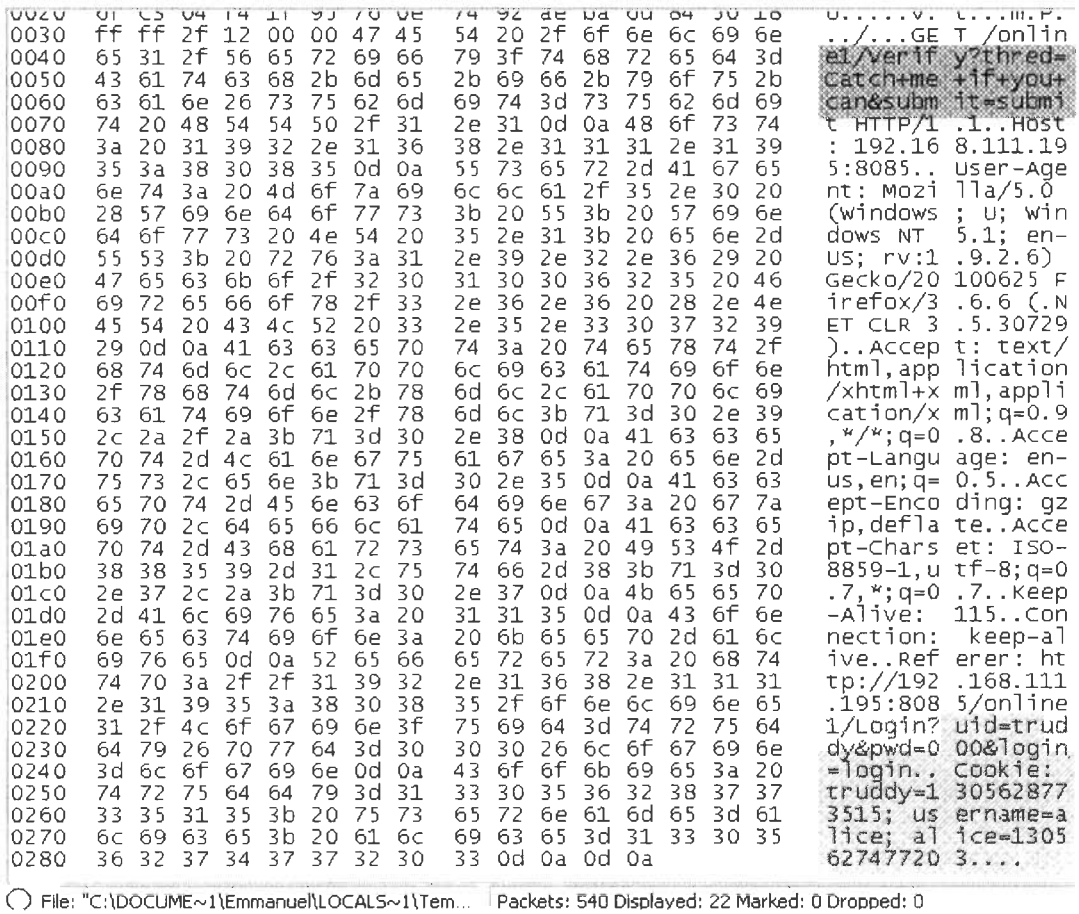

Figure 4.18 Attacker login as Victim and posting messages

4.7. Summary

The various attributes which are being misused at the top three layers of the TCP / IP protocol have been identified. The network features being manipulated by the attackers are correlated with a particular type of attack. These key attributes of the protocol are extracted from the packet capture file using the Net::pcap Perl language library and copied into a database. The statistical thresholds are calculated from these attributes for various attacks. Derived attribute values are calculated from basic attributes of a packet. This information is used to extract suspicious packets from the capture files. The attack packets are converted back into the packet capture format enabling analysis using existing open source tools.

A study was done on a specific type of attack for each of the three layers of TCP / IP protocol suite where attackers usually focus their strategies. It is not possible to make an extensive study of all the attacks and their variants at each of the three layers. Novel and 0-day attacks are developed – using slight variations or combinations of some of the attacks. A methodology was proposed to examine the attack features and arrive at a preliminary conclusion whether to continue with the attack investigation.

Identification and Correlation of protocol attributes which are manipulated by the attackers is a continuous process. Detecting the attack from packet capture files needs experience in monitoring traffic data for years. We have presented three attacks one at each of the network, transport and application layer. A datasheet of all the possible attacks and at all layers can be prepared over a period of time. The common techniques and behavioral patterns of the hackers also can be examined and correlated. This will help in analyzing novel and zero-day attacks.

Chapter 5

Multi Sensor Data Fusion

5.1. Introduction

Bass [14] predicted that the next-generation cyberspace IDS will require the fusion of data from myriad heterogeneous distributed network sensors to effectively create cyberspace situational awareness. The focus for network forensics is on developing a mechanism for integration of packet captures from various sources (hosts) and fusion of information from multiple sensors (tools). The motivation of our work is to develop a model for integration of packet captures from various hosts and fusion of information from multiple sensors to get a comprehensive picture [33].

There are many advantages of data fusion as mentioned by Esteban et al., [57] and these are found true in the case of fusion of network packet captures as well. The fusion of information from various sensors provides (a) increased robustness and reliability even in the case of sensor failure, (b) independent and specific features can be obtained, (c) extended parameter coverage rendering a complete picture, (d) improved resolution, reduced

uncertainty and increased confidence. These advantages drive the network based forensic analysis also to consider fusion involving multiple tools and hosts [37].

Two of the many challenging problems in network forensics are 'Complexity' and 'Quantity' [30]. Complexity problem is that the acquired data are at the lowest and most raw format, which is often too difficult for investigators to understand. It is not impossible but often requires great skills. Quantity problem is that the amount of data to analyze is massive. It is very difficult and inefficient to analyze every single piece of data.

The cost effective storage and faster processing resources available these days make it possible to collect full packet captures with existing open source sniffer tools. However storing large volume of network packets from high-speed networks is still a complex issue. The storage resources, though affordable, cannot be very large and are quickly exhausted by the enormous data. Thus archival time of network traffic logs is limited and the old data has to be overwritten.

These problems are multiplied when packet captures are collected from many clients in a LAN for investigation. These captures for multiple hosts can be evaluated independently or may be integrated to form a single file and evaluated. Integration will facilitate reduction in time for evaluation as data available in independent files is same as the data in the integrated file. The independent files may also have lot of redundant information when the packets are captured in promiscuous mode. Related data may also be gathered from various security tools which collect specific network event information and statistics.

We propose novel models for integration of the packet captures and fusion of alert information, while effecting data reduction. The libpcap files are collected from various clients in a LAN onto a forensic analysis server. These files are then integrated to form a single pcap file by removing duplicate packets using a database. Later, various security tools are used to identify useful network events occurring during the attack duration. This information is fused using the Dempster-Shafer (D-S) theory of evidence [199, 200] to get a comprehensive picture of the attack(s) and strategic information about the attacker(s).

Datasets in packet capture format are not available for validating models proposed for post-mortem forensic investigation. The most popular dataset, KDD Cup 99 dataset [85], is more than 10 years old and has attacks which are mostly obsolete. We create an attack dataset involving port scan attacks and DDoS attacks in our research lab.

5.2. Integration of Packet Captures

We integrate packet capture files from various hosts to effect data reduction and solve the quantity problem. We assume that packet capture and sniffer tools (tcpdump or wireshark) are installed and configured in the victim network as network forensics is possible only when the hosts are prepared to record evidence. When a particular attack is to be investigated, the packet captures on compromised machines are collected.

Libpcap [94] is the library developed by the developers of tcpdump to perform low-level packet (network traffic) capture, read, write and analyze capture files. This library provides the packet-capture and filtering engines of many open source and commercial network tools, including protocol analyzers / packet sniffers, and network intrusion detection systems. This library also specifies the most basic file format [95] used to save captured network data. The file extension is .pcap. The packet sniffer tools log the network traffic information on each host in a packet capture file.

5.2.1. Architecture

We propose to integrate the files collected from various hosts into a single file so that all the attack information is available at one place as shown in Figure 5.1. It is also easy to analyze a single packet capture file against a series of security tools. The integration will also result in data reduction as some of the packets collected by the multiple hosts will be similar. There will be broadcast and multicast packets which are logged by all hosts. The major issue to integrate the files will be to handle the timestamps of redundant packets with same

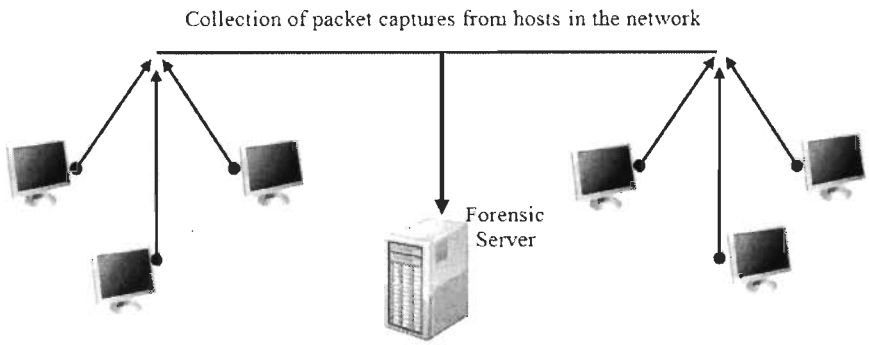

Figure 5.1. Model for Integration of Packet Captures

information. The other issue is to identify which files to be integrated directly and which files need a redundancy check before integration. This decision depends on the location of the compromised hosts from which the files are collected, whether the system is within a particular subnet.

The various steps involved are as follows:

1. The packet captures (pcap files) are collected from compromised systems which are identified in a network.
2. These files are integrated by converting them to a database, identifying unique packets and recreating a single file.

The Perl language module Net::Pcap [218] is used to convert the packet capture file contents into a database. The information contained in each packet header and payload is copied into a database. If 'n' packet captures are to be integrated, there will be 'n' database tables. The 'UNION' operation is performed on these 'n' tables resulting in a single table. When payload is same for a packet in two tables, only one copy is placed with a header from any file. This single table created is reconverted back into a packet capture file resulting in a single integrated file. Once the integrated packet capture file is ready for analysis, it is passed on to the fusion process.

5.2.2. Procedure

The two algorithms are developed to handle the redundancy and timestamp issues. The algorithm in Figure 5.2 decides which files are to be integrated directly and which files are to be checked for redundancy before integration. The algorithm in Figure 5.3 identifies the similar packets in all the files being integrated for a selected timestamp range. The similarity measure is the same header information and payload. One of the similar packets is included and the redundant packets are ignored. The algorithm does not drop duplicate packets within the same packet capture file and ensures a fair treatment to crafty packets created by the attacker with similar payload.

Algorithm in Figure 5.2 can be understood by using the rules for forming groups of hosts from which the files must be integrated, only after removing redundant packets:

1. The packets from hosts at different router level need not be in the same group. Level represents all hosts who are in same subnet or domain.
2. The packets collected by agents residing in different subnet or under a different NAT address need not be in the same group.
3. Packet logs existing in a particular group are only integrated after removing the redundancy.

```
program PacketLogging (Output)
begin
   for i:= 1 to k do     # For all levels, 'k' is the last
      if agent(host logging packet)is at level = i then
            collect packet logs only for all level i+1 routers
            by filtering out their IP/mask into a group
            # Only packet logs in level 'i' will be in a group
      end if
   end for
end.
```

Figure 5.2 Algorithm for identifying files for integration

The packet capture files are grouped according to the above rules and the program for integration is executed on files in the same group. Packet captures in one particular group will need a check for redundancy before integration.

All the 'm' pcap files are made into 'n' blocks each according to a fixed timestamp range. The number of packets in each block is also stored. All blocks with the same timestamp range are considered. The packets from the first are copied into a 'temp' database. The packets from second to the mth pcap are compared with packets in 'temp' and inserted if they do not already exist. Once all the pcap files are compared, the 'temp' database in moved to a 'final' database. The process is illustrated by the algorithm in Figure 5.3.

Though all the packets in each timestamp range are checked for redundant packets, there may be a possibility to miss out packets lying in the border ranges. This process of dropping same packets may result in loss of some information. However this value is very negligible compared to the strength obtained because of data reduction.

```
program Integration (Output)
   temp: temporary database
   final: final database
   var  m: int; # number of pcaps to be integrated
        n: int; # number of timestamp ranges in each file
        TotalNoPackets [1..m]: int; # Total number of
                  # packets in each of the m pcap files
begin
   for i = 0 to n-1 do
        INSERT all packets of the first pcap into temp
        for j = 1 to m-1 do
             for k = 0 to TotalNoPacket[j] do
                  INSERT pcap[k] IF NOT EXISTS in temp
             end for
        end for
        SELECT all packets from temp and INSERT into final
        EMPTY temp
   end for
end.
```

Figure 5.3 Algorithm for integration after checking for redundancy

5.2.3 Results

An attack dataset was created using 3 hosts in a LAN in our research laboratory. Two systems had *Ubuntu* v 9.10 and one system had *Windows* XP (SP3) as operating systems. Three systems were used to record packet captures using *tcpdump* [96]. Normal internet browsing and downloading of various files was carried on these three systems. Two systems were used to launch the port scanning and DDoS attacks.

Packet captures were collected from the three hosts of sizes 183.0 MB, 328.2 MB and 200.5 MB. These three packet capture files are now integrated to form a single file. Simply combining the files would result in a file size of 711.7 MB. Using our algorithms described in section 3.2, the redundant information is avoided and the resultant file is 622.7 MB.

Table 5.1. Reduction by Data Integration

Description	File 1	File 2	File 3
Initial Packet Capture Size (MB)	183.0	328.2	200.5.1
No of packets	405956	574945	551051
Combined file size (MB)		711.7	
No of packets		1531952	
Integrated file size (MB)		622.7	
No of packets		651357	
% of Reduction		12.50	

The results are shown in Table 5.1 and illustrated in Figure 5.4. There is a reduction of 12.5 %, which is not very significant. The reduction in packet capture file size is less but the number of redundant packets dropped is considerable. This is due to many attack packets being captured by all files but the effective size being very small (i. e. UDP Flood packets). Many networks using Layer 3 switches also exclude traffic capture in promiscuous mode and hence redundant traffic may be very less as well. The reduction is not drastic as well and may not be worth the effort in many cases. It is advantageous when multiple files are to be evaluated and when there are constraints in time and personnel.

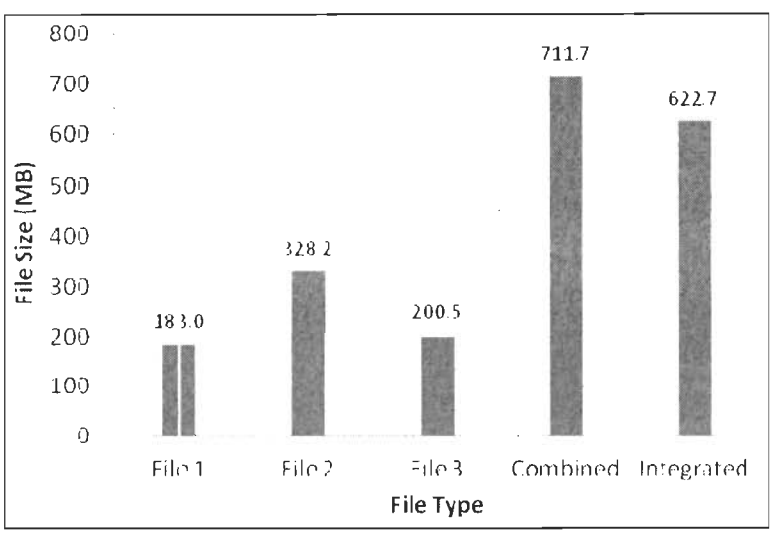

Figure 5.4. Graph illustrating the reduction using Data Integration

5.3. Fusion of Intrusion Alerts from Multiple Sensors

Many of the existing open source network security tools can be used for specific tasks in network forensics. However, they lack the functionality and strength of commercial tools that are specifically designed to process network traffic as evidence. We have used the open source network security and monitoring tools [167], to read the packet capture file post attack, and give various alerts and indicators. We propose that the attack information generated by 'm' number of security tools on 'n' number of packet captures will be same as the alerts generated by 'm' number of security tools on an integrated file.

Our model is built on the work proposed by Onwubiko [155]. There is no single security sensor, currently available, which can accurately detect, locate and identify all the attacks and give a complete picture of the attacker strategy. We use various security sensors to gather the alerts, indicators and statistics from the integrated file. These pieces of evidence are fused and analyzed using the D-S theory of evidence [217].

D-S theory is a mathematical technique based on the belief function and has the capability to combine diverse variety of evidence through its rule of combination. We use this combination rule to fuse information from multiple sensors and obtain a higher level of evidence. The information on the type and nature of attacks is also stored as a report, which can be used to collect all the suspicious packets in the ingress and egress traffic from the packet capture file. The various steps involved are as follows:

1. Various security tools are run on this file and information fusion is done using D-S theory to get a comprehensive picture on the attacks.
2. The fused information is used to identify the attack packets from the integrated packet capture file and mark them as suspicious packets.
3. A new packet capture file is created from the attack packets, which is minimum in size and with maximum possible information as evidence.

The architecture is shown in Figure 5.5. The packet capture files collected from many hosts are integrated and a single file is recreated. Various security tools are run on this file and information fusion is done using D-S theory of evidence to ascertain the validity of the attack occurrence. The IP addresses of suspicious attackers in the fused attack information are used to collect suspicious packet records in the integrated pcap file.

5.3.1. Architecture

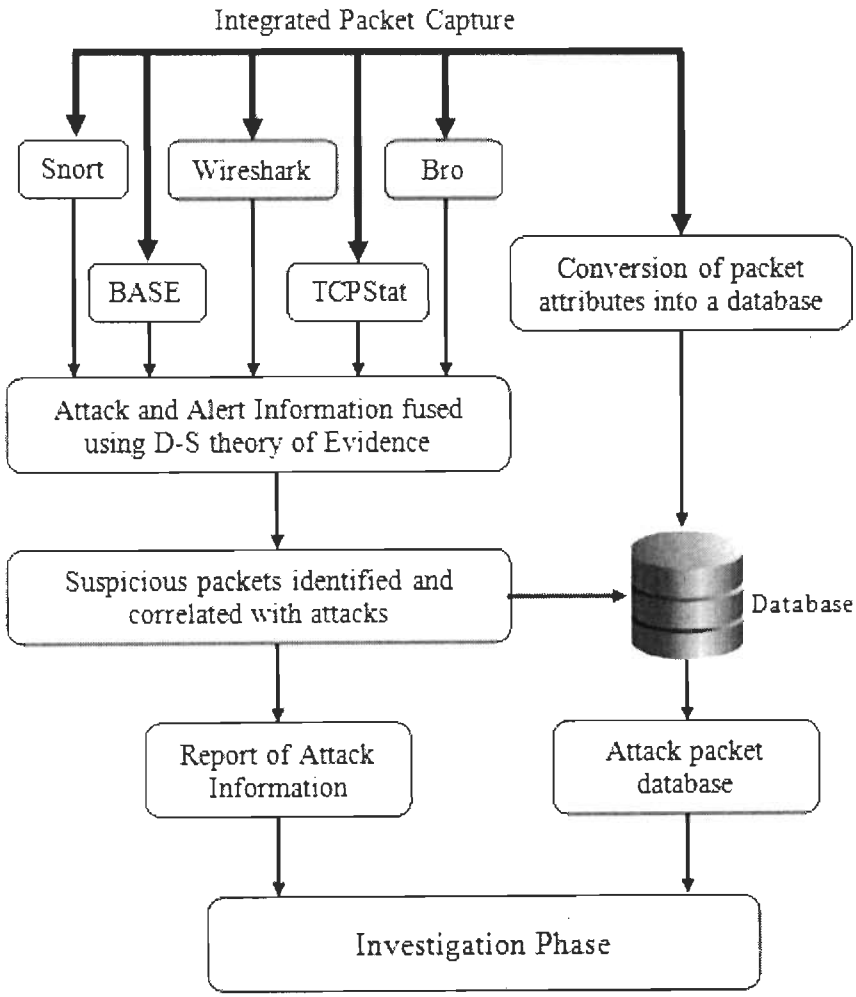

Figure. 5.5. Model for Fusion of Packet Captures

The model is built by aggregating the strengths of open source tools in accomplishing the task of collection and analysis. Security sensors with complementary and contradictory functions are used [214, 215]. Security tools with similarity build the redundancy and reliability of attack information. Diversity among the tools will ensure versatility. Data fusion is performed on the alert and attack information generated by these sensors so that the decision is more accurate. An IDS monitors network traffic and alerts when suspicious traffic is encountered. Packet capture and analysis tools or sniffers identify sessions or connections with anomalies in network traffic. Traffic statistics can be read from packet captures or from Netflow records taken from the network connection through a monitored device.

