2012

Minimization of DDoS false alarm rate in Network Security; Refining fusion through correlation

Faisal Mahmood

Minimization of DDoS false alarm rate in Network Security; Refining fusion through correlation

by

Faisal Mahmood

A Thesis
Submitted to the Faculty of Graduate Studies
through Computer Science
in Partial Fulfillment of the Requirements for
the Degree of Master of Science at the
University of Windsor

Windsor, Ontario, Canada

2012

Minimization of DDoS false alarm rate in Network Security; Refining fusion through correlation

by

Faisal Mahmood

APPROVED BY:

_____

Dr. Kemal Ertugrul Tepe, External reader
Department of Electrical and Computer Engineering

_____

Dr. Arunita Jaekel, Internal reader
School of Computer Science

_____

Dr. Robert Kent, Advisor
School of Computer Science

_____

Dr. Subir Bandyopadyhay, Chair of Defense
School of Computer Science

September 13, 2012

DECLARATION OF ORIGINALITY

I hereby certify that I am the sole author of this thesis and that no part of this thesis has been published or submitted for publication.

I certify that, to the best of my knowledge, my thesis does not infringe upon anyone's copyright nor violate any proprietary rights and that any ideas, techniques, quotations, or any other material from the work of other people included in my thesis, published or otherwise, are fully acknowledged in accordance with the standard referencing practices. Furthermore, to the extent that I have included copyrighted material that surpasses the bounds of fair dealing within the meaning of the Canada Copyright Act, I certify that I have obtained a written permission from the copyright owner(s) to include such material(s) in my thesis and have included copies of such copyright clearances to my appendix.

I declare that this is a true copy of my thesis, including any final revisions, as approved by my thesis committee and the Graduate Studies office, and that this thesis has not been submitted for a higher degree to any other University or Institution.

ABSTRACT

Intrusion Detection Systems are designed to monitor a network environment and generate alerts whenever abnormal activities are detected. However, the number of these alerts can be very large making their evaluation a difficult task for a security analyst. Alert management techniques reduce alert volume significantly and potentially improve detection performance of an Intrusion Detection System.

This thesis work presents a framework to improve the effectiveness and efficiency of an Intrusion Detection System by significantly reducing the false positive alerts and increasing the ability to spot an actual intrusion for Distributed Denial of Service attacks. Proposed sensor fusion technique addresses the issues relating the optimality of decision-making through correlation in multiple sensors framework. The fusion process is based on combining belief through Dempster Shafer rule of combination along with associating belief with each type of alert and combining them by using Subjective Logic based on Jøsang theory. Moreover, the reliability factor for any Intrusion Detection System is also addressed accordingly in order to minimize the chance of false diagnose of the final network state. A considerable number of simulations are conducted in order to determine the optimal performance of the proposed prototype.

DEDICATION

To the Open Source Community...

ACKNOWLEDGEMENTS

I am sincerely and heartily grateful to my advisor, Professor Dr. Robert Kent, for the support and guidance he showed me throughout my research work and dissertation writing. I am sure it would have not been possible without his help. I owe an earnest thankfulness to Dr. Akshai Kumar Aggarwal for providing me the opportunity to start my graduate degree under his supervision. Besides I would also like to thank to my lovely wife and sweet kids who boosted me morally though out my study work.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER I

INTRODUCTION

Overview

Nowadays, computers are interconnected through global Internet. These interconnections between computer systems offer numerous advantages in each life domain. However, computer networks are an easy target for various intrusion attempts. *Intrusion* is an unauthorized attempt to gain control of computer resources. These intrusions are growing more seriously, rapidly and aggressively with a passage of time. The most important challenge faced by network system administrators, since its invention is, the security and privacy of the data travel on the network. Information on these network systems is a valuable corporate asset and therefore more and more protective security efforts are being followed to protect these priceless assets. Organizations use various computer security tools to do damage control done by these intrusion attempts. Intrusion detection system (IDS) is one of the tools that determine unauthorized access attempts on the system network. IDS generate various intrusion detections reports and provide valuable information about the current system security status.

Generally, IDS generate large volume of *false alerts*. These alerts could be of two types, *false positive* or *false negative*. According to Anastasios et al. [1994] false positive alerts are mistakenly generated by the IDS. According to Axelsson [2000] false negative marked by the IDS as a non malicious event but in reality, it should be marked as malicious event. Over the time, researchers adopted various techniques to deal with the issue of large number of false positives alerts generation. These techniques include fusion, correlation and aggression, multi level alert clustering, probabilistic approach,

machine learning and data mining approaches. Each approach has its own advantages, disadvantages and alert management issues. Network designing, data rate, network data type and sensor selection also affects the final output of the IDS. This thesis work deals with the reduction of false positives through *alert correlation* and *fusion* process.

Problem Statement

According to Almgren et al. [2000] and [2001], most of the time IDS generate to many alerts through the process of one-to-one mapping between events and alerts. For system administrator, these overwhelming reports are difficult to read and take longer time to extract current situation of the network security. Additionally, the performance of IDS further affects by generating long list of false positive alerts. According to Alsubhai [2008], these false positives alerts are not real alerts. Investigating all these reported alerts manually and separating positive real alerts from the final listed alerts is extremely complicated and time consuming task for security experts. Moreover extra resources and care is required to accomplish this task error freely. In ideal situation, the IDS should detect all possible intrusions and report only real intrusions. Effectiveness of the IDS depends on its capability to detecting all malicious activities on the target systems. This capability is known as sensitivity of the IDS. According to Bass [1999] and [2000], sensitive IDS generates more alarms than the non sensitive one on daily basis. In an active operational middle size network, IDS could report several hundred thousand alerts a day, out of them 99 % might be the false alerts. Due to the huge volume of false alarms reported by any IDS, the possibility of ignoring the real threat by system administrator put data security in most dangerous situation.

The research reported in this thesis has focused on the ways to extend intrusion detection outcome with *alert correlation* and *alert fusion* process. The assumption has been that existing IDS are using various techniques to overcome the issue of false positives. The computer systems are itself under the protection of some basic security measures such as firewalls, authentication servers, IDS etc. When the security of a system is more important and possibility of making the current systems more secure exhausted through previous security techniques, other effective options need to be evaluated. This thesis has analyzed combination of two additional measures, which is correlation of intrusion alerts for more effective intrusion detection performance and fusion of alerts based on mathematical foundation to further minimize the negative effects of false alarm generation.

Research Questions

The process of surveying, proposing new methodology, testing and evaluating various alert correlation and fusion methods includes following research questions:

- What technique and methods exist for intrusion alert correlation?
- What technique and methods exist for intrusion alert fusion?
- What are the various researchers worked in this field?
- What are the short comings of previous work?
- What are the open problems in this problem domain?
- What are the various possibilities of data sets availability for experiments?
- Which dataset is acceptable for experiments in research community?
- What are the possible options for various sensors available for alert generation?

- What is the implication of combining alert correlation with fusion process?
- Is it possible to customize the current Open Source to create effective intrusion alert system?
- What are the possible computer attacks created so far and how their signature looks like?

Methods

To answer these research questions mentioned above, the following methods were used:

- Detailed survey the research topic, understand the work done by previous researchers along with their shortcomings.
- Design own system to address the issue of false alarms using system integration and system development techniques.
- Experimental evaluation of implemented prototype.
- Compare the proposed method result with other researchers output.

Contribution

We proposed new methodology for false positive alert reduction in Distributed Denial of Service (DDoS) attack by refining the fusion process through various correlation techniques. Our proposed fusion engine has strong mathematical foundation. Some small scale alert reduction rules and methods are also introduced on individual basis for various sensor types. For analysis and evaluation purpose, we used DARPA 1999 Dataset and two open source Intrusion Detections Systems, Bro and Snort. These selections are completely accordance with the acceptance of Intrusion Detection Research community for conducting and evaluating proposed methodologies by the researchers.

Thesis Outline

The rest of the thesis is organized as follows. Chapter 2 provides the literature survey of the research topic and detailed back ground study. Chapter 3 describes our proposed methodology along with some technical and configuration issues. Simulation results and their analysis are presented in chapter 4. Finally, Chapter 5 concludes this thesis with future recommendations for further possible improvements.

CHAPTER II

REVIEW OF LITERATURE

Brief Introduction

According to Siaterlis and Maglaris [2004] "data fusion is a process performed on multisource data towards detection, association, correlation, estimation and combination of several data streams into one with a higher level of abstraction and greater meaningfulness." The process of collecting information from multiple possibly sources and combining them leads to more authentic and significant results. According to Bass [2000], multi sensor data fusion is a relatively new discipline that is used to "combine data from multiple and diverse sensors and sources in order to make inferences about events, activities and situations". Bass [2000] states that this process can be compared to the human reasoning process where the brain receives various information from human body parts for evaluating their current state for making decision and to direct some specific actions. Bass [2000], Siaterlis and Maglaris [2004] [2005] give several examples that use data fusion in the real world. Bass [2000] claim that data fusion is widely used in military applications such as battlefield surveillance, tactical situation assessment and in commercial applications such as robotics, manufacturing, remote sensing and medical diagnosis. Military systems for threat assessment and weather forecast systems are the examples of such systems developed by Siaterlis and Maglaris [2004 and 2005] that are currently in use today.

Most modern IDS use multiple intrusion sensors to maximize their reliability level. The process of multi sensor data fusion, also known as distributed sensing is used to combine data from multiple intrusion sensors and various output sources of the

network in order to make more informed decision about events, activities and situations. Input to a data fusion module of the IDS consists of various sensors output data, users' commands, most of the time by system administrator and some other relevant data from the database through various data mining approaches. For example, the fusion system input could be the output data from various packet sniffers modules consisting of dedicated hardware and software, system log files, SNMP traps and queries, user profile databases, system messages and operator commands. The output of the data fusion module could be *estimates* of identity for a particular intruder, the various activities of an intruder, the location of an intruder, the observed threats, the attack rate and an assessment of the severity and strength of the cyber attack.

Network Types

Computer Networks are categorized based on their different sizes and network scalability. Selection among different network topologies also depends upon network type for sharing multiple information resources on the network. Some widely used network types are briefly discussed in following section.

Wired Networks: In wired networks, systems are physically connected to each other. These networks are much secure and support huge data rates. Wired networks are also cheap in cost and much faster as compared to wireless networks.

Wireless Networks: The simplest way to connect multiple devices is to use wireless networks. A wireless network uses radio waves for connection therefore the absence of physical media like wires makes this network much capable and easy to maintain. These types of networks are much slower and less secure than wired networks.

Infrastructure Networks: Infrastructure networks are the kind of a wireless network based on some infrastructure. It has servers and clients machines and is controlled by the main server. The whole network has a proper infrastructure and no other machine can enter the network without authentication and permission of the main server.

Infrastructure-less Networks (Ad hoc Networks): There is no specific server client relationship among various systems of infrastructure-less network. Each system can act as a server or a client. Each device in this network is self organized and connection among them is wireless. This network type is further sub categorize into following two types.

Static Ad hoc Networks: In a static ad hoc network, devices are not in continuous movement and they are connected among each other through a wireless network.

Mobile Ad hoc Networks: According to Corson et al. [1999] "a mobile ad hoc network (MANET) is an autonomous system of mobile routers. It is very flexible network and deployment is easy as compare to static ad hoc network". MANETs are suitable for emergency situations. This type of network is a relatively new innovation in the field of wireless network technology with new challenges and limitation. Various network operations are difficult to handle in a MANET because each node is independent and topologies change is very frequent. Reliable and efficient protocol is needed for smooth working of the mobile ad hoc network. A MANET consists of a set of communication devices able to indigenously interconnect without any pre existing infrastructure. The most challenging aspect in MANETs is communication of the devices within network range in point-to-point fashion and these devices are generally mobile.

Intrusion Detection System - IDS

The real-time monitoring and analysis of various network activities for potential vulnerability is known as intrusion detection. IDS have become an important component in the security of any network. According to Thakar et al. [2005] "IDS tools aim to detect computer attacks and/or computer misuse and to alert the proper individuals, most of the time a system administrator, upon detection". The IDS installed on a network serves the same purpose as a burglar alarm system installed in a house. Mainly IDS work in two modules, first when intruders or attackers are trying to enter in the network, secondly after they entered in the network. The IDS may be used with firewalls packages to enhance the network security and mainly to regulate the network traffic flow in and out of network. Firewalls and IDS are two different security tools and should never be considered the same.

Active and Passive IDS: According to the www.dummies.com web site "an active IDS (now more commonly known as an intrusion prevention system — IPS) is a system that is configured to automatically block suspected attacks in progress without any intervention by an operator". IPS is the main defense line for the intrusion that provides real-time response to an attack with in the network boundary. The main issue with an IPS is the vulnerability to cyber attack due to its activation within the network boundary.

According to CISSP for Dummies page 406 "a passive IDS is a system that is configured only to monitor and analyze network traffic activity and alert an operator to potential vulnerabilities and attacks". Unlike an IPS passive IDS only generate alerts relating to the network traffic. These IDS are not capable to take any decision on their own.

Development and deployment of these IDS are easy and quick compared to an IPS. These IDS are not vulnerable to intrusion because these systems mainly perform their role outside of the network.

Network-Based and Host-Based IDS: According to Sree, Babu, Murty, Ramachandran and Devi [2008] "Network-based intrusion detection analyzes data packets that travel over the actual network. These packets are examined and sometimes compared with empirical data to verify their nature. Because they are responsible for monitoring a network, rather than a single host, Network-based IDSs (NIDS) tend to be more distributed than host-based IDS". A network-based IDS has some network sensors connected with the network interface card working under unrestricted mode with their own management interface. The IDS monitors all data traffic within its boundary with the help of these network sensors.

According to the CISSP for Dummies "A host-based IDS requires small programs (or agents) to be installed on individual systems to be monitored. The agents monitor the operating system and write data to log files and/or trigger alarms". The host based IDS has limited surveillance to those host systems on which the agents are installed. Therefore entire network is not being monitored by host-based IDS. The very first type of IDS was the host based IDS. According to Sree et al. [2008] "these systems collect and analyze data that originate on a computer that hosts a service, such as a web server. Once this data is aggregated for a given computer, it can either be analyzed locally or sent to a separate/central analysis machine".

Knowledge-Based and Behavior-Based IDS: According to the CISSP for Dummies "a knowledge-based (or signature-based) IDS references a database of previous attack profiles and known system vulnerabilities to identify active intrusion attempts. Knowledge-based IDS is currently more common than behavior-based IDS". Knowledge-based IDS has many advantages over other categories of IDS. Most significant is lower false alarm rate. The understandings of these output alarms are more easy and standardized. The knowledge based IDS is completely dependent on signature database. Therefore continuous update and maintenance is needed to detect new and unique attacks patterns.

According to Desta [2007] "a behavior-based (or statistical anomaly–based) IDS references the baseline and learned pattern of normal system activity to identify active intrusion attempts. Deviations from this baseline or pattern cause an alarm to be triggered". The behavior-based IDS generate more false alarms than knowledge based IDS but these IDS are easily and dynamically able to detect new attacks.

Datasets available for experiments

Researchers who have conducted experiments to demonstrate their proposed methodology for reducing false alarm rate have utilized various datasets in their research work. The DARPA DDoS intrusion detection evaluation datasets are popular choice among many IDS testers. It is no different when it came to testing the Dempster-Shafer IDS models. Yu and Frincke [2005] used the DARPA 2000 DDoS intrusion detection evaluation dataset to test their model. Chou et al. [2007] and [2008] used the DARPA KDD99 intrusion detection evaluation dataset. The KDD99 dataset can be found at http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html.

According to Chou et al. [2007], the DARPA KDD99 data set is made up of a large number of network traffic connections and each connection is represented with 41 features. Further, each connection had a label of either normal or the attack type. They stated that the data set contained 39 attack types which fall into four main categories. They are, Distributed Denial of Service (DDoS), Probe, User to Root (U2R), and Remote to Local (R2L). The authors have reduced the size of the original data set by removing duplicate connections. They further modified the data set by replacing features represented by symbolic values and class labels by numeric values. Also, they normalized values of each feature to between 0 and 1 in order to offer equal importance among features. The 1998 DARPA intrusion detection evaluation data set was used by Katar [2006] for his experiments. Chen and Aickelin [2006] used the Wisconsin Breast cancer dataset and the Iris data set [Asuncion and Newman 2007] of the University of California, Irvine (UCI) machine learning repository for their research.

Some authors chose to generate their own data for the attacks and background traffic. For example, Siaterlis et al. [2003] used background traffic generated from more than 4000 computers in the National Technical University of Athens (NTUA) for their experiment. Various well known data sets used for experiments are mentioned in *appendix D*.

We used DARPA 1999 evaluation in our research work that is the most preferred evaluation dataset used by researcher to evaluate their proposed methodology. Despite some criticism by IDS research community, especially Mahoney and Chan [2003] relating data gathering and further use of DARPA dataset in anomaly detector, the dataset still remains the top choice for the new researcher in the field of IDS. As mentioned earlier, the main objective of this research work is to explore the issue of huge false alarm

generation and address specific methods to minimize the negative effects of these false alarms.

DARPA 1999 evaluation Dataset: It is difficult and very costly to perform reliable system evaluation for new IDS or new proposed methodology. The wide spread research in the field of IDS along with very high cost for development of these systems has led to a centralized idea to evaluate new IDS. DARPA consider most reliable and pioneer in this step by providing huge set of datasets to the research community. Like 1998 evaluation, 1999 evaluation dataset is also an off-line dataset. All critical points and issues raised in 1998 evaluation dataset are addressed individually in 1999 evaluation dataset. 1999 dataset covers more attack types than 1998 evaluation. Addition of Windows NT workstation as a victim and inside tcpdump sniffer machine also play significant role to make dataset type different from previous 1998 evaluation. New stealthy and inside attacks were design to include in 1999 evaluation in order to make network IDS evaluation more effective. 1999 evaluation dataset was designed mainly to detect the ability of new IDS for detecting more sophisticated distributed attacks without training the IDS on the instances of these new attacks.

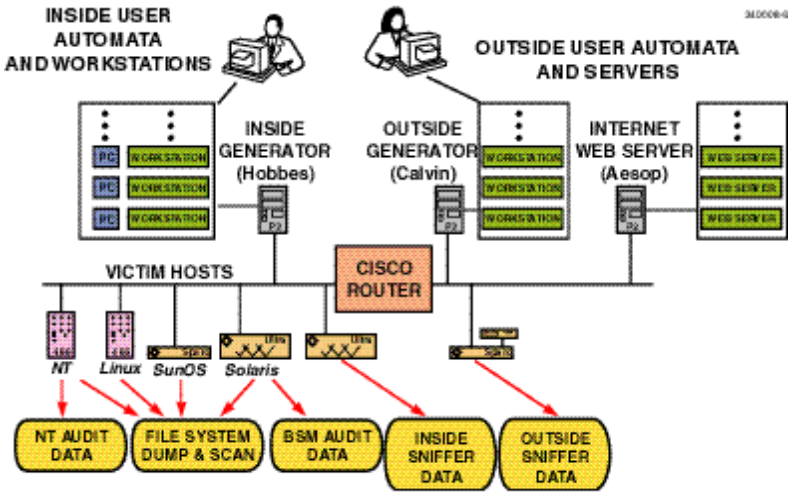

Figure 1Main Operational Network for DARPA 1999 Dataset

(Adapted from 1999 DARPA Intrusion Detection Evaluation:

Design and Procedures. Haines et al [2001 page 18])

Computer attacks in DARPA 1999 evaluation dataset

Computer attack is any malicious activity detected on any computer system. These attacks may include viruses, use of the system or its resources by any other unauthorized individual, probing of a system resources to get any unauthorized information, or any physical attack on computer hardware to interrupt the normal computer system. There are many ways that an attacker can either gain access to the system resources or stop the authorized user from gaining access of the system resources. The detailed attack instances for DARPA 1999 are mentioned in *appendix E*.

Selection of Network Intrusion Detection Systems

Normally, NIDS consists of five types of components including data collector, detector, user interface, storage and responder. Some NIDS perform these tasks in layers with addition of other supporting components. Detailed working and configuration of each NIDS is included in its documentation manual. We used Snort and Bro sensors for our

methodology test. A brief introduction of each of five major components of NIDS is mentioned, followed by some in depth knowledge of Snort and Bro NIDS used in this research work.

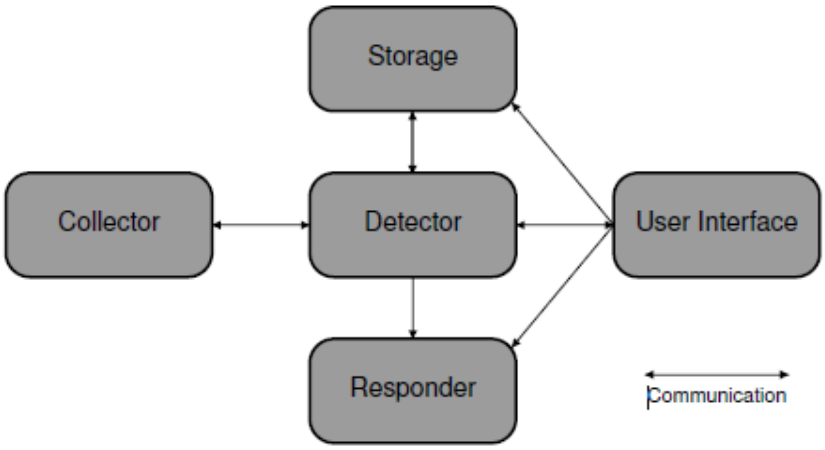

Figure 2Components of General Intrusion Detection System

(Adapted from Design Viable Intrusion Detection in High Performance Environments: Sommer [2005])

1) Collector

Collector provides an interface for accessing data used in detection process. That could in ASCII or binary format. For most of the NIDS, network tap is default type of data collector. Network tap provides interface access to all network packets traveling on some particular position of a network. Some collectors provide access to external databases for detection process.

2) Detector

Most important component where actual detection process takes place could be name as the central processing unit of whole NIDS like brain to human body. Collector and Storage units provide all necessary data to Detector unit for deciding what events are actual alerts.

3) User Interface

Provides output reports to the user and enable user to control NIDS. User interface is consisting of ASCII files and some complicated NIDS also provide graphical user interface.

4) Storage

Two types of data is saved in storage component. First, data may need by detector for further analysis and secondly for user interface to generate reports. This type of data is mainly provided by detector component and saved in database systems. Sometimes, user creates its own database as per requirements.

5) Responder

To save the system from an intrusion, NIDS reacts to the detected intrusions. Responder may actively respond to an intruder by dropping all connectivity with an attacker or connectivity within whole network depending on the severity of an attack.

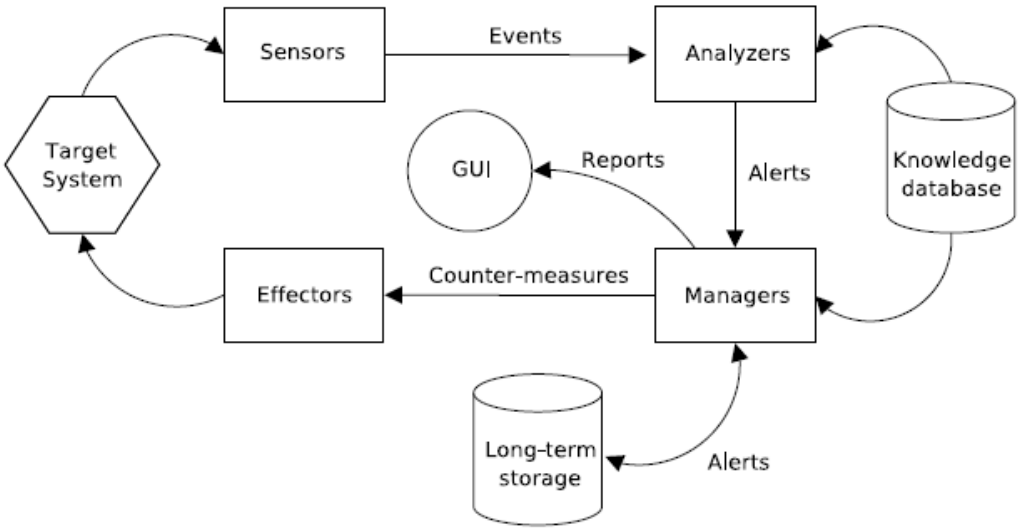

Figure 3Working module for a typical Intrusion Detection System

(Adapted from Design and Implementation of Intrusion Management: Alfaro [2006 page 22])

There is a wide range of NIDS available, and providing detailed overview of each NIDS is out of the scope of this thesis. Various NIDS are regularly evaluated by research community and computer network magazines. The most reputable research group that performs these tests on regular basis is NSS group [http://www.nsslabs.com]. Their reports are not freely available. Very brief introduction of some famous NIDS are mentioned in the table below followed by detailed overview for the two NIDS used in our thesis research work, Bro and Snort.

Table 1 Well known Network Intrusion Detection Systems

NIDS	Brief introduction
Bro	A very exible open-source research system. It also provides the starting point for much of our work relating correlation and fusion technique. Hence, we again take a close look at its design along with Snort.
Snort	Probably the most-widely deployed NIDS. Snort is the de-facto standard

	among open-source systems. This is used in our thesis work. Hence, we take a close look at its design and working.
Emerald, STAT	Two major research systems. They combine different detection approaches in a combined framework. While not purely network-based, they both contain major network-based components. While STAT is open-source, Emerald is not freely available.
Dragon, IntruShield	Two commercial NIDS developed by Enterasys Networks Inc. and McAfee Inc. respectively. These systems are purely network-based and are not available freely.

Intrusion Detection System used for research work.

Bro: Bro is widely used open source NIDS by research community. Vern Paxson is the main person behind the idea who developed Bro NIDS with his team. Compare to other NIDS, Bro is neither fully anomaly-based system nor a misused detection system. Bro has both capabilities, written in C++ under BSD-style license and is a Unix-based Network Intrusion Detection System.

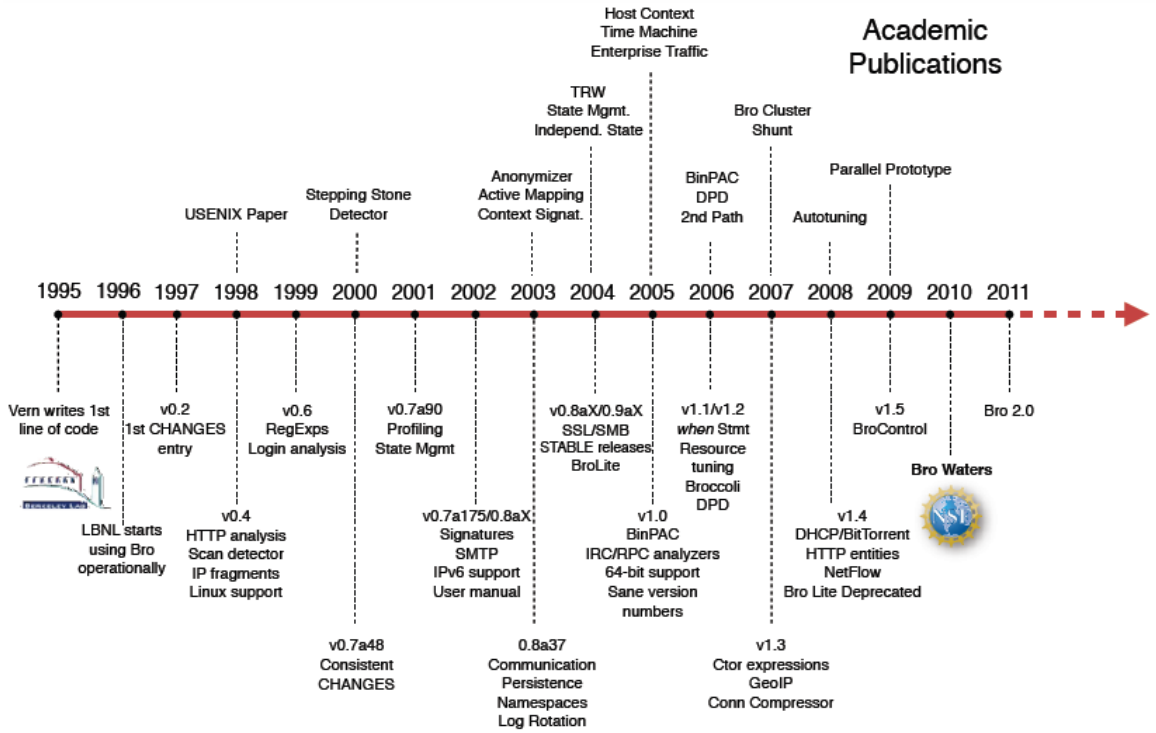

Figure 4 Bro development over the years

(Adapted from Bro Workshop [2011])

According to Sommer [2005] Bros' primary goal were 1) separation of mechanism and policy 2) efficiency operation suitable for high-volume network monitoring and 3) resistance to attacks directed at it. Bro works under real-time network analysis framework with low level detection capabilities. Bro works very well under high speed networks and emphasis on application-level semantics. Bro user need to specify what has to be detected by using very powerful Bro programming scripts.

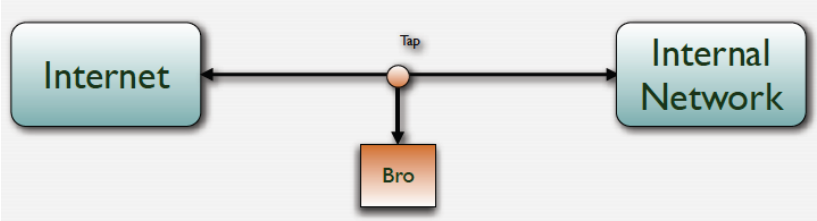

Figure 5 Bro tap in the network

(Adapted from Bro Workshop [2011])

Bro architecture consists of three main layers: packet filter, event generation and policy scripts execution. Packet filtering is done using a static BPF expression. Event generation layer generates events for various actions that take place within the network. For example, successful and unsuccessful user authentication, number of established connections, terminated connections, rejected connection and still alive connections. The user writes policy scripts using a high level language. These scripts contains event handler to deal with various events when they occur. Event handler could be program to response against an event. Event handler may also be program to update the data structure by sending out real time alerts against some event and same time executing some program to deal with the alert.