We use the following tools for data fusion:

- **Snort** [187, 188]: It is an open source network-based IDS (NIDS) and has the ability to perform real-time traffic analysis and packet logging on Internet Protocol (IP) networks. It analyzes network traffic for matches against a user-defined rule set and performs several actions based upon the match. The output of Snort can be configured to various logging and alerting mechanisms. By default, Snort logs are in decoded ASCII format and the full alert mechanism prints out the alert message in addition to the full packet headers. Fast alert mode writes the alert in a simple format with a timestamp, alert message, source and destination IPs or ports. Snort can also read packet captures and analyze the packets.

- **Wireshark** [45]: Wireshark is a network packet analyzer which captures live network packet data and displays the packet data with detailed protocol information. Packet data captured can be read, saved, imported and exported. Packets can be searched and filtered based on many criteria and create various statistics. There are a number of protocol dissectors to decode various protocols in Wireshark.

- **Tcpstat** [84]: It reports certain network interface statistics by either monitoring a specific interface or by reading previously saved tcpdump data from a file. It calculates statistics like bandwidth, number of packets, packets per second, average packet size, standard deviation of packet size, interface load, etc. Tcpstat is capable of handling large amounts of packets per second.

- **Bro** [163]: Bro is a Unix-based Network IDS, which monitors network traffic and detects intrusion attempts based on the traffic characteristics and content. It collects, filters, and analyzes traffic that passes through a specific network location. Bro comes with a rich set of policy scripts designed to detect the most common Internet attacks. These policies incorporate a signature matching facility that looks for specific traffic content. It can also analyze network protocols, connections, transactions, data amounts, and many other network characteristics.

- **BASE - Basic Security Analysis Engine** [105]: It is based on the code from the Analysis Console for Intrusion Databases (ACID) project. It provides a web front-end to query and analyze the alerts coming from a Snort system. BASE is a web interface to perform analysis of intrusions that snort has detected on your network. It uses a user authentication and role-base system, so that you as the security admin can decide what and how much information each user can see.

We convert the contents of the pcap file into a database and reconvert the attack packet records in the database to a new pcap file using the Net::Pcap module of the Perl language [218]. The packet headers in the libpcap file format contain information about the number of packets stored in the capture file and specify the capture length. Much of the network traffic comprises of IP packets based on the network layer protocols of the TCP/IP suite [102]. These IP packets are encapsulated in the packet capture file. The higher layer protocols (TCP, UDP, ICMP and application) information is wrapped in the IP packet. The various attributes in the protocol headers are copied into a database.

The attributes which make up the header information of these protocols is manipulated by the attacker to compromise the victim systems. A new packet capture file is created from the attack packets which is minimum in size and with maximum possible information as evidence. The pcap file with only attack packets is very much reduced in size when compared to the integrated file.

5.3.2. Attack Data Set

Real world attack datasets do not exist for validating models developed for network forensics. There are many incidents which occur in various commercial and governmental organizations. However, datasets in packet capture format contain the payload with confidential information as well. Even anonymized datasets are not made available for privacy reasons. We create an attack dataset using 3 hosts on a LAN in our research lab. Two systems had *Ubuntu* v 9.10 and one system had *Windows* XP (SP3) as operating systems. These three systems were used to record packet captures using *tcpdump* [96]. Normal internet browsing and downloading of various files was carried on these three systems.

Two systems were used to launch the port scanning and distributed denial of service (DDoS) attacks. *Nmap* 5.00-2 [128] and *hping* 3.a2.ds2-4 [191] were used for port scanning and flooding. Various executables of C programs were used for launching DDoS attacks. Brief description of the tools and sample code for some attacks is given below:

- **Nmap (Network Mapper Tool)** is a utility for network exploration or security auditing. It supports ping scanning (to determine which hosts are up), many port scanning techniques, and version detection (to determine service protocols and application versions listening behind ports). Zenmap [133], a GUI front end for Nmap, can also be used. Some attack scripts are given below:

  - SYN Scan:

```
nmap -sS -sU -sV -T4 -O -A -v -PE -PP -PS21, 22, 23, 25,
80, 113, 31339 -PA80, 113, 443, 10042 -PO --script all
192.168.100.111
```

  - ACK Scan:

```
nmap -sA -sS -sU -sV -T4 -O -A -v -PE -PP -PS 21, 22, 23,
25, 80, 113, 31339 -PA80, 113, 443, 10042 -PO --script
all  192.168.100.111
```

  - FIN Scan:

```
nmap -sF -sS -sU -sV -T4 -O -A -v -PE -PP -PS21, 22, 23,
25, 80, 113, 31339 -PA80, 113, 443, 10042 -PO --script
all  192.168.100.111
```

  - TCP Connect Scan:

```
nmap -sP -sT -PE -PA21, 23, 80, 3389 192.168.100.111
```

  - Xmas Scan:

```
nmap -sS -sU -sV -sX -T4 -O -A -v -PE -PP -PS21, 22, 23,
25, 80, 113, 31339 -PA80, 113, 443, 10042 -PO --script
all 192.168.100.111
```

  - Null Scan:

```
nmap -sN -sS -sU -sV -T4 -O -A -v -PE -PP -PS21, 22, 23,
25, 80, 113, 31339 -PA80, 113, 443, 10042 -PO --script
all 192.168.100.111
```

- Ping Scan:

```
nmap -sP -PE -PA21, 23, 80, 3389 192.168.100.111
```

- UDP Scan:

```
nmap -sP -sU -PE -PA21, 23, 80, 3389 192.168.100.111
```

- **Hping3 (Network Ping Tool) [209]:** Hping3 is able to send custom ICMP / UDP / TCP packets and to display target replies like ping does with ICMP replies. It handles fragmentation and arbitrary packet body and size, and can be used to transfer files under supported protocols. Hping3 can be used to perform (spoofed) port scanning, perform traceroute-like actions under different protocols, fingerprint remote operating systems and audit TCP/IP stacks. It can be used to launch Land attack and Xmas Scan.
- **Attack Codes in C:** Various C programs were collected from the Internet to launch attacks on various protocols in the TCP/IP suite. The Distributed Denial of Service attacks are launched with executable files of attacks like beer, jolt, bonk, boink, newtear, nestea, teardrop, syndrop, fraggle, smurf2. The usage of the files and examples are given in Table 5.2.

Table 5.2. Attack code parameters

Command	Parameters
./beer	< dstaddr > <number>
./jolt	<dstaddr> <srcaddr> [number]
./bonk	<dstaddr> <srcaddr> [number]
./boink	< srcaddr > < dstaddr > <startport> <stopport> [number]
./newtear	<srcaddr> <dstaddr> [-s srcport] [-t dstport] [-n number]
./nestea	<srcaddr> <dstaddr> [-s srcport] [-t dstport] [-n number]
./teardrop	<srcaddr> <dstaddr> [-s srcport] [-t dst port] [-n number]
./syndrop	<srcaddr> <dstaddr> [-s srcport] [-t dstport] [-n number]
./fraggle	< dstaddr > <spoofaddr file> <num> <packet delay> [dstport] [srcport]
./smurf2	< dstaddr > <spoofaddr file> <num> <packet delay> <packet size>

5.3.3. Results

The tools used are *Snort* [187, 188], *Wireshark* [45], *TCPstat* [84], *Bro* [163], and *BASE* [105], as described in section 5.3.1. We use the dataset created as described in section 5.3.2 and use the above tools on the packet capture files collected by *tcpdump* while the attacks were in progress. Snapshots of various alerts as indicated by the security sensors for port scans & DDoS attacks are given below. Figures 5.6 and 5.7 show the alerts as indicated by **BASE** of port scan traffic and DDoS attacks respectively.

Figures 5.8 and 5.9 show the information provided by Wireshark analyzing packet captures for DDoS Attack (NewTear) and Port Scan (TCP Connect) is shown

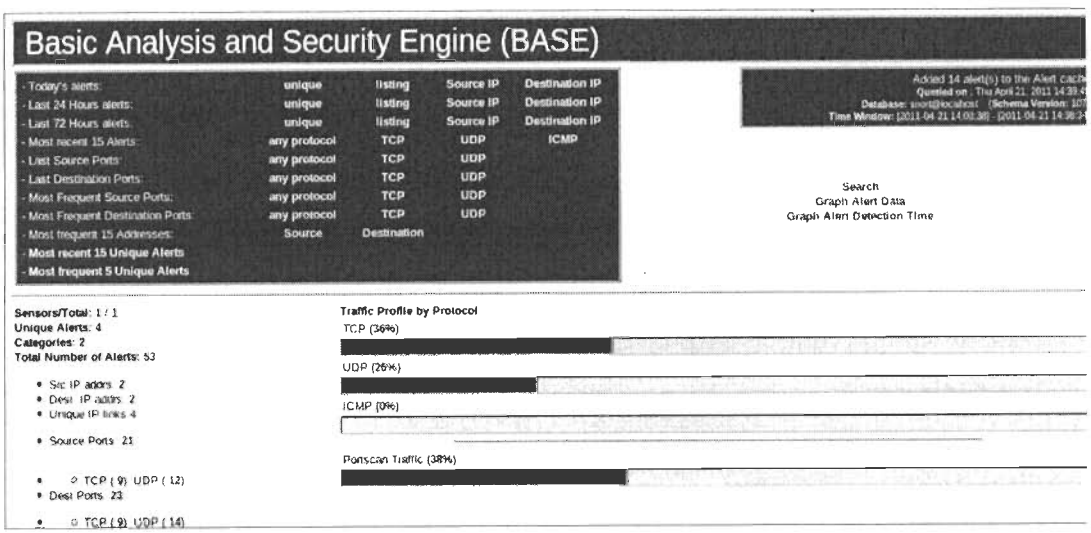

Figure 5.6. Port Scan Traffic

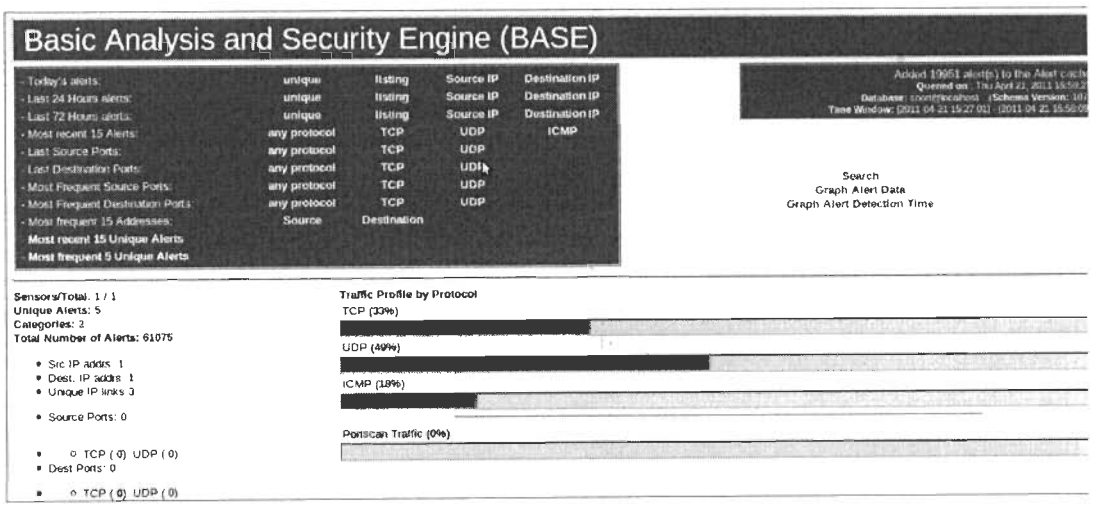

Figure 5.7. DDoS Traffic

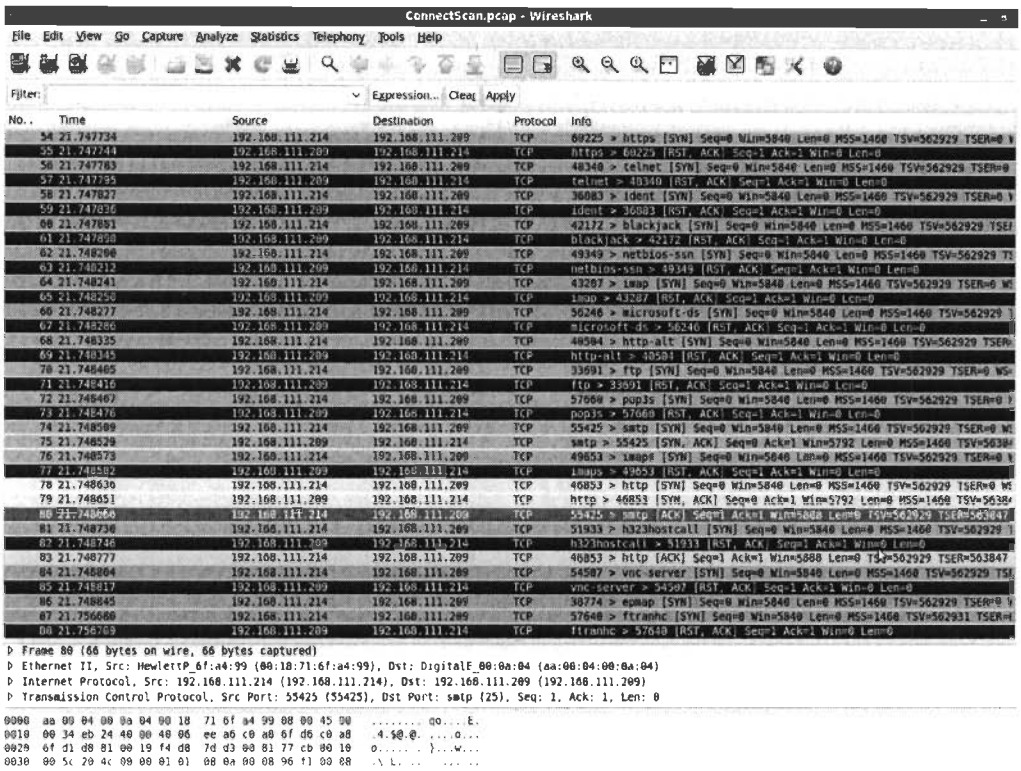

Figure 5.8. TCP Connect Scan

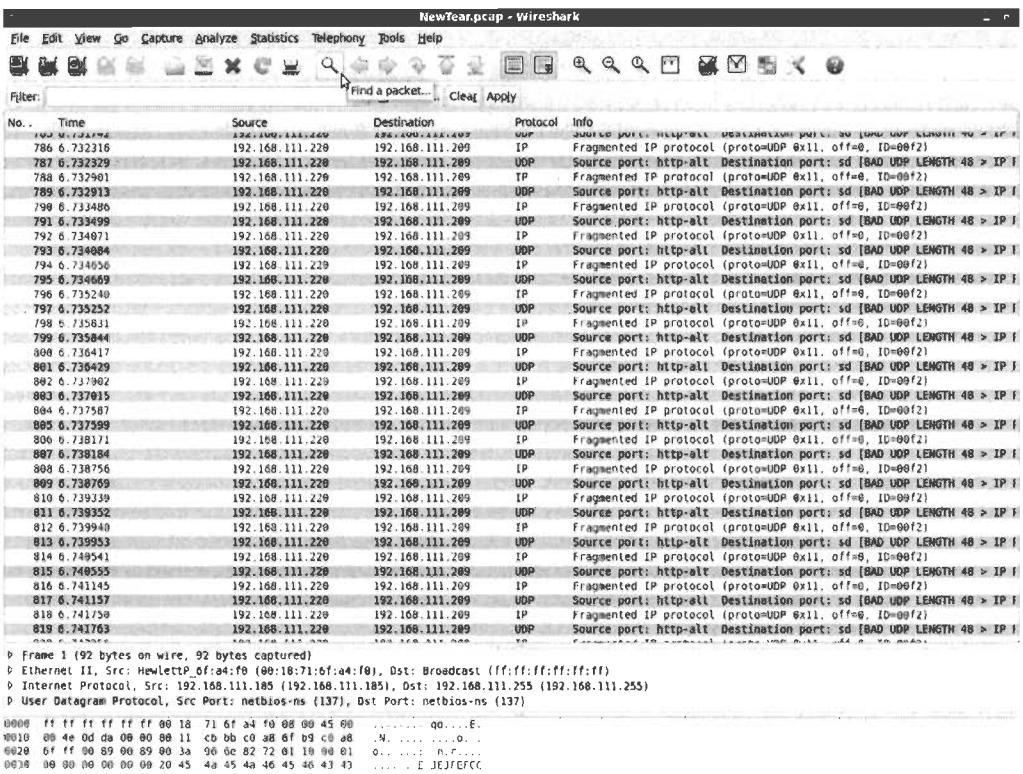

Figure 5.9. DDoS Attack (NewTear)

Snort alerts for some of the port scan and DDoS attacks are as follows:

- *UDP Scan:*

```
04/21-14:38:34.420257 [**] [1:100000160:2] COMMUNITY SIP
TCP/IP message flooding directed to SIP proxy [**]
[Classification: Attempted Denial of Service] [Priority:
2] {UDP} 192.168.111.214:34251 -> 192.168.111.209: 18994
Run time for packet processing was 0.41368 seconds
```

- *Xmas Scan:*

```
04/21-14:20:15.287561 [**] [1:100000160:2] COMMUNITY SIP
TCP/IP message flooding directed to SIP proxy [**]
[Classification: Attempted Denial of Service] [Priority:
2] {TCP} 192.168.111.209:1087 -> 192.168.111.214: 59592
04/21-14:20:28.443260 [**] [122:1:0] (portscan) TCP
Portscan [**] [Priority: 3] {PROTO:255} 192.168.111.214 -
> 192.168.111.209
04/21-14:20:28.518435 [**] [116:59:1] (snort_decoder):
Tcp Window Scale Option found with length > 14 [**]
[Priority: 3] {TCP} 192.168.111.214: 59864 ->
192.168.111.209:1
Run time for packet processing was 0.10479 seconds
```

- *Jolt Attack:*

```
04/21-15:27:01.213385 [**] [123:3:1] (spp_frag3) Short
fragment, possible DoS attempt [**] [Priority: 3] [20]
192.168.111.220 -> 192.168.111.209
```

- *Smurf Attack:*

```
04/21-16:01:12.311008 [**] [1:100000160:2] COMMUNITY SIP
TCP/IP message flooding directed to SIP proxy [**]
```

```
[Classification: Attempted Denial of Service] [Priority:
2] {ICMP} 192.168.111.209 -> 192.168.111.120
Run time for packet processing was 0.96950 seconds
```

- *SynDrop Attack:*

```
04/21-15:58:02.944992 [**] [123:5:1] (spp_frag3) Zero-
byte fragment packet [**] [Priority: 3] {TCP}
192.168.111.220 -> 192.168.111.209
04/21-15:58:02.945564 [**] [123:4:1] (spp_frag3) Fragment
packet ends after defragmented packet [**] [Priority: 3]
{TCP} 192.168.111.220 -> 192.168.111.209
```

Bro Alerts for the same port scan and DDoS attacks are as follows:

- *UDP Scan*

```
1303376290.650025 weird: bad_UDP_checksum
```

- *Xmas Scan*

```
1303375815.316225 weird: spontaneous_FIN
1303375825.534776 weird: bad_TCP_checksum
1303375826.286681 weird: baroque_SYN
```

- *Jolt*

```
weird: 1303379824.209294 excessively_large_fragment
weird: 1303379824.209294 fragment_overlap
```

- *Smurf*

```
1303381865.190161 weird: bad_UDP_checksum
```

- *SynDrop*

```
weird: 1303381689.530709 excessively_small_fragment
weird: 1303381689.530721 fragment_inconsistency
```

```
weird: 1303381689.530721 fragment_size_inconsistency
1303381689.530721 weird: bad_TCP_header_len
```

The output generated from **tcpstat** is also given for the same set of attacks:

- *UDP Scan*

```
$ tcpstat -o "Time:%S No of pkts:%n IPV4:%I TCP:%T UDP=%U
ICMP:%C\n" -r UDPScan.pcap -s 100
. . . . .
Time: 1303376039 No of pkts:119    IPV4:109  TCP:0
     UDP=104   ICMP:5
Time: 1303376044 No of pkts:110    IPV4:105  TCP:0
     UDP=100   ICMP:5
. . . . .
```

- *Xmas Scan*

```
$ tcpstat -o "Time:%S No of Packets:%n IPV4:%I TCP:%T
UDP=%U ICMP:%C\n"   -r XmasScan.pcap
. . . . .
Time: 1303375812 No of Packets:754  IPV4:746      TCP:746
     UDP=0     ICMP:0
Time: 1303375817 No of Packets:552  IPV4:543 TCP:543
     UDP=0     ICMP:0
. . . . .
```

- *Jolt*

```
$ tcpstat -o "Time:%S Packets:%n IPV4:%I TCP:%T UDP=%U
ICMP:%C\n" -r Jolt.pcap
. . . . .
Time: 1303379823 No of pkts:6366   IPV4:6360  TCP:0  UDP=3
     ICMP:6357
. . . . .
```

- *Smurf*

```
$ tcpstat -o "Time:%S No of pkts:%n IPV4:%I TCP:%T UDP=%U
ICMP:%C\n" -r Smurf.pcap
. . . . .
Time: 1303381872 No of pkts:10001  IPV4:9990 TCP:0
     UDP=0      ICMP:9990
. . . . .
```

- *SynDrop*

```
$ tcpstat -o "Time:%S No of pkts:%n IPV4:%I TCP:%T UDP=%U
ICMP:%C\n" -r SynDrop.pcap
. . . . .
Time: 1303381684 No of pkts:11250  IPV4:11237
     TCP:11237 UDP=0       ICMP:0
Time: 1303381689 No of pkts:8727         IPV4:8714
     TCP:8714  UDP=0       ICMP:0
. . . . .
```

5.3.4. Discussion

The attack dataset was created using tools and codes described in section 4.2. The snapshots of some tools and the alert logs of others were shown in section 4.3. We apply the D-S theory of evidence to validate the attacks and identify the suspicious addresses. We introduce few of the terms used in D-S Theory and give the calculations of the combination rule. The *frame of discernment* (FOD) Θ consists of all hypotheses for which the information sources can provide the evidence. This set is finite and consists of mutually exclusive propositions that span the hypotheses space.

A *basic probability assignment* (bpa) over a FOD Θ is defined as a mass function *m*, which assigns beliefs in a hypothesis and defines a mapping of the power set to the interval between 0 and 1. The sum of all *bpa* is equal to 1 and given in equation 1.

$$\sum_{A \subseteq \Theta} m(A) = 1 \qquad (1)$$

The *bpa* are assigned based on the information collected from the sensors. The *bpa* values are decided based on information specific to the environment in which the attack data was collected, type of attacks, correlation between attack and response generated by a particular tool and similar factors. Basic probability assignments given for various attack alerts generated by **Snort** are shown in Table 5.3. Similarly, *bpa* values were assigned based on the alert information collected from the other four tools.

Table 5.3. bpa's for various alerts and attack information given by **Snort**

Attack	Alert Information	bpa
UDP Scan	04/21-14:25:20.007275 [**] [1:100000160:2] COMMUNITY SIP TCP/IP message flooding directed to SIP proxy [**] [Classification: Attempted Denial of Service] [Priority: 2] {UDP} 192.168.111.214:46360 -> 192.168.111.209:17018	0.09
Xmas Scan	04/21-14:20:15.287561 [**] [1:100000160:2] COMMUNITY SIP TCP/IP message flooding directed to SIP proxy [**] [Classification: Attempted Denial of Service] [Priority: 2] {TCP} 192.168.111.209:1087 -> 192.168.111.214: 59592 04/21-14:20:28.443260 [**] [122:1:0] (portscan) TCP Portscan [**] [Priority: 3] {PROTO:255} 192.168.111.214 -> 192.168.111.209 04/21-14:20:28.518435 [**] [116:59:1] (snort_decoder): Tcp Window Scale Option found with length > 14 [**] [Priority: 3] {TCP} 192.168.111.214: 59864 -> 192.168.111.209:1	0.24
Jolt	04/21-15:27:01.213385 [**] [123:3:1] (spp_frag3) Short fragment, possible DoS attempt [**] [Priority: 3]{ICMP} 04/21-15:27:01.244772 [**] [123:8:1] (spp_frag3) Fragmentation overlap [**] [Priority: 3] {ICMP}	0.27
Smurf	04/21-17:23:04.164729 [**] [1:100000160:2] COMMUNITY SIP TCP/IP message flooding directed to SIP proxy [**] [Classification: Attempted Denial of Service] [Priority: 2] {ICMP} 192.168.111.209 -> 192.168.111.119	0.18
SynDrop	04/21-15:58:02.944992 [**] [123:5:1] (spp_frag3) Zero-byte fragment packet [**] [Priority: 3] {TCP} 192.168.111.220 -> 192.168.111.209 04/21-15:58:02.945564 [**] [123:4:1] (spp_frag3) Fragment packet ends after defragmented packet [**] [Priority: 3] {TCP} 192.168.111.220 -> 192.168.111.209	0.22

Basic probability assignments given for various attack alerts generated by **Bro** are shown in Table 5.4.

Table 5.4. bpa's for various alerts and attack information given by **Bro**

Attack	Alert Information	bpa
UDP Scan	1303376290.650025 weird: bad_UDP_checksum	0.06
Xmas Scan	1303375815.316225 weird: spontaneous_FIN 1303375825.534776 weird: bad_TCP_checksum 1303375826.286681 weird: baroque_SYN	0.27
Jolt	weird: 1303379824.209294 excessively_large_fragment weird: 1303379824.209294 fragment_overlap	0.25
Smurf	1303381865.190161 weird: bad_UDP_checksum	0.16
SynDrop	weird: 1303381689.530709 excessively_small_fragment weird: 1303381689.530721 fragment_inconsistency weird: 1303381689.530721 fragment_size_inconsistency 1303381689.530721 weird: bad_TCP_header_len	0.26

Bpa's given for various statistic information generated by **tcpstat** are shown in Table 5.5. Bpa's are assigned to alerts from BASE and Wireshark as well.

Table 5.5. bpa's for various attack statistical information given by **tcpstat**

Attack	Alert Information	bpa
UDP Scan	Time: 1303376039 No of pkts:119 IPV4:109 TCP:0 UDP=104 ICMP:5	0.10
Xmas Scan	Time: 1303375812 No of Packets:754 IPV4:746 TCP:746 UDP=0 ICMP:0	0.19
Jolt	1303379823 No of pkts:6366 IPV4:6360 TCP:0 UDP=3 ICMP:6357	0.28
Smurf	Time: 1303381872 No of pkts:10001 IPV4:9990 TCP:0 UDP=0 ICMP:9990	0.23
SynDrop	Time: 1303381684 No of pkts:11250 IPV4:11237 TCP:11237 UDP=0 ICMP:0	0.20

Belief measure is the degree of evidence that the alert information belongs to the attack information set A as well as to the various special subsets of A. *Plausibility measure* is the degree of evidence that the alert information belongs to the attack information set A or to any of its subsets or to any set that overlaps with A. Given several belief functions on the same frame of discernment based on the different evidences, we calculate a belief function using D-S theory's rule of combination as given in equation 2.

$$m(A) = \frac{\sum_{B \cap C=A} m_1(B) m_2(C)}{1-\sum_{B \cap C=\varphi} m_1(B) m_2(C)} \quad (2)$$

The numerator represents the accumulated evidence for the attack information sets B and C, which support the attack hypothesis A, and the denominator sum quantifies the amount of conflict between the two sets. Equation 2 is shown in [176] to be equivalent to equation 3.

$$m(A) = \frac{\text{Product of bpa's of the attack A given by all sensors}}{\text{Summation of the product of bpa's of the all the individual attacks given by all sensors}} \quad (3)$$

We include the bpa's for the five attacks and using the five tools. The values are shown in Table 5.6 as given below:

Table 5.6. bpa's for various alerts and information for 5 attacks using 5 tools

	Snort	Bro	tcpstat	Wireshark	BASE	Product
UDP Scan	0.09	0.06	0.10	0.12	0.15	0.0000097200
Xmas Scan	0.24	0.27	0.19	0.21	0.22	0.0005688144
Jolt	0.27	0.25	0.28	0.26	0.23	0.0011302200
Smurf	0.18	0.16	0.23	0.22	0.19	0.0002768832
SynDrop	0.22	0.26	0.20	0.19	0.21	0.0004564560

We show the calculations for a two attacks, Jolt and UDP Scan, as an illustration. The numerator will be product of all the bpa's for the Jolt and UDP Scan. The denominator will be summation of product of all bpa's of the individual sensors. The value was calculated as:

$$m(Jolt) = \frac{0.00113022}{0.00000972 + 0.0005688144 + 0.00113022 + 0.0002768832 + 0.000456456}$$

$$= \frac{0.000113022}{0.002419636} = 0.04670$$

$$m(UDP\ Scan) = \frac{0.0000097200}{0.00000972 + 0.0005688144 + 0.00113022 + 0.0002768832 + 0.000456456}$$

$$= \frac{0.0000097200}{0.00241963600} = 0.00402$$

The values appear to be very low when compared with the assigned bpa values because five tools were considered for data fusion. The value would be more than individual bpa if only two tools were considered. However, it is very trivial to note from the values that the Jolt attack has occurred definitely compared to the UDP Scan attack. The probability value of Jolt is ten times more than UDP scan. We can infer that this result obtained by fusion of information from several sensors gives a stronger belief that the attack has occurred when compared to the belief arrived from information collected from individual sensors. This process can be repeated for various other attacks. The above example illustrates that the validation of attack can be done accurately when data is fused from multiple sensors.

The values of bpa are assigned manually by a forensic investigator. The values are determined based on the years of experience in monitoring the network traffic. It is also based on the likelihood that the occurrence of a particular network event or an alert generated by specific tool. The value of 'm' depends on the number of security sensors and a threshold 'τ' is also established using the same criteria above. If the value of 'm' is greater than 'τ', for a given set of sensors, then we conclude that attack has taken place. If it is very less, it can be concluded that the attack has not taken place.

This information strengthens the evidence for the occurrence of an event and validates the same. Attack validation forms an important aspect in network forensics as it guides the important decision to carry on with the investigation or to consider the alert as a false alarm and ignore it. We have used similar and diverse tools to ensure redundancy and versatility. Data fusion of the alert information from various tools gives a strong evidence of the attack occurrence. The tools are all open-source and are suitable for post mortem analysis as all of them work with packet capture files. The outputs of all the tools can be redirected to logs or plain text files. These files can be searched manually to trace the attack information and alerts. Attack validation can be made using the combination rule of the D-S theory of evidence. The extra overhead is the processing time of performing the fusion of the attack information and alerts from various sensors. This is very negligible compared to the time spent on investigating an attack on the basis of a false alarm generated by one security sensor.

The validation of the attack also generates important information about the source of the attacks and facilitates data reduction. The report generated based on the alerts and attack information identifies the suspicious IP addresses and connections with those hosts are analyzed. The suspicious packets in the ingress and egress traffic are collected and a new packet capture file is created. The file size is reduced to 43.8 MB from the size of the integrated file of 622.7 MB, which is 92.96 % reduction. The results are given in Table 5.7. The reduced file can be easily analyzed because of the size and it contains all the information about the connections and communications with the attack source.

Table 5.7: Reduction after Data Fusion

Description	Integrated File
Integrated file size (MB)	622.7
Suspicious packets file size (MB)	43.8
% of Reduction	92.96

5.4. Summary

Network forensics analysis is performed using the integration of packet capture files and fusion of alert and attack information. Packets captures were collected from multiple sources and integrated to hold the entire information across the network. The redundant information is dropped during the integration process. The fusion of alert information and statistical values from various tools and application of D-S theory of evidence for fusion increased the confidence level. The attack is detected and validated and the crucial decision to proceed with the investigation is made. The attack information identifies suspicious packets and effects data reduction.

The proposed data integration and fusion models were validated on sample data sets with attack traffic generated in our research lab. The results were as expected for this particular situation with existing infrastructure. These models have to be tested against large datasets comprising of real time attack data.

Chapter 6

Source Traceback and Attribution

6.1. Introduction

IP traceback problem involves identifying the actual source of a packet across the Internet. Many techniques for traceback have been proposed, but all of them are focused on distributed denial of service (DDoS) attacks [117]. These techniques are aimed at prevention, detection and mitigation of the attacks. However many of these techniques are yet to be implemented and deployed. These techniques can be slightly modified to suit other attacks as well. IP traceback mechanisms aim at tracking back the source of attack, fix the accountability for the attack, implicate the attacker and prosecute them for any malicious actions. These mechanisms meet the ultimate goal of network forensics in providing sufficient evidence to allow the perpetrator to be prosecuted [237].

IP Traceback [221] is an important strategy to contain the ongoing attacks or to investigate and attribute the attacks in the post mortem stage. IP traceback problem is defined as "identifying the actual source of any packet sent across the Internet". IP traceback

techniques are not capable of preventing and mitigating the attack. They can only identify the source of attack packets. However, this information can be used to conduct post mortem investigation of the attack.

The weaknesses in TCP/IP facilitate IP spoofing where source address in the IP header can be manipulated. Network address translation (NAT) allows many hosts behind a gateway and all the packets sent by these hosts have the gateway's public IP address as their source address. Moreover, the attacker may not launch the attack directly and will use already compromised hosts and zombies. The attack flows through a chain of stepping stones (intermediate compromised hosts) before reaching the victim. There are also many anonymous systems which facilitate privacy and anonymity for legal users which become a convenient platform for stepping stone attacks. It is very difficult to trace the actual attacker who controlled the launch of attack(s) and the reconstruction of the path back to the attacker is a nontrivial task.

We propose two approaches, AS based Deterministic Packet Marking (ASDPM) and Deterministic Router and Interface Marking (DRIM). These two approaches are based on deterministic packet marking (DPM) as shown in Figure 6.1.

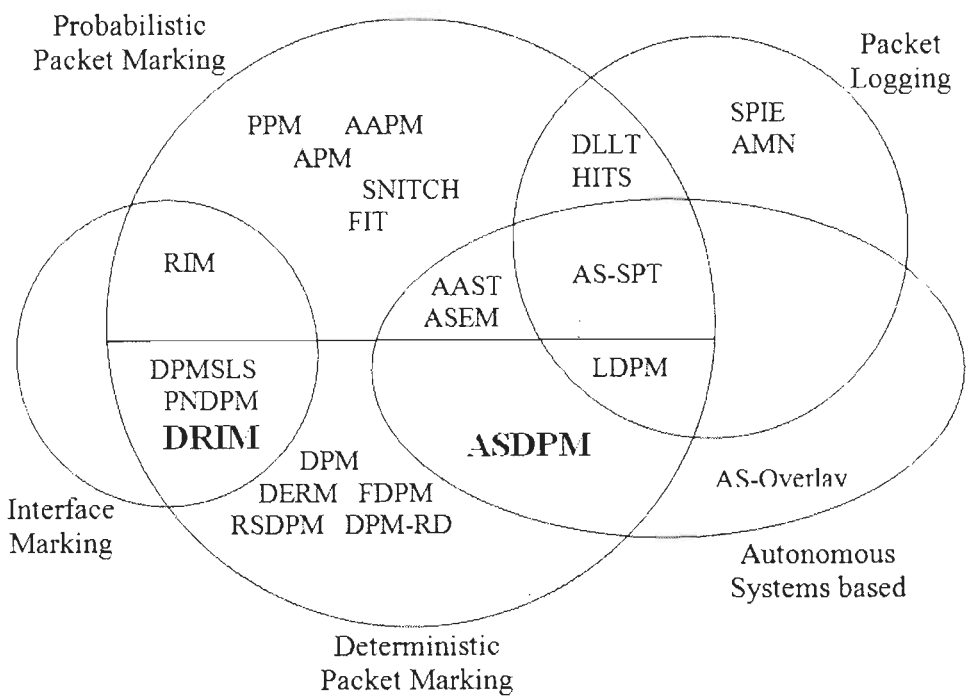

Fig. 6.1 Relation between proposed and existing Traceback Mechanisms

They use the following three values: (1) number associated with the source Autonomous System, (2) address of the first ingress edge router and (3) number associated with an interface through which the packet entered. The ASDPM approach uses a two level traceback mechanism. The first level involves deterministic marking of each packet with the hash of the IP address of the first internal router and the second level involves marking each packet by the AS Number (ASN) of the AS boundary router when it is leaving the source AS. The DRIM approach involves deterministic marking of each packet with the hash of the first ingress edge router's IP address and the number of the interface through which the packet reached the router. We present a detailed study of evaluation against various performance metrics, compare with related techniques and conduct simulations to validate the approaches.

The basic idea in both of our approaches is to record the access point as close as possible to the attacker. This information needs to be obtained from sources which the attacker cannot manipulate. We mark this information in each packet deterministically at the first ingress edge router. No other router modifies this marked information. A single packet is sufficient to detect the attack source as each packet carries the mark to traceback to the attack source. In ASDPM, we traceback to the internal router within the source AS of the attacker. We move closer, by identifying the interface on which the packet reaches the router, in DRIM.

Network forensic systems are classified into different types, based on various characteristics as discussed in section 2.1.2. We focus our work on the post mortem and packet based network forensics. The following assumptions are made similar to those in [50, 66, 193, 208] with simple modifications to suit traceback in the context of network forensics:

- attackers are able to generate and send any packet
- multiple attackers may act in a coordinated fashion
- attackers are aware of the traceback ability
- routers possess limited processing and storage capabilities
- routers in the attacker's network participate and all other routers may not participate
- routes between hosts are usually stable, but packets can be reordered or lost
- some of the attack packet streams may consist of only few packets and investigation needs to be performed using the limited evidence
- marking will be performed as close as possible to the attacker's network in comparison to the victim's network.