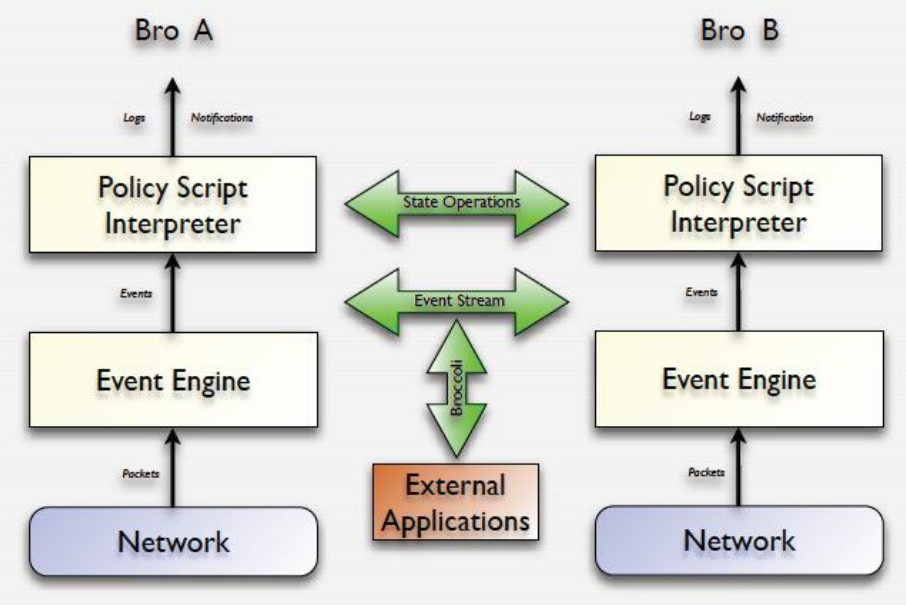

Figure 6 Bro layer architecture

(Adapted from Bro Workshop [2011])

Main advantages of Bro over the other NIDS are real-time network analysis beside intrusion detection, highly sophisticated and stateful that tracks extensive application-layer network state, policy-neutral at the core that can accommodate a range of detection approaches and supports wide range of forensic analysis by saving huge logs files on the network. Bro is mainly command line based and fully customizable that adds more power to its adoptability beside more than 20,000 lines of script prewritten functionality that can just be loaded. These prewritten scripts are responsible to generate logs and alarms as per user demand and requirements. Bro has some challenges when working on huge clusters. Mainly these challenges include communication capability and management of multi-machine setup. These challenges make setup tedious and complex. Bro system lacks proper documentations and functionality polishing that further upgrades the complexity level for some security experts. Currently NSF (National Science Foundation) is funding

Bro development team to tackle these issues by working on user experience and performance feed backs.

Bro is a fully distributed system and analyze the network connections to support almost all type of connections including TCP, UDP and ICMP. These connections types known as connections semantics are interpreted by analyzer through extracting basic semantic protocols elements and generating relating network events. Analyzer consists of two main components i.e. policy-neutral analysis and secondly policy script component that has various policy-specific actions. Analyzers can interact with wide range of protocols including HTTP, FTP, TCP, SMTP, DNS SSL and many more. Some analyzers are protocol independent for detecting scanning, stepping stones and backdoor attacks. Single line ASCII summary is generated by connection analyzer for each connection the system detects.

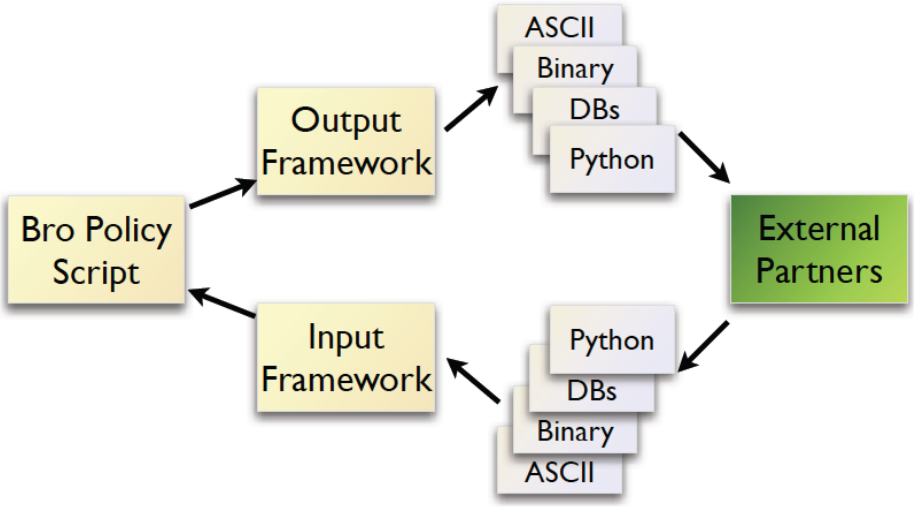

Figure 7 Various frame work of Bro NIDS

(Adapted from Bro web site)

Bro resists attacks against itself. Bro detection mechanism and all other documentations are public except the system core parameters in order to keep the Bro save from intrusion. Bro is a policy-neutral research system that provides huge flexibility to perform various experiments on different parameters in order to evaluate capabilities of new security policies. Therefore we carried out our work by using this highly sophisticated Network

Snort: Snort is highly sophisticated and widely acceptable open source network based IDS in computer networks intrusion research community. Snort uses signature detection method by sniffing network packets and afterward examining that data for the contents that matches known attacks. Snort was developed by Martin Roesch in 1998 and later by Sourcefire (http://www.sourcefire.com) a well know network Security Company based in USA. Snort is capable of performing real time traffic analysis with all capabilities that Bro NIDS capable of, like protocol analysis, events generations and saving the log files for further analysis. According to the Snort website, it can be used as straight packet sniffer like tcpdump, a packet logger or as a full blown network IDS.

Snort is logically divided into many components with different functionalities. The major components are 1) Packet Decoder that takes packets from different types of network interfaces and send them to detection engine for further process. 2) Preprocessors that modify the data before the detection engine perform some security operations in order to detect alerts. In other worlds, preprocessors are plug-ins that may find anomalies in packet headers. Snort Preprocessors has the ability to defragment the network packet, decode HTTP URI and reassembly various packet streams. 3) Detection Engine is like a brain to the Snort that is the most important and core part of NIDS. Detection Engine is

responsible to detect any intrusion activity within network packets. The processing power of Snort detection engine is directly link with the system power where the Snort is installed. Various Snort detection engine may take different amount of time to detect and response the network attacks. Detection engine's processing power mainly depends upon the number of loaded rules, machine power, network speed and network load.

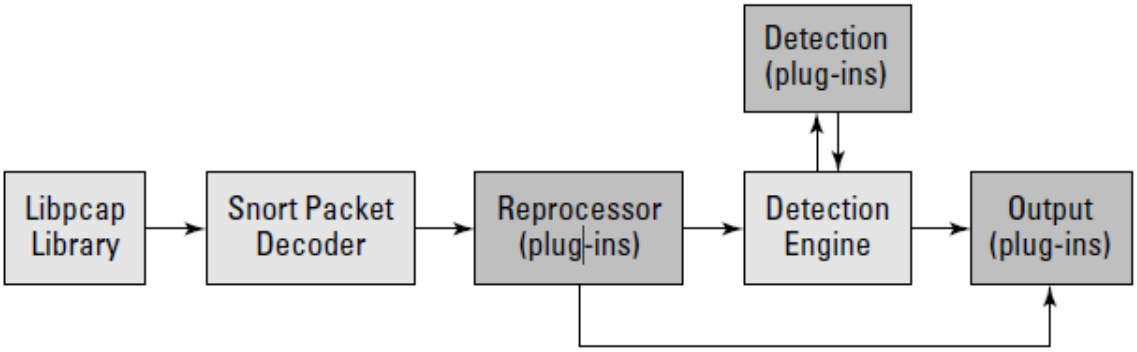

Figure 8 Various components of Snort NIDS

(Adapted from Snort user manual [July 2011])

4) Logging and Alerting System is another major component of Snort that creates various log files depending on the events detected by detection engine. Logs are kept in simple text files or tcp-dump format and may need further manipulation in order to apply various correlation rules. 5) Output Modules specify the format to save the output generated by logging and alerting system of Snort. Output modules may simply alerts in specified location, sending SNMP traps, sending messages to syslog facility, logging alerts in some database systems like MySQL or Oracle and generating XML output

Snort is being supported on number of hardware platforms and operating systems like Linux, OpenBSD, FreeBSD, NetBSD, Solaris, HP-UX, AIX, IRIX, MacOS and Windows. Like Bro, Snort also protects itself from intrusion.

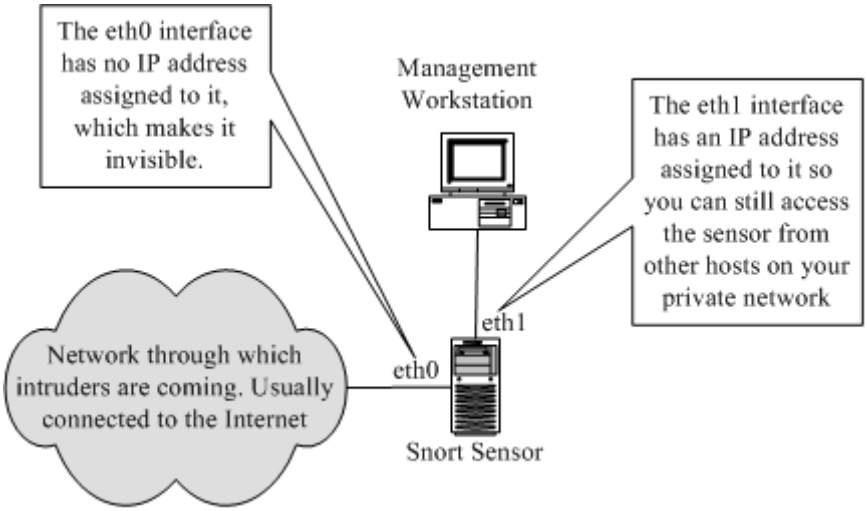

Figure 9 Snort in stealth mode to protect from outside attack

(Adapted from Advance Intrusion Techniques Rehman [2003])

User may run Snort in stealth mode for listening incoming traffic in order to hide its existence. Snort could be run without any IP address interface for keeping its identity hidden. Snort NIDS is well documented and several third-part tools provide interfacing bridge with Snort for administration tasks. Well known third party tools are Snorby (open source), BASE (open source), Sourcefile (commercial) and Aanval (commercial). Snort is most widely deployed NIDS worldwide with millions of downloads and over 400,000 registered users. That's why we selected Snort NIDS for our second open source sensor to carry out our research.

Standardized efforts for representing alerts in IDS - IDMEF

Intrusion Detection Message Exchange Format (IDMEF) defines common data formats and exchange procedures for sharing information among IDS and response systems. The Intrusion Detection Message Exchange Format (IDMEF) is a standard format (RFC 4765) developed by Curry and Debar in the Intrusion Detection Working Group (IDWG) [www.izerv.net/idwg-public]

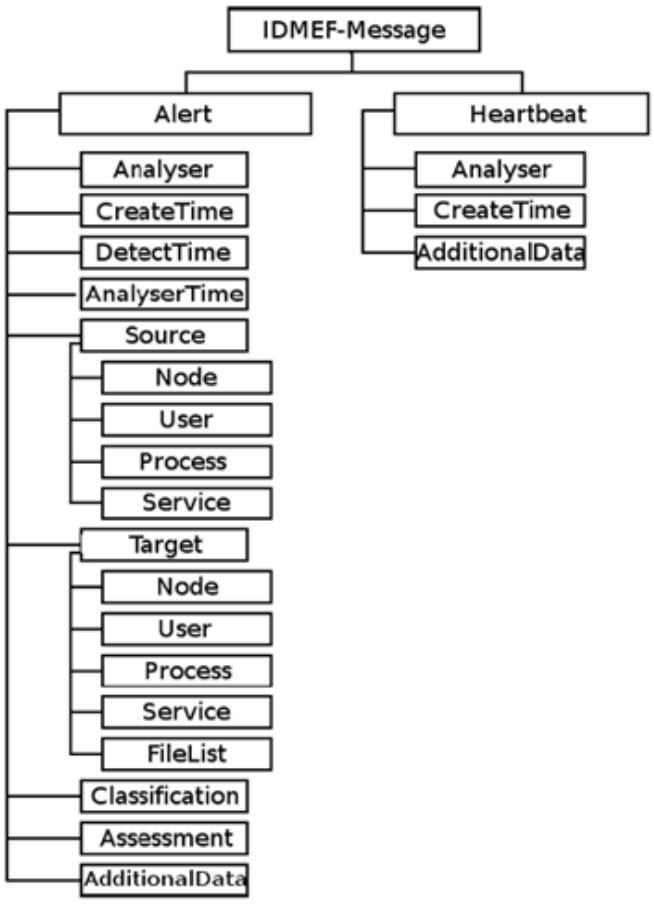

Figure 10 IDMEF data model

IDMEF manage different systems need to interact with IDS. In other words, IDMEF is a standard data format for alert representation used by various IDS for reporting various detected suspicious events. According to Debar et al. [2007], "the most obvious place to implement the IDMEF is in the data channel between an intrusion detection analyzer (or "sensor) and the manger (or "consol") to which it sends alarms."

The data stores in databases though many IDS in the form of IDMEF makes data analyzer (security administrator) able to decide about the system state more effetely on the basis of whole network picture instead of just a part of it. Another huge advantage of IDMEF is during the process of correlation, where various types of data though different sensors are cross-correlated based on some common calculating means. IDMEF also enables the data representation more easy to understand and more towards centralized interface. Data communication among various organizations and security agencies is also more effective and in more efficient way due to the implementation of IDMEF by them. IDMEF also has some limits. For example two or more IDS may name the same alert type differently due to limited capability of IDMEF in semantic representation. This issue further adds complexity in creating an equivalent object oriented representation of the same event in relational database such as MySQL.

Alert Management Techniques

After deploying IDS, alert handling is the next step. The challenge of IDS is not only detecting intrusions but also by managing alerts. Reducing the large number of alerts is known as alert management. Alert management techniques can be divided into two general classes:

Low-level alert management: According to the Alsubhi [2008] "the low-level alert operations deal with each alert individually to enrich its attributes or assign scores based on potential risk".

High-level alert management: According to the Alsubhi [2008] "high-level alert management techniques, such as aggregation, clustering, correlation, and fusion, deals with a set of alerts and provide an abstraction of these alerts. High-level techniques will improve their outcomes due to the early low-level evaluation steps". Various alert management techniques deals with huge number of unrelated alerts during the managing process that leads toward inaccurate final result for the IDS fusion process.

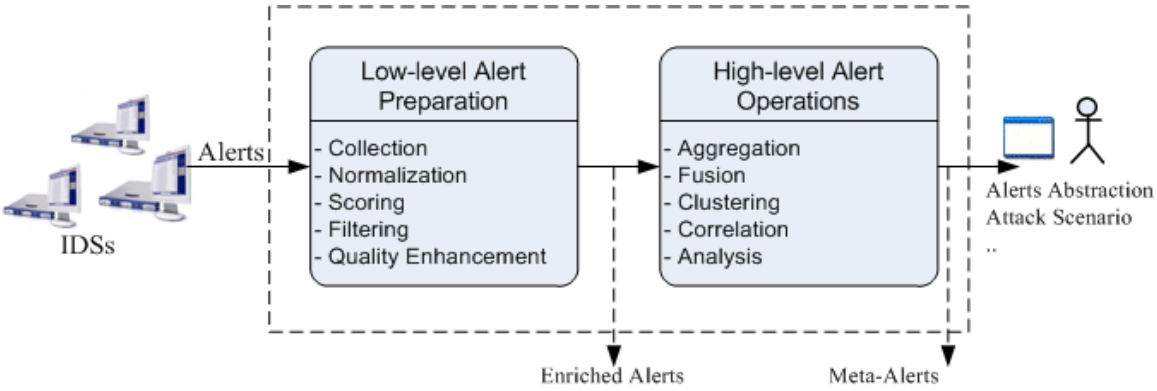

Figure 11 Various Alert Management techniques

(Adapted from A Fuzzy-logic based alert prioritization engine:

Alsubhi [2008])

Brief descriptions of the main alert management techniques are as follows:

Alert Reduction: Alert reduction is a very important technique that is done either before IDS generates alerts or afterwards. According to Alsubhi [2008] "alert reduction techniques includes alert aggregation technique, try to group a set of alerts together that

have some common characteristic, such as the source, the target and the type of the attack". The other important technique is Alert merging technique that combines all alerts in one single alert representation form to present an abstract view of the generated alerts. Alerts with common features such as IP addresses of the machines, their ports or domain name are grouped in alert clustering technique.

Alert Correlation: An alert correlation process discovers connection between various series of security events. To achieve the final goal, most of the time, intruders launch their attack in a series of steps. Traditional IDS are unable to detect these complex attacks series and deal with each small attack as an individual attack without correlating with other attacks. As a result, the IDS generates huge number of alerts and false positives making the analysis process more difficult for network administrator. Most of the time, alert correlation process eliminates huge number of false alarms.

Alert Visualization: Each alert has many related fields, for example the IP address for attack generated machine, the IP address for the target machine, domain name, port number, time stamps and many other related information. This information is shown to the network administrator in the best understandable and visual format. Simply by looking at alert records, it is difficult for the network administrator to make a decision without understanding completely what these alerts are addressing. Alert visualization helps network administrator to understand overall picture of network states.

Data Fusion Process for IDS

The main goal of the alert fusion process is to fix the problems of IDS that include generation of large number of alerts, large number of false positives, large number of non-relevant positives and alerts that do not contain enough information about their attributes. Alert fusion process not only fixes the problems of particular IDS but also process the output of other sensors to find the high-level of attack pattern by performing the process of alert prioritization. According to Helney et al. [2010] "data fusion is a formal framework in which are expressed means and tools for the alliance of data originating from different sources. It aims at obtaining information of greater quality; the exact definition of `greater quality' will depend upon the application."

Multi-Sensor Data Fusion: Around 1970, the first data fusion method was in the military domain for example in surface to air or air to air defense, various surveillance purposes, intelligent gathering for various warning, and defense systems in the battlefield. Later this process was also applied to civilian domains for example robotic engineering, automotive industry, manufacturing industry, medical applications and various construction sites. According to the Bass [2000] a more recent idea is the application of multi-sensor data fusion techniques to the area of information security.

Data Fusion Architecture: A basic and important aspect of a data fusion system is the selection of the architecture for developing the fusion system. Hall [1992] as well as Hall and Llinas [1997] describe three architectural approaches to the data fusion architecture model. According to Boer [2002] "the first approach involves the fusion of raw sensor data, and is called centralized fusion (with raw data), or data level fusion" (see figure

below). This method requires more bandwidth because all sensors data is required to transfer to central processing facility for the fusion process to start. It is important that all sensors in the network have equal capability and level of expertise.

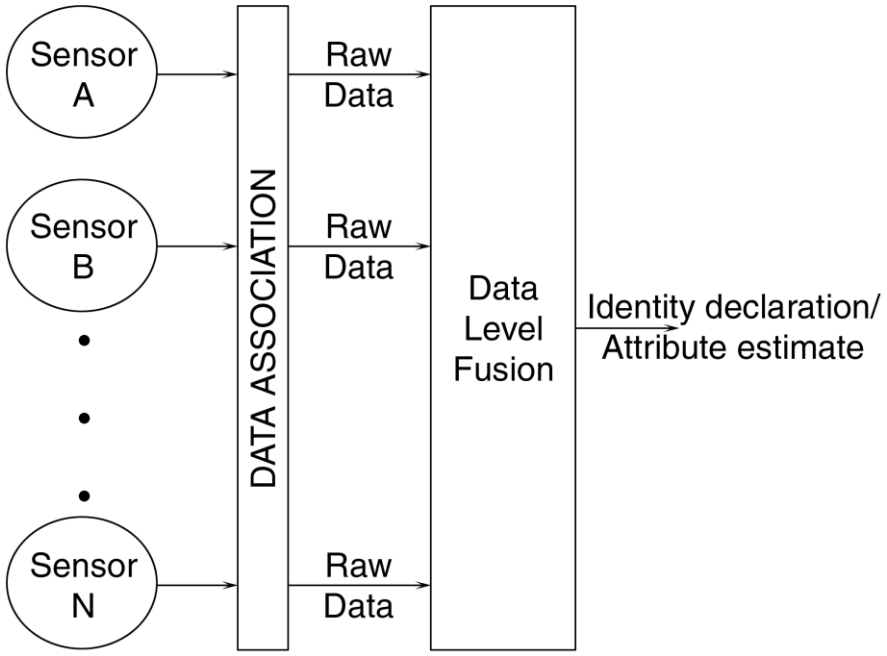

Figure 12 Data Level Fusion Approach

(Adapted from Hall [1992], Llinas [1997] page 17)

According to Boer [2002] "another possible architecture is centralized fusion with feature vector data (see figure below). This is also called feature level fusion. In this architecture, feature vectors rather than raw data are transmitted to the central fusion process". Sensors extract these vectors from the raw data. These feature vectors are refined representation of the raw data that results in some data lose. Because of the data lose in feature fusion

model during conversion of vector data from raw data, less bandwidth is require as compare to the data level fusion approach to transfer data to the central fusion facility.

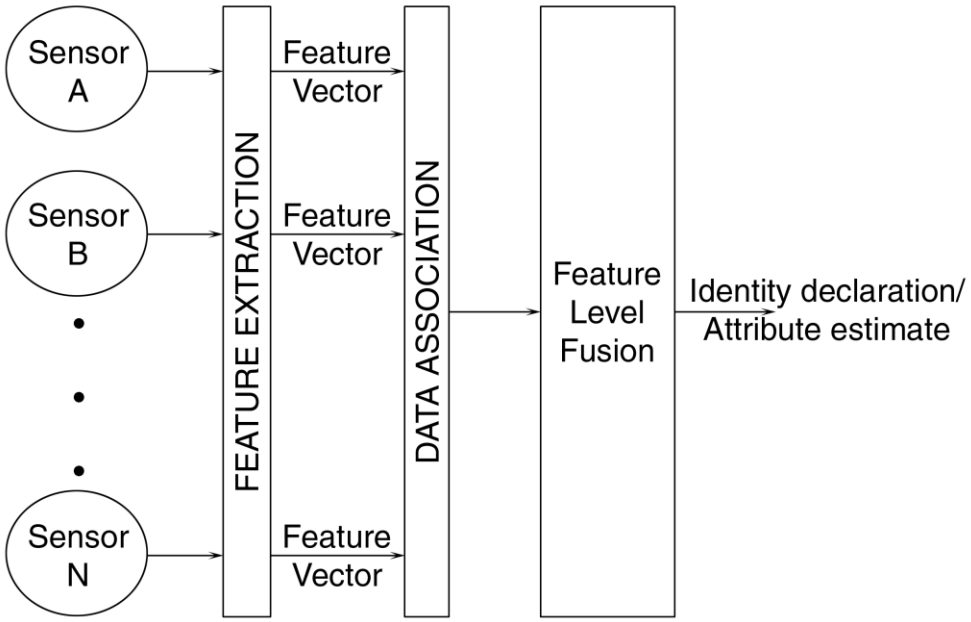

Figure 13 Feature Level Fusion Approach

(Adapted from Hall [1992], Llinas [1997] page 19)

The next fusion level model is known as decision level fusion approach model (see Figure below). In this approach sensors makes decision based their source data. This decision result of sensors goes into the fusion process for further refine fusion. This approach also deals with significant low volume of data transfer and requires less bandwidth as compare to previous two approaches.

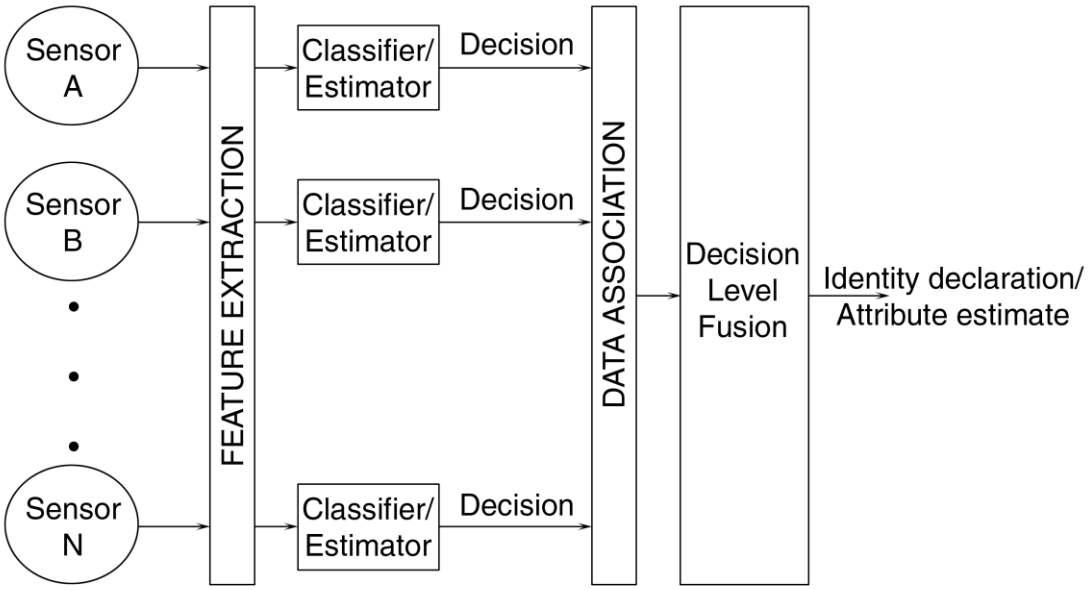

Figure 14 Decision Level Fusion Approach

(Adapted from Hall [1992], Llinas [1997] page 22)

The selection of right approach is purely depends on the requirements and various constraints, for example type of sensors and available bandwidth. Each architecture has some advantages and disadvantage over the others

Fusion-Based IDS

The main purpose of fusion-based IDS is to keep the network operators up to date relating to the current information of the network state. This information mainly relates to attacks, intruders and relationship among them. Fusion based IDS perform this task with greater quality than traditional non fusion based IDS. The success rate of a fusion based IDS is completely based on its multiple sensors and multiple levels of fusion processes known as abstraction levels. According to Blasch and Plano [2002] "the multiple levels of abstraction are supported by the Joint Director of Labs (JDL) [1999] data fusion model". Five more levels were introduced in 2002 by JDL. According to Basch and Plano

[2002] level 0 is for preprocessing, level 1 for object refinement, level 2 for situation refinement, level 3 for threat refinement, and level 4 for process refinement.

This means that the JDL model [2002] is applicable to intrusion detection. At each of the levels of the JDL model, functionality of fusion-based IDS can be identified. Filtration process is done at level 0. Unimportant and nonrelated are filtered out after series of events. These series of events generates alerts. These alerts are transferred to Level 1 fusion of JDL model. In level 1 these alerts combine on the basis on their similarities (alerts generated on the same event) and categorize in to groups. These groups of alerts are called alert tracks. Identity estimate process is done on each alert track by the process of fusion. On level 2 of JDL fusion model, reasoning process is applied in order to find the various categories of attacks and attackers (intruders) on the alert tracks. Threat assessment of current situation is done during level 3. Threat assessment of an attack and attacker is asses during this level. Level 4 generates the final output result after refining the threat assessment in an improved version after considering some additional external factors.

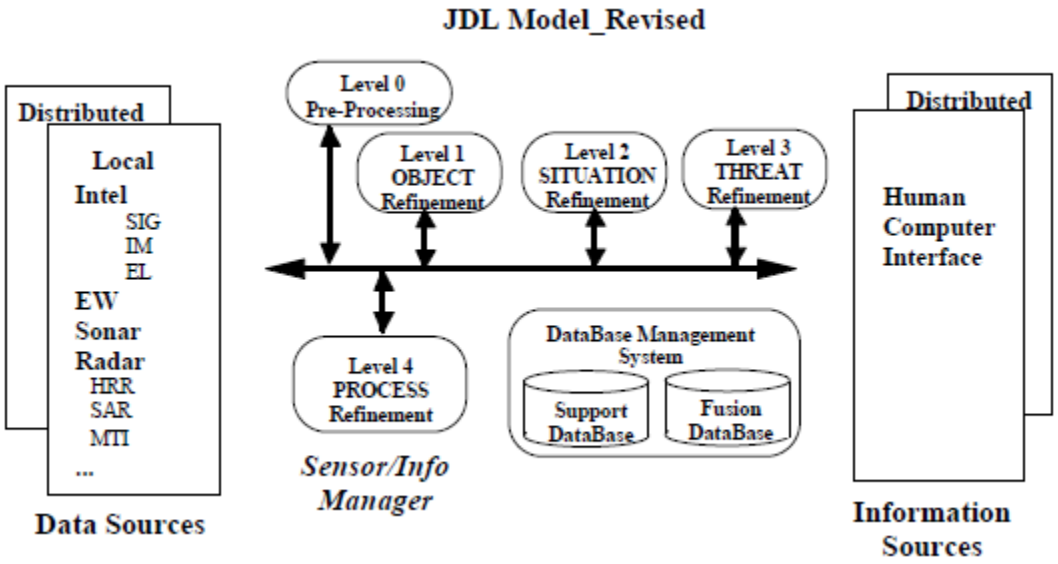

Figure 15 JDL Revised Model Data Level Fusion Approach

(Adapted from Blaschl E and Plano S [2002], page 2)

According to Valdes et al. [2001] "the probabilistic approach is used when a similarity function is defined for each attribute". Similarity function plays a vital role for controlling huge number of alerts. The overall similarity of alerts is calculating by applying similarity function using an expectation of similarity. According to Valdes et al. [2001] "the method considers appropriate fields in alert reports as features for matching algorithm. For each new alert, algorithm computes similarity to existing meta alerts, and merge the new alert with the best matching meta alert, as long as the match passes a threshold value".

Gula [2002] suggested a solution for preventing and reducing false positive alert by correlating IDS alert with vulnerability of that particular system. At a high level, if IDS knows that a system is vulnerable to a particular vulnerability, then it should only

concern itself with attacks against that particular vulnerability. According to the author, this alert fusion approach makes better decisions through an automated fashion.

According to the Burroughs [2002], network system should be secure and defended against attacker not the attack itself. To secure the system from an attacker, the intrusion detection is needed to be performed from an attacker's central viewpoint. According to Burroughs, central view point idea ignores the core issue of possibility of coordinated attack effort among separate attackers against a network.

Chaitanya Kumar [2003] explains another approach to handle false alarm through fusion process. He introduced E-IDS Data Collector that obtained input (alerts) from sensors, logs and device logs and perform multiple steps to reduce alert flooding and false positive.

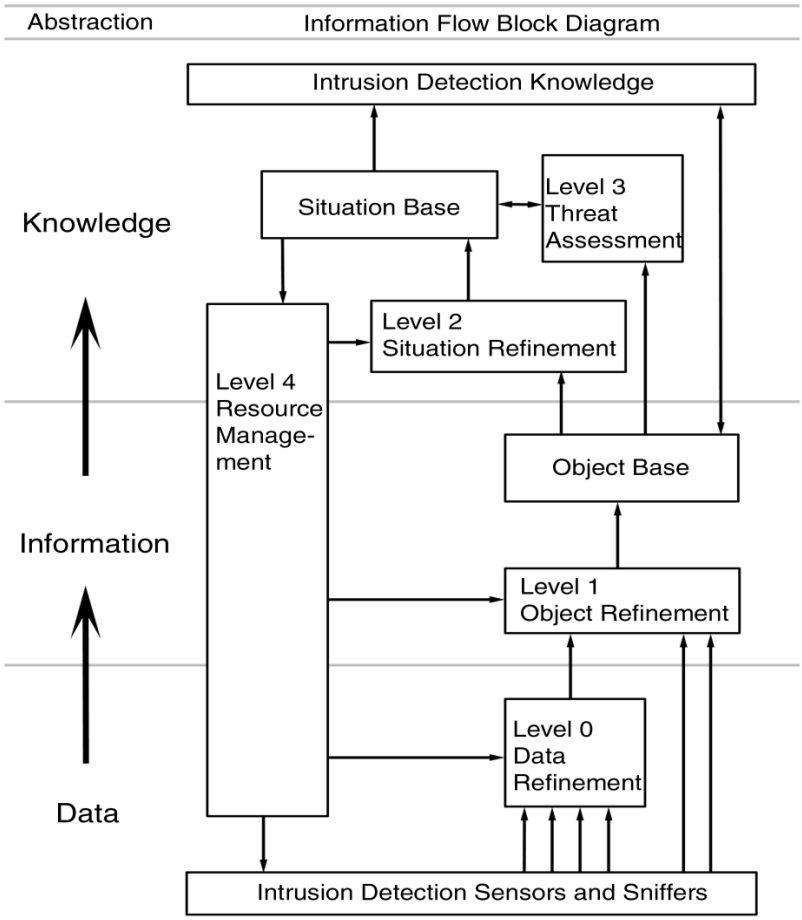

Figure 16 IDS Data Fusion Model – an implementation approach

(Adapted from Bassl [2000], page 14)

Probability Theory

Probability theory is branch of mathematics concerns with probability, the analysis of random formulas. Random variable and events are main inputs for any probability theory. This theory is very helpful to determine the non-deterministic events over time interval in random fashion. These random events repetition- occurrence lead towards evidence for certain patterns of occurrence. After detailed analysis, evidence helps for predicting random events occurrences. Probability theory is very helpful for partial knowledge

systems in order to describe the system state. Probability theory deals with events occur in countable time intervals, is known as discrete probability distribution theory. Continuous probability theory deals with events that occur in continuous time.

Detective logic: Detective logic creates arguments based on necessary hypotheses within set of premises. Premise is a statement about any argument based on rules of inference or symbolic logic for justifying the conclusion. Detective logic or reasoning considers being a sound reasoning if it is a valid reasoning argument and its premises are true.

Probability Logic: Probability logic handles uncertainty factor by combination of probability theory capacity with detective logic capacity. Probability logic is based on very complex formulas that further enhance computational complexity of probability and logical calculations. Various proposals are available to deal with these complex systems. There are two main classes of these proposals. One deals with probabilistic extension to logical entailment and other deals with uncertainty and lack of evidence.

Subjective logic: Subjective logic is a type of probability logic that deals with uncertainty and incomplete knowledge for any situation. Subjective logic works on opinion known as subjective opinion about any proposition. There are two types of opinions, one is binomial opinion applies to single preposition and represent as beta distribution. Other type of opinion is known as multinomial opinion represent as dirichlet distribution. Subjective logic has two types of subjective operators, standard operators and non standard operators. Belief in any proposition is subjective. Subjective opinion expresses

subjective belief relating the correctness of any proposition with degree of uncertainty. Proposition is content or meaning of any sentence. In mathematical form proposition is a combination of symbols and variable that make a meaningful declarative formula.

Uncertainty types: Uncertainty is the base of complex computational analysis that results in advancement of various fields of science. By handling uncertainty factor more efficiently, system becomes more accurate and effective. According to Helton [1997], uncertainty has two major types. Aleatory uncertainty results from the fact that a system can behave in a random way. Aleatory uncertainty is also known as Type A uncertainty, Irreducible uncertainty or objective uncertainty. The other type of uncertainty, known as Epistemic uncertainty results from the lack of knowledge about the system parameters. More in depth analysis needed for Epistemic uncertainty which is also known as Type B uncertainty, Reducible uncertainty, Subjective uncertainty or Ignorance.

Traditional probability theories can handle both uncertainty types. Application of traditional probability theories for Epistemic or Subjective uncertainty is known as Bayesian probability. In probabilistic analysis, analyst knows the exact probability of each event occurrence. When it is not available, Principle of Insufficient Reason is used to represent probability of un known events.

Types of Evidence: Combination of evidence from multiple sources has two major issues to deal with. One is the type of evidence involved and the other is to deal with conflicting evidence. The combination of evidence is based on various types of the evidence involved in the system‘s situation analysis. These types are consonant evidence (nested

structure), consistent evidence (at least one element is common to all subsets), arbitrary evidence (no element is common to all subsets) and disjoint evidence (any two subsets have no element in common).

Dempster Shafer Theory -DST

Dempster Shafer Theory (DST) is a mathematical theory of evidence. Initially Dempster [1967] provide the theory and later Shafer [1976] provide the extension of Dempster's work in more effective and influential way. In a finite discrete space, DST is a generalization of probability theory where different probabilities are being assigned to various sets of events unlike single possible event in traditional probability theories. Therefore, evidence in DST is more meaningful at higher level of abstraction. DST has three main functions to deal with uncertainty. One is basic probability assignment known as bpa or m. Secondly, the Belief function (Bel) and lastly, the Plausibility function (Pl). Dempster-Shafer theory of evidence is more general form of Bayesian theory that also deals with probability factor of an event that requires probability of each step for leading towards an event. Belief function is used in Bayesian theory that leads to various questions of interests like these degrees of belief may or may not have the mathematical properties of probabilities. The closeness of two related questions decides the difference of probability factor among them. According to Glenn Shafer [1976], subsequent work has made clear that the management of uncertainty inherently requires more structure than is available in simple rule-based systems, but the Dempster-Shafer theory remains attractive because of its relative flexibility.

The Dempster-Shafer theory works in two main steps. In first step the degree of belief for one question is gathered from subjective probabilities for a related question. In second

step the, Dempster's rule of combining is applied on these related questions on the basis of their probability factors. The Dempster-Shafer theory is mainly theory of reasoning in order to reach a final decision. The implementation of Dempster-Shafer theory involves the solution of two related problems. First, sorting the uncertainties factors of the problem into an independent item. Second, carry out Dempster's rule computationally. Sorting the uncertainties into independent items leads to a structure involving items of evidence that influence by different but related questions, and this structure can be used to make computations feasible.

Motivation for choosing Dempster-Shafer theory

A major advantage of Dempster Shafer theory over other approaches is narrowing down the hypothesis set with the accumulation of evidence. In Intrusion Detection environment, Dempster Shafer theory is useful for combing the evidence provided by different observers. These observers could even be completely different and located remotely from each other. Each observer could provide its own perceived state to a central server which will combine the evidence to determine the final state of the network. The most important part of the theory is Dempster's rule of combination is combining the evidence from two or more sources to form inferences. Various researchers discussed the advantages of D – S theory of evidence in their work. Some of them are discussed below.

According to Siaterlis et al. [2003], and Siaterlis and Maglaris [2004 and 2005], the Dempster-Shafer theory approach has significant advantages over the Bayesian approach. They state that in contrast to the Bayesian approach where one can only assign probabilities to single elements of the Frame of Discernment-FoD (Θ), the Dempster-

Shafer theory can assign probabilities to the states (elements) of the power set of Θ. Another advantage according to the authors is that Dempster-Shafer theory calculates the probability of the evidence supporting a hypothesis rather than calculating the probability of the hypothesis itself unlike the traditional probabilistic approach. Also, they say that Dempster-Shafer theory has a definite advantage in a vague and unknown environment.

According to Chen and Venkataramanan [2005] the Dempster-Shafer theory of evidence provides a mathematical way to combine evidence from multiple observers without the need to know about a priori or conditional probabilities as in the Bayesian approach.

According to Chen and Aickelin [2006], Dempster-Shafer theory is very well suited for anomaly detection because it does not require any prior knowledge. Another advantage of Dempster-Shafer theory according to Chen and Aickelin is that it can express a value of ignorance, giving information on the uncertainty of a situation. They state that Bayesian inference requires a priori knowledge and does not allow allocating probability to ignorance. So, the authors stated that, in their opinion, Bayesian approach is not always suitable for anomaly detection because prior knowledge may not always be provided. Especially, in casewhere the aim of anomaly detection is to discover previously unseen attacks, a system that relies on existing knowledge cannot be used.

According to Chatzigiannakis et al. [2007] the Dempster-Shafer theory of evidence has a clear advantage in an unknown environment when compared to inference processes like first order logic that assumes complete and consistent knowledge. They also stated that the Dempster-Shafer theory has an advantage when compared to probability theory which requires knowledge in terms of probability distributions.

Specifically relating to our research work, Dempster-Shafer method is a powerful tool that can deal with subject hypothesis for the evidence as well as statistical data combination. It is much like a voting mechanism. It does not have the requirement like Bayesian that the various sensor set be predefined and sensor's join observation probability distribution be known beforehand.

Disadvantages of D – S Theory: Various researchers discussed curtain disadvantages of D – S theory of evidence in their work. According to Siaterlis et al. [2003], Siaterlis and Maglaris [2004] and [2005], and Chatzigiannakis et al. [2007], the main disadvantage of the Dempster-Shafer theory is the assumption it makes, that the pieces of evidence is statistically independent from each other. Since sources of information are often linked with some sort of dependence in real life situations, this assumption does not always hold true. According to Siaterlis et al. [2003], Dempster-Shafer theory based system is unable to detect multiple simultaneous attacks because of consideration of mutually exclusive set of system states.

According to Chen and Aickelin [2006], Dempster-Shafer theory has two major problems. One is the computational complexity associated with Dempster-Shafer theory. The other is the management of conflicting beliefs. According to Chen and Aickelin the computational complexity of Dempster-Shafer theory increases exponentially with the number of elements in the frame of discernment. If there are n elements in frame of discernment set, there will be up to 2^n-1 focal elements that need to be check for any mass function like probability factor. Further the combination of two mass functions needs the computation of up to 2^n intersections.

Mathematics Computation for D – S Theory

The Frame of Discernment (θ): A complete (exhaustive) set describing all of the sets or states of a given system in the hypothesis space. Exactly one of them is true at a time. Generally, the frame is denoted as θ. The elements in the frame must be mutually exclusive. If the number of the elements in the set is n, then the power set (set of all subsets of (θ) will have 2^n elements.

Basic Probability Assignment (bpa): The Dempster-Shafer theory uses a number in the range [0, 1] to indicate the belief in a hypothesis given a piece of evidence. This number shows the degree to which the evidence supports the hypothesis. The impact of evidence on the subset θ is represented by the function called basic probability assignment (bpa). In other word, the basic probability assignment (bpa) represents by m defines the mapping of the power set to the interval between 0 and 1. The bpa to the null set is 0 and the summation of the bpa's of all subsets of the power set is 1. The bpa of any particular set A which is a subset of power set, represents all relevant and available evidence that supports the claim of particular element of X (known as universal set) belongs to the set A but not the subset of A. In other words, the value of m(A) only claims about the set A but not the subset of A. Any further evidence for subset of A would be represented in another bpa function.

$$m: P\ (X) \rightarrow [0,1] \text{ ..... (1)}$$

$$m(\emptyset) = 0 \text{..... (2)}$$

$$\Sigma_{A\ \epsilon\ P\ (X)}\, m(A) = 1 \text{ ..... (3)}$$

Belief and Plausibility: From the bpa assignment, upper and lower bounds of an interval can be defined. This interval contains the precise probability of the set of interest. The interval is bound by two measures knows as Belief and Plausibility. The lower bound Belief is defined for any set A, the subset of universal set X, as the sum of all the basic probability assignments (bpa) of the proper subset B of the set of interest A.

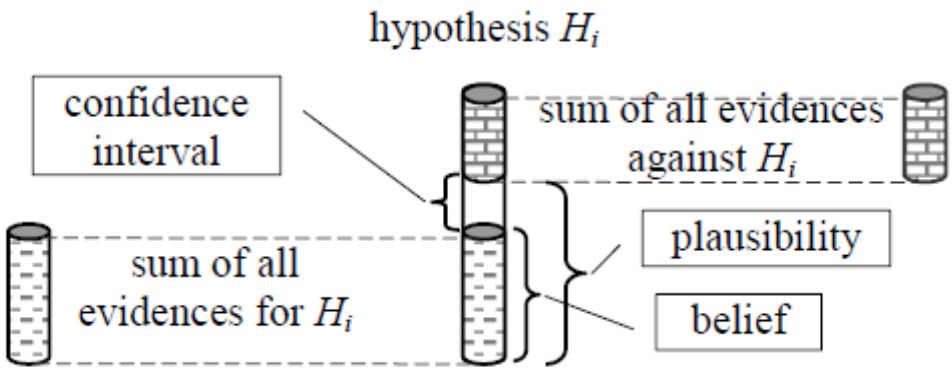

Figure 17 Confidence interval between "belief' and "plausibility"

(Adopted from Wu [2003])

The upper bound Plausibility for the set A, the subset of the universal set X, is the sum of all the basic probability assignments (bpa) of the set of B, the proposer subset of A, that interest the set of interest A.

$$Bel\,(A) = \sum_{B \mid B \subseteq A} m(B) \text{ ..... (4)}$$

$$Pl\,(A) = \sum_{B \mid B \cap A \neq \emptyset} m(B) \text{ ..... (5)}$$

It is possible to obtain the bpa from the Belief measure with the following inverse function

$$m(A) = \sum_{B \mid B \subseteq A} (-1)^{|A-B|}\ Bel(B) \text{ ..... (6)}$$

Where | A – B | is the difference f the cardinality of the two sets. Basic Probability must sum to 1. The two bounded measures, Belief and Plausibility can be drive from each other in the following way.

$$Pl(A) = 1 - Bel(\bar{A}) \text{ ..... (7)}$$

Where $\bar{A}$ is the classical component of A.

$$Bel(\bar{A}) = \sum_{B \mid B \subseteq \bar{A}} m(B) = \sum_{B \mid B \cap A = \emptyset} m(B) \text{ ..... (8)}$$

$$\sum_{B \mid B \cap A = \emptyset} m(B) = 1 - \sum_{B \mid B \cap A = \emptyset} m(B) \text{ ..... (9)}$$

From equation 6 and 7, given any one parameter measure among m(A), Bel(A) or Pl(A), it is possible to drive the other two. The exact probability of any event lies with the lower and upper bounds of Belief and Plausibility.

$$Bel(A) = P(A) = Pl(A) \text{ ..... (10)}$$

Rules for the Combination of Evidence

Multiple sources present different assessment of the same frame of discernment. Dempster Shafer Theory assumes that these sources are independent. From the set theory stand point, rules for the combination of evidence from different sources combine based on either conjunction or disjunction rules. Among the independent sources, if only one source is reliable source, we can justify the use of disjunction combination rules (OR-based on set union). On the other hand, where all sources are considered equally reliable, we can justify the use of conjunction combination rules (AND-based on set intersection). According to Doubious and Prade [1992], there are three types of combination,

conjunctive pooling, disjunctive pooling and a tradeoff. Various operators are available for each combination pooling to combine the data based on some algebraic properties. There are multiple possible ways to combine the evidence in Dempster Shafer Theory. Some combination rules beside Dempster rule of combination are Yager's rule, Inagaki's unified combination rule, Zhang's center combination rule and Doubios and Prade's disjunctive pooling rule. We will discuss Dempster's rule of Combination in detail.

The Dempster Rule of Combination: Basic probability assignment (m) plays vital role in Dempster's combination rule where multiple belief functions combines based on there bpa (m). These belief functions are based on the independent argument (evidence) for the same frame of discernment. According to Shafer [1986,p.132] the Dempster's combination rule is purely a conjunction pooled operation (AND). Dempster's combination rule combines two bpa's m_1 and m_2 in the following manners.

$$m_{12}(A) = \frac{\sum_{B \cap C = A} m_1(B) m_2(C)}{1-K} \text{ ..... (11)}$$

$$m_{12}(\emptyset) = 0 \text{ ..... (12)}$$

$$where\ K = \sum_{B \cap C = \emptyset} m_1(B) m_2(C) \text{ ..... (13)}$$

K represents basic probability mass associated with conflict and determined by the summing the products of the bpa's of all set with null intersection among them. The denominator 1-K is a normalization factor use to address the conflicting evidence, its probability mass and Dempster's rule.

Subjective Logic Theory

As mentioned before, various IDS development methods are based on anomaly or misuse techniques. These techniques and input data along with various configuration parameters of sensors create the system with different degrees of strength and weaknesses. In uncertain environment fuzzy logic, neural network and subjective logic are the most promising candidates for dealing with intrusion alarms. Each technique has its merits for improving the accuracy of detection mechanism. We used subjective logic that is a framework for artificial reasoning based on belief theory. Belief theory known as Dempster-Shafer theory was introduced as a framework for upper and lower probability bounds and a mathematical theory of evidence. According to Jøsang [1999] Classical Dempster-Shafer Theory suffers from the lack of traditional logical operators such as AND and OR. Subjective logic seems very suitable for reasoning about the intrusion detection attacks because an attack can be consider to be a crisp event, i.e. an attack either takes place or not. Mover over in our case, belief about intrusion reported by any sensor can have varying degree of certainty.

Opinion is the central concept of Jøsang's subjective logic theory. According to Jøsang [1999], let θ be a frame of discernment, and let m_θ be a Belief Mass Assignment (BMA) on θ. An opinion w_x on the binary frame of discernment θ with two atomic states (x) and $(\neg\, x)$ is define as

$$w_x = (b_x \,.d_x \,.u_x \,.a_x) \text{ ..... (14)}$$

b_x, d_x, u_x and a_x represents belief, disbelief, uncertainty and atomicity functions on (x) in 2^θ respectively.

The belief function b_x corresponding with m_θ is the function $b: 2^\theta \rightarrow [0,1]$ defined by:

$$b(x) = \sum_{y \subseteq x} m_\theta (y), \qquad x, y \in 2^\theta \text{ ..... (15)}$$

The disbelief function d_x corresponding with m_θ is the function $d: 2^\theta \rightarrow [0,1]$ defined by:

$$d(x) = \sum_{y \cap x = \emptyset} m_\theta (y), \qquad x, y \in 2^\theta \text{ ..... (16)}$$

The uncertainty function u_x corresponding with m_θ is the function $u: 2^\theta \rightarrow [0,1]$ defined by:

$$u(x) = \sum_{\substack{y \cap x = \emptyset \\ y \not\subseteq x}} m_\theta (y), \qquad x, y \in 2^\theta \text{ ..... (17)}$$

The relative atomicity function a_x of $x \; to \; y$ corresponding with m_θ is the function $a: 2^\theta \rightarrow [0,1]$ defined by:

$$a(x/y) = \frac{|x \cap y|}{|y|}, \qquad x, y \in 2^\theta \text{ ..... (18)}$$

The sum of the belief, disbelief and uncertainty function is equal to the sum of the belief mass in a BMA, equal to 1 according to following definition

$$b(x) + d(x) + u(x) = 1, \quad x, y \in 2^\theta \text{ ..... (19)}$$

The basic concept of opinion was initially introduced in Dempster's paper in 1967 and more refined form was introduced by Jøsang in his 1999 paper. Opinion can be presented in an opinion triangle as shown in the following figure.

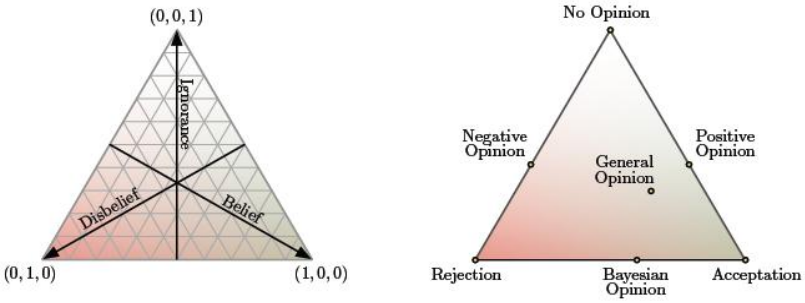

Figure 18 Jøsang's opinion triangle

(Adopted from Haenni [2005])

This triangle was extensively used by Jøsang to represent the opinion space known as opinion triangle. Initially the same concept was introduced by Dempster and he named the concept as a mathematical term barycentric coordinates. According to Jøsang [1999], every opinion corresponds to a normal mass function m_x on θ with $(m_x\{1\}) = (b_x), (m_x(\{0\}) = (d_x)$ and $(m_x(\{0,1\}) = (u_x)$. These relations create a connection between Dempster Shafer theory and Jøsang theory. Most of the researchers believe that Jøsang Subjective Logic is an enhanced special case of applying Dempster's Rule for the mass functions particularly conjunctions, disjunctions and negations but not for the consensus operators.

The Consensus Operator

According to Jøsang [1999], the consensus opinion of two possible conflicting argument opinions is an opinion that reflects both argument opinions in a fair and equal way. When two different observers have different beliefs about the truth of x based on different

evidence about x, the consensus operator produce a consensus belief that combines the two separate beliefs into one.

Let $w_x^A = (b_x^A , d_{x,}^A , u_{x,}^A , a_x^A)$ and $w_x^B = (b_x^B , d_{x,}^B , u_{x,}^B , a_x^B)$ be opinions respectively held by agents A and B about the same state x, and let $K = (u_x^A + u_{x,}^B - u_x^A \; u_x^B)$.

When $u_x^A, u_x^B \rightarrow 0$, the relative dogmatism between w_x^A and $w_x^B$is defined by γ so that $= \frac{u_x^B}{u_x^A}$. Let $w_x^{A,B} = (b_x^{A,B} , d_{x,}^{A,B} , u_{x,}^{A,B} , a_x^{A,B})$be the opinion such that :

$for\ K \neq 0$:

$$b_x^{A,B} = (b_x^A u_x^B + b_x^B \; u_x^A) \text{ / K } \ldots\ldots\ldots \text{ (20)}$$

$$d_{x,}^{A,B} = (d_x^A u_x^B + d_x^B u_x^A) \text{ / K } \ldots\ldots\ldots \text{ (21)}$$

$$u_x^{A,B} = (u_x^A u_x^B) \text{ / k } \ldots\ldots\ldots \text{ (22)}$$

$$a_x^{A,B} = (a_x^A u_x^B + a_x^B u_x^A - (a_x^A + a_x^B) \; u_x^A \; u_x^B) / u_x^A + u_x^B - 2u_x^A u_x^B \ldots\ldots\ldots \text{ (23)}$$

$$a_x^{A,B} = (a_x^A + a_x^B)/ 2 \quad \text{When } u_x^A = u_{x,}^B = 1 \ldots\ldots\ldots \text{ (24)}$$

$For\ K = 0$

$$b_x^{A,B} = (\gamma \, b_x^A + b_x^B) / \gamma + 1 \ldots\ldots\ldots \text{ (25)}$$

$$d_{x,}^{A,B} = (\gamma \, d_x^A + d_x^B) / \gamma + 1 \ldots\ldots\ldots \text{ (26)}$$

$$a_x^{A,B} = 0 \text{ ..... (27)}$$

$$u_x^{A,B} = (\gamma\ a_x^A + a_x^B)/\gamma + 1 \text{ ..... (28)}$$

Then $w_x^{A,B}$ is called the *consensus opinion* between w_x^A and w_x^B representing an imaginary agent [A, B]'s opinion about x. The mapping between the Evidence and Opinion Space can be drawn though following propositional expression where u ≠ 0 and C is a constant value.

$$b = \frac{r}{r+s+C} \text{ ..... (29)}$$

$$d = \frac{s}{r+s+C} \text{ ..... (30)}$$

$$u = \frac{2}{r+s+C} \text{ ..... (31)}$$

Where parameter b represents belief, d is disbelief, u is uncertainty, r represents the amount of evidence supporting the actual event and s represents the amount of evidence supporting its negative.

CHAPTER III

DESIGN AND METHODOLOGY

In this chapter, we present our proposed methodology with detailed architecture and various operating principles. We discuss in-depth analysis for each step and its impact on local and final output with some necessary arguments based on proven theorems and methodologies. We also provide different algorithms and pseudocode for each major independent module.

First we generally explain our proposed architecture and how it works. In next section, we will analyze our approach by describing different metrics used as standard preprocessing against every sensor's output. Multilevel correlation and fusion approaches are discussed and analyzed in next sections. As we discussed earlier in chapter 2, there are different approaches to minimize the false alarms rate, we will discuss how our approach is different from those and how shortcomings of earlier approaches are addressed in our proposed methodology.

Sensor output and Data Collection

IDS perform their network sniffing same time on different network locations. Therefore, their output might be different among themselves depending on the nature of traffic they are scanning. The nature of output data from each sensor completely depends on the internal processing of a particular sensor. After installing and configuring each sensor, their tap location should be carefully selected in order to maximize their utilization.

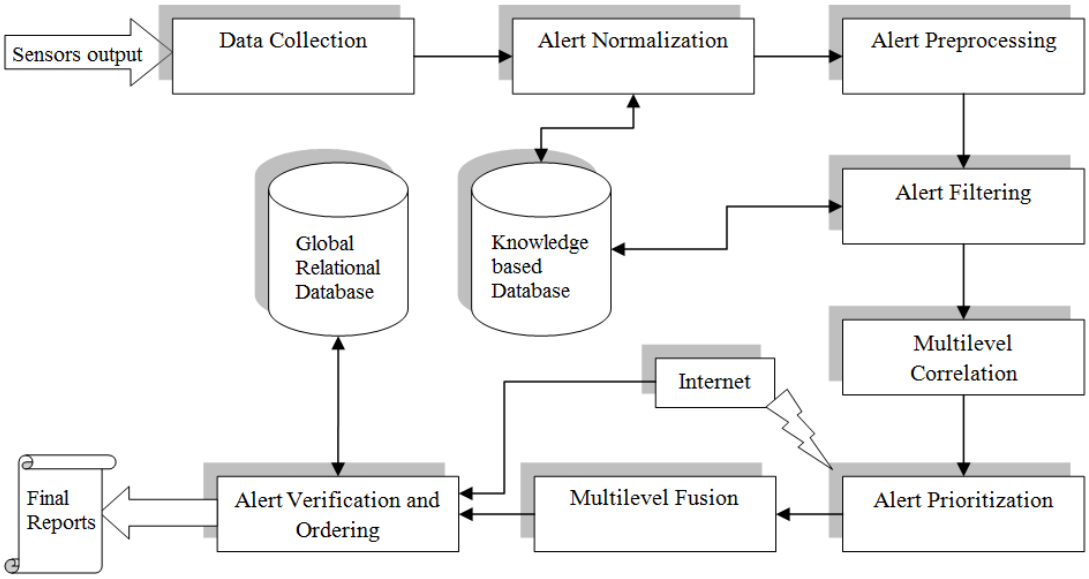

Figure 19 Proposed Architecture for Alert Reduction

Separate database was created for each sensor individually to minimize the possibility of output data lose generated through each sensor. Following are some basic steps for Data Collection.

- *Select location for sensor installation*
- *Install and configure sensors based on security requirements*
- *Create individual database for each sensor*
- *Connect the database engine with sensor's output module*

Alert Normalization

The correlation and fusion module receives alerts from different sensors. As mentioned earlier, different IDS use different format and attributes for alert representation. The basic purpose of this module is to translate every output alert from each sensor, into a common alert representation format with common representing fields. This common representation format is very important for understanding various semantics of each sensor by

correlation and fusion modules. Most of the IDS use Intrusion Detection Message Exchange Format (IDMEF) for their final alert output. Various IDS use different name for the same type of event or most of the time different fields for representing same event. Snort IDS uses well defined fields for an event representation. We followed Snort field set as our standard alert representation. Some sensors generate very detailed alert representation and might be helpful for further analysis. For this purpose, we create separate database for each sensor.

Our Normalization database engine is based on Snort IDS format as a standardized alert format for naming various alerts fields. All the alerts from different IDS are copied to the appropriate fields in order to adopt stand format. This scheme provides basic frame work for next alert reduction schemes especially multilevel correlation.