Two novel approaches for network forensic traceback, ASDPM and DRIM, are proposed in the next sections. They are compared and analyzed with related techniques against various performance metrics: level of involvement by the ISP , number of attacking packets needed for traceback, effect of partial deployment, processing overhead, bandwidth overhead, memory requirements, ease of evasion, protection, scalability, infrastructure changes, ability to handle major DDoS attacks, and ability to trace transformed packets. Simulations performed using ns-2 to validate the proposed approaches are explained.

6.2 ASDPM: Autonomous System based Deterministic Packet Marking

Autonomous Systems (ASes) are components that make up the Internet hierarchy and are regulated by one or more network operators. The operators enforce a consistent and clearly defined external routing policy. We use a two level traceback mechanism, where the first level involves deterministic marking of each packet by the first internal router within the AS and the second level involves marking each packet by the AS boundary router (ASBR). The architecture is shown in Figure 6.2.

Internal routers are directly connected to networks belonging to the same area. AS boundary routers exchange routing information with routers belonging to other Autonomous Systems. ASBRs advertise AS external routes throughout the AS and every router in a given AS knows the path to ASBR [141]. The first internal router marks each packet with the 12-bit hashed value of its 32-bit IP address. The AS boundary router marks the packet with its 16-bit AS number when it is leaving for the next AS. Once the packet is marked in either levels, the specific mark cannot be overwritten. Every outbound packet is marked and inbound packets are not marked. A single packet is sufficient to detect the source as each packet contains the information about the AS and the router.

Consider four ASes as shown in Figure 6.1 with various AS boundary routers and internal routers. Attacker is a host in AS1 and connects to the Internet through internal router R11. Packets from the attacker reaching R11 are marked deterministically and no other routers (R12, R13, etc) overwrite the mark. The packets travel within the AS and reach the AS boundary router ASBR1. If the packet is leaving to another AS, say AS3, then the packet is marked a second time. If it is travelling to another location within the AS, it is not marked.

Thus any packet is marked only twice, once by the first internal router (R11) and once by the AS boundary router (ASBR1) when it is leaving the AS.

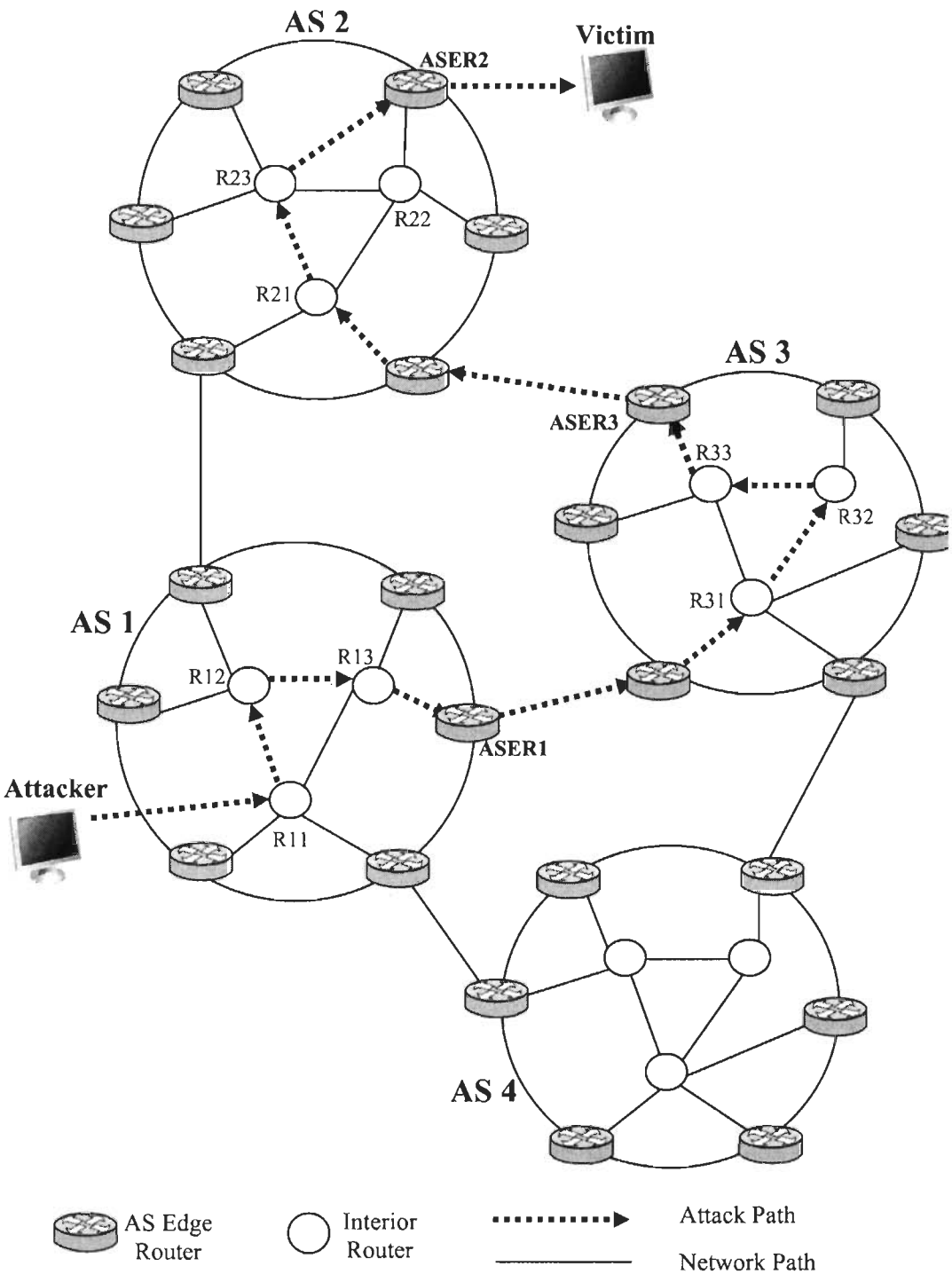

Figure 6.2. AS based Deterministic Packet Marking

6.2.1 Mark Information Encoding Scheme

The mechanism uses the 16-bit ID field, 3-bit fragment flag field and 13-bit fragment offset field in the IP header to store the marking information. The mapping between fields in the IP header and the marking fields is shown in Figure 6.2.

These 32 bits were designed in the IPv4 protocol to hold information about fragmentation. The flags and fragment offset fields can be used for fragmentation only in conjunction with the identification field as reassembly of the fragments will require that all fragments have the same ID number. We use all the 32 bits for marking as the fragmentation traffic is very rare in Internet (about 0.25% of all traffic) [193, 204].

The least significant 12 bits of the 13-bit offset are used to store the hashed IP address of the first internal router traversed by the packet as in AAST [162]. The most significant bit is used as a flag to indicate that the marking has taken place. All the 16 bits can also be used to store a 16-bit hash of the router address as in DERM [178], which reduces the number of false positives dues to hash collisions.

The AS number is stored in the 16 bit ID field and the reserved flag bit is set to 1. The total number of ASes in the Internet as on January 4, 2011 is 58000+ [89] and are globally identified by a 16-bit ASN. In order to reduce the false positives, we use the 16-bit ASN for marking rather than the 32-bit IP address of the AS boundary router. ASN can be uniquely represented using the 16-bit ID field [73]. It is proposed that ASN will become 32 bit once the 16 bit numbers are exhausted. Some ASes are issued 32 bit numbers already by IANA.

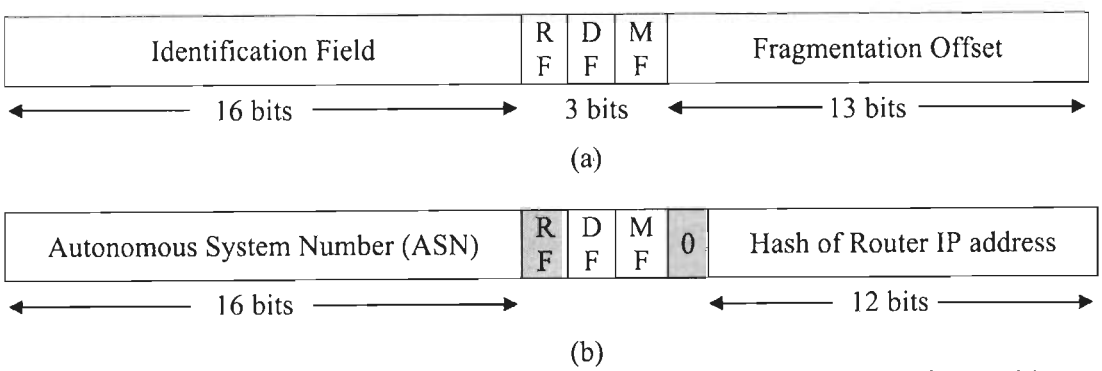

Fig. 6.3. Mark encoding (a) fields in the IP header (b) fields overloaded for marking

6.2.2 Marking Scheme

First level in the proposed two-level marking involves marking at the first internal router which the packet traverses. The internal routers examine the most significant bit of the offset field to check if the packet has been marked by previous routers and then forward it. If the packet has not been marked, then the 12-bit hash value of the 32-bit IP address is copied into the 12 least significant bits of the offset field before forwarding. The algorithm 6.1 handles the marking mechanism at the first internal router:

```
foreach outbound packet P do
   if P.offset[0] = '0' then
        P.offset[0] ← '1';
        write HashIP12(Ri) into P.offset[1..12];
   end
   forward (P);
end
```

Figure 6.4: Marking at the first internal router

The second level marking is at the AS boundary router (ASBR) from which the packet is leaving to another AS. The AS boundary routers examine the reserved flag field to check if the packet has been marked by previous ASBRs and then forward it. If the packet has not been marked and if the packet is traversing into another AS, then the 16-bit AS number of the ASBR is copied into the 16-bit identification field before forwarding. The algorithm 6.2 handles the marking mechanism at the AS boundary router:

```
foreach outbound packet P do
   if P.flag[0] = '0' and P has the destination
        address in another AS, ASBRj then
        P.flag[0] ← '1';
        write ASN (ASBRi) into P.Identification;
   end
   forward (P);
end
```

Figure 6.5: Marking at the AS boundary router

6.2.3. Traceback Scheme

Traceback operation is simple as each packet holds the information required to identify the AS and the first internal router connected to the attacker. The 16-bit identification field in the IP header gives the ASN, identifying the source AS of the packet. The 12-bit hash value in the offset field is used to extract the internal router IP address using the hash function. Algorithm 6.3 explains the traceback operation extracting the ASN and IP address of internal routers at the victim side:

```
foreach attack packet P reaching Victim V do
    read HashIP12(Ri) from P.offset[1..12];
    extract IP from HashIP12(Ri);
    read ASN(ASBRi) from P.Identification ;
    return (IP, ASN(ASBRi)) ;
end
```

Figure 6.6: Reconstruction at the Victim V

6.3 DRIM: Deterministic Router and Interface Marking

Our second approach is based on deterministic marking of the address of the router and the interface number through which the packet enters the network. The packet is marked by the first ingress edge router, which places both the marks. This marking information is not overwritten. Every outbound packet is marked and inbound packets are not marked. The architecture of the model is shown in Figure 6.3.

A router connects to two or more logical interfaces, represented by IP subnets or unnumbered point to point lines. Thus, it has at least one physical interface. Forwarding an IP datagram generally requires the router to choose the address and relevant interface of the next-hop router or the destination host (for the final hop) [12]. The packet is delivered locally and not considered for forwarding if the destination is an IP multicast address and one of the logical interfaces associated with the physical interface on which the packet arrived is a member of the destination multicast group. The router can identify the interface through which the packet reaches it. This identity is associated with a number called as interface number (IN). The router can also identify which interfaces are local to the internal network and which interfaces are connected to the external network. The various interfaces may

connect a router to a host, a LAN switch or another router. The marking algorithm ensures that only packets reaching the router from the internal set of local interfaces Ilocal will be marked.

Consider the architecture in Figure 5 with various hosts, switches, routers and interfaces. Attacker is a host connecting to the Internet through ingress edge router R1. Packets reach the first router R1 through interface I2. The other interfaces I1, I3, I4 and I5 of the router R1 are connected to a switch S1, a host and two routers R2, R3, respectively. The interface number I2 and a hash value of the router R1's IP address are marked deterministically into

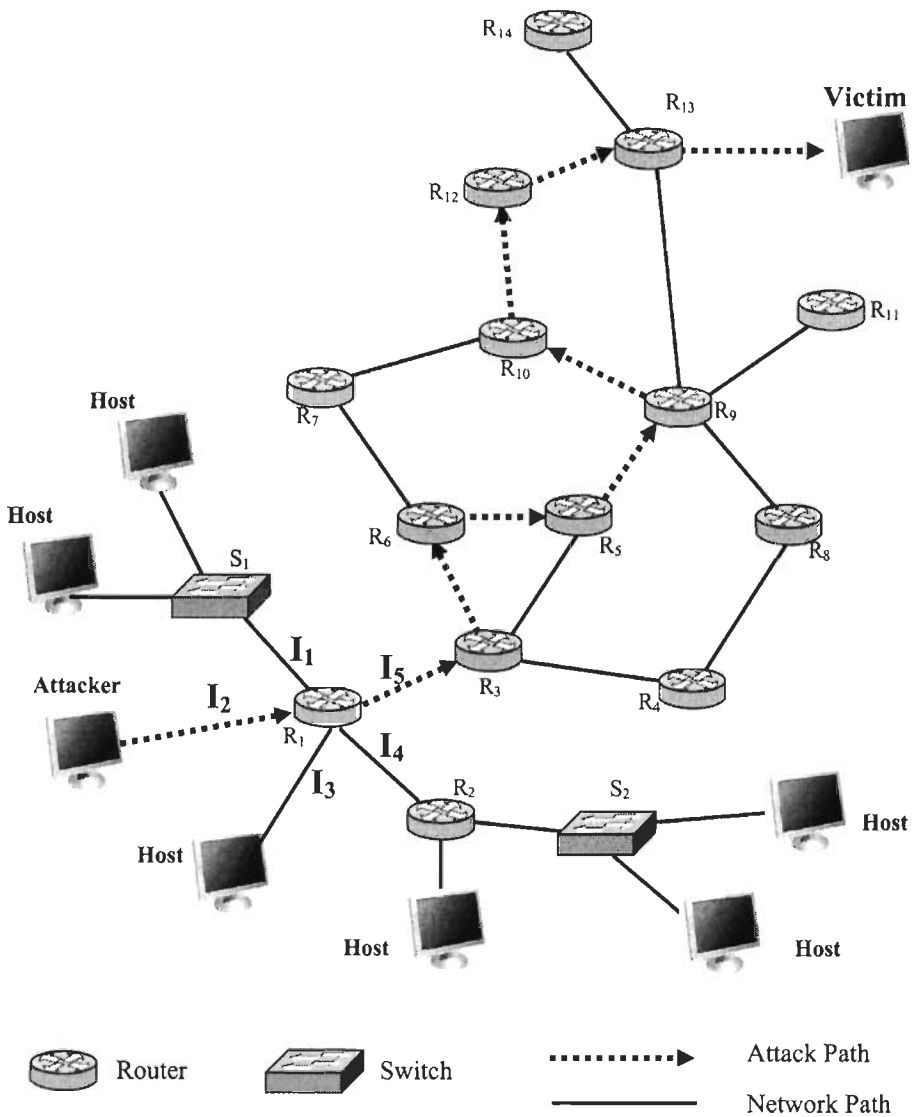

Figure 6.7. Deterministic Router and Interface Marking

each packet on the attack path. No other routers (R3 to R13) overwrite the mark. Packets arriving through interfaces I1, I2, and I3 only are marked by router R1 as they belong to the internal network. Packets arriving through I4 and I5 connected respectively to routers R2 and R3 are not marked. Each packet is marked only once with two values, interface number and a hash of the router IP address. A single packet is sufficient to detect the source as it contains the information about the interface and the router through which the packet entered the Internet.

6.3.1. Mark Information Encoding Scheme

The 16-bit ID field, 3-bit fragment flag field and 13-bit fragment offset field in the IP header are used to store the marking information. The mapping between fields in the IP header and the marking fields is shown in Figure 6.4.

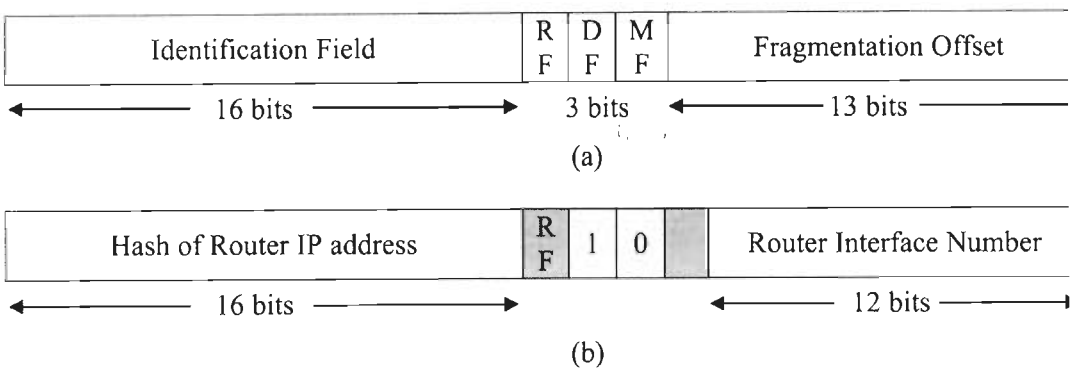

Figure 6.8 Mark encoding (a) fields in the IP header (b) fields overloaded for marking

The reason for overloading these 32 bits with marking information has been already explained in section 6.2. Identification field is used to store a 16-bit hash of the 32-bit IP address of the first ingress edge router. DERM [178] used the same marking scheme but included only the hash of router's address. The hash function used for converting the 32-bit IP address into a 16-bit value may result in some collisions and hence yield some false positives.

In our approach we also store the interface number in the least significant 12 bits of the offset field. A Cisco router can connect to a maximum number of 4096 interfaces. Hence, any router can remember each interface with a unique number using a maximum of 12 bits. The DF bit is set to 1 and MF bit to 0 to indicate packet marking and disable fragmentation.

6.3.2. Marking Scheme

The proposed marking mechanism enables the first ingress edge router to place two marks in each of the packet traversing it. The 16-bit hash of its 32-bit IP address and the number I_j , associated with the interface through which the packet has reached are marked. The router has information about all the interfaces connected to it and can recognize packets arriving on interfaces, I_{local}, connected to local components like hosts and switches which are internal to the network. The router copies the two marks into such packets before forwarding them. Packets arriving on other interfaces are forwarded without any change. The algorithm 6.4 illustrates the marking mechanism:

```
foreach outbound packet P reaching router Ri through
                                    interface Ij ε Ilocal do
    write HashIP16(Ri) into P.Identification;
    write Ij into P.offset[1..12];
    P.DF ← 1;     P.MF ← 0;
end
```

Figure 6.9: Marking of the router's address and interface number

6.3.3. Traceback Scheme

Each packet holds the information required to identify the first ingress router and the interface through which the packet reached the router. The identification field gives the 16-bit hash value of the routers IP address. The 12-bit value in the offset field indicates the interface number. The identification of the interface through which the attack packets enter the network will move us one step closer to the attacker as most techniques traceback to the first ingress edge router and stop. The algorithm 6.5 explains the traceback mechanism:

```
foreach attack packet P reaching Victim V do
    read HashIP16(Ri) from P.Identification;
    extract IP from HashIP16(Ri);
    read IN from P.offset[1..12];
    return (IP, IN);
end
```

Figure 6.10: Reconstruction at the Victim V

6.4. Performance Evaluation

The two proposed traceback approaches are compared with each other and their strengths and weaknesses are highlighted. The approaches are also compared individually with their related models. Simulations are performed to examine the feasibility of the approaches and validate them. ASDPM is compared with DRIM and the observations are shown in Table 6.1.