- *Extract detail of each field represented in sensor output report*
- *Compare them with Snort sensor standard fields*
- *Create fields for new sensor output according to Snort format*
- *Copy all appropriate fields of selected sensor that match with Snort representation fields*

As mentioned earlier, we used Snort standard formatting and naming convention, there are some available schemes for naming convention. Most of the researcher used Common Vulnerability and Exposure (*CVE) naming scheme* for sensors output. According to CVE web site, CVE provides a list of information security vulnerabilities which is free to use. The prime goal of CVE is to make it easier to share data across separate vulnerability capabilities tools. According to Snort official web site, some Snort signatures and Snort rules have CVE field's specification.

Various sensors don't follow IDMEF scheme. For this purpose, simple unification format is an option for correlation and fusion processes, in case those sensors are selected for data sniffing. Following table describe some general fields used for unification alert format scheme.

Table 2 Unification alert format

Field Name	**Description**
analyzerID	*Intrusion detection system ID*
alerted	*Alert ID*
alertType	*Class of alert*
srcIP	*Source IP address of alert*
dstIP	*Destination IP address of alert*
srcPort	*Source port of the alert*
dstPort	*Destination port of alert*
alertTime	*Alert time stamp*
alertInfo	*Alert information and additional description*

Alert Preprocessing

Sensors generate multiple information for any detected event during network sniffing. Most of the time, the information in different fields is not in a standard format. This non standard information is difficult to understand by correlation and fusion processes. For example, Bro use UNIX time stamp and numeric IP address that could not be used for further alert reduction schemes without conversion. In our Alert Preprocessing module, all data representation formatting schemes for various sensors are being examined. If

needed, further processing and conversion is done on the final output in order to achieve standard format for each field in the sensor's local knowledge based database. .

- Check data formatting for standard data representation in each filed of local database for every sensor.
- *If needed, convert non-standard formatting into standard data formatting*

Following is the algorithm for creating knowledge based database for each sensor individually.

- *INPUT TCPDUMP list file.*
- *INPUT alert logs.*
- *Create separate database for each sensor based on the IDMEF.*
- *For each row in TCPDUMP/alert logs files.*
- *IF, the row is labeled with proper attack information and standard data representation*
    - *THEN, Send to local knowledge based database.*
    - *ELSE Convert non standard data representation for each column into standard representation.*
- *RETURN Knowledge based database for each sensor.*

Alert Filtering

Alert filtering module filters unnecessary and unimportant alerts before sending them to the next processing module. In some cases, alert filtering module has zero effect on alert reduction as a whole. In our case, Snort and Bro generates many alerts which can be filter through this procedure. This is very simple and state forward unless attacks are

distributed and complex. For more complex alerts, *alert reconstruction module* could be added, but this is not the scope of this thesis work. We filter those particular events which has only one corresponding alert (not more than one alert) in global relational database. In our module, we extract this information by considering a particular unique alert ID, its corresponding source ip, destination ip, alert time and attack type or signature type. If the alert is generated only once and it is not the part of any further distributed attack, the alert is marked as false positive alert and can be filter from local alert pool (knowledge base database) for each sensor.

- *Select all alerts with one time occurrence in knowledge based database.*
- *Check their participation within any further attack.*
- *Delete the selected alerts that are not the part of any further attack.*

Following is the algorithm for alert filtering based on source and destination IPs.

- *INPUT Knowledge based database for each sensor*
- *FOR every event in the knowledge based database*
- *IF the appearance of Source IP/ Destination IP is only once and IP is not the part of other distributed attack*
    - *THEN Label the event as False Positive alert*
    - *ELSE send alert to Knowledge based database*
- *Return Filtered Knowledge based database*

Multilevel correlation and Alert prioritization

The purpose for this level is to reduce false alarms through various correlation techniques based on the data traffic type and relationships among attacks pattern and their signatures. On the basis of these relationships, important attacks can be separated from the rest.

Various IDS provides detail analysis of alert signature that their sensor is capable of detecting. For this purpose, we considered all signatures with their priority list and correlated all alerts generated with those signatures type. We mainly considered following signatures types by carefully examining their time stamp to detect number of occurrences for any particular signature type alert that might be used in further attacks.

- *ICMP Timestamp Request*
- *ICMP Timestamp Reply*
- *ICMP Destination Unreachable Host Unreachable*
- *ICMP Destination Unreachable Port Unreachable*
- *Protocol Mismatching*
- *Echo reply*

In next levels of correlation, we performed detailed analysis of ICMP traffic. Our correlation rules for this level are based on ICMP type and code number.

- *Correlation based on ICMP type and code*
- *Correlation on the basis of signature priority*
- *Correlation based on source IP and time stamp (specifically for TCP and UDP traffic)*

Following is the algorithm for False Positive Reduction based on ICMP traffic.

- *INPUT Filtered Knowledge based database for each sensor*
- *FOR ICMP traffic,*
- *separate the alerts based on*
    - *ICMP Timestamp Request*
    - *ICMP Timestamp Reply*

    - *ICMP Destination Unreachable Host Unreachable*
    - *ICMP Destination unreachable Port Unreachable*
    - *Protocol Mismatching*
    - *ICMP Echo reply*
- *IF events are part of distributed attack*
    - *THEN send to Global database*
    - *ELSE Label the events as False Positive alerts*
- *RETURN Global database*

Following is the algorithm for False Positive Reduction based on source IP and time stamp for ICMP, UDP and TCP traffic.

- *INPUT Global database*
- *FOR each time slot*
    - *Separate time frames having more than one event*
- *IF events has same source IP, same destination IP and same alert type*
    - *THEN Correlate alerts into single alert*
    - *ELSE send them to global database*
- *RETURN Correlated Global database*

Each signature has priority level assigned to it. There are 3 priority levels, 1 to 3 for any signature type. We used these predefined priority levels for each signature and further perform our analysis to determine how many alerts were generated in a fixed time window. Based on these fixed time windows and alert signature priority, various alerts belong to same attack type in a specific time window were correlated. In other words, different alerts belong to same attack type in a fixed time window were correlated. This

proposed correlation technique is very useful for correlating different alerts constitute same attack and has huge impact for minimizing false positive alerts.

Alert Fusion Engine based on Dempster Shafer Theory

Building the Frame of Discernment (Ω): D-S theory of evidence need to construct the frame of discernment, also known as a universe of disclosure. FoD is a set of mutually exclusive and exhaustive possibilities denoted by (Ω). For this purpose, the very first step is to determine, where a network attack took place or not. This is determined through the process of evidence collection by network sniffing through sensor (observer). If any network attack associated with these alerts has not happened, these alerts are considered to be the false alerts.

Therefore our FoD for attack happened or no attack happened is

$$\textit{Frame of Discernment} = \Omega = \{ A, \bar{A} \}$$

Any hypothesis *H* will refer to the subset of Ω for which the observer can present the evidence. The set of all possible subsets of Ω, including itself and the null set ∅ is called the power set and designated by 2^{Ω}. We can say, the power set consists of all possible hypothesis H_i and known as focal elements $2^{\Omega} = \{\emptyset, A, \Omega, \bar{A}\}$. In other worlds, we may call A_i to the subset of Ω carrying hypothesis H_i. *(Where i is any number starting from 1 to n)*

Our main focus is to detect the flooding attack and try to eliminate the false positives. Therefore, our main focus is on the following type of traffic:

$$\{\text{normal, ICMP, TCP, UDP}\}$$

Every sensor $S_1 .. S_n$ collects the evidence of these attacks on the basis of their observation and assigns the basic probability assignment (bpa) function. Our goal is to find the true state of the system based on some evidence $E_1 .. E_n$. This evidence has some degree of uncertainty towards the system state. Based on single evidence E_i, we assign a probability that it supports a certain hypothesis H_i for a particular system state preposition A_i.

Evidence collection

The very first step is to collect the evidence. For this purpose, we use two types of observations, *r and s*. The observation *r* represents the amount of evidence supporting the actual event happening and *s* represent the observation for negation of any event occurrence. Both *r* and *s* are known as amount of observation representing opposite evidence about an event occurrence. Degree of belief, disbelief or uncertainty may be assign to any event based on the evidence in favor or against an event happening.

Reliability of sensors and certainty level of collected evidence

To be the useful fusion process, the measurement of the evidence must be reliable. Having incorrect information is more damaging than no information. To address this issue, we used two very reliable sensors (Snort and Bro) for collecting the evidence. Moreover, we used MIT DARPA dataset 1999 for our proposed prototype evaluation. This dataset is the most reliable source available to the researchers for conducting their

experiments. As mentioned in earlier chapter, there is no perfect answer and no perfect measure for any evidence. The relationship between accuracy and the uncertainty is always been an issue for concern, especially in digital evidence collection. Reliability of the sensors and certainty levels of their digital evidences are two different domains. If the evidence collected from a sensor can be modified by external unauthorized factors, its reliability is compromised. On the other hand, the certainty level of collected evidence depends on the technical capabilities of a particular sensor to monitor specific type of traffic as well as the right placement of the sensor.

Reliability of any sensor depends on these questions.

- Is it possible to attack the sensor itself?
- Is it possible that during the evidence collection by the sensor, the whole process can be influence by external factors?
- Is it possible that evidence stored location can be attacked and its security might be compromised?

Certainty level of any sensor's digital evidence depends on some following points.

Selecting the right category/type of the sensor based on the nature of traffic being monitor on the network. Various sensors and IDS are best option for one category of attacks but not suitable for other categories of attacks and could pass on wrong, incomplete and uncertain evidence to the system administrator. For example, to analyze SNMP traffic, any sensor with capabilities of detecting and analyzing SNMP packets is much better option than the other sensor with same reliability but no capability on detecting SNMP traffic. These sensors having same sensor reliability factor but might

collect different certainty level of evidence for the same data traffic due to their different detection capabilities.

Sensor's configuration is another important factor for judging the certainty level of any sensor. Right configuration of a particular sensor enhances the certainty level of a sensor's evidence. Sensors with high degree of sensitivity generates huge number of false positives where as low sensitive sensors might ignore the true positives. Through our series of experiments, we observed that if reliability factor of any sensor decreases, the uncertainty factor relating any proposition increases.

As we mentioned earlier, in our research we used Snort and Bro sensors. Both are configured correctly and work well on our Frame of Discernment set Ω for the ICMP, TCP and UDP traffic. Further, there is no chance to compromise the collected evidence by external unauthorized factors like intrusion on stored evidence itself. During our research process, the configuration of each sensor remains the same. All data gathering is done by keeping the configuration static. Therefore we are not dealing with *reliability of sensor* during the research process. In DS rule of combination, we used reliability factor for each sensor as 1 mentioning out sensors are 100 % reliable.

To address the issue of certainty level for each sensor, we defined *False Positive Rate (FPR), True Positive Rate (TPR)* and *True Deduction Rate (TDR)* for each sensor used in our research process as follows;

$$\left(\mathrm{FPR} = \frac{\mathrm{NOFA}}{\mathrm{NOA}} * \%\right)$$

$$\left(\mathrm{TPR} = \frac{\mathrm{NOTA}}{\mathrm{NOA}} * \%\right)$$

$$\left(\mathrm{TDR} = \frac{\mathrm{NOTA}}{\mathrm{RA}} * \%\right)$$

Where

FPR = False Positive Rate of any sensor

NOFA = Number of False Alert

NOA = Number of Alert

TPR = True Positive Rate

TDR = True Deduction Rate

NOTA = Number of True Alerts

RA = Real Alerts

By conducting series of experiments, we calculated the FPR, TPR and TDR for each sensor. Our experiments show that Bro has huge volume of false positive generation as compare to Snort. After multiple correlation techniques, Snort has high TDR than Bro despite poor performance of both sensors in detecting true alerts. Numerical values are given in *appendix-I*. On the basis of FPR, TPR and TDR values, it is obvious that each sensor produce correct information or faulty information. Here our FoD is θ = (Correct, Fault) and the power set is three elements excluding empty set is: 2^{θ} = {correct, fault, θ}.

For the purpose of independence, sensors observe the network traffic separately. Correct represents the hypothesis for correctness of any evidence collected by any sensor. Fault represents its negation and universal set θ represents the uncertainty. Certainty level of each sensor is calculated after completing the process of multiple correlations. NOTA (Number of True Alerts) and RA (Real Alerts) contribute for calculating the certainty

level of any sensor. We calculated the certainly level of each sensor through following proposed methodology;

Certainty level for Snort:

Number of True Alerts (NOTA)-detected by Snort = 840

Real Alerts (RA)-Actual alerts in Dataset = 1482

True Detection Rate (TDR) = 56 %

We assigned belief value 0.56 for sensor Snort by calculating its TDR that shows the capability of Snort for detecting the real attacks from provided Dataset. The remaining portion of total belief (1 - 0.56) i.e. 0.44 is divided equally in disbelief and uncertainty level of sensor Snort. To support our argument, we performed series of experiments and each time, Snort generated false positives with almost same degree of TDR. Therefore, we can say

$$\omega_{S1}^{F} = (b_{S1}^{F}, d_{S1}^{F}, u_{S1}^{F}, a_{S1}^{F}) = (0.56, 0.22, 0.22, 0.5)$$ where S_1= Snort

Please note that we put relative atomicity to the default base rate of 0.5. To support our selection for the value of a_{S1}^{F}= 0.5, we considered the original Jøsang [1999] definition 5

of relative atomicity. According to Jøsang [1999], relative atomicity of *x* to *y* is the function defined by:

$$a\left(\frac{x}{y}\right) = \frac{|x \cap y|}{|y|}, \text{ x y} \in 2^{\theta}$$

In our case,

x = (correct value) or x= (fault value)

y = (correct, fault)

$$a\left(\frac{x}{y}\right) = \frac{|x \cap y|}{|y|}, \text{ x y} \in 2^{\theta}$$

$$a\left(\frac{x}{y}\right) = \frac{1}{2}, \text{ x y} \in 2^{\theta}$$

Therefore, in our case, a_{S1}^{F}= 0.5 that is equal to the default base rate for atomicity value.

Certainty level for Bro:

Number of True Alerts (NOTA) = 690

Real Alerts (RA) = 1482

True Detection Rate (TDR) = 46

We assigned belief value 0.46 for sensor Bro by calculating its TDR that shows the capability of Bro for detecting the real attacks from provided Dataset. The remaining portion of total belief (1 – 0.46) i.e. 0.54 is divided equally in disbelief and uncertainty level of sensor Snort. To support our argument, we performed series of experiments and each time, Bro generated false positives with almost same degree of TDR. Therefore, we can say

$$\omega_{S2}^{F} = (\, b_{S2}^{F}, d_{S2}^{F}, u_{S2}^{F}, a_{S2}^{F}\,) = (0.46, 0.27, 0.27, 0.5)$$ where S_2= Snort

Please note that we put relative atomicity to the default base rate of 0.5. To support our selection for the value of a_{S2}^{F} 0.5, we considered the original Jøsang [1999] definition 5 of relative atomicity. According to Jøsang [1999], relative atomicity of *x* to *y* is the function defined by:

$$a\left(\frac{x}{y}\right) = \frac{|x \cap y|}{|y|}, \text{ x y} \in 2^{\theta}$$

In our case,

x = (correct value) or x= (fault value)

y = (correct, fault)

$$a\left(\frac{x}{y}\right) = \frac{|x \cap y|}{|y|}, \text{ x y} \in 2^{\theta}$$

$$a\left(\frac{x}{y}\right) = \frac{1}{2}, \text{ x y} \in 2^{\theta}$$

Therefore, in our case, a_{S2}^{F} = 0.5 that is equal to the default base rate for atomicity value.

Evidence rules:

The evidence gathered by intrusion detection sensors were transformed into evidence variables r and s where

r = amount of evidence - supporting the actual event e occurrence

s = amount of evidence - supporting the negation of the event e occurrence

We are dealing with DDoS attack type which are ICMP flooding, UDP flooding or SYN flooding known as TCP flooding. Each of these flooding attacks is final result of many distributed attacks. In other words, presence of various distributed attacks within the fixed time slot contributes finally flooding attack. To utilize our proposed theory of evidence collection, in first phase, we collected all supporting distributed attacks with their signature types that contribute to ICMP flooding, UDP flooding and SYN flooding individually for each sensor type.

During our second phase of evidence collection, we assign a numeric value r for each distributed attack generated by our sensor and a numeric value to s for each absence of the evidence. For Snort and Bro IDS, these supporting attacks for each type of flooding are given in *appendix F*. As we mentioned earlier, we used two types of observations, *r and s*. Both *r* and *s* are known as amount of observation representing opposite evidence about an event occurrence. Degree of belief, disbelief or uncertainty may be assign to any event based on the evidence in favor or against an event happening. Evidence gathering process was done for 3 types of attacks detected by each sensors (Snort and Bro) separately.

Proposed r-rule

Our proposed r - rule for any attack type detected through any sensor is defined in the following 2 cases.

Case I) For Number of Alerts (NOA) < 10

r = 1

Case II) For Number of Alerts (NOA) ≥ 10

$$r = \frac{Number\ of\ Total\ Detected\ Alerts}{50} + 2$$

Drop all digits after decimal point

Example: Total detected attacks by Snort on March 29, 1999 slot for ICMP flooding against attack type s-1 are 40 i.e. r ≥ 10 therefore, case II is being followed. According to above rule, value of r for s-6 can be calculated as follows

$$r = \frac{40}{50} + 2 = 2.8$$

r = 2 (by dropping 0.8 i.e. drop all digits after decimal point)

Finally, r value of s-6 attack on March 29, 1999 is 2 in this particular example.

Proposed s-rule

Initially, value 1 is given to all supporting attack types for *s type observation* mentioning no attack is detected for specific category. The s value converts to 0, in case attack for a particular category is detected.

Example: Total attacks detected by Snort sensor for March 29, 1999 time slot again attack type s-6 is 0. In this case, s-6 has 0 values for observation s.

Network traffic is analysed based on equal time interval slots for each attack type. We used week 4 and week 5 of DARPA dataset 1999, total 10 days of dataset. In order to collect alerts, we run our sensors against each trace file of the dataset. In total 19 trace files were used, 2 for each day. One trace file for the inside network data collection and one for the outside network data collection except for March 30, 1999 where only outside dataset trace file is available. We performed whole process of data gathering for more than ten times and extracted the average values to show in our results.

Basic Probability Assignment (bpa):

The Dempster-Shafer theory uses a number in the range [0, 1] to indicate the belief in a hypothesis given a piece of evidence. This number shows the degree to which the evidence supports the hypothesis. The impact of evidence on the subset Ω is represented by the function called basic probability assignment (bpa). In other word, the basic probability assignment (bpa) represents by m defines the mapping of the power set of Ω

to the interval between 0 and 1. The bpa to the null set is 0 and the summation of the bpa's of all subsets of the power set is 1. The detailed bpa function is defined in the previous chapter.

According to our observation, every alert generated by a sensor might be a true alert (true positive) or might be a false alert (false positive). Therefore we can say that each alert generated by any sensor has the degree of belief, degree of disbelief and degree of uncertainty. Dempster Shafer rule of combination do not limit any specific method for assigning bpa to an event. We collected the mapping between the Evidence and Opinion Space though following propositional expression. Note that we followed the Jøsang[1999] for collecting our parameters for bel*ief, disbelief* and *uncertainty;*

$$b = \frac{r}{r+s+C} \text{ ..... (29')}$$

$$d = \frac{s}{r+s+C} \text{ ..... (30')}$$

$$u = \frac{2}{r+s+C} \text{ ..... (31')}$$

Where parameter

C represents a constant,

b represents the belief mass in support of x ∈ FoD (Ω) being true,

d is the belief mass in the support of x being false,

u ≠ 0 and represents uncertainty or the amount of uncommitted belief mass,

r represents the amount of evidence supporting the actual event and

s represents the amount of evidence supporting its negative.

In our research, constant C has the default value of 2 because according to Josang [1999], the default value 2 for the constant C ensures the probability density function with default base rate *a=0.5* is a uniform probability density function. We put the base rate (*a*) value equal to 0.5 for representing opinion of any sensor relating an attack happening. The basic probability assignment (bpa) is assign to each alert pool generated by any sensor. In other words, bpa for each sensor (Snort and Bro) against each targeted alert (ICMP flooding, SYN flooding, UDP flooding) during every time slot duration is calculated and stored in a separate database. According to Jøsang [1999], the sum of the belief, disbelief and uncertainty function is equal to the sum of the belief mass in a BMA, equal to 1 according to following definition

$$b(x) + d(x) + u(x) = 1, \quad x, y \in 2^{\theta}$$

The Reassignment of Mass Function between 2 sensors:

After assigning the appropriate bpa value to each alert domain generated by two sensors separately, the combine mass function was reassigned between these two sensors based on following table.

Table 3The reassignment of mass function between two sensors

	m1(S1)	A	¬A	Ω
m2(S2)				
A		*m(A) = m1(A).m2 (A)*	*m(φ)=m1(¬A).m2(A)*	*m(A)=m1(Ω).m1(A)*
¬A		*m(φ)=m1(A).m2(¬A)*	*m(¬A)=m1(¬A).m2(¬A)*	*m(¬A)=m1(Ω).m2(¬A)*
Ω		*m(A)=m1(A).m2(Ω)*	*m(¬A)=m1(¬A).m2(Ω)*	*m(Ω)=m1(Ω).m2(Ω)*

Where

A represent the probability that the bpa module assigns relating the current state of the network attack A_i *happening,*

¬ *A represent the probability that the bpa module assigns relating the current state of the network attack* A_i *refutation,*

Ω represents the probability that the bpa module assigns relating the current state of the network attack A_i *uncertainty.*

We used two separate combination rules for our research in order to reduce false alarm rate and to spot the actual intrusion happening for DDoS attacks. One is the Dempster rule of Combination and other is consensus operator by Jøsang. We also proposed our final rule of decision for both Dempster Combination and Consensus operator. The final results of both rules are compared in the next chapter with detailed analysis. In previous chapter, both rules are discussed in detail.

Dempster-Shafer Applied to Distributed Intrusion Detection

Our Frame of Discernment consists of two possibilities concerning network events:

$$\textit{Frame of Discernment} = \Omega = \{\mathrm{A}, \neg\mathrm{A}\} \text{ ..... (34)}$$

Where A means event is an attack and $\neg\mathrm{A}$ means that event is not an attack. For this, Ω the power set has three focal elements: hypothesis H associated with (A) that event is a real attack; hypothesis $\neg\mathrm{H}$ associated with (¬A) that event is not an attack and universe hypothesis Ω that event is either attack or not an attack.

Next, the combination belief for hypothesis allocated to power set is define as

$$\mathrm{bel(H)} = \mathrm{m(H)} = \mathrm{m_1(H)} \oplus \mathrm{m_2(H)} \text{ ..... (35)}$$

By Dempster rule of combination, we combined belief of two hypotheses as follows.

$$m_1(H) \oplus m_2(H) = \frac{1}{K} [\, m_1(H)m_2(H) + m_1(H)m_2(\Omega) + m_1(\Omega)\, m_2(H)] \text{ ..... (36)}$$

$$m_1(\neg H) \oplus m_2(\neg H) = \frac{1}{K} [\, m_1(\neg H)m_2(\neg H) + m_1(\neg H)m_2(\Omega) + m_1(\Omega)\, m_2(\neg H)]$$

$$m_1(\Omega) \oplus m_2(\Omega) = \frac{1}{K} m_1(\Omega)m_2(\Omega) \text{ ..... (38)}$$

Where

$$K = m_1(H)m_2(H) + m_1(H)m_2(\Omega) + m_1(\Omega)\, m_2(H) + m_1(\neg H)m_2(\neg H) + m_1(\neg H)m_2(\Omega) + m_1(\Omega)\, m_2(\neg H) + m_1(\Omega)\, m_2(\Omega) \text{.....(39)}$$

We are using two sensors in our research therefore the combination is between two sources. In case of more than two evidence collection sources, we can combine hypothesis of more resources the same way by combining the final result of two sensors with the third one and so on.

To combine the various statements about the hypothesis H_i, the Dempster-Shafer approach must know the reliability or trustworthiness of each sensor S_i participating in evidence collection E_i. This is a big advantage of Dempster Shafter theory of combination over Bayesian approach where (in Bayesian approach) combining the evidence relating each hypothesis H_i, we need to know the prior probability *P(H)* as well as every conditional probability that sensor S_i would offer relating evidence E_i when hypothesis H_i is true and when it is not. Compare to Dempster-Shafer, the Bayesian

approach requires much more information that most of the time might not be available for combination process.

Final Decision Rule (FDR) for Dempster Shafer Combination rule.

After applying DS rule of combination to all collected evidences for each hypothesis domain, we obtained final fusion belief results for

$$m_i(H), m_i(\neg H) \text{ and } m_i(\Omega) \text{ where } i = 1..n$$

Where $m_i(H)$ is the final belief in support of an attack $\mathbf{A_i}$ occurrence, $m_i(\neg H)$ represents the combined belief for not supporting the attack happening and $m_i(\Omega)$ is combined uncertain about the current state of 2 or more prepositions. Number of total attacks for each belief level decides the final result of a fusion module. Our fusion module is based on the following rules.

- If max $(m_i(H), m_i(\neg H), m_i(\Omega)) = m_i(H)$

  Strong combined evidence that shows the attack happening, therefore

  Final Result = True Alerts.

- If max $(m_i(H), m_i(\neg H), m_i(\Omega)) = m_i(\neg H)$

  Strong combined evidence that shows the attack did not happened, therefore

  Final Result= False Alerts.

- If max $(m_i(H), m_i(\neg H), m_i(\Omega)) = m_i(\Omega)$

  Strong combined evidence that shows the uncertainty of the current situation,

  Final Result = True Alerts.

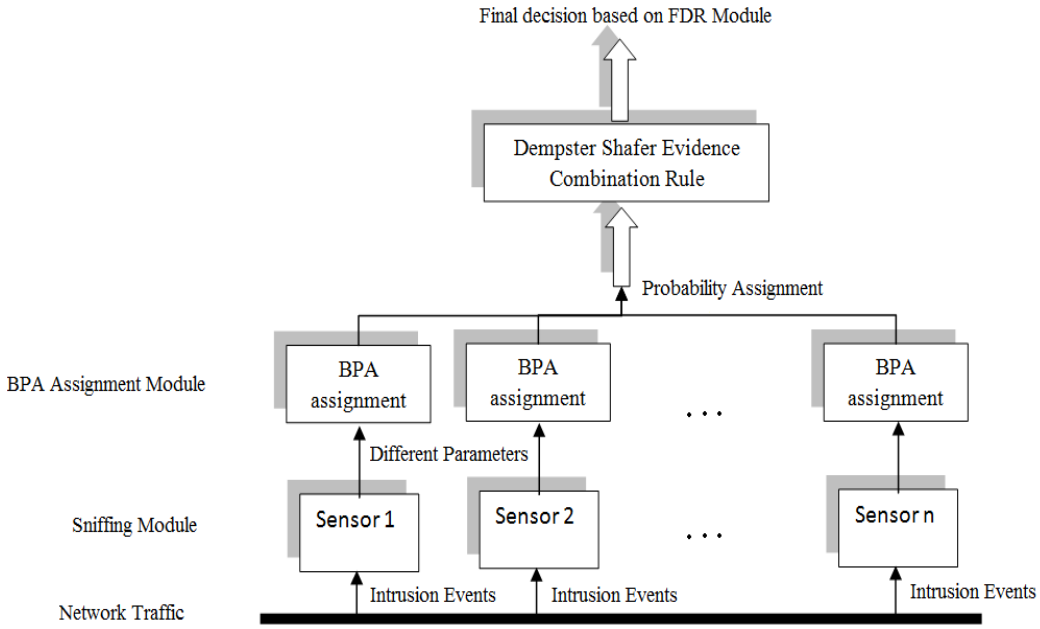

Figure 20 Fusion Module based on Dempster Shafer Theory

The architecture of Fusion Module based on Dempster Shafer Rule of Combination

Alert Fusion Engine based on Subjective Logic

In our fusion frame work, various sensors collect the data and raise an alarm if necessary. The fusion component processes the generated alarms and if necessary, generates an alert. The generated alerts could be handled manually by system administrator or by other automatic response as per requirement. The opinion relating the accuracy level of an alert generated by fusion component helps the system administrator to deal with the alert in manual or automatic manner. In our frame work, we apply subjective logic on various sensors used for data collection and on the fusion component itself. When subjective

logic is applied to sensors, the collected data represent an opinion about the presence of an attack through collected evidence. Some evidence can generate positive opinion and other can generate negative opinion. In other words, each attack pool has belief, disbelief and uncertainty level for the presence of an attack. These levels of opinion are used by fusion component for fusing multiple alarms.

Our fusion component can produce alerts based on single type or multiple types of alerts. This is done by calculating belief or opinion about every preposition associated with each alarm generated through various sensors. In our research, we have crisp preposition, an attack or not an attack, for the category ICMP flooding, UDP flooding or SYN flooding that is being attempted. By observing alarms from multiple sensors (in our prototype testing - Bro and Snort) opinion about the correctness of the preposition (attack or not attack) can be generated. Some alerts contribute in the negation of an event happening and on the other hand, some alerts can generate positive opinion, i.e. belief that the preposition is true.

We also note that the more alarms generating the positive opinions results final opinion relating a preposition stronger. In our research work, we assume that our sensors are independent and generate independent alarms depending on their own analysis. Therefore, we used consensus operator to model our prototyping.

The Consensus Operator

According to Jøsang [1999], the consensus opinion of two possible conflicting argument opinions is an opinion that reflects both argument opinions in a fair and equal way. When two different observers have different beliefs about the truth of x based on different evidence about x, the consensus operator produce a consensus belief that combines the

two separate beliefs into one. This interpretation of belief combination can be supported by classical statistical inference principles discussed in Josang [2001] with formal justification of the consensus operator.

Let $w_x^A = (b_x^A , d_{x,}^A , u_{x,}^A , a_x^A)$ and $w_x^B = (b_x^B , d_{x,}^B , u_{x,}^B , a_x^B)$ be opinions respectively held by agents A and B about the same state x, and let $K = (u_x^A + u_{x,}^B - u_x^A u_x^B)$. When $u_x^A, u_x^B \rightarrow 0$, the relative dogmatism between w_x^A and w_x^B is defined by γ so that $\gamma = \frac{u_x^B}{u_x^A}$. Let $w_x^{A,B} = (b_x^{A,B} , d_{x,}^{A,B} , u_{x,}^{A,B} , a_x^{A,B})$ be the opinion such that :

$for\ K \neq 0$:

$$b_x^{A,B} = (b_x^A u_x^B + b_x^B u_x^A) / K$$

$$d_{x,}^{A,B} = (d_x^A u_x^B + d_x^B u_x^A) / K$$

$$u_x^{A,B} = (u_x^A u_x^B) / k$$

$$a_x^{A,B} = (a_x^A u_x^B + a_x^B u_x^A - (a_x^A + a_x^B) u_x^A u_x^B) / u_x^A + u_x^B - 2u_x^A u_x^B$$

$$a_x^{A,B} = (a_x^A + a_x^B)/ 2 \quad \text{When } u_x^A = u_{x,}^B = 1$$

$For\ k = 0$

$$b_x^{A,B} = (\gamma\, b_x^A + b_x^B) / \gamma + 1$$

$$d_{x,}^{A,B} = (\gamma\, d_x^A + d_x^B) / \gamma + 1$$

$$a_x^{A,B} = 0$$

$$u_x^{A,B} = (\gamma\, a_x^A + a_x^B)/\,\gamma + 1$$

Then $w_x^{A,B}$ is called the consensus opinion between w_x^A and w_x^B representing an imaginary agent [A,B]'s opinion about x. In other words, by using $\oplus$ symbol for consensus operation, we can define;

$$\omega_x^{A,B} = \omega_x^A \oplus \omega_x^B$$

We used the same mapping between the Evidence and Opinion Space that was used for belief, disbelief and uncertainty representation in previous section for Dempster Shafer Theory. Josang [1999] proved that consensus operator is commutative as well as associative. Therefore, according to Josang [1999], we can say that the order in which the opinions are combined has no importance and significance. It is also noted that during the combination, opinion of an agent should be count only once. The fusion process should not allow an agent's opinion to be counted more than once.

Dealing with uncertainty of the sensor

As we mentioned earlier, there are always the risks that an attacker may mislead the prime source of data gathering like network sensor. This process of misleading results false positive alarms generations by those sensors. These false positive alerts affect the right decision process of the fusion engine. In other words, if the protection of the sensors needs to be uncompromised or if an attacker might be able to mislead the fusion engine by generating huge false alarms, the alarms generated though those type of sensors should be discounted accordingly.

Practically speaking, no sensor generates 100 % true alerts. This means that all the generated alarms might not be correct. The capability of a sensor could be less effective for some types of attack domains. To address this issue, final result from any sensor should be degraded accordingly before sending the result into the fusion engine. Without addressing this issue, we might mislead the fusion engine for making the right decision.

Let $S_1$and S_2 are two independent sensors. We introduced Subjective Logic for each sensor such that the fusion engine has predefined subjective opinion relating both sensors. We deal fusion engine opinion relating any sensor through subjective logic discount operator. We then used subjective logic consensus operator in order to find the final opinion about an attack happening by eliminating the distrust of fusion engine on any sensor. Let Fusion engine F receive alert a_1 from sensor $S_1$and a_2 from sensor S_2 relating an attack A_1occurrence. Fusion System F would have formed opinion $\omega_{A1}^{F(a1)}$ and $\omega_{A1}^{F(a2)}$ about the attack A_1. But, in our case, sensors are not fully trustworthy; therefore Fusion Engine's opinion relating both sensors, ω_{S1}^{F} for S_1 and ω_{S2}^{F} for S_2 should be discounted in order to receive correct fusion result of both independent sensors. We have final opinion as follows:

$$\omega_{A1}^{F(S_1(a1)),(S_2(a2))} = \omega_{a1}^{F,S1} \oplus \omega_{a2}^{F,S2}$$

Where

$$\omega_{a1}^{F,S1} \equiv (\omega_{S1}^{F} \otimes \omega_{a1}^{S1}) = (b_{a1}^{F,S1},\ d_{a1}^{F,S1},\ u_{a1}^{F,S1},\ a_{a1}^{F,S1})$$

$$b_{a1}^{F,S1} = b_{S1}^{F} b_{a1}^{S1}$$

$$d_{a1}^{F,S1} = b_{S1}^{F} d_{a1}^{S1}$$

$$u_{a1}^{F,S1} = d_{S1}^{F} + u_{S1}^{F} + b_{S1}^{F} u_{a1}^{S1}$$

$$a_{a1}^{F,S1} = a_{a1}^{S1}$$

and

$$\omega_{a2}^{F,S2} \equiv (\omega_{S2}^{F} \otimes \omega_{a2}^{S2}) = (b_{a2}^{F,S2},\ d_{a2}^{F,S2},\ u_{a2}^{F,S2},\ a_{a2}^{F,S2})$$

$$b_{a2}^{F,S2} = b_{S2}^{F} b_{a2}^{S2}$$

$$d_{a2}^{F,S2} = b_{S2}^{F} d_{a2}^{S2}$$

$$u_{a2}^{F,S2} = d_{S2}^{F} + u_{S2}^{F} + b_{S2}^{F} u_{a2}^{S2}$$

$$a_{a2}^{F,S2} = a_{a2}^{S2}$$

In more compact form, we can say;

$$\omega_{A1}^{F(S_1(a1)),(S_2(a2))} = \omega_{a1}^{F,S1} \oplus \omega_{a2}^{F,S2} = (\omega_{S1}^{F} \otimes \omega_{a1}^{S1}) \oplus (\omega_{S2}^{F} \otimes \omega_{a2}^{S2})$$

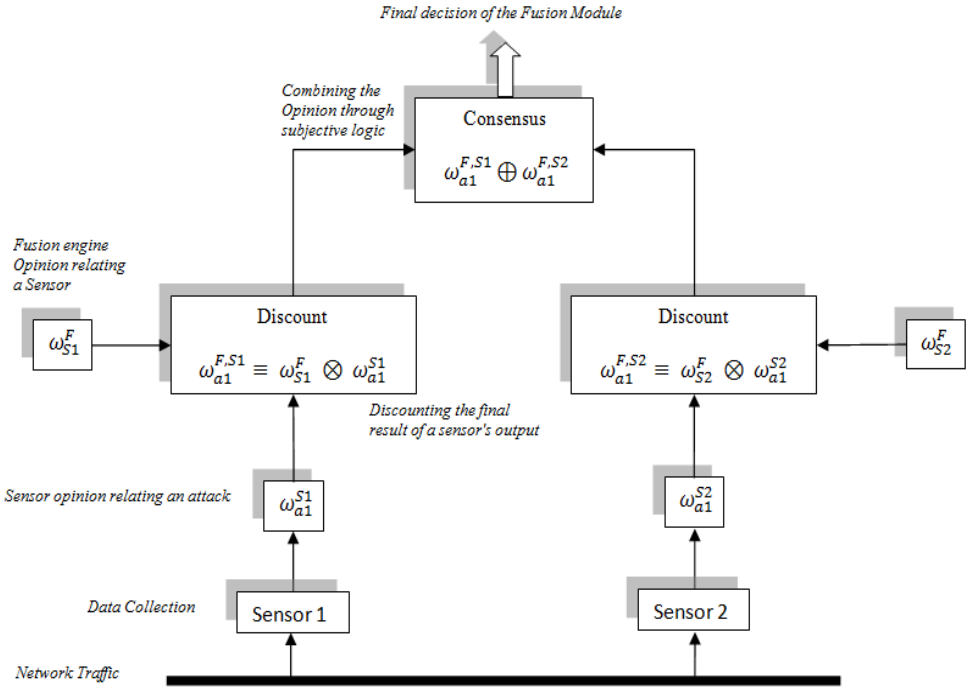

Figure 21 Fusion Module based on Subjective Logic

The architecture of Fusion Module based on Subjective logic

The final result of $\omega_{a1}^{F,S1}$ and $\omega_{a2}^{F,S2}$ are joined through consensus operator in order to conclude the final possible state of an attack A_1 reported by 2 uncertain sensors S_1 and S_2 with different degree of reliability in an independent environment.

$\omega_{A1}^{F(S_1(a1)),(S_2(a2))}$ can be defined in two separate ways. One for K = 0 and other for K ≠ 0

where $K = (u_{a1}^{F,S1} + u_{a2}^{F,S2} - u_{a1}^{F,S1} . u_{a2}^{F,S2})$

For K ≠ 0

$$b_{A1}^{F(S_1(a1)),(S_2(a2))} = (b_{a1}^{F,S1} . u_{a2}^{F,S2} + b_{a2}^{F,S2} . u_{a1}^{F,S1})/ K$$

$$d_{A1}^{F(S_1(a_1)),(S_2(a_2))} = (d_{a1}^{F,S1}.u_{a2}^{F,S2} + d_{a2}^{F,S2} . u_{a1}^{F,S1}) / K$$

$$u_{A1}^{F(S_1(a_1)),(S_2(a_2))} = (u_{a1}^{F,S1} . u_{a2}^{F,S2}) / K$$

$$a_{A1}^{F(S_1(a_1)),(S_2(a_2))} = a_{a2}^{F,S2}.u_{a1}^{F,S1} + a_{a1}^{F,S1}.u_{a2}^{F,S2} - (a_{a1}^{F,S1} + a_{a2}^{F,S2})u_{a1}^{F,S1}.u_{a2}^{F,S2} /$$

$$u_{a1}^{F,S1} + u_{a2}^{F,S2} - 2\,u_{a1}^{F,S1}.u_{a2}^{F,S2}$$

and for K = 0:

$$b_{A1}^{F(S_1(a_1)),(S_2(a_2))} = (b_{a1}^{F,S1} + b_{a2}^{F,S2}) / 2$$

$$d_{A1}^{F(S_1(a_1)),(S_2(a_2))} = (d_{a1}^{F,S1} + d_{a2}^{F,S2}) / 2$$

$$u_{A1}^{F(S_1(a_1)),(S_2(a_2))} = 0$$

$$a_{A1}^{F(S_1(a_1)),(S_2(a_2))} = (a_{a1}^{F,S1} + a_{a2}^{F,S2}) / 2$$

CHAPTER IV

ANALYSIS OF RESULTS

This chapter reports the analysis of evaluation methods conducted to validate the effectiveness of our proposed approach. We also present the complete analysis of our reported results to judge the impact of our proposed methodology towards the final goal of the thesis. Complete set of accomplished results for data collection, alert normalization, alert preprocessing and alert filtering are presented in Appendix-J along with multilevel correlation results. Appendix-J also presents the complete series of results for fusion process based on DS theory and Jøsang subjective logic.

The proposed approach for each component is discussed in previous chapter. For the sake of testing and evaluating any new the proposed framework, DARPA has provided a number of Intrusion Detection Evaluation Data Sets including 1998, 1999 2000 Datasets. In order to validate the effectiveness and efficiency of our proposed framework, the DARPA 1999 intrusion detection dataset was used and the generated alerts were stored in a MySQL database. For this purpose, separate database was created for each sensor. Java was used to implement various proposed algorithms. The final output of each sensor is converted into the standard formats in order to apply the various correlation and fusion rules.

Result Analysis:

To analyze the impact of our proposed prototype towards the final goal of our research work i.e. minimize the false positive alerts through correlation and fusion process, in this section we present the detail analysis of our final results for each major component. Mainly the whole process is divided into two major components i.e. alert correlation and

alert fusion. We present analysis of these two components individually for each sensor (Snort and Bro) including every attack category (ICMP flooding, UDP flooding and TCP flooding)

Analysis of multilevel correlation component.

Multilevel correlation for Snort sensor: The following horizontal plotted graph of DARPA 1999 dataset for 9 different days from March 29, 1999 to April 09, 1999 shows various alerts number. Dark horizontal line represents total alerts generated by Snort sensor. The light color horizontal line beside each dark line, against each date shows the remaining alerts after applying proposed multilevel correlation rules for Snort sensor.

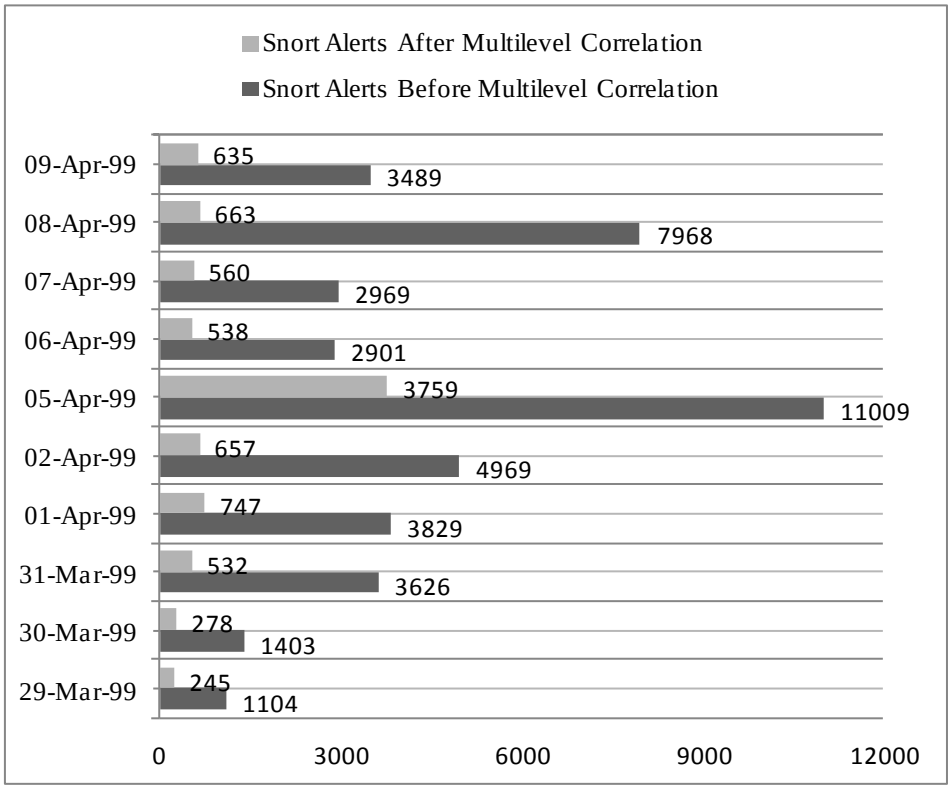

Figure 22 (a) Multilevel Correlation - Snort

The same information is represented in the following figure. It is obvious form the graph that our proposed correlation technique for Snort sensor reduced huge number of false

positives alerts that were not the actual attacks. The vertical straight line between the two lines (dark color and light color) shows the actual false alerts, reduced through proposed methodology.

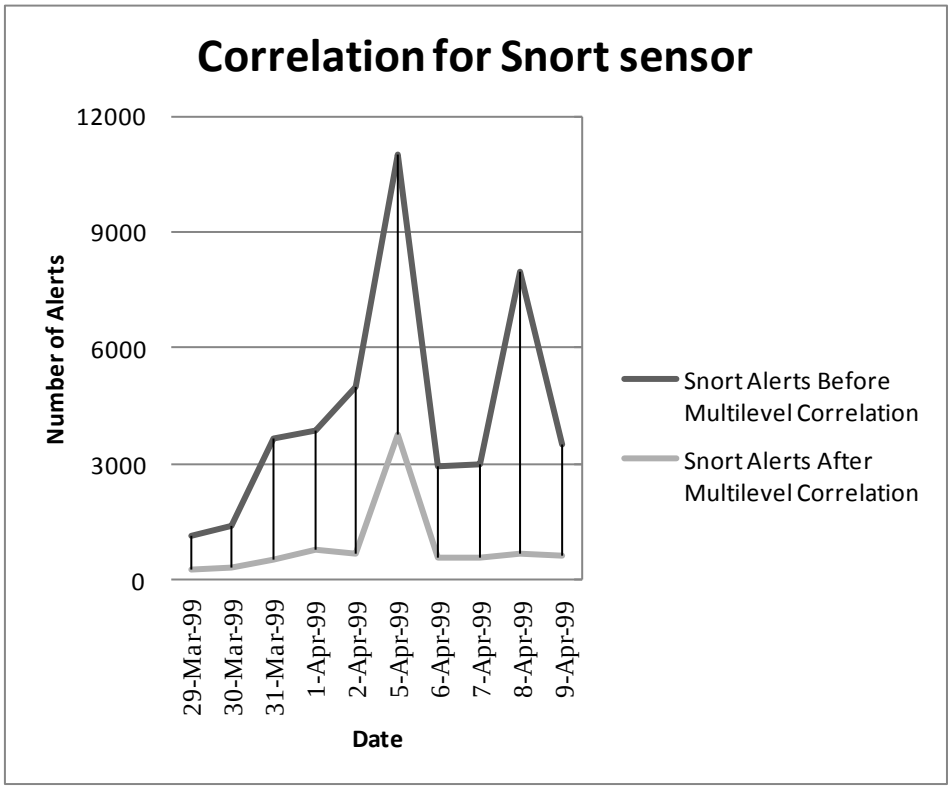

Figure 23(b) Multilevel Correlation - Snort

In the following figure, alert reduction rate for Snort senor is drawn to show the effectiveness of our proposed framework towards reduction of false positive alerts. Alert reduction rate for each day mention in the graph. The average reduction is more than 80 % for Snort sensor.

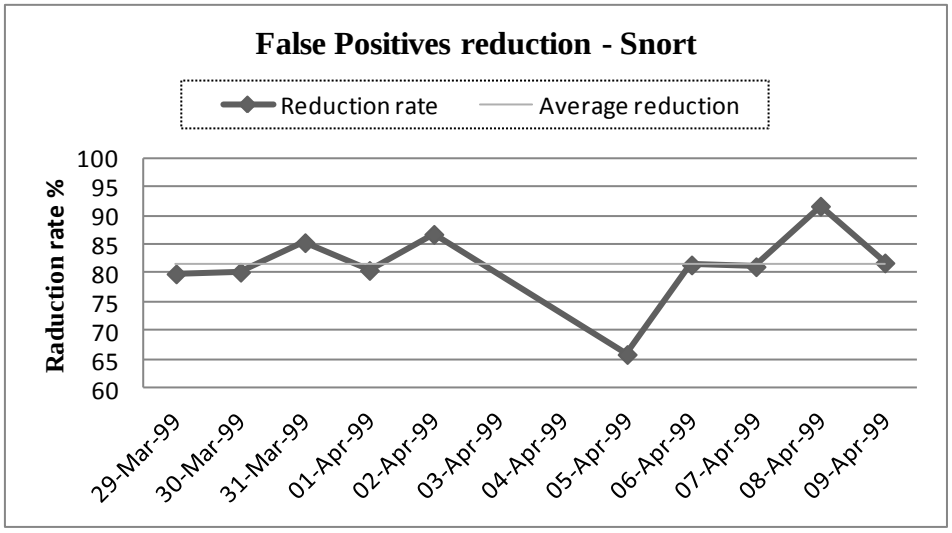

Figure 24 False positive reduction rate - Snort

Multilevel correlation for Bro sensor: The following horizontal plotted graph of DARPA 1999 dataset for 9 different days from March 29, 1999 to April 09, 1999 shows various alerts number. Dark horizontal line represents total alerts generated by Bro sensor. The light color horizontal line beside each dark line, against each date shows the remaining

alerts after applying proposed multilevel correlation rules for Bro sensor.

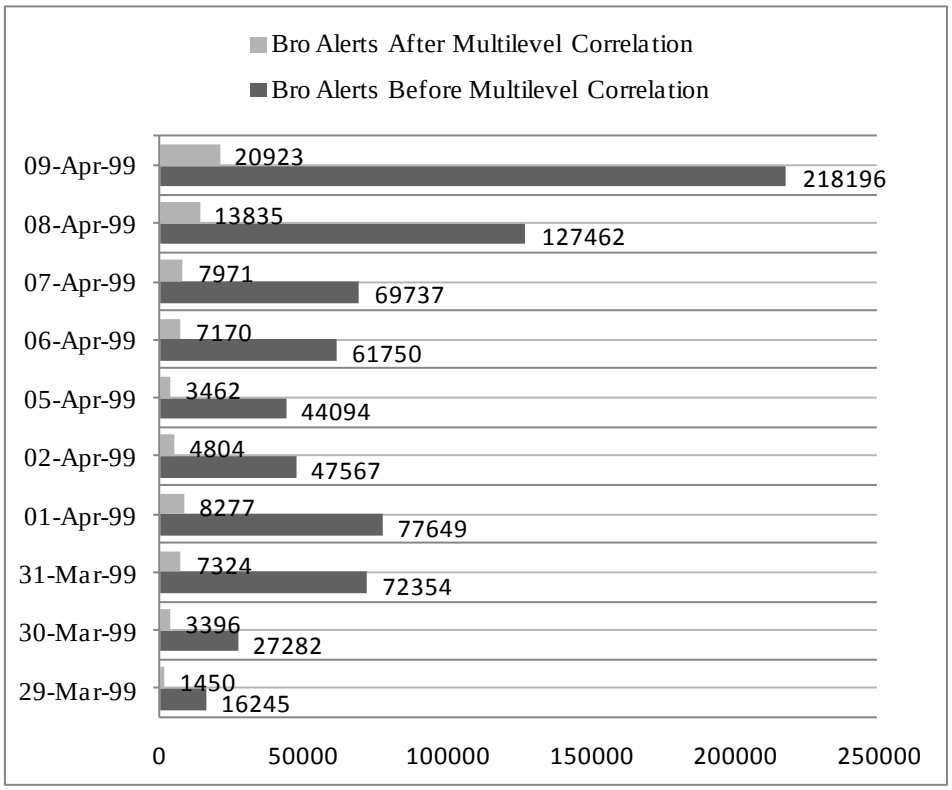

Figure 25 (a) Multilevel Correlation - Bro

The same information is presented in the following figure. It is obvious form the graph that our proposed correlation technique for Bro sensor also reduced huge number of false positives alerts that were not the actual attacks. Overall reduction is more than 89 % in this case. The vertical straight line between the two lines (dark color and light color) shows the actual false alerts, reduced through our proposed methodology.

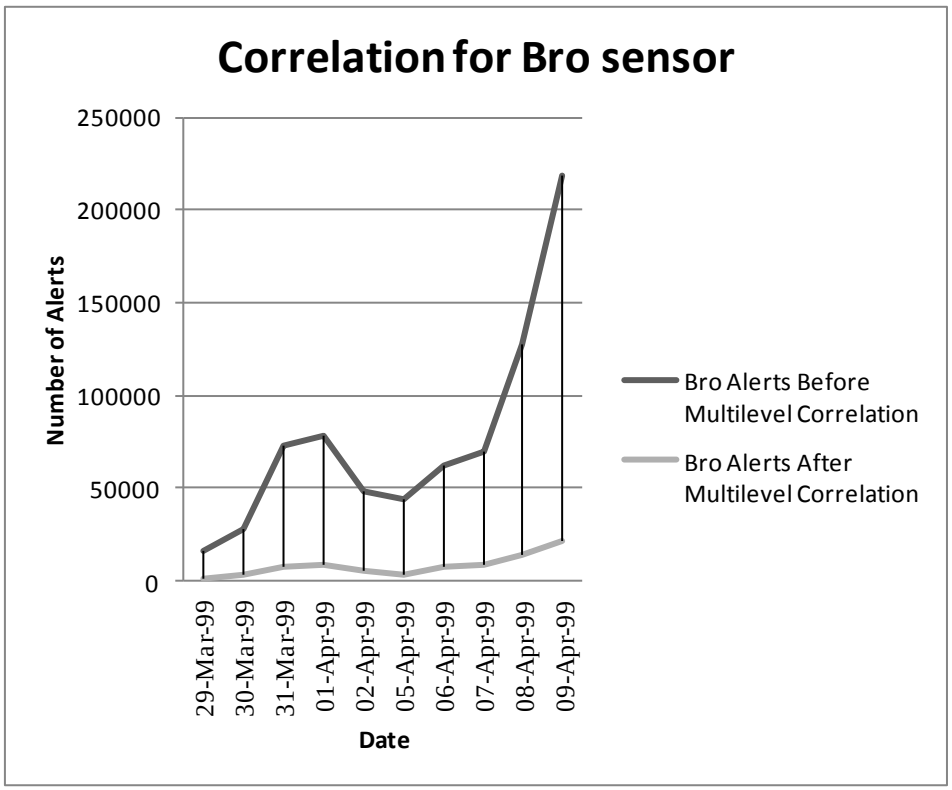

Figure 26 (b) Multilevel Correlation -Bro

In the following line graph, alert reduction rate for Bro senor is drawn to illustrate the effectiveness of our proposed framework towards reduction of false positive alerts. Alert reduction rate for each day mention in the graph. The average reduction is more than 89 % for Snort sensor.

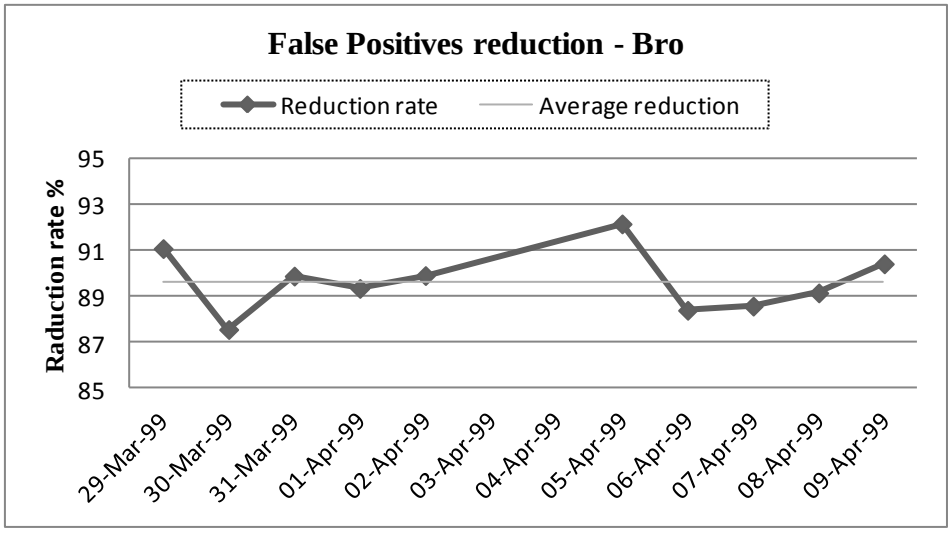

Figure 27 False positive reduction rate - Bro

Analysis of fusion component

Following figures shows the final result of attack scenario derived through combining the bpa values of each sensor (Snort and Bro) for each attack category (ICMP and SYN). UDP alerts are not combined due to absent of any UDP alert detection by sensor Snort.

DS Rule of Combination: From the following two figures, it is obvious that by combining the evidence using Dempster Shafter Rule of Combination, collected through two independent sources provide us more certain situation of any attack happening. Based on these fusion engine decisions, we can further mark those alerts which are associated with No-attack entry as false positives. 30-Mar-99 for ICMP flooding and 29-Mar-99, 30-Mar-99 and 05-Apr-99 for SYN flooding are the entries with No attack. Some slots has high certainty level of attack happening like 29-Mar-99 for ICMP flooding with .9275 certain value of an ICMP DDoS attack happening. The alerts associated with high level of certainty level most likely contribute in DDoS attack in the particular time slot.