Table 6.1. Comparison of ASDPM and DRIM

ASDPM	DRIM
First internal router marks each packet with the 12-bit hash of its address and first AS boundary router marks the 16-bit AS number when the packet is leaving the AS	First ingress edge router marks each packet with the 16-bit hash of its address and 12-bit interface number through which it reached the router
12-bit hash of the 32-bit router address is stored	16-bit hash of the 32-bit router address is stored
Traceback identifies the source AS and the first internal router through which the packet entered the network using a single packet	Traceback identifies the first ingress router and the interface through which the packet reached the router using a single packet
Insider attacks can be easily identified if the AS number is not marked in the 16-bit identifier field	Identification of the interface from which the packet arrived narrows down search, even in case of networks involving NAT
There is a possibility for the attackers to spoof the packet with marking information as the flags can be set if they know the traceback mechanism	Mark spoofing can be totally avoided as the router identifies all local interfaces and can enforce marking even if flags are set in advance

6.4.1. Comparison with existing techniques

Belenky and Ansari [17] suggest 13 evaluation metrics for comparing the various traceback approaches. The metrics are level of involvement by the ISP , number of attacking packets needed for traceback, effect of partial deployment, processing overhead, bandwidth overhead (additional traffic generated during traceback), memory requirements, ease of

evasion, protection, scalability (number of attackers), infrastructure changes (number of functions needed to be implemented in the network devices), ability to handle major DDoS attacks, and ability to trace transformed packets. We compare the proposed approaches with related techniques against parameters specific to network forensics. ASDPM technique is compared with DPM [16, 18], ASEM [66] and ASSPT [113] as shown in Table 6.2. DRIM is compared with RIM [38, 39], DPM [16, 18] and DPMLS [41] as shown in Table 6.3.

6.4.2. Simulation Setup

Simulations were performed to examine the feasibility of the approaches and validate them using discrete event network simulator ns-2 [58] version 2.34 on ubuntu 9.10 operating system. The first approach is based on Autonomous System and a simple topology was generated for ns-2 using Boston university Representative Internet Topology gEnerator (BRITE) [136]. The second approach was simulated using topology as in Figure 6.3, which was manually generated in ns-2.

Changes are made in the source code of the ns-2 simulator to facilitate the requirements of proposed marking mechanisms. The IP header in ns-2 has only six fields, namely, src_, dst_, ttl_ fid_, prio_ and offset_. Additional variables for identification, flags_ and offset_ fields need to be added to facilitate marking. The offset_ variable has a different purpose in ns-2 simulator. The two 16-bit variables, fid_ and prio_ are placed for usage with IPv6 protocol. These variables can be used to accommodate 16-bit identification, 3-bit flag and 13-bit offset variables for our proposed models. New variables can also be created for identification, flags and offset fields and added to the IP header. These changes are to be made in the ip.h file, available in ns-2.34/common.

The important part of any simulation code for traceback model is the mark encoding mechanism at the ingress router. In ns-2, routers are represented by a node object. The router nodes have multiple links and can be easily distinguished from the host nodes, which have only a single link. The router node receives, processes and forwards the packet to the next node. The code for packet marking is added to the recv function of the classifier, which is responsible for making the forwarding decision at the nodes. It is located in the classifier.cc file in ns-2.34/classifier. The marking code is placed in this function so that the marks are placed before forwarding the packets to the next hop.

Table 6.2. Comparison of ASDPM with related techniques

Evaluation Metric	Deterministic Packet Marking (DPM)	AS based Edge Marking (ASEM)	AS level Single Packet Traceback (ASSPT)	AS based DPM (ASDPM)
Mechanism	Pure deterministic marking	AS based PPM	AS based packet logging	AS based DPM
Packets for traceback	7 packets	68 packets	Single packet	Single packet
Marking field length	34 bits in 2 consecutive packets	32 bits	Not applicable as packets are logged	32 bits
Processing overhead	Each packet is marked with either the first or last 16 bits of the edge router's address. Two consecutive packets carry the total IP address.	AS PATH information is calculated and updated at each AS ingress edge router. Three other marks are placed by the first marking router.	Each packet is logged at the border AS router and the logs can be queried during traceback	Two marks will be placed in each packet at the internal router and the AS boundary router
Storage overhead	A table is used for matching source with ingress addresses at the victim.	The hashing of router IP addresses can be calculated in advance and stored	Logging requires storage space at the AS border routers.	The 12-bit hash of router address is calculated in advance and stored

Table 6.2. Comparison of ASDPM with related techniques (continued)

Evaluation Metric	Deterministic Packet Marking (DPM)	AS based Edge Marking (ASEM)	AS level Single Packet Traceback (ASSPT)	AS based DPM (ASDPM)
Traceback overhead	Source addresses can be identified from the Ingress Table	Source address can be calculated from the hashed IP available in ID field	AS traceback server recursively queries upstream ASes till it identifies the egress router of the attack packet	Source address can be calculated from the hash value available in offset field and the ASN in the ID field
Infrastructure changes	One function is added in the networking devices	Two functions are added in the networking devices	Logging mechanism is implemented at the AS border routers	One function each is added in the internal and AS boundary routers
Scalability	Thousands of simultaneous attackers can be traced	Large scale attacks can be handled	No. of attackers handled are limited by storage space for packet logs	Any number of attackers can be handled
ISP involvement	Very limited as changes are less	Limited as AS level deployment is needed to calculate PATH information	Limited as AS level deployment is needed for logging and querying	Very limited as changes are less
False Positives	Few false positives	False positives are effectively suppressed	Bloom filters used for storing logs yield false positives	Hashing of the routers' address yields few false positives.
Extent of Traceback	Ingress router closest to the attacker	Entire path of ASes can be reconstructed	AS and the egress router closest to the attacker	AS and the internal router closest to the attacker

Table 6.3. Comparison of DRIM with related techniques

Evaluation Metric	**Router Interface Marking (RIM)**	**Deterministic Packet Marking (DPM)**	**DPM with Link Signatures (DPMLS)**	**Deterministic Router and Interface Marking (DRIM)**
Mechanism	Interface based PPM	Pure deterministic marking	Link Signatures based DPM	Interface based DPM
Packets for traceback	1 packet per attack path	7 packets	Single packet	Single packet
Marking field length	17 bits	34 bits in 2 consecutive packets	16 bits	31 bits
Processing overhead	Packets will be probabilistically marked with XOR and IID values and value in XOR field is updated otherwise	Each packet is marked only once with either the first or last 16 bits of the edge router's address	Each packet is marked by every router	Two marks will be placed in each packet by the first ingress edge router. No other routers mark the packet.
Storage overhead	A trace table is maintained with hop count, interface id and XOR value	A table is used for matching source with ingress addresses at the victim.	Each router stores the signatures of all the adjacent links.	The 16-bit hash of the router address is calculated in advance and stored

Table 6.3. Comparison of DRIM with related techniques (continued)

Evaluation Metric	Deterministic Packet Marking (DPM)	AS based Edge Marking (ASEM)	AS level Single Packet Traceback (ASSPT)	AS based DPM (ASDPM)
Traceback overhead	Records in trace table is sorted, grouped and associated with an attack path	Source addresses can be identified from the Ingress Table	Each router participates in traceback and 1.4% additional traffic is generated	Each packet can give the information of the ingress edge router and the interface number
Infrastructure changes	One function is added in the networking devices	One function is added in the networking devices	Two functions are added to the networking devices	One function is added in the networking devices
Scalability	Large number of attackers can be handled	Thousands of simultaneous attackers can be traced	Bandwidth overhead increases with number of attackers	Any number of attackers can be handled
ISP involvement	Limited	Very limited	Very limited	Very limited
False Positives	Few false positives as router interface IDs may not be unique	Few	Almost zero false positives	Hashing of the routers' address yields few false positives.
Extent of Traceback	Ingress router closest to the attacker	Ingress router closest to the attacker	Ingress router closest to the attacker	Ingress router and the interface closest to the attacker

6.4.3 Simulation for AS-DPM:

BRITE topology generator was used to generate a two level hierarchy of Autonomous Systems and Routers. A simple topology was created with 4 ASes and 5 routers in each AS. The lengths of planes specified by HS and LS were both set to 10. The number of nodes for AS was set to 4 and for routers, 5. The default value of Waxman was selected for model. The other option, BarabasiAlbert (BA) could also be used. Random orientation was used in placing the nodes in the plane. The number of links per node was set to the default value of 2. Incremental growth type was used for joining the nodes in the topology. A random edge connection method was used for interconnection of the router topologies. The inter and intra bandwidth distance was set to the default constant value. The topology was generated and imported to the tcl file format of ns-2. The network animator (nam) snapshot of the ns-2 is shown in Figure 6.5.

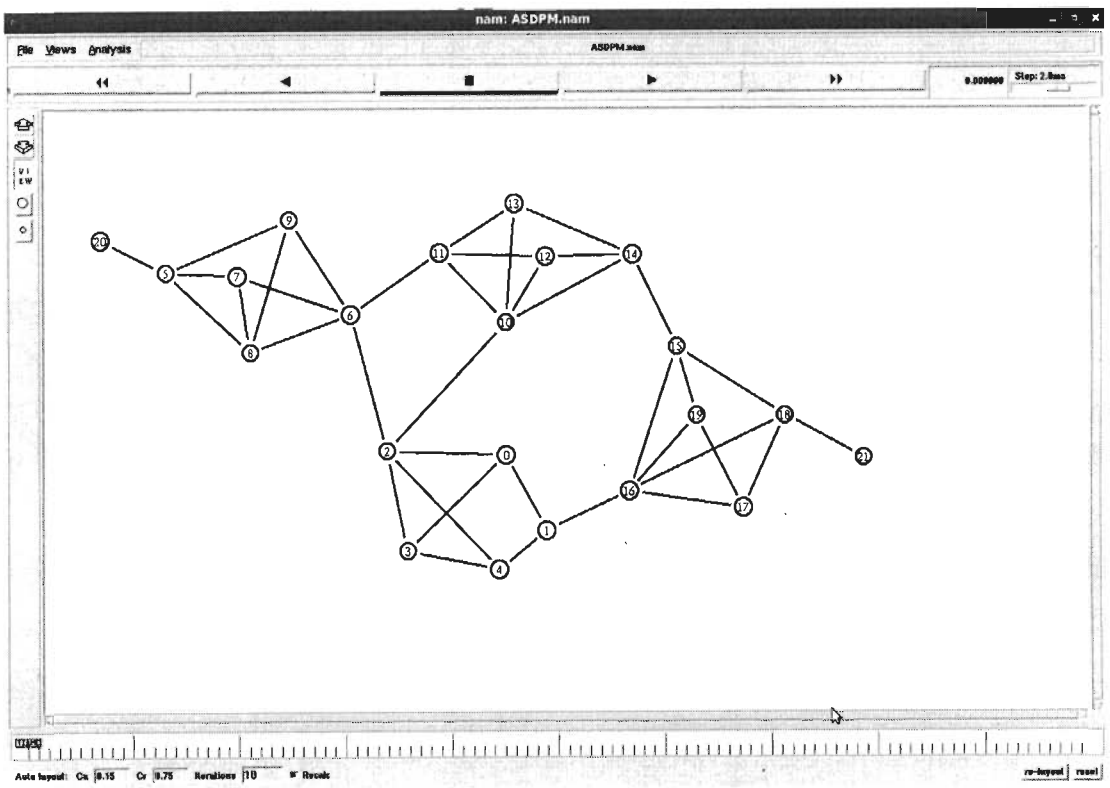

Figure 6.11. NAM output of topology generated in ns-2 using BRITE

Two new agents were created and placed in ns-2.34/apps. One UDP Agent is responsible for generating packets and is attached to the node designated as Attacker. The other UDP Sink Agent, attached to the node designated as Victim, receives the packets. Algorithms 6.1 and 6.2 are used for marking the packets at the internal router and the AS boundary router respectively. The code is placed in the recv function of the classifier. The

node.cc file available in ns-2.34/common is modified to differentiate between router node and AS node. All the packets arriving at the first router are marked with its node id. All other routers check the mark placed and ensure that packets traversing them from other routers will not be marked. The packets leaving the first AS boundary router are also marked in the same way. The other AS boundary routers do not mark the packet. The victim receives the packets and the algorithm 6.3 is used to extract the values. The simulation shows that ASDPM is able to perform the traceback even with a single packet.

6.4.3 Simulation for DRIM:

The topology for Figure 6.3 was manually generated in ns-2. Two new agents were created in same way as described in the previous section. The code of algorithm 6.4 is placed in the recv function of the classifier and is used for marking the packets. All the packets arriving at the first router are marked with its node id and the previous node id. The interface value is not available directly but the value of previous node specifies the direction of packet arrival. All other routers check the mark placed and ensure that packets traversing them from other routers will not be marked. The victim receives the packets and the algorithm 6.5 is used to extract the values. The simulation shows that DRIM is able to perform the traceback effectively with each packet. The network animator (nam) snapshot of the ns-2 is shown in Figure 6.6.

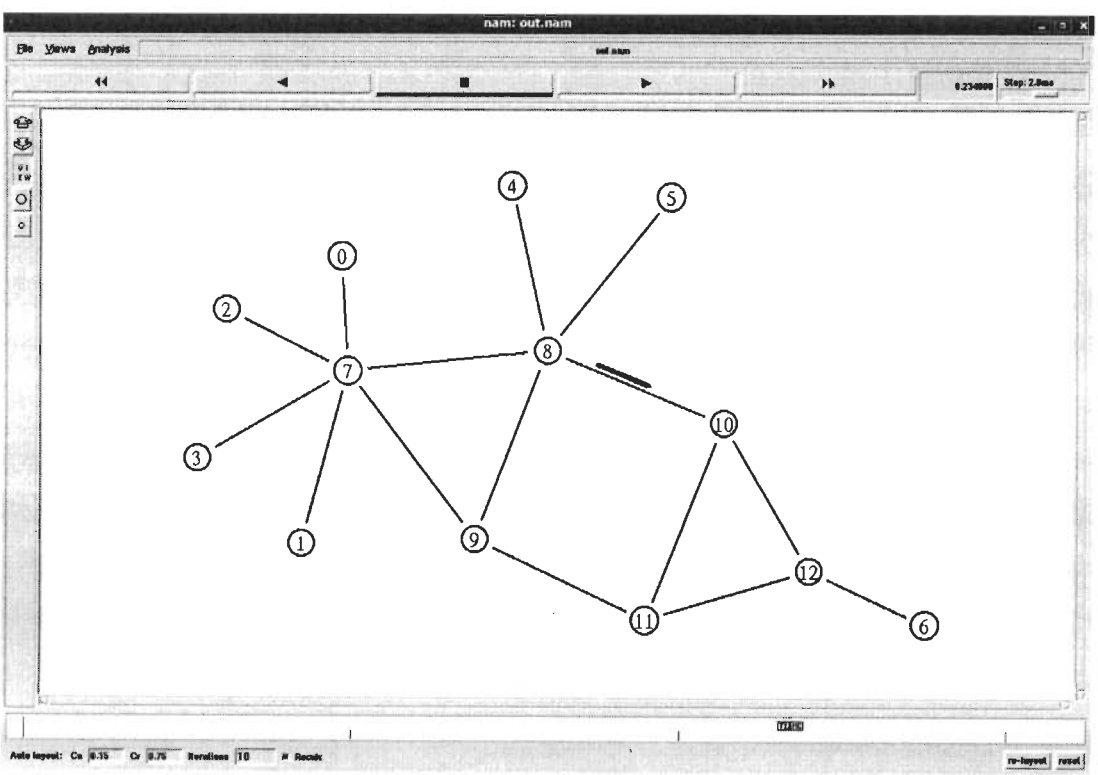

Figure 6.12. NAM output of topology generated in ns-2

6.5. Practical Considerations

We have discussed many of the performance issues in the previous section in comparison with related techniques. ASDPM and DRIM approaches were found to be advantageous in many metrics. We also discuss the two important considerations:

6.5.1. Ease of Evasion

An attacker can inject a packet marked with erroneous information into a stream of packets. If the attackers are aware of the marking technique being used, they can place misleading information in the fields being used to store the encoded marks. This is called mark spoofing. Both our proposed techniques overcome mark spoofing. In ASDPM, the first internal router deterministically marks every packet which reaches it and sets the first bit in the offset field to '1'. Even if the attacker tries to spoof the mark, the spoofed mark will be overwritten with a correct mark.

In DRIM, the first ingress edge router not only marks every packet reaching it with its hashed IP address. It also identifies the interface from which the packet is reaching it and includes the interface number in the encoded mark. The router can totally avoid mark spoofing by ensuring that all packets arriving from an interface connected to the internal network will be definitely marked. Any information placed by the attacker to mislead will definitely be overwritten. Hence there is 100% protection from mark evasion techniques.

6.5.2. Gradual Deployment

Gradual deployment of the proposed technique is possible with only partial routers along the path enforcing the marking mechanism. All the packets are marked when they reach the first internal router or ingress edge router and the AS boundary router (ASBR). The packets are not marked by any other router and the encoded mark information cannot be overwritten. The assumption that that marking mechanism is deployed in the attacker's network or autonomous system (AS), ensures that all the packets are marked once by the first router and AS boundary router. The other routers do not overwrite the mark and partial deployment does not affect the traceback in any way. However if the mechanism is not deployed in the attacker network, traceback may result is a large number of false positives. This situation is similar to all the deterministic packet marking techniques. Any network forensic investigation will attribute the attack to a legitimate user, if the attacker network cannot be accessed.

6.6 Summary

IP traceback problem for network forensics was examined. Two novel approaches based on deterministic packet marking were proposed, where the access point as close as possible to the attacker is recorded. In the ASDPM approach, traceback to the internal router within the source AS of the attacker is possible. The DRIM approach will move one step closer to the attacker by identifying the interface on which the packet reaches the router. A single packet is sufficient to detect the attack source as each packet carries the information to traceback to the attack source. This meets the requirement of network forensics, where the investigation of attacks may involve only few packets.

Chapter 7

Conclusion and Scope for Future Work

7.1 Conclusion

The existing research challenges in network forensic analysis have been identified in the literature survey in chapter 2. They are improving the various maturing phases of examination, analysis and investigation. In this thesis, we develop a network forensic framework which will capture and perform fusion of network and traffic data, classify, correlate, and analyze this data in order to investigate the source of attack, attribute the attack or crime while generating an incident response. The availability of a framework with thoroughly researched and practical solutions for the examination, data analysis and investigation phases will strengthen the mechanisms for network forensics. The major contributions of our work can be summarized as follows:

- A generic process model for network forensics with nine phases is proposed. This model is built over the existing models for digital forensics by considering phases specific to network forensics. The proposed model is generic as it handles network forensics both in real-time and post attack scenarios.