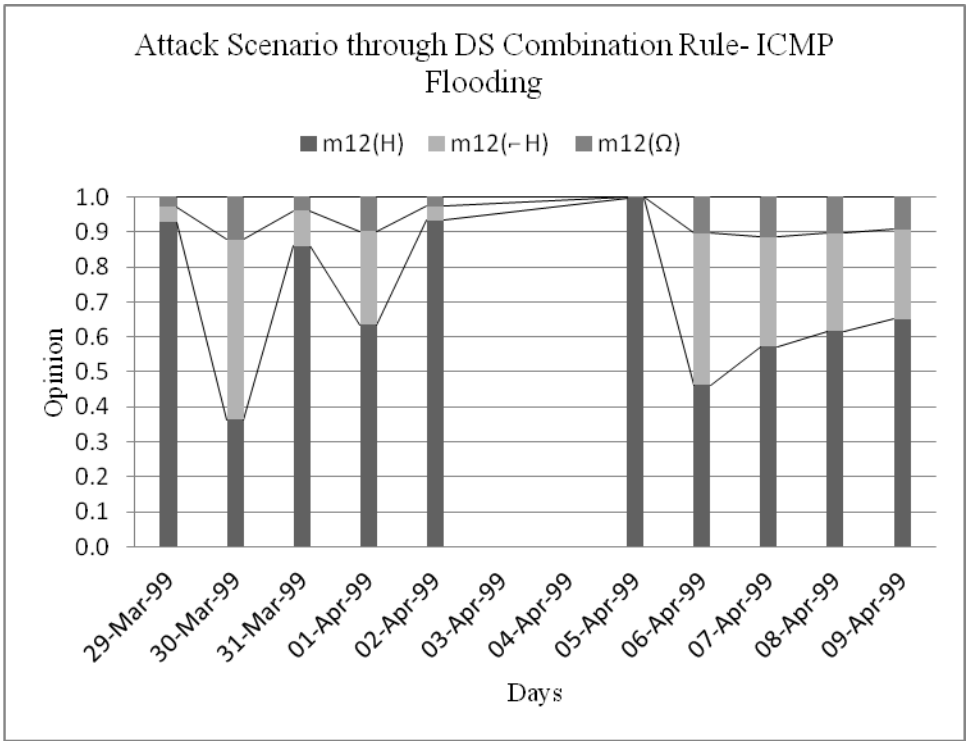

Figure 28 Attack Scenario through DS combination rule - ICMP Flooding

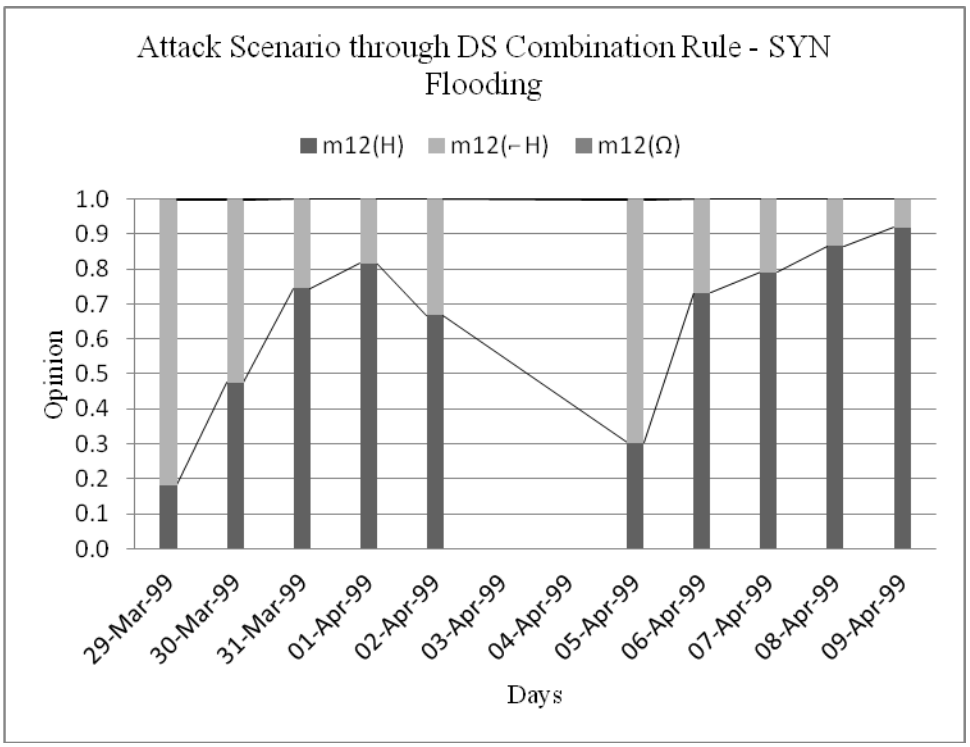

Figure 29 Attack Scenario through DS combination rule - SYN Flooding

Jøsang Subjective Logic: The following figures represents final decision relating any attack happening derived through Jøsang subjective logic approach. Higher the belief level (b) represent the actual attack happening. The more value for b shows more certain, the attack occurred. The state of the network is undecided when uncertain value (u) is higher. When the disbelief value (d) is higher, it shows the network is not under DDoS attack. 30-Mar-99, 01-Apr-99, 06-Apr-99, 07-Apr-99, 08-Apr-99 and 09-Apr-99 for ICMP flooding attack. In the case of SYN flooding attack 29-Mar-99, 30-Mar-99 and 05-Apr-99 are time slots (or the date) when state of the network is undecided because of higher the value of u. Please note that the same time slots in Dempster Shafer Rule of Combination are marked as No-Attack. As a comparison of both fusion rules, one thing is obvious that the Jøsang subjective approach gives more undecided states then Dempster

Shafer Rule of Combination. The reason behind this high level of undecided states is the use of less trusted sensor.

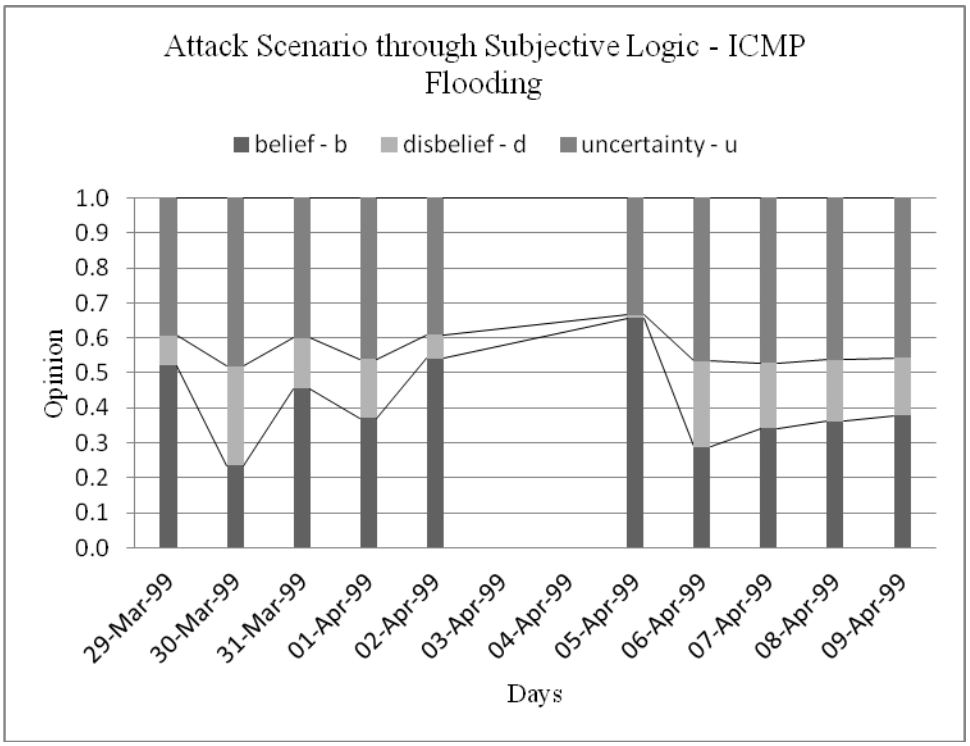

Figure 30 Attack Scenario through Subjective Logic - ICMP Flooding

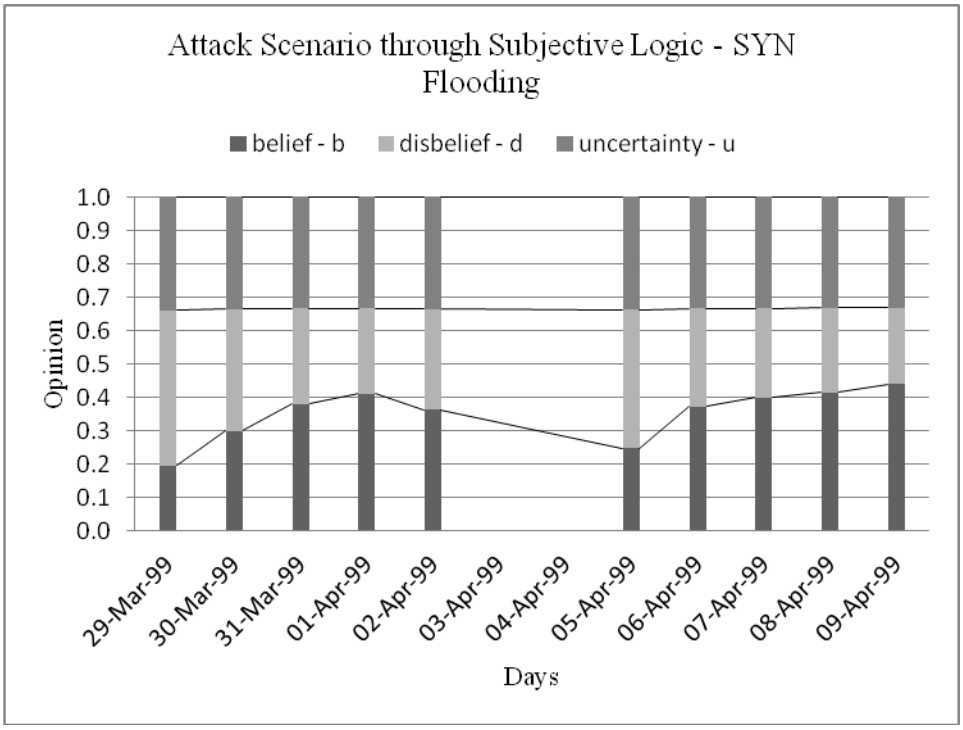

Figure 31 Attack Scenario through Subjective Logic - SYN Flooding

Discussion

There are many challenges for applying this type of attack analysis for determining the final opinion values of the observed alarms especially when the sensors are not fully trustworthy. The major challenge is to set a threshold value for any sensor for generating wrong information relating any situation assessment. Most of the sensors can be tunes (in case, these options are available to the end user) for generating dogmatic opinion where uncertainty level for their assessment is zero. In the case of consensus, only the sources with some degree of uncertainty can be used to generate a final consensus result. There are many ways to discount the opinion of a sensor, for example, one way is to drive the opinion through previous ability of the sensor for correctly detecting specific type of alarms. Another way is based on the various attributed that might affect the overall

performance of the sensor i.e. network type, network traffic type, location of the sensor installation, configuration changes within the sensor itself and various components configuration within the whole intrusion detection systems. We put our selected sensors on default setting and based on their degree of correct alert detection rate, discount their opinion. Through our research work, we concluded the right choice of different sensors for the fusion component is based on their independence on each other and also on their ability to detect various attack types. Our sensor selection options were very limited due to their availability. We used open source sensors with fully network support and used DARPA offline dataset to analyse their ability for detecting various attacks. Some researchers create their own test bed raw data to evaluate the proposed fusion component. In that case, realistic data and alerts should be included in the network traffic. This whole test bed requires huge system requirements and hardware support that was not possible in our case.

CHAPTER V

CONCLUSIONS AND RECOMMENDATIONS

This chapter revisits the goal of this thesis and state the final conclusions. Furthermore, it discusses the major contribution of this thesis work in the field of research domain. Finally, possible directions for future research and recommendations are given.

Conclusion and summary of contribution:

Deployment of an Intrusion Detection System is not sufficient for detecting a real intrusion. The issue of managing large number of generated alerts is a challenging task for any security analyst. The goal of this work, as stated earlier, is to propose, design and develop architecture for a fusion-based false alarm reduction module that work with existing Intrusion Detection Systems. Our proposed methodology has the ability to deal with all these aspects by introducing following modules

- Alert collection module
- Alert Normalization module
- Alert Preprocessing module
- Alert Filtering module

Each of these modules has positive effects on reduction of false positive alerts up to some extent. The major goal that is, false positive reduction is achieved through

- Multi level correlation module
- Multi level fusion module

Our correlation module is based on some simple and more complex correlated rules based on the signature types of the detected events. The alert fusion module is based on Dempster Shafer Theory. The collected evidence from various sensors is combined

through DS Rule of Combination. The proposed prototype introduces subjective logic for each sensor to deal with uncertainty level of each source individually. We also combined our collected belief through consensus operator proposed by Jøsang [1999] and compare the final results with DS Rule of Combination.

Our proposed architecture for multi sensor alert reduction and fusion frame work is discussed in chapter 3, seems promising in its ability to eliminate the false positive alerts and minimizing the huge number of false alerts. Moreover, the initial steps in the proposed architecture are critical for the smooth functioning of our correlation and fusion modules.

In this thesis work on correlation and fusion approaches, more time was devoted towards the combination rules in Dempster Shafer theory and Jøsang subjective logic with binominal opinion representation. More consideration was given to the accurate mathematical representation of conflict in the intrusion domain, amount and accuracy of evidence and reliability of independent evidence collection sources. We also noticed that Dempster's rule of combination is the most appropriate combining rule in case of all the evidence collection sources are reliable. This also leads to the conclusion that various combination rules handle the same situation differently. We concluded through our research work that there are number of considerations that need to be address during combining the evidence through Dempster-Shafer theory. These considerations include the level of evidence itself, the source of information, the particular domain and various operation used for combining the evidence.

Our thesis also presents the method to fuse the false alarms for presenting real attack scenario. This process is being achieved through reasoning method relating various

intrusion attacks by considering belief factor associated with these alerts and later combining the alarm with subjective logic. These processes reduce the false alarms rate and increase the trustworthiness of the final result. Our system also considers the reliability associated with each sensor and discounts its result accordingly. By applying subjective logic to our fusion system, we noticed that fusion module draw its own opinion about the situation by using alarms generated through the sensors and the reliability of a sensor.

Recommendations and Future Work

Intrusion detection is very important in order to preventing the system again an attack. Future generation Network Intrusion Detection Systems (NIDS) will be more incorporated into the integrated hybrid network security systems. Separate security solution will not be a solution any more. For developing such systems, combination of more complex mathematical theories and various self sustain modules would be more helpful in order to obtain true information relating an attack instance. The network attacks are becoming more distributed and complex. To detect such complex intrusions, NIDS need to report intrusion information in a timely manner and compact to make the IDS response more effective. When we talk about the compact form, it means with less false positive alerts. To build an effective security suit, reliable security technologies need to be developed and tested.

As an extension of this work presented in chapter 3, we may consider the study of various more effective security rules where the architecture include firewalls, more sophisticated IDS and intrusion prevention devices. Our system addresses to single final result i.e, an attack or not an attack based on evidence collection and combination rules.

By introducing more devices in the proposed system, more complex system policies are needed to address the issue of intrusion prevention along with intrusion detection and minimizing huge false positive alerts.

In parallel to the above proposed possible extension, the dynamic real intrusion detection system may be aligning with false positive reduction module. This system will help to refine the final result about the current system intrusion along with other available IDS like Snort and Bro.

The current system can be extending to detect and eliminate false positives for not only ICMP, UDP and SYSN flooding but also for other categories. More efficient database mining approaches can be introduced for efficient data manipulation in order to decide about the current state of an event.

Another possible extension to the current proposed architecture would be the detailed analysis of the current situation of the system and let the system decide to launch an automatic response. This work needs to reconfigure the security policies randomly in order to prevent new occurrence of a particular intrusion. This random reconfiguration of system policies may result in complete failure of some system components if proper data follow among these components are not carefully handled through low level of software design policies.

Our proposed system of alert reduction may be implemented on the other combination rules (other than Dempster Shafer Rule of Combination and Jøsang Consensus and Discount operators) like Yager;s rule, Inagaki's rule and Zhang's rule. Various situations can be implemented individually on a single system for making more refined final decision.

Our proposed correlation techniques can be further enhanced by introducing more rules based on other possible intrusion signatures. Different IDS can be used for this purpose for generating different signature types for other type of network traffic (other than ICMP, UDP and TCP).

In our proposed framework, materialized view of the database object can be introduced. It will solve the delay issue for database query response time especially in rapid data growth. This model will enable much more efficient access of the query at low cost.

Finally, our proposed prototype can be tested on real data in a real network environment. At the moment, this was not possible for us. Therefore DARPA 1999 dataset was used which is the number 1 choice for all researchers in the research domain. Our system can be further extending by including other Open Source IDS beside existing one. Moreover, we can further divide our time frame window into smaller intervals. This might increase our algorithm complexity but on the other hand will be helpful for modeling final decision with more confident.

The main purpose for the development of the proposed architecture for false positive reduction through alert correlation and alert fusion methodology with collaboration with existing IDS is to enhance the performance of the security suit in terms of less memory utilization, speed up the intrusion detection procedure and to minimize the negative effects of huge false event. Our whole system is mainly based on the fact that using multiple IDS in a network environment is a good solution to increase the intrusion detection rate. The final output of this research work will definitely simplify the network administrator's job for maintaining the network in top possible secure level.

APPENDICES

APPENDIX A

System Configuration and Experimental Test Bed

We used three laptop systems for our test beds. Centralized database engine is running on one machine. Both IDS were installed for scanning/sniffing and data collection on the fastest possible machines. Third laptop is used to perform various tests and system parameters for smooth execution of the whole test bed. The detailed configuration with system specification is mention in the following section

System Specification

Table 4 System Specification

Sr. #	Hardware	System Specification	Plate form
1	Laptop 1	64 bit Intel (R) Core (TM) i5 @ 2.4 GHz. 8 GB RAM with 500 GB available disk space	Ubuntu Release 11.10 64 bit
2	Laptop 2	64 bit Intel (R) Core (TM) i5 @ 2.4 GHz 8 GB RAM with 500 GB available disk space	Ubuntu Release 11.10 64 bit
3	Laptop 3	32 bit Intel (R) Dual Core (TM) Dual Core @ 2.00 GHz 4 GB Ram with 100 GB available disk space	Ubuntu Release 11.04 Kernal Linux 2.6.32-40-generic GNOME 2.30.2

4	Desktop	32 bit Intel (R) Dual Core @ 3.2 GHz 4 GB Ram with 125 GB available disk space	Ubuntu Release 11.04 Kernal Linux 2.6.32-40-generic GNOME 2.30.2
5	Network card		
6	Router	Linksys WRT54G wifi gateway with Broadcom BCM5354 @ 240 MHz with 8 MB RAM	4+1 port network switch

Table 5 Services running on each system

Sr. #	Systems	Services running on each System
1	Laptop 1	Snort package, Barnyard2, Acid Base, MySQL
2	Laptop 2	Bro package with all dependences, MySQL
3	Laptop 3	Nessus, Eclipse, JDBC

Acquiring Ubuntu Linux.

Ubuntu's download procedure changes over the time so going directly to Ubuntu's website is much prefer choice than installing Ubuntu's through its repository. Open the download address *http://www.ubuntu.com/download/ubuntu/download* and carefully select latest download version or older version of your choice depending on your system specification and architecture. Other option for getting Ubuntu is either through installation CD or by downloading through BitTorrent and burn the CD or install through flash drive.

To ensure that the Ubuntu operating system has all latest patches installed, enter following commands on terminal prompt.

Sudo apt-get update

Sudo apt-get upgrade

Follow the various steps for Ubuntu's installation and configuration depending on your version of choice. Whole process takes almost 30 minutes to an hour depending on your system specification and personal choice of selecting various packages.

APPENDIX B

IDS Installation and Configuration

As mentioned earlier, we used Snort and Bro, open source IDS for our experiments. For this purpose, we select the best possible available systems (laptop1 and laptop2) and installed both sensors on each of them. Installation process and configuration process is briefly mention in the following section.

Snort

Pre installation packages for Snort IDS: Some basic packages are needed to install and configure before starting Snort installation process. These packages can be installing through their repositories. Installing via RPM is quite easy and quick method. Remotely Package Manager known as RPM in Linux operating system is a powerful command line driven package management system capable of installing, verifying, querying and updating computer software packages. These processes also download the package to the system by providing correct URLs directly though machine terminal command prompt provided that system are connected to the internet.

The following are some basic packages listing, need to install and configure before Snort installation process begin. Installation can be done by entering command sudo apt-get *install <package name>. You need Ubuntu's root password.*

sudo apt-get install nmap

sudo apt-get install nbtscan

sudo apt-get install apache2

sudo apt-get install php5

sudo apt-get install php5-mysql

sudo apt-get install php5-gd

sudo apt-get install libpcap0.8-dev

sudo apt-get install libpcre3-dev

sudo apt-get install install g++

sudo apt-get install bison

sudo apt-get install flex

sudo apt-get install wireshark

sudo apt-get install libpcap-rubyNmap

For installing *MySQL* database package, password must be selected carefully for root user and should not be forgotten. To install MySQL enter the following command on terminal

sudo apt-get install mysql-server

sudo apt-get install libmysqlclient16-dev

Our choice for selecting the Snort IDS is described in chapter 2 page-25 along with its detailed architecture and advantages over the remaining available IDS. We used a dedicated system laptop 1 for Snort installation with 2 network cards. One network card is built in and the other was installed on external port eth0 for testing purpose and system load balancing.

Supporting tools/packages for Snort: Snort also support graphic library for representing Snort reports in the form of pie chart. For this purposed, JpGraph package is used. After downloading the file, execute the following commands form the terminal shell to install JpGraph package.

sudo wget http://hem.bredband.net/jpgraph/jpgraph-1.27.1.tar.gz

sudo mkdir /var/www/jpgraph

sudo tar zxvf jpgraph-1.27.1.tar.gz

sudo cp -r jpgraph-1.27.1/src /var/www/jpgraph/

Snort Report package is an add on package module for the IDS Snort. Latest Snort Report should be selected and download to the system. We used Snortreport-1.3.3 for our system. After downloading the package, execute the command from that downloaded location.

sudo tar zxvf Snortreport-1.3.3.tar.gz -C /var/www/

Snort report need to be configured to reflect the results generated by Snort IDS into MySQL databases. The main configuration file for Snort report package is srconf.php located on location */var/www/Snortreport-1.3.3/srconf.php*. Edit this file in any editor of your choice for configure purpose.

sudo pico /var/www/Snortreport-1.3.2/srconf.php

MySQL password is required to access data from MySQL to Snort reports. Locate the line *$pass = "YOURPASS"*; by pressing <ctrl+w> and enter MySQL password to the *$pass field.*

$pass = "YOURPASS";

Change this to

$pass = "YOURMYSQLPASSWORD";

Save the file and exit the editor mode.

Installing new Data Acquisition API: Latest Snort versions support Data Acquisition Application Program Interface known as API. This API is necessary to use with Snort version 2.9.0 and above. This API helps Snort to capture the network traffic more

efficiently. Download the latest Data Acquisition API to your system. Executing following command from the downloaded location will install it.

sudo tar zxvf daq-0.6.2.tar.gz

cd daq-0.6.2

sudo ./configure

sudo make

sudo make install

sudo ldconfig

Installing libdnet

According to Snort web site, Unix Systems use pcap in the libpcap library and Windows System use port of libpcap known as WinPcap. Our system is Ubuntu, therefore we used libdnet package. Download the file and save to the system. Install libdnet by executing the following commands from the download location.

sudo tar zxvf libdnet-1.12.tgz

cd libdnet-1.12/

sudo ./configure

sudo make

sudo make install

sudo ln -s /usr/local/lib/libdnet.1.0.1 /usr/lib/libdnet.1

Download and install Snort: Download the Snort IDS's latest version from the web site (http://www.Snort.org/start/download) depending on your system hardware architecture. Another option for installation is to install the Snort package from the Ubuntu's repositories. There are chances that latest version of Snort may not be downloaded from

this method. Therefore, we use compile and install procedure of the source code for Snort IDS installation.

From the downloaded location of Snort source code, execution of following commands on the command terminal will install Snort in the selected location. We select location < */home/faisal/Snort* > for Snort installation.

sudo tar zxvf Snort-2.9.1.2.tar.gz

cd Snort-2.9.1

sudo ./configure /home/faisal/Snort

sudo make

sudo make install

sudo mkdir /var/log/Snort

sudo mkdir /var/Snort

sudo groupadd Snort

sudo useradd -g Snort Snort

sudo chown Snort:Snort /var/log/Snort

Downloading and install the latest Snort Rules: Rules for generating any alert against any event are mention in Snort rule sets. According to Snort web site, Sourcefire Vulnerability Research Team (VRT) Rules are the official rules of Snort IDS. There are two types of Snort rules releases. One for registered user and the other for non registered known as subscriber release. We download latest rule set for register user < *Snortrules-snapshot-2921.tar.gz* > from Snort web site.

From the download location, execute the following command to install the package in </*home/faisal/Snort*> location.

sudo tar zxvf Snortrules-snapshot-2910.tar.gz -C /home/faisal/Snort

sudo mkdir /home/faisal/Snort/lib/Snort_dynamicrules

sudo cp /home/faisal/Snort/so_rules/precompiled/Ubuntu-10-4/i386/2.9.1.0/ \ /home/faisal/Snort/lib/Snort_dynamicrules*

sudo touch /home/faisal/Snort/rules/white_list.rules

sudo touch / home/faisal/Snort/rules/black_list.rules

Snort Configuration: The basic file for Snort configuration is < Snort.conf > located at location < /*home/faisal/Snort/etc/Snort.conf* >. We edit this file for making all necessary changes in order to configure Snort according to our requirement and system configuration.

sudo pico /home/faisal/Snort/etc/Snort.conf

Locate the following lines by pressing <ctrl+w> and change following lines

var WHITE_LIST_PATH /etc/Snort/rules

var BLACK_LIST_PATH /etc/Snort/rules

To these lines

var WHITE_LIST_PATH /home/faisal/Snort/rules

var BLACK_LIST_PATH /home/faisal/Snort/rules

Once again locate the following lines by pressing <ctrl+w> and change following

dynamicpreprocessor directory /home/faisal/lib/Snort_dynamicpreprocessor/

dynamicengine /home/faisal/lib/Snort_dynamicengine/libsf_engine.so

dynamicdetection directory /home/faisal/lib/Snort_dynamicrules

To these lines

Dynamicpreprocessordirectory /home/faisal/Snort/lib/Snort_dynamicpreprocessor/

Dynamicengine/ home/faisal/Snort/lib/Snort_dynamicengine/libsf_engine.so

dynamicdetection directory /home/faisal/Snort/lib/Snort_dynamicrules

Locate the following line in <Snort.conf> file. Below this line

*#output unified2: filename merged.log, limit 128, nostamp, *

mpls_event_types, vlan_event_types

Add following line to limit the output file size to Snort logs.

output unified2: filename Snort.u2, limit 128

To implement these changes, save the file and exit the editor.

Downloading and install Barnyard2: Snort is a huge package with very complicated detection rules. To minimize the processing load on the Snort detection engine, Barnyard2 package is being installed. The basic task of Barnyard2 is to read the log files generated by Snort IDS and enter them into the database. Other advantage of using Barnyard2 is to eliminate the chance of alerts lose generated by Snort. If database service is not available, Barnyard2 writes all non written entries to the database, once it is again available. Download the current version of Barnyard2 package and save to the system.

We unpack the Barnyard2 package through following commands for installation on specified location *</home/faisal/Snort/barnyard2-1.8>*

sudo tar zxvf barnyard2-1.8.tar.gz

cd barnyard2-1.8

sudo ./configure --with-mysql

sudo make

sudo make install

sudo cp etc/barnyard2.conf /home/faisal/Snort/etc

sudo mkdir /var/log/barnyard2

sudo chmod 666 /var/log/barnyard2

sudo touch /var/log/Snort/barnyard2.waldo

sudo chown Snort.Snort /var/log/Snort/barnyard2.waldo

Configuring Barnyard2: After the Barnyard2 installation process, necessary changes and editing was made in Barnyard configuration file <*barnyard2.conf*> in order to configure the package according to our test bed parameters.

sudo pico /home/faisal/Snort/etc/barnyard2.conf

Locate the following line in the file by pressing <ctrl+w> and change them

config reference_file: /etc/Snort/reference.config

config classification_file: /etc/Snort/classification.config

config gen_file: /etc/Snort/gen-msg.map

config sid_file: /etc/Snort/sid-msg.map

#config hostname: thor

#config interface: eth0

#output database: log, mysql, user=root password=test dbname=db host=localhost

To following lines

config reference_file: /home/faisal/Snort/etc/reference.config

config classification_file: /home/faisal/Snort/etc/classification.config

config gen_file: /home/faisal/Snort/etc/gen-msg.map

config sid_file: /home/faisal/Snort/etc/sid-msg.map

config hostname: localhost

config interface: eth1

*output database: log, mysql, user=Snort password=YOURMySQLPASSWORD dbname=Snort *

host=localhost

Creating MySQL Database for Snort: Snort need separate database to store output alerts. We create separate Snort user and separate database name Snort. To execute these steps, we perform following steps from Ubuntu's terminal. First step is to create Snort database.

echo "create database Snort;" | mysql -u root -p

mysql -u root -p -D Snort < ./schemas/create_mysql

For creating Snort user with privileges to manipulate Snort database, we used following command on the terminal

echo "grant create, insert, select, delete, update on Snort.* to Snort@localhost identified by \ 'MySQLPASSWORD'" | mysql -u root –p

These steps will create user Snort, a database name Snort with following tables.

Table 6 Snort tables list

acid_ag	acid_ag_alert	acid_event	acid_ip_cache
base_roles	base_user	data	detail
encoding	event	icmphdr	iphdr
opt	reference	reference_system	schema
sensor	sig_class	sig_reference	schema
sensor	sig_class	sig_reference	signature
tcphdr	udphdr		

Pre Installation package for Bro IDS : Like Snort installation, Bro installation also requires some basic package to be install for system preparation as IDS node. The following are some basic packages listing, need to install and configure before Snort installation process begin. Installation can be done by entering command *sudo apt-get install <package name>*. You need Ubuntu's root password.

sudo ap-get install libncurses5-dev

sudo ap-get install g++

sudo ap-get install bison

sudo ap-get install flex

sudo ap-get install lib-magic-dev

sudo ap-get install libgeoip-dev

sudo ap-get install libssl-dev

sudo ap-get install build-essential

sudo ap-get install python-dev

sudo ap-get install libpcap-dev

sudo ap-get install cmake

sudo ap-get install swig2.0 (we installed swig1.3 due to unavailability of swig2.0)

Bro Installation: After successfully installing these packages, we install latest version of Bro IDS that is Bro2.0. We download the package file from Bro web site < http://www.Bro-ids.org > in the download section. Installation location for Bro IDS in our system is < */home/faisal/Bro-2.0* >

tar xvzf Bro-2.0.tar.gz

sudo ./configure --prefix=/home/faisal/Bro-2.0

sudo make

sudo make install

sudo make install Brolite

BroControl installation: BroControl is an interactive shell for operating Bro installation process. Two separate modes are provided by BroControl, first is stand-alone mode for single system and second is cluster mode. We used stand-alone mode our system preparation. Download the BroControl and execute following commands from terminal shell will install BroControl. The configuration process is bit tedious for BroControl.

tar xvfz Broctl-1.0.tar.gz

~/ cd Broctl-1.0

sudo ./configure --prefix=/home/faisal/Bro-2.0

make

sudo make install Broctl

For setting path variables include bin.

export PATH=$PATH:$/home/faisal/Bro-2.0/bin

BroControl Configuration: Three main files Broctl.cng, nides.cnf and networks.cfg need to be configuring according to the network architecture and system requirements. These files are located in </home/faisal/Bro-2.0/etc>. These file can be edited to reflect all necessary changes.

Broccoli installation and configuration: According to Bro web site < http://www.Bro-ids.org > Brocoli is a client communication library use for create client sensors for Bro IDS. We followed following steps from download location of Broccoli and then

configuring it with Bro sensor by editing three files, Broctl.cng, nodes.cnf and networks.cfg.

tar xvfz Broccoli-1.8.tar.gz

<Download location> cd Broccoli-1.8

sudo ./configure --prefix=/home/faisal/Bro-2.0

make

sudo make install

For setting path variables include bin.

export PATH=$PATH:$/home/faisal/Bro/bin

For configuring Broccoli package, edit following files located on location *</home/faisal//Bro-2.0/etc >*

Broctl.cng8

nodes.cnf

networks.cfg

Installation and configuration for hf-1.3 package: According to Bro web site, hf is a command line tool for replacing numeric IP address with resolved hostname. Simply from hf file download location, execute following commands will install it.

tar xvzf hf.tar.gz

cd hf-1.3

sudo ./configure --prefix=/home/faisal/Bro-2.0

sudo make

sudo make install

Installation and configuration for cf package: According to Bro web site, cf is a command line tool for replacing numeric Unix timestamp with a readable representation. These easy to understand readable real time stamps are very important for correlation and various fusion techniques.

tar xvzf cf.tar.gz

cd cf-1.2.3

sudo ./configure --prefix=/home/faisal/Bro-2.0

sudo make

sudo make install

We create separate database with the name Bro in MySQL database for Bro IDS. Following tables shows the various Bro database tables names created to save Bro IDS output.

Table 7 Bro-Database tables list

conn	dns	dpd	ftp
http	irc	notice_policy	notice
packet_filter	smtp_entries	smtp	ssh
syslog	weird		

APPENDIX C

Table 8 Attack Dictionary

Alert ID	Signature Name	Catagory
s-1	ICMP Destination Unreachable Port Unreachable	ICMP
s-2	ICMP Destination Unreachable Host Unreachable	ICMP
s-3	ICMP Echo Reply	ICMP
s-4	ICMP Fragment Reassembly Time Exceeded	ICMP
s-5	ICMP PING	ICMP
s-6	ICMP PING *NIX	ICMP
s-7	ICMP PING BSDtype	ICMP
s-8	ICMP Time-To-Live Exceeded in Transit	ICMP
b-1	bad_ICMP_checksum	ICMP
b-2	bad_UDP_checksum	UDP
b-3	Attacks Contribute to SYN Flooding	UDP
s-9	WEB-CLIENT Telnet protocol specifier in web page attempt	SYN
s-10	CHAT IRC nick change	SYN
s-11	CHAT IRC channel join	SYN
s-12	CHAT IRC message	SYN
s-13	ATTACK-RESPONSES Invalid URL	SYN
s-14	ATTACK-RESPONSES 403 Forbidden	SYN
s-15	SHELLCODE x86 NOOP	SYN
s-16	WEB-CLIENT Script Engine Stack Exhaustion Denial of Service	SYN
s-17	SHELLCODE x86 inc ecx NOOP	SYN
s-18	X11 xopen	SYN
s-19	ATTACK-RESPONSES directory listing	SYN
s-20	WEB-CLIENT BIN file download request	SYN
s-21	SHELLCODE x86 inc ebx NOOP	SYN
s-22	WEB-CLIENT overflow attempt	SYN
s-23	WEB-CGI phf arbitrary command execution attempt	SYN
s-24	EXPLOIT javascript handler in URI XSS attempt	SYN
s-25	WEB-CLIENT Portable Executable binary file transfer	SYN
s-26	POLICY potentially executable file upload via FTP	SYN
s-27	BAD-TRAFFIC SSH brute force login attempt	SYN
s-28	FINGER / execution attempt	SYN
s-29	FINGER 0 query	SYN
s-30	FINGER redirection attempt	SYN
s-31	FINGER root query	SYN
s-32	WEB-ACTIVEX DirectAnimation.SequencerControl ActiveX CLSID	SYN
s-33	SHELLCODE Linux shellcode	SYN
s-34	WEB-MISC cookiejacking attempt	SYN
s-35	WEB-MISC Generic HyperLink buffer overflow attempt	SYN
s-36	WEB-CLIENT ExEmbed container buffer overflow attempt	SYN
s-37	WEB-CLIENT integer underflow attempt	SYN

s-38	WEB-CLIENT null string overflow attempt	SYN
s-39	WEB-CLIENT string overflow attempt	SYN
s-40	WEB-CLIENT structure memory corruption attempt	SYN
s-41	SHELLCODE sparc NOOP	SYN
s-42	SHELLCODE x86 setuid	SYN
s-43	RSERVICES rlogin	SYN
s-44	RPC portmap listing TCP	SYN
s-45	SPECIFIC-THREATS	SYN
b-4	id.resp_p	SYN
b-5	port	SYN
b-6	non_IPv4_packet	SYN
b-7	DNS_RR_unknown_type	SYN
b-8	line_terminated_with_single_CR	SYN
b-9	unescaped_special_URI_char	SYN
b-10	irc_line_too_short	SYN
b-11	HTTP_version_mismatch	SYN
b-12	unmatched_HTTP_reply	SYN
b-13	data_after_reset	SYN
b-14	unescaped_%_in_URI	SYN
b-15	window_recision	SYN
b-16	SYN_after_reset	SYN
b-17	DNS_truncated_len_lt_hdr_len	SYN
b-18	DNS_Conn_count_too_large	SYN
b-19	DNS_truncated_RR_rdlength_lt_len	SYN
b-20	excessively_small_fragment	SYN
b-21	fragment_inconsistency	SYN
b-22	fragment_size_inconsistency	SYN
b-23	DNS_label_len_gt_pkt	SYN
b-24	DNS_truncated_ans_too_short	SYN
b-25	binpac exception: out_of_bound: Sys	SYN
b-26	above_hole_data_without_any_acks	SYN
b-27	NUL_in_line	SYN
b-28	fragment_overlap	SYN
b-29	SYN_inside_connection	SYN
b-30	excessively_large_fragment	SYN
b-31	possible_split_routing	SYN
b-32	data_before_established	SYN
b-33	excessive_data_without_further_acks	SYN
b-34	connection_originator_SYN_ack	SYN
b-35	SYN_after_close	SYN
b-36	bad_TCP_checksum	SYN
b-37	binpac exception: string mismatch a	SYN
b-38	malformed_ssh_identification	SYN

APPENDIX D

Various Datasets and Research Institutes

Table 9Datasets with corresponding download addresses

Dataset	Website
Masquerading User Data	http://www.schonlau.net/intrusion.html
DARPA 1998 Evaluation data sets (MIT) Intrusion Detection and Evaluation (IDEVAL)	http://www.ll.mit.edu/mission/communications/ist/index.html
DARPA 1999 Evaluation data sets (MIT) Intrusion Detection and Evaluation (IDEVAL)	http://www.ll.mit.edu/mission/communications/ist/index.html
Knowledge Discovery and Data Mining (KDD) CUP 1999 competition data set	http://kdd.ics.uci.edu/databases/kddcup99/kddcup99.html
TIAA – A Toolkit for Intrusion Alert Analysis	http://discovery.csc.ncsu.edu/software/correlator/ver1.0/
Defcon11 and 17	http://www.pcapr.net/forensics

Table 10 Research Institute with corresponding reference addresses

Institute name	Reference address
Computer Security and Intrusion Detection	http://www.acm.org/crossroads/xrds11-1/csid.html
DEFCON Inc	http://www.defcon.com/index.php/def

	con/
LINCOLN LABORATORY (DARPA) IDEVAL	http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html
UCI KDD Archive (KDD)	http://kdd.ics.uci.edu/
Cyber Defense Laboratory (TIAA)	http://discovery.csc.ncsu.edu/software/correlator/ver1.0/
Mu Dynamic Research Labs	http://labs.mudynamics.com/
Xtractr (Hybrid Cloud Application)	http://www.pcapr.net/xtractr
PCAP	http://sourceforge.net/apps/mediawiki/networkminer/index.php?title=Publicly_available_PCAP_files

APPENDIX E

DARPA 1999 Attacks

Five attacks types are added in DARPA 1999 evaluation dataset to cover wide range of various attacks instances. These five attack types are probe or scan attacks (37 instance), DoS (65 instance), R2L (56 instance), U2R (37 instances) and finally Data (13 instances). These attacks are briefly discussed in the following section with first table for probe and DoS attacks. The second table shows Remote to Local (R2L), User to Root (U2R) and Data attacks. In each of the following 2 tables, first row mention the attack victims. These include Solaris, SunOS, NT, Linux and all other. Brief introduction of all these attack categories and their instances are mention below. More detailed information including detailed signature of each attack instance for DARPA 1999 dataset is available in [3][9][10][12].

Denial of Service Attacks: A denial of service attack is an attack in which the attacker makes some computing or memory resource too busy or too full to handle legal authorize requests, or denies legitimate users access to a machine. There are many varieties of denial of service (or DoS) attacks. Some DoS attacks (like a mailbomb, neptune, or smurf attack) abuse a perfectly legitimate feature. Others (teardrop, Ping of Death) create malformed packets that confuse the TCP/IP stack of the machine that is trying to reconstruct the packet.

Probes: In recent years, a growing number of programs have been distributed that can automatically scan a network of computers to gather information or find known vulnerabilities. These network probes are quite useful to an attacker who is staging a future attack. An attacker with a map of which machines and services are available on a

network can use this information to look for weak points. Some of these scanning tools (satan, saint, mscan) enable even a very unskilled attacker to very quickly check hundreds or thousands of machines on a network for known vulnerabilities.

Table 11 Probes and DoS attacks in DARPA 1999 Dataset

	Solaris	**SunOS**	**NT**	**Linux**	**All**
Probe (37)	portsweep queso	portsweep queso	ntinfoscan portsweep	lsdomain mscan portsweep queso satan	illegal- Sniffer ipsweep portsweep
DoS(65)	neptune pod processtable selfping smurf syslog tcprest warezclient	arppoison land mailbomb neptun pod processtable	arppoison crashiis dosnuke smurf tcpreset	apache2 arppoison nack mailbomb neptune pod processtble smurf tcpreset teardrop udpstorm	

Remote to Local Attacks: A Remote to User attack occurs when an attacker who has the ability to send packets to a machine over a network—but who does not have an account on that machine—exploits some vulnerability to gain local access as a user of that machine. There are many possible ways an attacker can gain unauthorized access to a local account on a machine.

User to Root Attacks: User to Root exploits are a class of exploit in which the attacker starts out with access to a normal user account on the system (perhaps gained by sniffing

passwords, a dictionary attack, or social engineering) and is able to exploit some vulnerability to gain root access to the system.

Data Attacks: This is new attack type added in DARPA 1999 dataset. The prime aim of these attacks is to target those special files that various security policies specifies should be remain on the host machine. Other attack types are prevented to access those high security files but data attacks can have direct access of those files through bypassing the security policies on the host machine.

Table 12 U2L, U2R and Data attacks in DARPA 1999 Dataset

	Solaris	**SunOS**	**NT**	**Linux**	**Cisco**
R2L (37)	dict ftpwrite guest httptunel xlock xsnoop	dict xsnoop	dict framespof netbus netcat ppmarco	dict imap named ncftp phf sendmail sshtrojan xlock xsnoop	snmpget
U2R (37)	eject fdformat ffbconfig ps	loadmodule	casesen ntfsdos nukepw sechole yaga	perl sqlattack xterm	
Data(13)	secret	ntfdos ppmacro	secret sqlattack		

APPENDIX F

Table 13 Attack list detected by Bro sensor

Bro - ICMP Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
b-1	398	0	253	0	253	2172	0	0	0	0
Total	**398**	**0**	**253**	**0**	**253**	**2172**	**0**	**0**	**0**	**0**

Bro - UDP Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
b-2	4	0	12	1	5	2	2	5	11	11
b-3	0	0	0	1	7	12	0	3	4	0
Total	**4**	**0**	**12**	**2**	**12**	**14**	**2**	**8**	**15**	**11**

Bro - SYN Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
b-4	0	0	0	0	0	0	0	0	0	0
b-5	0	0	0	0	0	0	0	0	0	0
b-6	0	0	0	0	0	0	0	0	0	0
b-7	0	0	0	0	0	0	0	0	0	0
b-8	0	0	159	172	92	97	109	112	261	119
b-9	0	72	573	590	242	130	361	233	329	328
b-10	55	0	41	55	58	63	0	58	59	56
b-11	0	0	0	0	0	0	0	0	0	0
b-12	248	1018	2994	2338	1028	395	2047	1693	4926	15498
b-13	745	2306	3292	5120	3119	591	4651	5867	8245	4911
b-14	0	0	0	0	0	0	0	0	0	0
b-15	0	0	0	0	0	0	0	0	0	0
b-16	0	0	0	0	0	0	0	0	0	0
b-17	0	0	0	0	0	0	0	0	0	0
b-18	0	0	0	0	0	0	0	0	0	0
b-19	0	0	0	0	0	0	0	0	0	0
b-20	0	0	0	0	0	0	0	0	0	0
b-21	0	0	0	0	0	0	0	0	0	0
b-22	0	0	0	0	0	0	0	0	0	0
b-23	0	0	0	0	0	0	0	0	0	0
b-24	0	0	0	0	0	0	0	0	0	0
b-25	0	0	0	0	0	0	0	0	0	0
b-26	0	0	0	0	0	0	0	0	0	0
b-27	0	0	0	0	0	0	0	0	0	0
b-28	0	0	0	0	0	0	0	0	0	0
b-29	0	0	0	0	0	0	0	0	0	0
b-30	0	0	0	0	0	0	0	0	0	0
b-31	0	0	0	0	0	0	0	0	0	0
b-32	0	0	0	0	0	0	0	0	0	0
b-33	0	0	0	0	0	0	0	0	0	0
b-34	0	0	0	0	0	0	0	0	0	0
b-35	0	0	0	0	0	0	0	0	0	0
b-36	0	0	0	0	0	0	0	0	0	0
b-37	0	0	0	0	0	0	0	0	0	0
b-38	0	0	0	0	0	0	0	0	0	0
Total	**1048**	**3396**	**7059**	**8275**	**4539**	**1276**	**7168**	**7963**	**13820**	**20912**

APPENDIX G

Table 14 Attack list detected by Snort sensor

Snort - ICMP Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
s-1	40	21	92	56	61	492	73	38	41	72
s-2	47	18	77	73	42	1199	57	42	29	45
s-3	17	11	49	130	211	1484	98	88	138	155
s-4	23	0	0	0	18	153	0	19	17	14
s-5	11	0	0	0	31	198	0	15	11	11
s-6	0	0	0	30	0	0	0	0	0	0
s-7	0	0	0	0	0	51	0	0	0	0
s-8	0	0	0	3	0	0	0	0	0	0
Total	**138**	**50**	**218**	**292**	**363**	**3577**	**228**	**202**	**236**	**297**

Snort - UDP Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
-	0	0	0	0	0	0	0	0	0	0
Total	0	0	0	0	0	0	0	0	0	0

Snort - SYN Flooding										
ID	**29-Mar-99**	**30-Mar-99**	**31-Mar-99**	**01-Apr-99**	**02-Apr-99**	**05-Apr-99**	**06-Apr-99**	**07-Apr-99**	**08-Apr-99**	**09-Apr-99**
s-9	0	0	0	0	0	0	0	16	0	0
s-10	0	0	12	21	19	19	20	13	13	17
s-11	20	0	19	17	13	15	18	14	15	19
s-12	38	33	91	152	99	59	42	37	51	28
s-13	10	0	54	28	21	52	65	31	68	31
s-14	0	21	54	21	27	37	74	41	101	29
s-15	16	82	73	151	76	0	0	28	65	23
s-16	0	0	0	0	0	0	0	0	0	17
s-17	0	0	0	14	12	0	0	0	0	41
s-18	0	0	0	0	0	0	0	0	0	0
s-19	0	11	0	51	27	0	0	31	114	0
s-20	0	0	0	0	0	0	0	0	0	0
s-21	0	0	11	0	0	0	0	0	0	0
s-22	0	0	0	0	0	0	0	0	0	0
s-23	0	0	0	0	0	0	0	0	0	0
s-24	0	0	0	0	0	0	28	0	0	0
s-25	0	0	0	0	0	0	0	0	0	0
s-26	0	0	0	0	0	0	0	0	0	0
s-27	0	81	0	0	0	0	0	0	0	0
s-28	0	0	0	0	0	0	0	0	0	0
s-29	0	0	0	0	0	0	0	0	0	0
s-30	0	0	0	0	0	0	0	0	0	0
s-31	0	0	0	0	0	0	0	0	0	0
s-32	0	0	0	0	0	0	0	0	0	0
s-33	0	0	0	0	0	0	0	0	0	0
s-34	0	0	0	0	0	0	0	0	0	0
s-35	0	0	0	0	0	0	63	147	0	62
s-36	0	0	0	0	0	0	0	0	0	71
s-37	0	0	0	0	0	0	0	0	0	0
s-38	0	0	0	0	0	0	0	0	0	0
s-39	0	0	0	0	0	0	0	0	0	0
s-40	0	0	0	0	0	0	0	0	0	0
s-41	0	0	0	0	0	0	0	0	0	0
s-42	0	0	0	0	0	0	0	0	0	0
s-43	0	0	0	0	0	0	0	0	0	0
s-44	0	0	0	0	0	0	0	0	0	0
s-45	0	0	0	0	0	0	0	0	0	0
Total	84	228	314	455	294	182	310	358	427	338

APPENDIX H

Table 15 Evidence Collection - Snort and Bro sensors

Snort-ICMP Flooding						
Days	r	s	A (icmp)	⌐A (icmp)	Ω (icmp)	Belief Mass
29-Mar-99	10	3	0.6667	0.2000	0.1333	1
30-Mar-99	6	5	0.4615	0.3846	0.1538	1
31-Mar-99	8	5	0.5333	0.3333	0.1333	1
01-Apr-99	13	3	0.7222	0.1667	0.1111	1
02-Apr-99	14	3	0.7368	0.1579	0.1053	1
05-Apr-99	80	2	0.9524	0.0238	0.0238	1
06-Apr-99	9	5	0.5625	0.3125	0.1250	1
07-Apr-99	10	3	0.6667	0.2000	0.1333	1
08-Apr-99	12	3	0.7059	0.1765	0.1176	1
09-Apr-99	14	3	0.7368	0.1579	0.1053	1

Bro-ICMP Flooding						
Days	r	s	A (icmp)	⌐A (icmp)	Ω (icmp)	Belief Mass
29-Mar-99	9	0	0.8182	0.0000	0.1818	1
30-Mar-99	0	1	0.0000	0.3333	0.6667	1
31-Mar-99	7	0	0.7778	0.0000	0.2222	1
01-Apr-99	0	1	0.0000	0.3333	0.6667	1
02-Apr-99	7	0	0.7778	0.0000	0.2222	1
05-Apr-99	45	0	0.9574	0.0000	0.0426	1
06-Apr-99	0	1	0.0000	0.3333	0.6667	1
07-Apr-99	0	1	0.0000	0.3333	0.6667	1
08-Apr-99	0	1	0.0000	0.3333	0.6667	1
09-Apr-99	0	1	0.0000	0.3333	0.6667	1

Snort-UDP Flooding						
Days	r	s	A (udp)	⌐A (udp)	Ω (udp)	Belief Mass
29-Mar-99	0	0	0.0000	0.0000	0.0000	0
30-Mar-99	0	0	0.0000	0.0000	0.0000	0
31-Mar-99	0	0	0.0000	0.0000	0.0000	0
01-Apr-99	0	0	0.0000	0.0000	0.0000	0
02-Apr-99	0	0	0.0000	0.0000	0.0000	0
05-Apr-99	0	0	0.0000	0.0000	0.0000	0
06-Apr-99	0	0	0.0000	0.0000	0.0000	0
07-Apr-99	0	0	0.0000	0.0000	0.0000	0
08-Apr-99	0	0	0.0000	0.0000	0.0000	0
09-Apr-99	0	0	0.0000	0.0000	0.0000	0

Bro-UDP Flooding						
Days	r	s	A (udp)	⌐A (udp)	Ω (udp)	Belief Mass
29-Mar-99	1	0	0.3333	0.0000	0.6667	1
30-Mar-99	0	1	0.0000	0.3333	0.6667	1
31-Mar-99	1	0	0.3333	0.0000	0.6667	1
01-Apr-99	2	0	0.5000	0.0000	0.5000	1
02-Apr-99	2	0	0.5000	0.0000	0.5000	1
05-Apr-99	2	0	0.5000	0.0000	0.5000	1
06-Apr-99	1	0	0.3333	0.0000	0.6667	1
07-Apr-99	2	0	0.5000	0.0000	0.5000	1
08-Apr-99	2	0	0.5000	0.0000	0.5000	1
09-Apr-99	1	0	0.3333	0.0000	0.6667	1

Snort-SYN Flooding						
Days	r	s	A (syn)	⌐A (syn)	Ω (syn)	Belief Mass
29-Mar-99	8	32	0.1905	0.7619	0.0476	1
30-Mar-99	12	31	0.2667	0.6889	0.0444	1
31-Mar-99	18	29	0.3673	0.5918	0.0408	1
01-Apr-99	23	28	0.4340	0.5283	0.0377	1
02-Apr-99	18	28	0.3750	0.5833	0.0417	1
05-Apr-99	12	31	0.2667	0.6889	0.0444	1
06-Apr-99	17	29	0.3542	0.6042	0.0417	1
07-Apr-99	20	27	0.4082	0.5510	0.0408	1
08-Apr-99	21	29	0.4038	0.5577	0.0385	1
09-Apr-99	22	26	0.4400	0.5200	0.0400	1

Bro-SYN Flooding						
Days	r	s	A (syn)	⌐A (syn)	Ω (syn)	Belief Mass
29-Mar-99	25	33	0.4167	0.5500	0.0333	1
30-Mar-99	73	33	0.6759	0.3056	0.0185	1
31-Mar-99	148	31	0.8177	0.1713	0.0110	1
01-Apr-99	173	31	0.8398	0.1505	0.0097	1
02-Apr-99	98	31	0.7481	0.2366	0.0153	1
05-Apr-99	32	31	0.4923	0.4769	0.0308	1
06-Apr-99	150	32	0.8152	0.1739	0.0109	1
07-Apr-99	167	32	0.8308	0.1592	0.0100	1
08-Apr-99	284	31	0.8959	0.0978	0.0063	1
09-Apr-99	431	31	0.9289	0.0668	0.0043	1

APPENDIX I

Table 16 Calculation for Certainty level – Snort sensor

Snort - Before Correlation							
Days	**RA**	**NOA**	**NOTA**	**NOFA**	**FPR**	**TPR**	**TDR**
29-Mar-99	71	1104	43	1061	96.11	3.89	61
30-Mar-99	104	1403	78	1325	94.44	5.56	75
31-Mar-99	143	3626	102	3524	97.19	2.81	71
01-Apr-99	71	3829	38	3791	99.01	0.99	54
02-Apr-99	327	4969	204	4765	95.89	4.11	62
05-Apr-99	111	11009	81	10928	99.26	0.74	73
06-Apr-99	285	2901	139	2762	95.21	4.79	49
07-Apr-99	90	2969	71	2898	97.61	2.39	79
08-Apr-99	159	7968	95	7873	98.81	1.19	60
09-Apr-99	121	3489	97	3392	97.22	2.78	80
Total	**1482**		**948**				**63**

Snort - After Correlation							
Days	**RA**	**NOA**	**NOTA**	**NOFA**	**FPR**	**TPR**	**TDR**
29-Mar-99	71	245	40	205	83.67	16.33	56
30-Mar-99	104	278	67	211	75.90	24.10	64
31-Mar-99	143	532	91	441	82.89	17.11	64
01-Apr-99	71	747	27	720	96.39	3.61	38
02-Apr-99	327	657	181	476	72.45	27.55	55
05-Apr-99	111	3759	67	3692	98.22	1.78	60
06-Apr-99	285	538	137	401	74.54	25.46	48
07-Apr-99	90	560	63	497	88.75	11.25	70
08-Apr-99	159	663	82	581	87.63	12.37	52
09-Apr-99	121	635	85	550	86.61	13.39	70
Total	**1482**		**840**				**56**

RA	Real Attacks
NOA	Number of Alerts
NOTA	Number of True Alerts
NOFA	Number of False Alerts
FPR	False Positive Rate
TPR	True Positive Rate
TDR	True Detection Rate

Table 17 Calculation for Certainty level – Bro sensor

Bro - Before Correlation							
Days	**RA**	**NOA**	**NOTA**	**NOFA**	**FPR**	**TPR**	**TDR**
29-Mar-99	71	16245	39	16206	99.76	0.24	55
30-Mar-99	104	27282	42	27240	99.85	0.15	40
31-Mar-99	143	72354	94	72260	99.87	0.13	66
01-Apr-99	71	77649	26	77623	99.97	0.03	37
02-Apr-99	327	47567	130	47437	99.73	0.27	40
05-Apr-99	111	44094	69	44025	99.84	0.16	62
06-Apr-99	285	61750	193	61557	99.69	0.31	68
07-Apr-99	90	69737	49	69688	99.93	0.07	54
08-Apr-99	159	127462	68	127394	99.95	0.05	43
09-Apr-99	121	2181196	51	2181145	100.00	0.00	42
Total	**1482**	**2725336**	**761**	**2724575**			**51**

Bro - After Correlation							
Days	**RA**	**NOA**	**NOTA**	**NOFA**	**FPR**	**TPR**	**TDR**
29-Mar-99	71	1450	35	1415	97.59	2.41	49
30-Mar-99	104	3396	37	3359	98.91	1.09	36
31-Mar-99	143	7324	91	7233	98.76	1.24	64
01-Apr-99	71	8277	26	8251	99.69	0.31	37
02-Apr-99	327	4804	92	4712	98.08	1.92	28
05-Apr-99	111	3462	67	3395	98.06	1.94	60
06-Apr-99	285	7170	190	6980	97.35	2.65	67
07-Apr-99	90	7971	49	7922	99.39	0.61	54
08-Apr-99	159	13835	61	13774	99.56	0.44	38
09-Apr-99	121	20923	42	20881	99.80	0.20	35
Total	**1482**	**78612**	**690**	**77922**			**46**

RA	Real Attacks
NOA	Number of Alerts
NOTA	Number of True Alerts
NOFA	Number of False Alerts
FPR	False Positive Rate
TPR	True Positive Rate
TDR	True Detection Rate

APPENDIX J

Due to the nature of the massive data was collected in order to present this in easy accessible form, we devote this chapter to describing how the different categories of data can be presented in manageable tables. Appendix-I contains the complete set of all such tables along with their brief descriptions

Following first three tables show the result after data collection, alert normalization and alert preprocessing steps. First table shows the total connections in DARPA 1999 dataset detected by Bro IDS.

Table 18 Total connections in DARPA 1999 Datasets

Date	Total connection in DARPA 1999		
Time Slot	in	out	Total
29-Mar-99	107526	21452	128978
30-Mar-99		38693	38693
31-Mar-99	171764	48312	220076
01-Apr-99	254576	54137	308713
02-Apr-99	209919	35188	245107
05-Apr-99	231765	54858	286623
06-Apr-99	209116	94252	303368
07-Apr-99	220886	48422	269308
08-Apr-99	349098	137739	486837
09-Apr-99	322408	118478	440886
Total	**2077058**	**651531**	**2728589**

Please note that there is no connection available for March 30 (in) slot due to unavailability of dataset for the same date. The reason to include this table in very beginning is to provide a general idea relating the complexity of the dataset along with its volume.

In the next two tables, we provide the attack lists detected by Snort and Bro sensors separately. These detected alerts were stored in their respective databases after applying alert normalization and alert preprocessing algorithms. The selected sensors

were used with their default detection capabilities to scan and detect intrusions within the binary tcpdump file of the inside and outside dataset traffics. The reason to use the sensors in their default settings is to minimize the effect of their own capabilities on the proposed prototype throughout the research process.

In the following table, we present the total alerts detected by IDS snort. Snort reported total of 43267 alerts (37271 ICMP, no UDP and with 5996 TCP alerts).

Table 19 Attacks detected by Snort sensor

Date	Darapa attacks detected by Snort			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	916	0	188	1104
30-Mar-99	944	0	459	1403
31-Mar-99	3054	0	572	3626
01-Apr-99	2720	0	1109	3829
02-Apr-99	4364	0	605	4969
05-Apr-99	10720	0	289	11009
06-Apr-99	2453	0	448	2901
07-Apr-99	2272	0	697	2969
08-Apr-99	7059	0	909	7968
09-Apr-99	2769	0	720	3489
Total	**37271**	**0**	**5996**	**43267**

In the next table, we present the total alerts detected by Bro. Bro reports huge number of alerts i.e. 762336 with 3076 ICMP, 85 UDP and 759175 alerts of TCP type. It is obvious from these tables that both selected sensors create huge volume of alerts in their default settings.

Table 20 Attacks detected by Bro sensor

Date	Darapa attacks detected by Bro IDS			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	398	4	15843	16245
30-Mar-99	0	0	27282	27282
31-Mar-99	253	12	72089	72354
01-Apr-99	0	2	77647	77649
02-Apr-99	253	12	47302	47567
05-Apr-99	2172	14	41908	44094
06-Apr-99	0	2	61748	61750
07-Apr-99	0	8	69729	69737
08-Apr-99	0	18	127444	127462
09-Apr-99	0	13	218183	218196
Total	**3076**	**85**	**759175**	**762336**

J.1 Alert Filtering component for Snort sensor

Following table shows the alert filtering result based on Source IP address for Snort. Here, we may notice that for Snort sensor, only 1 false alert is eliminated through Alert Filtering component based on source IP address. This number has almost no effect on alert reduction process as a whole but might be helpful for other sensors that will be used in the future system expansion.