- The first five phases (including incident response) handle real-time network traffic. The preparation phase ensures the monitoring tools are in place. Detection phase helps in attack discovery and collection phase captures network packets ensuring integrity of data. A suitable incident response is generated based on the nature of attacks. A hash of the data is created and a copy is made in the preservation phase.
- The post attack investigation begins at the examination phase, where a copy of the packet capture (libpcap) file is given for investigation. The examination phase identifies the attack using features of the protocols used. The attack indicators are correlated. The analysis phase fuses inputs from various security sensors and validates the attack. It also classifies attack patterns using data mining, soft computing or statistical approaches. The investigation phase involves traceback and attribution. The final presentation phase results in the prosecution of the attacker.

- The proposed model is the first comprehensive model on network forensics covering most of the phases needed for analyzing network traffic. It is built on the well researched phases of computer forensics. The generic process model brings out the clear distinction of network forensics and other forms of digital forensics.
- A novel framework for network forensic analyses comprising of challenging phases, which do not have mature implementation, is also proposed. The framework will capture network traffic data, examine the protocol attributes with attack features, analyze data by performing fusion of alerts and attack information, in order to investigate the source of attack.
- The well researched phases like preparation, collection, detection, preservation, and presentation in the previous section. Standard techniques have been developed which are well tested by time. The remaining phases in our proposed generic process model, namely examination, analysis, and investigation are addressed.
- A practical approach is presented to identify network events at the application, transport and network layer of the TCP/IP protocol stack and correlate them with attacks. The events specific to distributed denial of service (DDoS) attacks, port scan attacks and cross-site Scripting (XSS) attacks are identified and correlated.

- A technique for performing data fusion of information from multiple security sensors is proposed. Tools with complementary and contradictory functions are identified. Data fusion was performed on the alert and attack information generated by these sensors so that the attack evidence is more accurate. D-S Theory of evidence was used to perform fusion of the alert information and the vaidity of the attack occurrence is ascertained. The proposed technique is validated by applying it on an attack dataset generated in the lab.
- Two novel approaches for network forensic traceback, Autonomous System based Deterministic Packet Marking (ASDPM) and Deterministic Router and Interface Marking (DRIM) – are proposed. ASDPM involves deterministic marking of each packet with the hash of the IP address of the first internal router and AS Number (ASN) of the AS boundary router when it is leaving the source AS. The DRIM approach involves deterministic marking of each packet with the hash value of the address of the first ingress router and the number of the interface through which the packet reached it.
- As part of investigation phase, the packets are deterministically marked by the first router to facilitate traceback to the source of attack packets. Simulations are performed using network simulator ns-2 to validate both the approaches.

7.2 Scope for future work

We started with the research challenges and attempted to improve existing solutions and propose a novel technique. A number of issues crop up in addressing the problems. Some of them are as follows:

- Correlation of attack features needs to be done using classification techniques in data mining and soft computing techniques.
- Data Fusion of attack alerts and logs can be automated. The process of generating bpa values can also be automated.
- Suitability of alternative statistical methods similar to D-S theory of evidence need to be investigated for validating the attack. Other combination of tools can also be tried for more accuracy.

- Fragmentation, though only 0.25 % of internet traffic, also needs to be facilitated while marking packets in order to account for many DDOS attack strategies.
- Another important task that remains for network forensic traceback is to move closer to the attacker in a network using NAT.
- The data sets used are locally generated in the academic environment. The proposed techniques can also be validated on real world datasets.
- As part of the ongoing work, a prototype implementation for incidence response will be developed. This will complete the framework of four maturing phases..

References

[1] AccessData, "SilentRunner Sentinel," 2011, Available: http://accessdata.com/downloads/media/SilentRunner_BROCHURE.pdf, [April 30, 2011]

[2] B. Al-Duwairi and M. Govindarasu, "Novel hybrid schemes employing packet marking and logging for IP traceback," *IEEE Transactions on Parallel and Distributed Systems,* vol. 17, no. 5, pp. 403-418, 2006.

[3] W. Alcorn, "Cross-site scripting viruses and worms - a new attack vector," *Network Security,* vol. 2006, no. 7, pp. 7-8, 2006.

[4] H. Aljifri, "IP traceback: a new denial-of-service deterrent?," *IEEE Security & Privacy,* vol. 1, no. 3, pp. 24-31, 2003.

[5] H. Aljifri, M. Smets, and A. Pons, "IP Traceback using header compression," *Computers & Security,* vol. 22, no. 2, pp. 136-151, 2003.

[6] A. Almulhem, "Network forensics: Notions and challenges," in *Proc. IEEE International Symposium on Signal Processing and Information Technology (ISSPIT' 09)*, Ajman, UAE 2009, pp. 463-466.

[7] A. Almulhem and I. Traore, "Experience with engineering a network forensics system," in *Proc. International Conference on Information Networking, Convergence in Broadband and Mobile Networking (ICOIN 05)*, Jeju Island, Korea, 2005, pp. 62-71.

[8] E. A. Anaya, M. Nakano-Miyatake, and H. M. Perez Meana, "Network forensics with Neurofuzzy techniques," in *Proc. 52nd IEEE International Midwest Symposium on Circuits and Systems (MWSCAS '09)*, Cancun, Mexico, 2009, pp. 848-852.

[9] E. Athanasopoulos, A. Krithinakis, and E. P. Markatos, "Hunting Cross-Site Scripting Attacks in the Network," in *Proc. Web 2.0 Security and Privacy (W2SP' 10)*, Oakland, California, USA, 2010, pp. 1-8.

[10] S. Axelsson, "Intrusion detection systems: A survey and taxonomy," Department of Computer Engineering, Chalmers University, Gothenburg, Technical Report 99-15, March 14, 2000.

[11] T. Baba and S. Matsuda, "Tracing network attacks to their sources," *IEEE Internet Computing,* vol. 6, no. 2, pp. 20-26, 2002.

[12] F. Baker, "RFC1812: Requirements for IP version 4 routers," 1995Available: http://tools.ietf.org/html/rfc1812, [April 30, 2011]

[13] V. Baryamureeba and F. Tushabe, "The enhanced digital investigation process model," in *Proc. 4th Digital Forensic Research Workshop*, Maryland, USA, 2004.

[14] T. Bass, "Intrusion detection systems and multisensor data fusion," *Communications of the ACM,* vol. 43, no. 4, pp. 99-105, 2000.

[15] N. L. Beebe and J. G. Clark, "A hierarchical, objectives-based framework for the digital investigations process," *Digital Investigation,* vol. 2, no. 2, pp. 147-167, 2005.

[16] A. Belenky and N. Ansari, "IP traceback with deterministic packet marking," *IEEE Communications Letters,* vol. 7, no. 4, pp. 162-164, 2003.

[17] A. Belenky and N. Ansari, "On IP traceback," *IEEE Communications Magazine,* vol. 41, no. 7, pp. 142-153, 2003.

[18] A. Belenky and N. Ansari, "On deterministic packet marking," *Computer Networks,* vol. 51, no. 10, pp. 2677-2700, 2007.

[19] S. M. Bellovin, "A look back at "security problems in the TCP/IP protocol suite," in *Proc. 20th Annual Computer Security Applications Conference (ACSAC' 04)*, Tucson, USA, 2004, pp. 229-249.

[20] S. M. Bellovin, M. Leech, and T. Taylor, "ICMP traceback messages," Internet Draft: draft-bellovin-itrace-00. txt 2000.

[21] H. Berghel, "The discipline of Internet forensics," *Communications of the ACM,* vol. 46, no. 8, pp. 15-20, 2003.

[22] J. Bhattacharya, R. Dass, V. Kapoor, and S. K. Gupta, "Utilizing Network Features for Privacy Violation Detection," in *Proc. First International Conference on Communication System Software and Middleware (Comsware' 06)*, New Delhi, India, 2006, pp. 1-10.

[23] R. Braganza, "Cross-site scripting - an alternative view," *Network Security,* vol. 2006, no. 9, pp. 17-20, 2006.

[24] J. Broadway, B. Turnbull, and J. Slay, "Improving the Analysis of Lawfully Intercepted Network Packet Data Captured for Forensic Analysis," in *Proc. Third International Conference on Availability, Reliability and Security (ARES' 08)*, Barcelona, Spain, 2008, pp. 1361-1368.

[25] V. Broucek and P. Turner, "Forensic computing: Developing a conceptual approach for an emerging academic discipline," in *Proc. 5th Australian Security Research Symposium*, Perth, Australia, 2001, pp. 55-68.

[26] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," in *Proc. 14th Systems Administration Conference (LISA 2000)*, New Orleans, Louisiana, USA, 2000, pp. 319-327.

[27] M. Bykova, S. Ostermann, and B. Tjaden, "Detecting network intrusions via a statistical analysis of network packet characteristics," in *Proc. 33rd Southeastern Symposium on System Theory*, Ohio, USA, 2001, pp. 309-314.

[28] B. Carrier, "Defining Digital Forensic Examination and Analysis Tools Using Abstraction Layers," *International Journal of Digital Evidence,* vol. 1, no. 4, pp. 1-12, 2003.

[29] B. Carrier and C. Shields, "The session token protocol for forensics and traceback," *ACM Transactions on Information and System Security (TISSEC),* vol. 7, no. 3, pp. 333-362, 2004.

[30] 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, 2003.

[31] E. Casey, "Network traffic as a source of evidence: tool strengths, weaknesses, and future needs," *Digital Investigation,* vol. 1, no. 1, pp. 28-43, 2004.

[32] E. Casey, "Case study: Network intrusion investigation - lessons in forensic preparation," *Digital Investigation,* vol. 2, no. 4, pp. 254-260, 2005.

[33] E. Casey, C. Daywalt, A. Johnston, and T. Maguire, "Network Investigations," in *Handbook of digital forensics and investigation*, E. Casey, Ed. London, UK: Elsevier Academic Press, 2010, pp. 437-516.

[34] E. Casey and G. Palmer, "The investigative process," in *Digital evidence and Computer crime: Forensic science, Computers and the Internet*, E. Casey, Ed. London: Academic Press, 2004.

[35] A. Castelucio, A. Ziviani, and R. Salles, "An AS-level overlay network for IP traceback," *IEEE Network,* vol. 23, no. 1, pp. 36-41, 2009.

[36] V. Chatzigiannakis, G. Androulidakis, K. Pelechrinis, S. Papavassiliou, and V. Maglaris, "Data fusion algorithms for network anomaly detection: classification and evaluation," in *Proc. Third International Conference on Networking and Services (ICNS 07)*, Athens, Greece, 2007, pp. 50-56.

[37] L. Chen, Z. Li, and C. Gao, "Automated Analysis of Multi-Source Logs for Network Forensics," in *Proc. First International Workshop on Education Technology and Computer Science (ETCS '09)*, 2009, pp. 660-664.

[38] R. Chen, J.-M. Park, and M. Randolph, "RIM: Router Interface Marking for IP Traceback," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM '06)*, San Francisco, California, USA, 2006, pp. 1-5.

[39] R. Chen, J. M. Park, and R. Marchany, "A Divide-and-Conquer Strategy for Thwarting Distributed Denial-of-Service Attacks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 18, no. 5, pp. 577-588, 2007.

[40] B.-C. Cheng and H. Chen, "Quality Assurance for Evidence Collection in Network Forensics," in *Information Security Applications*. vol. 4298, J. Lee, O. Yi, and M. Yung, Eds. Berlin / Heidelberg: Springer, 2007, pp. 121-132.

[41] S. Ciardhuáin, "An extended model of cybercrime investigations," *International Journal of Digital Evidence,* vol. 3, no. 1, pp. 1-22, 2004.

[42] F. Cohen, "Internet holes -- Part 2: Packet fragmentation attacks," *Network Security,* vol. 1995, no. 9, pp. 14-16, 1995.

[43] M. I. Cohen, "PyFlag - An advanced network forensic framework," *Digital Investigation,* vol. 5, no. Supplement 1, pp. S112-S120, 2008.

[44] M. I. Cohen, "Source attribution for network address translated forensic captures," *Digital Investigation,* vol. 5, no. 3-4, pp. 138-145, 2009.

[45] G. Combs, "Wireshark," 2007, Available: http://www.wireshark.org, [April, 30, 2011]

[46] V. Corey, C. Peterman, S. Shearin, M. S. Greenberg, and J. Van Bokkelen, "Network forensics analysis," *IEEE Internet Computing,* vol. 6, no. 6, pp. 60-66, 2002.

[47] CWE-SANS, "2010 CWE/SANS Top 25 Most Dangerous Software Errors," 2010, Available: http://cwe.mitre.org/top25, http://www.sans.org/top25-software-errors, [April 30, 2011]

[48] T. E. Daniels, "A functional reference model of passive systems for tracing network traffic," *Digital Investigation,* vol. 1, no. 1, pp. 69-81, 2004.

[49] M. Davis, G. Manes, and S. Shenoi, "A Network-Based Architecture for Storing Digital Evidence," in *Advances in Digital Forensics*. vol. 194, M. Pollitt and S. Shenoi, Eds. Boston: Springer 2005, pp. 33-42.

[50] D. Dean, M. Franklin, and A. Stubblefield, "An algebraic approach to IP traceback," *ACM Transactions on Information and System Security (TISSEC),* vol. 5, no. 2, pp. 119-137, 2002.

[51] O. Demir, P. Ji, and J. Kim, "Session Based Packet Marking and Auditing for Network Forensics," *International Journal of Digital Evidence,* vol. 6, no. 1, pp. 1-15, 2007.

[52] T. T. P. Desk, "New cyber attack targets Iran," 2011, Tehran, Tehran TimesAvailable: http://www.tehrantimes.com/PDF/11135/11135-1.pdf, [April 30, 2011]

[53] F. deSouza, "Safeguarding critical infrastructure from the next Stuxnet " 2011, Network World, Available: http://www.networkworld.com/news/tech/2011/042711-infrastructure-stuxnet-safeguard.html, [April 30, 2011]

[54] G. A. Di Lucca, A. R. Fasolino, M. Mastoianni, and P. Tramontana, "Identifying cross site scripting vulnerabilities in Web applications," in *Proc. Sixth IEEE International Workshop on Web Site Evolution (WSE' 04)*, Chicago, IL, USA, 2004, pp. 71-80.

[55] EMC, "NetWitness," 2011, Available: http://www.netwitness.com/products-services/investigator, [April 30, 2011]

[56] B. Endicott-Popovsky, D. A. Frincke, and C. A. Taylor, "A theoretical framework for organizational network forensic readiness," *Journal of Computers,* vol. 2, no. 3, pp. 1–11, 2007.

[57] J. Esteban, A. Starr, R. Willetts, P. Hannah, and P. Bryanston-Cross, "A review of data fusion models and architectures: towards engineering guidelines," *Neural Computing & Applications,* vol. 14, no. 4, pp. 273-281, 2005.

[58] K. Fall and K. Varadhan, "Network Simulator 2 (NS2)," 2009, The VINT ProjectAvailable: http://www.isi.edu/nsnam, [April 30, 2011]

[59] N. Falliere, L. O. Murchu, and E. Chien, "W32. Stuxnet Dossier," 2011, Symantec Security Response, Available: http://www.symantec.com/en/ca/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf, [April 30, 2011]

[60] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, and T. Berners-Lee, "RFC 2616: Hypertext Transfer Protocol - HTTP/1.1," 1999, Defense Advanced Research Projects Agency, Available: http://www.ietf.org/rfc/rfc2616.txt, [April 30, 2011]

[61] B. A. Forouzan, *TCP/IP protocol Suite*, Third ed. California: McGraw Hill Publications, 2006.

[62] D. Forte, "Fragmentation Attacks: Protection Tools and Techniques," *Network Security,* vol. 2001, no. 12, pp. 12-13, 2001.

[63] D. Forte, "The Future of Computer and Network Forensics," *Network Security,* vol. 2002, no. 10, pp. 13-15, 2002.

[64] J. Gadge and A. A. Patil, "Port scan detection," in *Proc. 16th IEEE International Conference on Networks (ICON' 08)*, New Delhi, India, 2008, pp. 1-6.

[65] Z. Gao and N. Ansari, "Tracing cyber attacks from the practical perspective," *IEEE Communications Magazine,* vol. 43, no. 5, pp. 123-131, 2005.

[66] Z. Gao and N. Ansari, "A practical and robust inter-domain marking scheme for IP traceback," *Computer Networks,* vol. 51, no. 3, pp. 732-750, 2007.

[67] J. Garcia-Alfaro and G. Navarro-Arribas, "A Survey on Detection Techniques to Prevent Cross-Site Scripting Attacks on Current Web Applications," in *Critical Information Infrastructures Security.* vol. 5141, J. Lopez and B. Hämmerli, Eds. Berlin / Heidelberg: Springer 2008, pp. 287-298.

[68] S. Garfinkel, "Network forensics: Tapping the internet," 2002, Sebastopol, CA, O'Reilly Network, Available: http://www.oreillynet.com/pub/a/network/2002/04/26/nettap.html, [April 30, 2011]

[69] S. L. Garfinkel, "Digital forensics research: The next 10 years," *Digital Investigation,* vol. 7, no. Supplement 1, pp. S64-S73, 2010.

[70] M. B. Ghorbel, M. Talbi, and M. Mejri, "Specification and Detection of TCP/IP Based Attacks Using the ADM-Logic," in *Proc. The Second International Conference on Availability, Reliability and Security (ARES' 07)*, Vienna, Austria, 2007, pp. 206-212.

[71] G. Giacinto, F. Roli, and L. Didaci, "Fusion of multiple classifiers for intrusion detection in computer networks," *Pattern Recognition Letters,* vol. 24, no. 12, pp. 1795-1803, 2003.

[72] P. Giura and N. Memon, "NetStore: An Efficient Storage Infrastructure for Network Forensics and Monitoring," in *Recent Advances in Intrusion Detection.* vol. 6307, S. Jha, R. Sommer, and C. Kreibich, Eds. Berlin / Heidelberg: Springer, 2010, pp. 277-296.

[73] C. Gong and K. Sarac, "A More Practical Approach for Single-Packet IP Traceback using Packet Logging and Marking," *IEEE Transactions on Parallel and Distributed Systems,* vol. 19, no. 10, pp. 1310-1324, 2008.

[74] J. Govil, J. Govil, N. Kaur, and H. Kaur, "An examination of IPv4 and IPv6 networks : Constraints and various transition mechanisms," in *Proc. IEEE Southeastcon 08*, Huntsville, Alabama, USA, 2008, pp. 178-185.

[75] Y. Guan, "Network Forensics," in *Computer and Information Security Handbook*, R. V. John, Ed. Boston: Morgan Kaufmann, 2009, pp. 339-347.

[76] Y. Guan and L. Zhang, "Attack Traceback and Attribution," in *Wiley Handbook of Science and Technology for Homeland Security*, J. G. Voeller, Ed. New York: John Wiley & Sons, Inc., 2008.

[77] R. Guo, T. Cao, and X. Luo, "Application Layer Information Forensics Based on Packet Analysis," in *Proc. International Conference of Information Science and Management Engineering (ISME' 10)*, Xian, China, 2010, pp. 206-209.

[78] R. Hadjidj, M. Debbabi, H. Lounis, F. Iqbal, A. Szporer, and D. Benredjem, "Towards an integrated e-mail forensic analysis framework," *Digital Investigation,* vol. 5, no. 3-4, pp. 124-137, 2009.

[79] I. Hamadeh and G. Kesidis, "A taxonomy of internet traceback," *International Journal of Security and Networks,* vol. 1, no. 1, pp. 54-61, 2006.

[80] S. Hansman and R. Hunt, "A taxonomy of network and computer attacks," *Computers & Security,* vol. 24, no. 1, pp. 31-43, 2005.

[81] B. Harris and R. Hunt, "TCP/IP security threats and attack methods," *Computer Communications,* vol. 22, no. 10, pp. 885-897, 1999.

[82] J. S. Haugdahl, "Network Forensics: Methods, Requirements, and Tools," 2007, Bitcricket LLC, Available: http://www.bitcricket.com/downloads/Network%20Forensics.pdf, [April, 30, 2011]

[83] J. Hawkinson and T. Bates, "RFC 1930: Guidelines for creation, selection, and registration of an Autonomous System (AS)," 1996, Available: http://tools.ietf.org/html/rfc1930, [April 30, 2011]

[84] P. Herman, "The tcpstat tool," 2003, Available: http://www.frenchfries.net/paul/tcpstat, [April 30, 2011]

[85] S. Hettich and S. D. Bay, "KDD Cup 1999 Dataset," 1999, Irvine, CA, The UCI KDD Archive, Available: http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html,

[86] K. Himanshu, B. Jim, B. Mehedi, F. Mike, W. Von, and B. Randy, "Palantir: a framework for collaborative incident response and investigation," in *Proc. Proc. of the 8th Symposium on Identity and Trust on the Internet*, Gaithersburg, Maryland, 2009, pp. 38-51.

[87] HP, "ProCurve Secure Router OS Firewall," 2010, Available: ftp://hp.com/pub/networking/software/A-C04-Firewall.pdf, [April 30, 2011]

[88] R. Hunt and J. Slay, "Achieving critical infrastructure protection through the interaction of computer security and network forensics," in *Proc. Eighth Annual International Conference on Privacy Security and Trust (PST '10)*, Ottawa, Ontario, Canada, 2010, pp. 23-30.

[89] IANA, "16-bit Autonomous System Numbers," 2008, Available: http://www.iana.org/assignments/as-numbers/as-numbers.xml, [April 30, 2011]

[90] R. S. C. Ieong, "FORZA - Digital forensics investigation framework that incorporate legal issues," *Digital Investigation,* vol. 3, no. Supplement 1, pp. 29-36, 2006.

[91] V. Igure and R. Williams, "Taxonomies of attacks and vulnerabilities in computer systems," *IEEE Communications Surveys & Tutorials,* vol. 10, no. 1, pp. 6-19, 2008.

[92] Infosecurity, "Network forensics helps bolsters confidence in cloud computing security," 2010, Available: http://www.infosecurity-us.com/view/13252/network-forensics-helps-bolsters-confidence-in-cloud-computing-security/, [April 30, 2011]

[93] ISO, "ISO/IEC 27001:2005 Information technology -- Security techniques -- Information security management systems -- Requirements," 2005, International Organization for Standardization, Available: http://www.iso.org/iso/catalogue_detail.htm?csnumber=42103, [April 30, 2011]

[94] V. Jacobson, C. Leres, and S. McCanne, "libpcap," 1994, Berkeley, CA, LBNL, Available: http://wiki.wireshark.org/Development/LibpcapFileFormat, [April 30, 2011]

[95] V. Jacobson, C. Leres, and S. McCanne, "Pcap and Libpcap," 2003, Berkeley, CA, LBNL, University of California, Available: http://www.tcpdump.org/pcap3_man.html, [April 30, 2011]

[96] V. Jacobson, C. Leres, and S. McCanne, "tcpdump," 2009, Berkeley, CA, LBNL, University of CaliforniaAvailable: http://www.tcpdump.org/,

[97] A. Jain, G. Andreys, and G. Sivakumar, "Intelligent real-time reactive Network Management," in *European Conference on Computer Network Defence (EC2ND' 05)*, A. Blyth, Ed. London: Springer 2006, pp. 61-72.

[98] P. Jayashree and K. S. Easwarakumar, "Network Anomaly Detector System for Active Networks," *International Journal of Imaging Science and Engineering,* vol. 2, no. 2, pp. 149-153, 2008.

[99] G. Jin and J. Yang, "Deterministic packet marking based on redundant decomposition for IP traceback," *IEEE Communications Letters,* vol. 10, no. 3, pp. 204-206, 2006.

[100] W. X. Jing and X. Y. Lin, "IP Traceback Based on Deterministic Packet Marking and Logging," in *Proc. Eighth International Conference on Embedded Computing, Scalable Computing and Communications (SCALCOM-EMBEDDEDCOM '09)*, Dalian, China, 2009, pp. 178-182.

[101] Y.-N. Jing, P. Tu, X.-P. Wang, and G.-D. Zhang, "Distributed-log-based scheme for IP traceback," in *Proc. The Fifth International Conference on Computer and Information Technology (CIT' 05)*, Shanghai, China, 2005, pp. 711-715.

[102] W. John and T. Olovsson, "Detection of malicious traffic on back-bone links via packet header analysis," *Campus-Wide Information Systems,* vol. 25, no. 5, pp. 342-358, 2008.

[103] W. John and S. Tafvelin, "Analysis of internet backbone traffic and header anomalies observed," in *Proc. 7th ACM SIGCOMM Conference on Internet measurement*, San Diego, California, USA, 2007.

[104] M. Johns, B. Engelmann, and J. Posegga, "XSSDS: Server-Side Detection of Cross-Site Scripting Attacks," in *Proc. Annual Computer Security Applications Conference (ACSAC' 08)*, Anaheim, California, USA, 2008, pp. 335-344.

[105] K. Johnson, "BASE - Basic Analysis and Security Engine," 2009, Available: http://base.secureideas.net/, [April 30, 2011]

[106] A. Johnston and J. Reust, "Network intrusion investigation - Preparation and challenges," *Digital Investigation,* vol. 3, no. 3, pp. 118-126, 2006.

[107] G. Kessler, "Stuxnet worm possibly made to cripple Iran centrifuges," 2010, Washington Post, Available: http://www.washingtonpost.com/wp-dyn/content/article/2010/11/15/AR2010111506768.html, [April 30, 2011]

[108] T. Kilpatrick, J. Gonzalez, R. Chandia, M. Papa, and S. Shenoi, "An Architecture for SCADA Network Forensics," in *Advances in Digital Forensics II*. vol. 222, M. Olivier and S. Shenoi, Eds. Boston: Springer 2006, pp. 273-285.

[109] T. Kilpatrick, J. Gonzalez, R. Chandia, M. Papa, and S. Shenoi, "Forensic analysis of SCADA systems and networks," *International Journal of Security and Networks,* vol. 3, no. 2, pp. 95-102, 2008.

[110] J.-S. Kim, M. Kim, and B.-N. Noh, "A Fuzzy Expert System for Network Forensics," in *Computational Science and its Applications*. vol. 3043, A. Laganà, M. L. Gavrilova, V. Kumar, Y. Mun, C. J. K. Tan, and O. Gervasi, Eds. Berlin / Heidelberg: Springer 2004, pp. 175-182.

[111] T.-H. Kim and H.-Y. Kwon, "Applying Security Engineering to Build Security Countermeasures: An Introduction," in *Applied Parallel Computing. State of the Art in Scientific Computing*. vol. 3732, J. Dongarra, K. Madsen, and J. Wasniewski, Eds. Berlin / Heidelberg: Springer, 2006, pp. 957-963.

[112] E. Kirda, C. Kruegel, G. Vigna, and N. Jovanovic, "Noxes: a client-side solution for mitigating cross-site scripting attacks," in *Proc. 2006 ACM Symposium on Applied Computing (SAC' 06)*, Dijon, France, 2006, pp. 330-337.

[113] T. Korkmaz, C. Gong, K. Sarac, and S. G. Dykes, "Single packet IP traceback in AS-level partial deployment scenario," *International Journal of Security and Networks,* vol. 2, no. 1, pp. 95-108, 2007.

[114] U. Lamping, R. Sharpe, and E. Warnicke, *Wireshark User's Guide: for Wireshark 1.5*: Wireshark Foundation, 2008.

[115] B. Laurie, "Network Forensics," *ACM Queue,* vol. 2, no. 4, pp. 50-56, 2004.

[116] S.-Y. Lee, M.-C. Shin, J.-S. Cha, and T.-H. Kim, "Threat Description for Developing Security Countermeasure," in *Advances in Multimedia Information Processing (PCM' 04)*. vol. 3331, K. Aizawa, Y. Nakamura, and S. I. Satoh, Eds. Berlin / Heidelberg: Springer 2005, pp. 548-555.

[117] S. C. Lee and C. Shields, "Tracing the source of network attack: A technical, legal and societal problem," in *Proc. IEEE Workshop on Information Assurance and Security*, West Point, NY, 2001, pp. 239-246.

[118] B. Leiner and Y. Rekhter, "RFC1560: The MultiProtocol Internet," 1993Available: http://tools.ietf.org/html/rfc1560, [April 30, 2011]

[119] N. Liao, S. Tian, and T. Wang, "Network forensics based on fuzzy logic and expert system," *Computer Communications,* vol. 32, no. 17, pp. 1881-1892, 2009.

[120] T. V. Lillard, C. P. Garrison, C. A. Schiller, and J. Steele, "The Future of Network Forensics," in *Digital Forensics for Network, Internet, and Cloud Computing* Boston: Syngress, 2010, pp. 341-347.

[121] T. V. Lillard, C. P. Garrison, C. A. Schiller, and J. Steele, "What Is Network Forensics?," in *Digital Forensics for Network, Internet, and Cloud Computing* Boston: Syngress, 2010, pp. 3-20.

[122] C. Lin, L. Zhitang, and G. Cuixia, "Automated Analysis of Multi-Source Logs for Network Forensics," in *Proc. First International Workshop on Education Technology and Computer Science (ETCS '09)*, Wuhan, China, 2009, pp. 660-664.

[123] I. Lin and T.-H. Lee, "Robust and Scalable Deterministic Packet Marking Scheme for IP Traceback," in *Proc. IEEE Global Telecommunications Conference (GLOBECOM '06)*, San Francisco, California, USA, 2006, pp. 1-6.

[124] M. Liu, Q. Zhang, H. Zhao, and D. Yu, "Network Security Situation Assessment Based on Data Fusion," in *Proc. First International Workshop on Knowledge Discovery and Data Mining (WKDD 08)*, Adelaide, Australia, 2008, pp. 542-545.

[125] Z. Liu and D. Feng, "Incremental Fuzzy Decision Tree-Based Network Forensic System," in *Computational Intelligence and Security*. vol. 3802, Y. Hao, J. Liu, Y.-P. Wang, Y.-m. Cheung, H. Yin, L. Jiao, J. Ma, and Y.-C. Jiao, Eds. Berlin / Heidelberg: Springer 2005, pp. 995-1002.

[126] M. Locasto, M. Burnside, and A. Keromytis, "Online Network Forensics for Automatic Repair Validation," in *Advances in Information and Computer Security*. vol. 5312, K. Matsuura and E. Fujisaki, Eds. Berlin / Heidelberg: Springer, 2008, pp. 136-151.

[127] C. Low, "ICMP Attacks Illustrated," 2003, SANS Institute Reading Room, Available: http://www.sans.org/reading_room/whitepapers/threats/icmp-attacks-illustrated_477, [April, 30, 2011]

[128] G. Lyon, "Nmap - Network Mapper," 2010, Available: http://www.nmap.org/, [April 30, 2011]

[129] P. Mag, "Twitter Hit with 'Mouse Over' XSS Attack," 2010, Available: http://www.pcmag.com/article2/0,2817,2369438,00.asp, [April 30, 2011]

[130] M. Mahoney and P. K. Chan, "PHAD: Packet header anomaly detection for identifying hostile network traffic," Florida Institute of Technology, Melbourne, FL, USA, Technical Report CS-2001-04, 2001.

[131] K. Mandia, C. Prosise, and M. Pepe, *Incident Response and Computer Forensics*, Second ed. California: McGraw-Hill Osborne Media, 2003.

[132] A. Mankin, D. Massey, W. Chien-Lung, S. F. Wu, and Z. Lixia, "On design and evaluation of "intention-driven" ICMP traceback," in *Proc. Tenth International Conference on Computer Communications and Networks (ICCCN 01)*, Arizona, USA, 2001, pp. 159-165.

[133] A. M. Marques, "Zenmap - Graphical Nmap frontend " 2010, Available: http://www.nmap.org/zenmap, [April 30, 2011]

[134] A. Matrosov, E. Rodionov, D. Harley, and J. Malcho, "Stuxnet Under the Microscope," 2011, ESET, Available: http://www.eset.com/us/resources/white-papers/Stuxnet_Under_the_Microscope.pdf, [April 30, 2011]

[135] V. M. Matthew, "Network traffic anomaly detection based on packet bytes," in *Proc. ACM Symposium on Applied Computing (SAC' 03)*, Melbourne, Florida, 2003, pp. 346-350.

[136] A. Medina, A. Lakhina, I. Matta, and J. Byers, "BRITE: Boston university Representative Internet Topology gEnerator," 2001, Boston UniversityAvailable: http://www.cs.bu.edu/brite, [April 30, 2011]

[137] L. D. Merkle, "Automated network forensics," in *Proc. Genetic and Evolutionary Computation Conference (GECCO 08)*, Atlanta, Georgia, USA, 2008, pp. 1929-1932.

[138] C. Meyler and I. Sutherland, "A Flexible, Open Source Software Architecture for Network-Based Forensic Computing & Intelligence Gathering," in *European Conference on Computer Network Defence (EC2ND' 05)*, A. Blyth, Ed. London: Springer, 2006, pp. 253-262.

[139] E. Mills and R. Langner, "Ralph Langner on Stuxnet, copycat threats (Q&A)," 2011, CNET, Available: http://news.cnet.com/8301-27080_3-20061256-245.html, [April 30, 2011]

[140] S. Mitropoulos, D. Patsos, and C. Douligeris, "Network forensics: towards a classification of traceback mechanisms," in *Proc. Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm' 05)*, Athens, Greece, 2005, pp. 9-16.

[141] J. Moy, "RFC1247: OSPF Version 2," 1991, Available: http://tools.ietf.org/pdf/rfc1583, [April 30, 2011]

[142] S. Mukkamala and A. H. Sung, "Identifying significant features for network forensic analysis using artificial intelligent techniques," *International Journal of Digital Evidence,* vol. 1, no. 4, pp. 1-17, 2003.

[143] A. Nagesh, "Distributed Network Forensics using JADE Mobile Agent Framework," M. S. thesis, Dept of Computing Studies, Arizona State University, Mesa, AZ, 2006.

[144] S. Naqvi, P. Massonet, and A. Arenas, "Scope of Forensics in Grid Computing – Vision and Perspectives," in *ISPA' 06 Workshop on Frontiers of High Performance Computing and Networking.* vol. 4331, G. Min, B. Di Martino, L. Yang, M. Guo, and G. Rünger, Eds. Berlin / Heidelberg: Springer 2006, pp. 964-970.

[145] netForensics, "Security Compliance Management," 2010, Available: http://www.netforensics.com/compliance/, [April 30, 2011]

[146] Netresec, "NetworkMiner," 2011M M Available: http://www.netresec.com/?page=NetworkMiner, [April 30, 2011]

[147] NetScout, "InfiniStream Console," 2011Available: http://www.netscout.com/products/service_provider/nSAS/sniffer_analysis/Pages/InfiniStream_Console.aspx, [April 30, 2011]

[148] S. Networks, "Unveiling the Security Illusion " 2010Available: http://www.soleranetworks.com/resources/wp_need_for_net_forens_web.pdf, [April 30, 2011]

[149] S. Networks, "DS 5200 Integrated Network Forensics Appliance," 2011Available: http://www.soleranetworks.com/resources/datasheet5200.pdf, [April 30, 2011]

[150] B. J. Nikkel, "Domain name forensics: a systematic approach to investigating an internet presence," *Digital Investigation,* vol. 1, no. 4, pp. 247-255, 2004.

[151] B. J. Nikkel, "A portable network forensic evidence collector," *Digital Investigation,* vol. 3, no. 3, pp. 127-135, 2006.

[152] B. J. Nikkel, "An introduction to investigating IPv6 networks," *Digital Investigation,* vol. 4, no. 2, pp. 59-67, 2007.

[153] Niksun, "NetDetector," 2011, A,vailable: http://www.niksun.com/collateral/NIKSUNDatasheet_NetDetector_Alpine_0511.pdf, [April 30, 2011]

[154] I. V. Onut and A. A. Ghorbani, "A feature classification scheme for network intrusion detection," *International Journal of Network Security,* vol. 5, no. 1, pp. 1-15, 2007.

[155] C. Onwubiko, "Data Fusion in Security Evidence Analysis," in *Proc. 3rd Conference on Advances in Computer Security and Forensics (ACSF' 08)*, Liverpool, UK, 2008, pp. 1-6.

[156] A. Otaka, T. Takagi, and O. Takahashi, "Network Forensics on Mobile Ad-Hoc Networks," in *Knowledge-Based Intelligent Information and Engineering Systems*. vol. 5179, I. Lovrek, R. Howlett, and L. Jain, Eds. Berlin / Heidelberg: Springer 2008, pp. 175-182.

[157] OWASP, "OWASP Top 10 Web Application Security Risks for 2010," 2010, Available: http://www.owasp.org/index.php/Category: OWASP_Top_Ten_ Project, [April 30, 2011]

[158] G. Palmer, "Digital Forensic Science in Networked Environments (Network Forensics)," in *Proc. 1st Digital Forensic Research Workshop (DFRWS' 01)*, Utica, New York, USA, 2001, pp. 27-30.

[159] B. Pande, D. Gupta, D. Sanghi, and S. K. Jain, "The Network Monitoring Tool: PickPacket," in *Proc. Third International Conference on Information Technology and Applications (ICITA' 05)* Sydney, Australia 2005, pp. 191-196.

[160] N. Pandey, S. Gupta, and S. Leekha, "Algebra for Capability Based Attack Correlation," in *Information Security Theory and Practices. Smart Devices, Convergence and Next Generation Networks*. vol. 5019, J. Onieva, D. Sauveron, S. Chaumette, D. Gollmann, and K. Markantonakis, Eds. Berlin / Heidelberg: Springer 2008, pp. 117-135.

[161] D. Parikh and C. Tsuhan, "Data Fusion and Cost Minimization for Intrusion Detection," *IEEE Transactions on Information Forensics and Security*, vol. 3, no. 3, pp. 381-389, 2008.

[162] V. Paruchuri, A. Durresi, R. Kannan, and S. S. Iyengar, "Authenticated autonomous system traceback," in *Proc. 18th International Conference on Advanced Information Networking and Applications (AINA 04)*, Fukuoka, Japan, 2004, pp. 406-413.

[163] V. Paxson, "Bro - Intrusion Detection System," 2008Available: http://www.bro-ids.org, [April 30, 2011]

[164] J. C. Pelaez and E. B. Fernandez, "Wireless VoIP Network Forensics," in *Proc. Fourth LACCEI International Latin American and Caribbean Conference for Engineering and Technology (LACCET' 06)*, Mayagüez, Puerto Rico, 2006, pp. 1-12.

[165] D. Peng, Z. Shi, L. Tao, and W. Ma, "Enhanced and Authenticated Deterministic Packet Marking for IP Traceback," in *Advanced Parallel Processing Technologies*. vol. 4847, M. Xu, Y. Zhan, J. Cao, and Y. Liu, Eds. Berlin / Heidelberg: Springer 2007, pp. 508-517.

[166] S. Perry, "Network forensics and the inside job," *Network Security,* vol. 2006, no. 12, pp. 11-13, 2006.

[167] E. S. Pilli, R. C. Joshi, and R. Niyogi, "Network forensic frameworks: Survey and research challenges," *Digital Investigation,* vol. 7, no. 1-2, pp. 14-27, 2010.

[168] M. Ponec, P. Giura, J. Wein, and H. Bronnimann, "New payload attribution methods for network forensic investigations," *ACM Transactions on Information and System Security (TISSEC),* vol. 13, no. 2, pp. 1-32, 2010.

[169] J. Postel, "RFC 768: User Datagram Protocol," 1980, Defense Advanced Research Projects AgencyAvailable: http://tools.ietf.org/pdf/rfc768.pdf, [April 30, 2011]

[170] J. Postel, "RFC 791: Internet Protocol," 1981, Defense Advanced Research Projects AgencyAvailable: http://www.ietf.org/rfc/rfc791.txt, [April 30, 2011]

[171] J. Postel, "RFC 792: Internet Control Message Protocol," 1981, Defense Advanced Research Projects AgencyAvailable: http://tools.ietf.org/html/rfc0792, [April 30, 2011]

[172] J. Postel, "RFC 793: Transmission Control Protocol," 1981, Defense Advanced Research Projects AgencyAvailable: http://tools.ietf.org/pdf/rfc793.pdf, [April 30, 2011]

[173] PyFlag, "PyFlag - Python Forensic and Log Analysis GUI," 2011Available: http://www.pyflag.net/, [April 30, 2011]

[174] A. Qureshi, "802.11 Network Forensic Analysis," 2009, SANS Institute Reading RoomAvailable: http://www.sans.org/reading_room/whitepapers/wireless/80211-network-forensic-analysis_33023, [April, 30, 2011]

[175] S. Raghavan and S. V. Raghavan, "Digital Evidence Composition in Fraud Detection," in *Digital Forensics and Cyber Crime*. vol. 31, S. Goel and others, Eds. Berlin Heidelberg: Springer 2009, pp. 1-8.

[176] U. K. Rakowsky, "Fundamentals of the Dempster–Shafer Theory and its Applications to System Safety and Reliability Modelling," *Reliability: Theory & Applications (Gnedenko Forum Journal),* vol. 3-4, no. 1, pp. 173-185, 2007.

[177] M. J. Ranum, "Intrusion detection and network forensics," in *Proc. 2nd USENIX Symposium on Internet Technologies and Systems*, Colorado, USA, 1999.

[178] S. Rayanchu and G. Barua, "Tracing Attackers with Deterministic Edge Router Marking (DERM)," in *Distributed Computing and Internet Technology*. vol. 3347, R. Ghosh and H. Mohanty, Eds. Berlin / Heidelberg: Springer 2005, pp. 400-409.

[179] F. Raynal, Y. Berthier, P. Biondi, and D. Kaminsky, "Honeypot forensics, Part 1: Analyzing the network," *IEEE Security & Privacy,* vol. 2, no. 4, pp. 72-78, 2004.

[180] F. Raynal, Y. Berthier, P. Biondi, and D. Kaminsky, "Honeypot forensics, Part II: Analyzing the compromised host," *IEEE Security & Privacy,* vol. 2, no. 5, pp. 77-80, 2004.

[181] M. Reith, C. Carr, and G. Gunsch, "An examination of digital forensic models," *International Journal of Digital Evidence,* vol. 1, no. 3, pp. 1-12, 2002.

[182] S. Rekhis, J. Krichene, and N. Boudriga, "DigForNet: Digital Forensic in Networking," in *IFIP TC-11 23rd International Information Security Conference*. vol. 278, S. Jajodia, P. Samarati, and S. Cimato, Eds. Boston: Springer 2008, pp. 637-651.

[183] W. Ren, "On A Network Forensics Model For Information Security," in *Proc. 3rd International Conference on Information Systems Technology and its Applications (ISTA 2004)*, Utah, USA, 2004, pp. 229-234.

[184] W. Ren, "On The Reference Model of Distributed Cooperative Network Forensics System," in *Proc. 6th International Conference on Information Integration and Web-based Application & Services (iiWAS 04)*, Jakarta, Indonesia, 2004, pp. 771-775.

[185] W. Ren and H. Jin, "Distributed agent-based real time network intrusion forensics system architecture design," in *Proc. 19th International Conference on Advanced Information Networking and Applications (AINA 05)*, Taipei, Taiwan, 2005, pp. 177-182.

[186] W. Ren and H. Jin, "Modeling the network forensics behaviors," in *Proc. Workshop of the 1st International Conference on Security and Privacy for Emerging Areas in Communication Networks (SecureComm' 05)*, Athens, Greece, 2005, pp. 1-8.

[187] M. Roesch, "Snort - lightweight intrusion detection for networks," in *Proc. 13th USENIX Conference on System Administration*, Seattle, Washington, 1999, pp. 229–238.

[188] M. Roesch, "Snort " 2010Available: http://www.snort.org/snort, [April 30, 2011]

[189] R. Rowlingson, "A ten step process for forensic readiness," *International Journal of Digital Evidence,* vol. 2, no. 3, pp. 1-28, 2004.

[190] S. Saad and I. Traore, "Method ontology for intelligent network forensics analysis," in *Proc. Eighth Annual International Conference on Privacy Security and Trust (PST' 10)*, Ottawa, Ontario, Canada, 2010, pp. 7-14.

[191] S. Sanfilippo, "Hping: an active network security tool," 2005Available: http://www.hping.org/, [April 30, 2011]

[192] L. Santhanam, A. Kumar, and D. P. Agrawal, "Taxonomy of IP traceback," *Journal of Information Assurance and Security,* vol. 1, no. 1, pp. 79-94, 2006.

[193] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," *IEEE/ACM Transactions on Networking,* vol. 9, no. 3, pp. 226-237, 2001.

[194] SecureList, "Live Twitter XSS," 2010Available: http://www.securelist.com/en/blog/2297/Live_Twitter_XSS [April 30, 2011]

[195] e. D. Security, "Iris Network Traffic Analyzer," 2011Available: http://www.eeye.com/iris, [April 30, 2011]

[196] W. Security, "WhiteHat Website Security Statistic Report 10th Edition Fall 2010," 2010Available: http://www.whitehatsec.com/home/resource/stats.html, [April 30, 2011]

[197] V. Sekar, Y. Xie, D. A. Maltz, M. K. Reiter, and H. Zhang, "Toward a Framework for Internet Forensic Analysis," in *Proc. Proc. ACM SIGCOMM Hot Topics in Networks (HotNets' 04)*, San Diego, CA, USA, 2001, pp. 83-97.

[198] S. R. Selamat, R. Yusof, and S. Sahib, "Mapping process of digital forensic investigation framework," *International Journal of Computer Science and Network Security,* vol. 8, no. 10, pp. 163-169, 2008.

[199] K. Sentz and S. Ferson, "Combination of evidence in Dempster-Shafer theory," Sandia National Laboratories, Livermore, California 2002.

[200] G. Shafer, *A mathematical theory of evidence* vol. 1. Princeton, NJ: Princeton University Press 1976.

[201] K. Shanmugasundaram, H. Brönnimann, and N. Memon, "Integrating Digital Forensics in Network Infrastructures," in *Advances in Digital Forensics.* vol. 194, M. Pollitt and S. Shenoi, Eds. Boston: Springer 2005, pp. 127-140.

[202] K. Shanmugasundaram and N. Memon, "Network Monitoring for Security and Forensics," in *Information Systems Security*. vol. 4332, A. Bagchi and V. Atluri, Eds. Berlin / Heidelberg: Springer, 2006, pp. 56-70.

[203] K. Shanmugasundaram, N. Memon, A. Savant, and H. Bronnimann, "ForNet: A Distributed Forensics Network," in *Computer Network Security*. vol. 2776, V. Gorodetsky, L. Popyack, and V. Skormin, Eds. Berlin / Heidelberg: Springer 2003, pp. 1-16.

[204] C. Shannon, D. Moore, and k. claffy, "Characteristics of fragmented IP traffic on Internet links," in *Proc. 1st ACM SIGCOMM Workshop on Internet Measurement*, San Francisco, California, USA, 2001, pp. 83-97.

[205] R. Sira, "Network Forensics Analysis Tools: An Overview of an Emerging Technology," *GSEC (1.4)*, 2003.

[206] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Hash-based IP traceback," in *Proc. ACM Annual Conference of the Special Interest Group on Data Communication (SIGCOMM' 01)*, San Diego, California, USA, 2001, pp. 3-14.

[207] M. Solon and P. Harper, "Preparing evidence for court," *Digital Investigation*, vol. 1, no. 4, pp. 279-283, 2004.

[208] D. X. Song and A. Perrig, "Advanced and authenticated marking schemes for IP traceback," in *Proc. Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 01)*, Anchorage, Alaska, 2001, pp. 878-886.

[209] J. Stanger, "Security testing with hping: at the hop," in *Linux-magazine* Kansas, USA: CPAN, 2009, pp. 39-41.

[210] R. Stone, "CenterTrack: An IP overlay network for tracking DoS floods," in *Proc. 9th Usenix Security Symposium*, Denver, USA, 2000.

[211] B. K. Sy, "Integrating intrusion alert information to aid forensic explanation: An analytical intrusion detection framework for distributive IDS," *Information Fusion*, vol. 10, no. 4, pp. 325-341, 2009.

[212] P. Tae-Kyou and R. Ilkyeun, "Design and Evaluation of a Network Forensic Logging System," in *Proc. Third International Conference on Convergence and Hybrid Information Technology (ICCIT '08)*, Busan, Korea, 2008, pp. 1125-1130.

[213] Y. Tang and T. E. Daniels, "A simple framework for distributed forensics," in *Proc. 25th IEEE International Conference on Distributed Computing Systems Workshops (ICDCS 05)*, Columbus, OH, USA, 2005, pp. 163-169.

[214] C. Thomas and N. Balakrishnan, "Performance enhancement of Intrusion Detection Systems using advances in sensor fusion," in *Proc. 11th International Conference on Information Fusion*, Cologne, Germany, 2008, pp. 1-7.

[215] C. Thomas and N. Balakrishnan, "Improvement in Intrusion Detection With Advances in Sensor Fusion," *IEEE Transactions on Information Forensics and Security,* vol. 4, no. 3, pp. 542-551, 2009.

[216] O. Thonnard and M. Dacier, "A framework for attack patterns' discovery in honeynet data," *Digital Investigation,* vol. 5, no. Supplement 1, pp. S128-S139, 2008.

[217] J. Tian, W. Zhao, R. Du, and Z. Zhang, "D-S Evidence Theory and its Data Fusion Application in Intrusion Detection," in *Proc. Sixth International Conference on Parallel and Distributed Computing, Applications and Technologies (PDCAT 05)*, Dalian, China, 2005, pp. 115-119.

[218] S. A. Tramoni, "Net::Pcap 0.16 - Interface to pcap(3) LBL packet capture library," 2005, CPANAvailable: http://search.cpan.org/~saper/Net-Pcap-0.16/Pcap.pm,

[219] B. Turnbull and J. Slay, "Wi-Fi Network Signals as a Source of Digital Evidence: Wireless Network Forensics," in *Proc. Third International Conference on Availability, Reliability and Security (ARES 08)*, Barcelona, Spain, 2008, pp. 1355-1360.

[220] Twitter, "XSS attack identified and patched," 2010Available: http://status.twitter.com/post/1161435117/xss-attack-identified-and-patched, [April 30, 2011]

[221] T. Udaya Kiran and V. Vijay, "Analysis of Traceback Techniques," in *Proc. 2006 Australasian Workshops on Grid Computing and e-research*, Hobart, Tasmania, Australia, 2006, pp. 115-124.

[222] T. Udaya Kiran, V. Vijay, and P. Srini Rao, "DoSTRACK: a system for defending against DoS attacks," in *Proc. ACM Symposium on Applied Computing (SAC' 09)*, Honolulu, Hawaii, 2009, pp. 47-53.

[223] G. Vandenberghe, "Network Traffic Exploration Application: A Tool to Assess, Visualize, and Analyze Network Security Events," in *Visualization for Computer Security*. vol. 5210, J. Goodall, G. Conti, and K.-L. Ma, Eds. Berlin / Heidelberg: Springer 2008, pp. 181-196.

[224] G. Vigna, "A Topological Characterization of TCP/IP Security," in *FME 2003: Formal Methods*. vol. 2805, K. Araki, S. Gnesi, and D. Mandrioli, Eds. Berlin / Heidelberg: Springer, 2003, pp. 914-939.

[225] M. Vivo, E. Carrasco, G. Isern, and G. O. Vivo, "A review of port scanning techniques," *SIGCOMM Computer Communication Review*, vol. 29, no. 2, pp. 41-48, 1999.

[226] I. Vural and H. Venter, "Mobile Botnet Detection Using Network Forensics," in *Future Internet - FIS 2010*. vol. 6369, A. Berre, A. Gómez-Pérez, K. Tutschku, and D. Fensel, Eds. Berlin / Heidelberg: Springer, 2010, pp. 57-67.

[227] I. Vural and H. S. Venter, "Using Network Forensics and Artificial Intelligence Techniques to Detect Bot-nets on an Organizational Network," in *Proc. Seventh International Conference on Information Technology: New Generations (ITNG' 10)*, Las Vegas, Nevada, USA, 2010, pp. 725-731.

[228] D. Wang, T. Li, S. Liu, J. Zhang, and C. Liu, "Dynamical Network Forensics Based on Immune Agent," in *Proc. Third International Conference on Natural Computation (ICNC 2007)*, Haikou, Hainan, China, 2007, pp. 651-656.

[229] S.-J. Wang, Y.-H. Chang, W.-Y. Chiang, and W.-S. Juang, "Investigations in Cross-site Script on Web-systems Gathering Digital Evidence against Cyber-Intrusions," in *Proc. Future Generation Communication and Networking (FGCN 07)*, Jeju, Korea, 2007, pp. 125-129.

[230] W. Wang and T. E. Daniels, "A graph based approach toward network forensics analysis," *ACM Transactions on Information and System Security (TISSEC)*, vol. 12, no. 1, p. 4, 2008.

[231] Y. Wang, H. Yang, X. Wang, and R. Zhang, "Distributed intrusion detection system based on data fusion method," in *Proc. Fifth World Congress on Intelligent Control and Automation (WCICA 04)*, Hangzhou, China, 2004, pp. 4331-4334

[232] WildPackets, "OmniPeek Network Analyzer," 2011, Available: http://www.wildpackets.com/products/network_analysis_and_monitoring/omnipeek_network_analyzer, [April 30, 2011]

[233] Y. Xia, J. Shang, J. Chen, and G.-P. Liu, "Networked Data Fusion With Packet Losses and Variable Delays," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics,* vol. 39, no. 5, pp. 1107-1120, 2009.

[234] Y. Xiang, W. Zhou, and M. Guo, "Flexible Deterministic Packet Marking: An IP Traceback System to Find the Real Source of Attacks," *IEEE Transactions on Parallel and Distributed Systems,* vol. 20, no. 4, pp. 567-580, 2009.

[235] A. Yaar, A. Perrig, and D. Song, "FIT: fast Internet traceback," in *Proc. 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 05)*, Miami, FL, USA, 2005, pp. 1395-1406 vol. 2.

[236] S. J. Yang, A. Stotz, J. Holsopple, M. Sudit, and M. Kuhl, "High level information fusion for tracking and projection of multistage cyber attacks," *Information Fusion,* vol. 10, no. 1, pp. 107-121, 2009.

[237] A. Yasinsac and Y. Manzano, "Policies to Enhance Computer and Network Forensics," in *Proc. IEEE Workshop on Information Assurance and Security*, New York, USA, 2001, pp. 289–295.

[238] A. Yasinsac and Y. Manzano, "Honeytraps, a network forensic tool," in *Proc. 6th World Multi-Conference on Systemics, Cybernetics, and Informatics (SCI 02)*, Florida, USA, 2002.

[239] Z. Ye, W. Shi, and D. Ye, "DDoS Defense Using TCP_IP Header Analysis and Proactive Tests," in *Proc. International Conference on Information Technology and Computer Science (ITCS 2009)*, Kiev, Ukraine, 2009, pp. 548-552.

[240] S. Yi, Y. Xinyu, L. Ning, and Q. Yong, "Deterministic Packet Marking with Link Signatures for IP Traceback," in *Information Security and Cryptology*. vol. 4318, H. Lipmaa, M. Yung, and D. Lin, Eds. Berlin / Heidelberg: Springer 2006, pp. 144-152.

[241] G. Yinghua and M. Simon, "Network Forensics in MANET: Traffic Analysis of Source Spoofed DoS Attacks," in *Proc. 4th International Conference on Network and System Security (NSS' 10)*, Melbourne, Australia, 2010, pp. 128-135.

[242] S. Yong-Dal, "New Digital Forensics Investigation Procedure Model," in *Proc. Fourth International Conference on Networked Computing and Advanced Information Management (NCM '08)*, Gyeongju, Korea, 2008, pp. 528-531.

[243] L. Zhang and Y. Guan, "TOPO: A Topology-aware Single Packet Attack Traceback Scheme," in *Proc. Workshop of the 1st International Conference on*

Security and Privacy for Emerging Areas in Communication Networks (SecureComm' 06), 2006, pp. 1-10.

[244] Y. Zhang, Y. Ren, J. Wang, and L. Fang, "Network Forensic Computing Based on ANN-PCA," in *Proc. International Conference on Computational Intelligence and Security Workshops (CISW 07)*, Harbin, Heilongjiang, China, 2007, pp. 942-945.

Publications out of the work

Journals:

1. E. S. Pilli, R. C. Joshi, and R. Niyogi, "Network forensic frameworks: Survey and research challenges," *Digital Investigation,* vol. 7, no. 1-2, pp. 14-27, 2010.
2. E. S. Pilli, R. C. Joshi, and R. Niyogi, "A generic framework for network forensics," *International Journal of Computer Applications (IJCA),* vol. 1, no. 11, pp. 1-6, 2010.

International Conferences:

3. E. S. Pilli, R. C. Joshi, and R. Niyogi, "Data Reduction by Identification and Correlation of TCP/IP Attack Attributes for Network Forensics," in *Proc. International Conf. and Workshop on Emerging Trends in Technology (ICWET' 11)*, Mumbai, India, 2011, pp. 276-283 [ACM Digital Library].
4. E. S. Pilli, R. C. Joshi, and R. Niyogi, "Deterministic Router and Interface Marking for Network Forensics," in *Proc. Seventh Annual IFIP WG 11.9 International Conf. on Digital Forensics*, Orlando, Florida, USA, 2011 (in press) [Springer].
5. E. S. Pilli, R. C. Joshi, and R. Niyogi, "An IP Traceback Model for Network Forensics," in *Proc. 2nd International ICST Conf. on Digital Forensics & Cyber Crime (ICDF2C' 10)*, Abu Dhabi, UAE, 2010, pp. 129-136 [Springer].
6. E. S. Pilli, R. C. Joshi, and R. Niyogi, "A Framework for Network Forensic Analysis," in *Proc. International Conf. Information and Communication Technologies*, Kochi, India, 2010, pp. 142-147 [Springer].

Journals (Communicated):

7. E. S. Pilli, R. C. Joshi, and R. Niyogi, "Network Forensic Analysis by Integration of Packet Captures and Fusion of Attack Information," *The Journal of Digital Forensics, Security and Law,* 2011 (Submitted Manuscript ID: 37-130-1-SM).

Journals (To be communicated):

8. E. S. Pilli, R. C. Joshi, and R. Niyogi, "Novel Deterministic Packet Marking Approaches for Network Forensic Traceback," Digital Investigation / Information Security Journal, 2011.