Table 21 Alert Filtering based on Source IP – Snort sensor

Date	Snort - Alert Filtering based on Source IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	915	0	188	1103
30-Mar-99	944	0	459	1403
31-Mar-99	3054	0	572	3626
01-Apr-99	2720	0	1109	3829
02-Apr-99	4364	0	605	4969
05-Apr-99	10720	0	289	11009
06-Apr-99	2453	0	448	2901
07-Apr-99	2272	0	697	2969
08-Apr-99	7059	0	909	7968
09-Apr-99	2769	0	720	3489
Total	**37270**	**0**	**5996**	**43266**

The following table shows the alert filtering based on destination IP address for Snort. 239 false alerts are filtered based on single entry destination IP address. Please note that these IP addresses are carefully checked for their further participation in any distributed attack in Snort database.

Table 22 Alert Filtering based on Destination IP – Snort sensor

Date	Snort - Alert Filtering based on Destination IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	915	0	188	1103
30-Mar-99	944	0	459	1403
31-Mar-99	3054	0	572	3626
01-Apr-99	2720	0	1109	3829
02-Apr-99	4323	0	605	4928
05-Apr-99	10642	0	289	10931
06-Apr-99	2453	0	448	2901
07-Apr-99	2236	0	697	2933
08-Apr-99	7007	0	909	7916
09-Apr-99	2737	0	720	3457
Total	**37031**	**0**	**5996**	**43027**

The following table describes the total reduction rate of Alert Filtering component for Snort sensor. Please note that the overall reduction rate for false alert is only .5555 % for Snort sensor with total of 240 false alarms filtered though Alert Filtering component.

Table 23 Alert reduction rate 1 - Snort sensor

Date	Snort - Alert Reduction rate of Alert Filtering Module				
Time Slot	icmp	udp	tcp	total	rate
29-Mar-99	1	0	0	1	0.0000
30-Mar-99	0	0	0	0	0.0000
31-Mar-99	0	0	0	0	0.0000
01-Apr-99	0	0	0	0	0.0000
02-Apr-99	41	0	0	41	0.8320
05-Apr-99	78	0	0	78	0.7136
06-Apr-99	0	0	0	0	0.0000
07-Apr-99	36	0	0	36	1.2274
08-Apr-99	52	0	0	52	0.6569
09-Apr-99	32	0	0	32	0.9257
Total	**240**	**0**	**0**	**240**	**0.5555**

J.2 Alert Filtering component for Bro sensor

In the following table for Bro sensor, 1846 false alerts are eliminated through Alert Filtering component based on source IP address.

Table 24 Alert Filtering based on Source IP – Bro sensor

Date	Bro - Alert Filtering based on Source IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	398	4	15840	16242
30-Mar-99	0	0	27277	27277
31-Mar-99	253	12	71834	72099
01-Apr-99	0	2	77645	77647
02-Apr-99	253	12	47048	47313
05-Apr-99	2172	14	40589	42775
06-Apr-99	0	2	61745	61747
07-Apr-99	0	8	69728	69736
08-Apr-99	0	17	127444	127461
09-Apr-99	0	13	218180	218193
Total	**3076**	**84**	**757330**	**760490**

The following table shows the alert filtering based on destination IP address for Bro sensor. 64 false alerts are filtered based on single entry destination IP address.

Table 25 Alert Filtering based on Destination IP – Bro sensor

Date	Bro - Alert Filtering based on Destination IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	398	4	15837	16239
30-Mar-99	0	0	27228	27228
31-Mar-99	253	12	71833	72098
01-Apr-99	0	2	77644	77646
02-Apr-99	253	12	47047	47312
05-Apr-99	2172	14	40588	42774
06-Apr-99	0	2	61742	61744
07-Apr-99	0	8	69727	69735
08-Apr-99	0	15	127444	127459
09-Apr-99	0	11	218180	218191
Total	**3076**	**80**	**757270**	**760426**

The following table shows the total reduction rate of Alert Filtering component for Bro sensor. Please note that the overall reduction rate for false alert is only .2505 % for the Bro sensor with total of 1910 false alarms filtered though Alert Filtering component.

Table 26 Alert reduction rate 1 - Bro sensor

Date	Bro - Alert Reduction rate of Alert Filtering Module				
Time Slot	icmp	udp	tcp	Total	rate
29-Mar-99	0	0	6	6	0.0369
30-Mar-99	0	0	54	54	0.1979
31-Mar-99	0	0	256	256	0.3538
01-Apr-99	0	0	3	3	0.0039
02-Apr-99	0	0	255	255	0.5361
05-Apr-99	0	0	1320	1320	2.9936
06-Apr-99	0	0	6	6	0.0097
07-Apr-99	0	0	2	2	0.0029
08-Apr-99	0	3	0	3	**0.0024**
09-Apr-99	0	2	3	5	0.0023
Total	**0**	**5**	**1905**	**1910**	**0.2505**

J.3 Multilevel Correlation for Snort sensor

This following table shows the final output of the alert correlation techniques for sensor Snort, based on selected signatures types. These signature types are mentioned in the previous chapter along with the algorithm. It is quite obvious from the table that the total alerts are now reduced to 17133 after correlating and eliminating almost 25894 alerts. This proposed technique has huge impact for the reducing of false positive alerts generated by our selected sensor Snort.

Table 27 Correlation based on Signature type - Snort sensor

Date	Snort - Correlation based on Signature type			
Time Slot	icmp	udp	tcp	total
29-Mar-99	349	0	188	537
30-Mar-99	226	0	459	685
31-Mar-99	409	0	572	981
01-Apr-99	497	0	1109	1606
02-Apr-99	561	0	605	1166
05-Apr-99	6946	0	289	7235
06-Apr-99	456	0	448	904
07-Apr-99	539	0	697	1236
08-Apr-99	638	0	909	1547
09-Apr-99	516	0	720	1236
Total	**11137**	**0**	**5996**	**17133**

The following table shows the further false positive reduction for sensor Snort based on timestamp and source IP address. Different alerts belong to same attack type in a fixed time intervals were correlated. This proposed correlation technique is very useful for correlating different alerts constitute same attack and has huge impact for minimizing false positive alerts. Finally, 8614 alerts are leftover for fusion process to deal with. Total of 34653 false positive alerts (almost 80% of total generated alerts) has been eliminated by proposed multilevel correlation techniques for Snort sensor.

Table 28 Correlation based on time stamp and IP - Snort sensor

Date	Snort - Correlation based on timestamp and IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	138	0	107	245
30-Mar-99	50	0	228	278
31-Mar-99	218	0	314	532
01-Apr-99	292	0	455	747
02-Apr-99	363	0	294	657
05-Apr-99	3577	0	182	3759
06-Apr-99	228	0	310	538
07-Apr-99	202	0	358	560
08-Apr-99	236	0	427	663
09-Apr-99	297	0	338	635
Total	**5601**	**0**	**3013**	**8614**

J.4 Multilevel Correlation for Bro sensor

The following table shows the further false positive reduction for sensor Bro based on timestamp and source IP address. Different alerts belong to same attack type in a fixed time intervals were correlated. This proposed correlation technique is very useful for correlating different alerts constitute same attack and has huge impact for minimizing false positive alerts. Finally, 78548 alerts are leftover for fusion process to deal with. Total of 683788 false positive alerts (almost 89% of total generated alerts) has been eliminated by proposed false alert generation techniques for Bro sensor.

Table 29 Correlation based on time stamp and IP - Bro sensor

Date	Bro - Correlation based on timestamp and IP			
Time Slot	icmp	udp	tcp	Total
29-Mar-99	398	4	1045	1447
30-Mar-99	0	0	3347	3347
31-Mar-99	253	12	7058	7323
01-Apr-99	0	2	8274	8276
02-Apr-99	253	12	4538	4803
05-Apr-99	2172	14	1275	3461
06-Apr-99	0	2	7165	7167
07-Apr-99	0	8	7962	7970
08-Apr-99	0	13	13820	13833
09-Apr-99	0	9	20912	20921
Total	**3076**	**76**	**75396**	**78548**

In the following series of tables, we present the final results of our fusion engine. In appendices section, appendix H presents the values of bpa for each attack type detected by both sensors for each time slot individually. In the following first section, these calculated bpa values (belief, disbelief and uncertainty) are used in DS combination rule to draw the final result of the network state (attack or no attack). In second section, we present detailed calculation through subjective logic for leading toward the final attack scenario.

J.5 DS Rule of Combination – Demspter Shafer Theory

ICMP Flooding

Table 30 DS rule of combination - ICMP Flooding

DS Combination Rule-ICMP Flooding - March 29, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.1636
Bro	Reliability		1	1	1	m12 (H)	0.9275
		Mass	0.6667	0.2000	0.1333	m12 (⌐H)	0.0435
m2 (H)	1	0.8182	0.5455	0.1636	0.1091	m12(Ω)	0.0290
m2(⌐ H)	1	0.0000	0.0000	0.0000	0.0000		
m2(Ω)	1	0.1818	0.1212	0.0364	0.0242	Decision	Attack

DS Combination Rule-ICMP Flooding - March 30, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.1538
Bro	Reliability		1	1	1	m12 (H)	0.0000
		Mass	0.4615	0.3846	0.1538	m12 (⌐H)	0.5152
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.1212
m2(⌐ H)	1	0.3333	0.1538	0.1282	0.0513		
m2(Ω)	1	0.6667	0.3077	0.2564	0.1026	Decision	No-Attack

DS Combination Rule-ICMP Flooding - March 31, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.2593
Bro	Reliability		1	1	1	m12 (H)	0.8600
		Mass	0.5333	0.3333	0.1333	m12 (⌐H)	0.1000
m2 (H)	1	0.7778	0.4148	0.2593	0.1037	m12(Ω)	0.0400
m2(⌐ H)	1	0.0000	0.0000	0.0000	0.0000		
m2(Ω)	1	0.2222	0.1185	0.0741	0.0296	Decision	Attack

DS Combination Rule-ICMP Flooding - April 01, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.2407
Bro	Reliability		1	1	1	m12 (H)	0.6341
		Mass	0.7222	0.1667	0.1111	m12 (⌐H)	0.2683
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.0976
m2(⌐ H)	1	0.3333	0.2407	0.0556	0.0370		
m2(Ω)	1	0.6667	0.4815	0.1111	0.0741	Decision	Attack

DS Combination Rule-ICMP Flooding - April 02, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.1228
Bro	Reliability		1	1	1	m12 (H)	0.9333
		Mass	0.7368	0.1579	0.1053	m12 (⌐H)	0.0400
m2 (H)	1	0.7778	0.5731	0.1228	0.0819	m12(Ω)	0.0267
m2(⌐ H)	1	0.0000	0.0000	0.0000	0.0000		
m2(Ω)	1	0.2222	0.1637	0.0351	0.0234	Decision	Attack

DS Combination Rule-ICMP Flooding - April 05, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.0228
Bro	Reliability		1	1	1	m12 (H)	0.9979
		Mass	0.9524	0.0238	0.0238	m12 (⌐H)	0.0010
m2 (H)	1	0.9574	0.9119	0.0228	0.0228	m12(Ω)	0.0010
m2(⌐ H)	1	0.0000	0.0000	0.0000	0.0000		
m2(Ω)	1	0.0426	0.0405	0.0010	0.0010	Decision	Attack

DS Combination Rule-ICMP Flooding - April 06, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.1875
Bro	Reliability		1	1	1	m12 (H)	0.4615
		Mass	0.5625	0.3125	0.1250	m12 (⌐H)	0.4359
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.1026
m2(⌐ H)	1	0.3333	0.1875	0.1042	0.0417		
m2(Ω)	1	0.6667	0.3750	0.2083	0.0833	Decision	Attack

DS Combination Rule-ICMP Flooding - April 07, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.2222
Bro	Reliability		1	1	1	m12 (H)	0.5714
		Mass	0.6667	0.2000	0.1333	m12 (⌐H)	0.3143
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.1143
m2(⌐ H)	1	0.3333	0.2222	0.0667	0.0444		
m2(Ω)	1	0.6667	0.4444	0.1333	0.0889	Decision	Attack

DS Combination Rule-ICMP Flooding - April 08, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.2353
Bro	Reliability		1	1	1	m12 (H)	0.6154
		Mass	0.7059	0.1765	0.1176	m12 (⌐H)	0.2821
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.1026
m2(⌐ H)	1	0.3333	0.2353	0.0588	0.0392		
m2(Ω)	1	0.6667	0.4706	0.1176	0.0784	Decision	Attack

DS Combination Rule-ICMP Flooding - April 09, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.2456
Bro	Reliability		1	1	1	m12 (H)	0.6512
		Mass	0.7368	0.1579	0.1053	m12 (⌐H)	0.2558
m2 (H)	1	0.0000	0.0000	0.0000	0.0000	m12(Ω)	0.0930
m2(⌐ H)	1	0.3333	0.2456	0.0526	0.0351		
m2(Ω)	1	0.6667	0.4912	0.1053	0.0702	Decision	Attack

Table 31DS Combination rule result - ICMP Flooding

DS Combination Rule-ICMP Flooding				
Date	**m12(H)**	**m12(⌐ H)**	**m12(Ω)**	**Final Decision**
29-Mar-99	0.9275	0.0435	0.0290	Attack
30-Mar-99	0.3636	0.5152	0.1212	No-Attack
31-Mar-99	0.8600	0.1000	0.0400	Attack
01-Apr-99	0.6341	0.2683	0.0976	Attack
02-Apr-99	0.9333	0.0400	0.0267	Attack
05-Apr-99	0.9979	0.0010	0.0010	Attack
06-Apr-99	0.4615	0.4359	0.1026	Attack
07-Apr-99	0.5714	0.3143	0.1143	Attack
08-Apr-99	0.6154	0.2821	0.1026	Attack
09-Apr-99	0.6512	0.2558	0.0930	Attack

SYN Flooding

Table 32 DS rule of combination - SYN Flooding

DS Combination Rule-SYN Flooding - March 29, 1999					
Sensors	**Snort**		**m1 (H)**	**m1(⌐ H)**	**m1(Ω)**
Bro	**Reliability**		1	1	1
		Mass	0.1905	0.7619	0.0476
m2 (H)	1	0.4167	0.0794	0.3175	0.0198
m2(⌐ H)	1	0.5500	0.1048	0.4190	0.0262
m2(Ω)	1	0.0333	0.0063	0.0254	0.0016

Final Values	
K	0.4222
m12 (H)	0.1827
m12 (⌐H)	0.8146
m12(Ω)	0.0027
Decision	No-Attack

DS Combination Rule-SYN Flooding - March 30, 1999					
Sensors	**Snort**		**m1 (H)**	**m1(⌐ H)**	**m1(Ω)**
Bro	**Reliability**		1	1	1
		Mass	0.2667	0.6889	0.0444
m2 (H)	1	0.6759	0.1802	0.4656	0.0300
m2(⌐ H)	1	0.3056	0.0815	0.2105	0.0136
m2(Ω)	1	0.0185	0.0049	0.0128	0.0008

Final Values	
K	0.5471
m12 (H)	0.4752
m12 (⌐H)	0.5229
m12(Ω)	0.0018
Decision	No-Attack

DS Combination Rule-SYN Flooding - March 31, 1999					
Sensors	**Snort**		**m1 (H)**	**m1(⌐ H)**	**m1(Ω)**
Bro	**Reliability**		1	1	1
		Mass	0.3673	0.5918	0.0408
m2 (H)	1	0.8177	0.3004	0.4839	0.0334
m2(⌐ H)	1	0.1713	0.0629	0.1014	0.0070
m2(Ω)	1	0.0110	0.0041	0.0065	0.0005

Final Values	
K	0.5468
m12 (H)	0.7455
m12 (⌐H)	0.2535
m12(Ω)	0.0010
Decision	Attack

DS Combination Rule-SYN Flooding - April 01, 1999					
Sensors	**Snort**		**m1 (H)**	**m1(⌐ H)**	**m1(Ω)**
Bro	**Reliability**		1	1	1
		Mass	0.4340	0.5283	0.0377
m2 (H)	1	0.8398	0.3644	0.4437	0.0317
m2(⌐ H)	1	0.1505	0.0653	0.0795	0.0057
m2(Ω)	1	0.0097	0.0042	0.0051	0.0004

Final Values	
K	0.5090
m12 (H)	0.8153
m12 (⌐H)	0.1839
m12(Ω)	0.0007
Decision	Attack

DS Combination Rule-SYN Flooding - April 02, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.5251
Bro	Reliability		1	1	1	m12 (H)	0.6685
		Mass	0.3750	0.5833	0.0417	m12 (⌐H)	0.3302
m2 (H)	1	0.7481	0.2805	0.4364	0.0312	m12(Ω)	0.0013
m2(⌐ H)	1	0.2366	0.0887	0.1380	0.0099		
m2(Ω)	1	0.0153	0.0057	0.0089	0.0006	Decision	Attack

DS Combination Rule-SYN Flooding - April 05, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.4663
Bro	Reliability		1	1	1	m12 (H)	0.3024
		Mass	0.2667	0.6889	0.0444	m12 (⌐H)	0.6951
m2 (H)	1	0.4923	0.1313	0.3391	0.0219	m12(Ω)	0.0026
m2(⌐ H)	1	0.4769	0.1272	0.3285	0.0212		
m2(Ω)	1	0.0308	0.0082	0.0212	0.0014	Decision	No-Attack

DS Combination Rule-SYN Flooding - April 06, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.5541
Bro	Reliability		1	1	1	m12 (H)	0.7324
		Mass	0.3542	0.6042	0.0417	m12 (⌐H)	0.2666
m2 (H)	1	0.8152	0.2887	0.4925	0.0340	m12(Ω)	0.0010
m2(⌐ H)	1	0.1739	0.0616	0.1051	0.0072		
m2(Ω)	1	0.0109	0.0038	0.0066	0.0005	Decision	Attack

DS Combination Rule-SYN Flooding - April 07, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.5228
Bro	Reliability		1	1	1	m12 (H)	0.7902
		Mass	0.4082	0.5510	0.0408	m12 (⌐H)	0.2089
m2 (H)	1	0.8308	0.3391	0.4578	0.0339	m12(Ω)	0.0009
m2(⌐ H)	1	0.1592	0.0650	0.0877	0.0065		
m2(Ω)	1	0.0100	0.0041	0.0055	0.0004	Decision	Attack

DS Combination Rule-SYN Flooding - April 08, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.5391
Bro	Reliability		1	1	1	m12 (H)	0.8653
		Mass	0.4038	0.5577	0.0385	m12 (⌐H)	0.1341
m2 (H)	1	0.8959	0.3618	0.4996	0.0345	m12(Ω)	0.0005
m2(⌐ H)	1	0.0978	0.0395	0.0545	0.0038		
m2(Ω)	1	0.0063	0.0025	0.0035	0.0002	Decision	Attack

DS Combination Rule-SYN Flooding - April 09, 1999						Final Values	
Sensors	Snort		m1 (H)	m1(⌐ H)	m1(Ω)	K	0.5124
Bro	Reliability		1	1	1	m12 (H)	0.9183
		Mass	0.4400	0.5200	0.0400	m12 (⌐H)	0.0813
m2 (H)	1	0.9289	0.4087	0.4830	0.0372	m12(Ω)	0.0004
m2(⌐ H)	1	0.0668	0.0294	0.0347	0.0027		
m2(Ω)	1	0.0043	0.0019	0.0022	0.0002	Decision	Attack

Table 33 DS Combination rule result - SYN Flooding

DS Combination Rule-SYN Flooding				
Date	**m12(H)**	**m12(⌐ H)**	**m12(Ω)**	**Final Decision**
29-Mar-99	0.1827	0.8146	0.0027	No-Attack
30-Mar-99	0.4752	0.5229	0.0018	No-Attack
31-Mar-99	0.7455	0.2535	0.0010	Attack
01-Apr-99	0.8153	0.1839	0.0007	Attack
02-Apr-99	0.6685	0.3302	0.0013	Attack
05-Apr-99	0.3024	0.6951	0.0026	No-Attack
06-Apr-99	0.7324	0.2666	0.0010	Attack
07-Apr-99	0.7902	0.2089	0.0009	Attack
08-Apr-99	0.8653	0.1341	0.0005	Attack
09-Apr-99	0.9183	0.0813	0.0004	Attack

J.6 Attack scenario through Subjective Logic – Jøsang

ICMP Flooding

Table 34 Jøsang Subjective Logic - ICMP Flooding

March 29, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.6667	0.2000	0.1333	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8182	0.0000	0.1818	0.5000	
Discount					
	0.3733	0.1120	0.5147	0.5000	
	0.3764	0.0000	0.6236	0.5000	
Consensus					
	0.5218	0.0855	0.3927	0.5000	Decision
K	0.8173				**Attack**

March 30, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.4615	0.3846	0.1538	0.5000	
ω_{B2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{A1}^{E(S1)}$	0.2585	0.2154	0.5262	0.5000	
$\omega_{B2}^{E(S2)}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.2360	0.2836	0.4804	0.5000	Decision
K	0.9273				**Undecided**

March 31, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.5333	0.3333	0.1333	0.5000	
ω_{B2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.7778	0.0000	0.2222	0.5000	
Discount					
$\omega_{A1}^{E(S1)}$	0.2987	0.1867	0.5147	0.5000	
$\omega_{B2}^{E(S2)}$	0.3578	0.0000	0.6422	0.5000	
Consensus					
	0.4549	0.1451	0.4000	0.5000	Decision
K	0.8264				**Attack**

April 01, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.7222	0.1667	0.1111	0.5000	
ω_{B2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{A1}^{E(S1)}$	0.4044	0.0933	0.5022	0.5000	
$\omega_{B2}^{E(S2)}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.3707	0.1689	0.4604	0.5000	Decision
K	0.9237				**Undecided**

April 02, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
$\omega_{A1}^{F_{S1}}$	0.5600	0.2200	0.2200	0.5000	
$\omega_{A1}^{F_{S2}}$	0.7368	0.1579	0.1053	0.5000	
$\omega_{A2}^{F_{S2}}$	0.4600	0.2700	0.2700	0.5000	
$\omega_{A1}^{F_{S3}}$	0.7778	0.0000	0.2222	0.5000	
Discount					
$\omega_{S1}^{F_{S1}}$	0.4126	0.0884	0.4989	0.5000	
$\omega_{S2}^{F_{S2}}$	0.3578	0.0000	0.6422	0.5000	
Consensus					
	0.5404	0.0692	0.3904	0.5000	Decision
K	0.8207				**Attack**

April 05, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
$\omega_{S1}^{F_{S1}}$	0.5600	0.2200	0.2200	0.5000	
$\omega_{A1}^{F_{S2}}$	0.9524	0.0238	0.0238	0.5000	
$\omega_{A2}^{F_{S2}}$	0.4600	0.2700	0.2700	0.5000	
$\omega_{A1}^{F_{S3}}$	0.9574	0.0000	0.0426	0.5000	
Discount					
$\omega_{S1}^{F_{S1}}$	0.5333	0.0133	0.4533	0.5000	
$\omega_{S2}^{F_{S2}}$	0.4404	0.0000	0.5596	0.5000	
Consensus					
	0.6561	0.0098	0.3341	0.5000	Decision
K	0.7592				**Attack**

April 06, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
$\omega_{S1}^{F_{S1}}$	0.5600	0.2200	0.2200	0.5000	
$\omega_{A1}^{F_{S2}}$	0.5625	0.3125	0.1250	0.5000	
$\omega_{A2}^{F_{S2}}$	0.4600	0.2700	0.2700	0.5000	
$\omega_{A1}^{F_{S3}}$	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{S1}^{F_{S1}}$	0.3150	0.1750	0.5100	0.5000	
$\omega_{S2}^{F_{S2}}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.2884	0.2448	0.4669	0.5000	Decision
K	0.9249				**Undecided**

April 07, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E1}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.6667	0.2000	0.1333	0.5000	
ω_{A2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{A1}^{E\backslash E1}$	0.3733	0.1120	0.5147	0.5000	
$\omega_{A2}^{E\backslash E2}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.3415	0.1877	0.4708	0.5000	Decision
K	0.9256				**Undecided**

April 08, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E1}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.7059	0.1765	0.1176	0.5000	
ω_{A2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{A1}^{E\backslash E1}$	0.3953	0.0988	0.5059	0.5000	
$\omega_{A2}^{E\backslash E2}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.3621	0.1745	0.4634	0.5000	Decision
K	0.9242				**Undecided**

April 09, 1999-ICMP					
Opinion	**b**	**d**	**u**	**a**	
ω_{A1}^{E1}	0.5600	0.2200	0.2200	0.5000	
ω_{A1}^{E}	0.7368	0.1579	0.1053	0.5000	
ω_{A2}^{E}	0.4600	0.2700	0.2700	0.5000	
ω_{A1}^{E}	0.0000	0.3333	0.6667	0.5000	
Discount					
$\omega_{A1}^{E\backslash E1}$	0.4126	0.0884	0.4989	0.5000	
$\omega_{A2}^{E\backslash E2}$	0.0000	0.1533	0.8467	0.5000	
Consensus					
	0.3784	0.1640	0.4576	0.5000	Decision
K	0.9232				**Undecided**

Table 35 Jøsang Subjective Logic result - ICMP Flooding

Subjective Logic-ICMP Flooding					
Date	**b**	**d**	**u**	**a**	**Final Decision**
29-Mar-99	0.5218	0.0855	0.3927	0.5000	Attack
30-Mar-99	0.2360	0.2836	0.4804	0.5000	Undecided
31-Mar-99	0.4549	0.1451	0.4000	0.5000	Attack
01-Apr-99	0.3707	0.1689	0.4604	0.5000	Undecided
02-Apr-99	0.5404	0.0692	0.3904	0.5000	Attack
05-Apr-99	0.6561	0.0098	0.3341	0.5000	Attack
06-Apr-99	0.2884	0.2448	0.4669	0.5000	Undecided
07-Apr-99	0.3415	0.1877	0.4708	0.5000	Undecided
08-Apr-99	0.3621	0.1745	0.4634	0.5000	Undecided
09-Apr-99	0.3784	0.1640	0.4576	0.5000	Undecided

SYN Flooding

Table 36 Jøsang Subjective Logic - SYN Flooding

March 29, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
ω_{K_1}	0.5600	0.2200	0.2200	0.5000	
ω_{K_2}	0.1905	0.7619	0.0476	0.5000	
ω_{K_3}	0.4600	0.2700	0.2700	0.5000	
ω_{K_4}	0.4167	0.5500	0.0333	0.5000	
Discount					
$\omega_{K_1 K_2}$	0.1067	0.4267	0.4667	0.5000	
$\omega_{K_3 K_4}$	0.1917	0.2530	0.5553	0.5000	
Consensus					
	0.1949	0.4654	0.3397	0.5000	Decision
K	0.7628				**No Attack**

March 30, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
ω_{K_1}	0.5600	0.2200	0.2200	0.5000	
ω_{K_2}	0.2667	0.6889	0.0444	0.5000	
ω_{K_3}	0.4600	0.2700	0.2700	0.5000	
ω_{K_4}	0.6759	0.3056	0.0185	0.5000	
Discount					
$\omega_{K_1 K_2}$	0.1493	0.3858	0.4649	0.5000	
$\omega_{K_3 K_4}$	0.3109	0.1406	0.5485	0.5000	
Consensus					
	0.2986	0.3652	0.3362	0.5000	Decision
K	0.7584				**No Attack**

March 31, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.3673	0.5918	0.0408	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8177	0.1713	0.0110	0.5000	
Discount					
	0.2057	0.3314	0.4629	0.5000	
	0.3761	0.0788	0.5451	0.5000	
Consensus					
	0.3788	0.2873	0.3339	0.5000	Decision
K	0.7556				**Attack**

April 01, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.4340	0.5283	0.0377	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8398	0.1505	0.0097	0.5000	
Discount					
	0.2430	0.2958	0.4611	0.5000	
	0.3863	0.0692	0.5445	0.5000	
Consensus					
	0.4115	0.2558	0.3328	0.5000	Decision
K	0.7545				**Attack**

April 02, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.3750	0.5833	0.0417	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.7481	0.2366	0.0153	0.5000	
Discount					
	0.2100	0.3267	0.4633	0.5000	
	0.3441	0.1089	0.5470	0.5000	
Consensus					
	0.3624	0.3027	0.3349	0.5000	Decision
K	0.7569				**Attack**

April 05, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.2667	0.6889	0.0444	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.4923	0.4769	0.0308	0.5000	
Discount					
	0.1493	0.3858	0.4649	0.5000	
	0.2265	0.2194	0.5542	0.5000	
Consensus					
	0.2469	0.4147	0.3383	0.5000	Decision
K	0.7614				**No Attack**

April 06, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.3542	0.6042	0.0417	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8152	0.1739	0.0109	0.5000	
Discount					
	0.1983	0.3383	0.4633	0.5000	
	0.3750	0.0800	0.5450	0.5000	
Consensus					
	0.3729	0.2930	0.3341	0.5000	Decision
K	0.7558				**Attack**

April 07, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.4082	0.5510	0.0408	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8308	0.1592	0.0100	0.5000	
Discount					
	0.2286	0.3086	0.4629	0.5000	
	0.3822	0.0732	0.5446	0.5000	
Consensus					
	0.3990	0.2673	0.3337	0.5000	Decision
K	0.7554				**Attack**

April 08, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.4038	0.5577	0.0385	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.8959	0.0978	0.0063	0.5000	
Discount					
	0.2262	0.3123	0.4615	0.5000	
	0.4121	0.0450	0.5429	0.5000	
Consensus					
	0.4152	0.2524	0.3324	0.5000	Decision
K	0.7539				**Attack**

April 09, 1999-TCP					
Opinion	**b**	**d**	**u**	**a**	
	0.5600	0.2200	0.2200	0.5000	
	0.4400	0.5200	0.0400	0.5000	
	0.4600	0.2700	0.2700	0.5000	
	0.9289	0.0668	0.0043	0.5000	
Discount					
	0.2464	0.2912	0.4624	0.5000	
	0.4273	0.0307	0.5420	0.5000	
Consensus					
	0.4393	0.2282	0.3325	0.5000	Decision
K	0.7538				**Attack**

Table 37 Jøsang Subjective Logic result - SYN Flooding

Subjective Logic-SYN Flooding					
Date	**b**	**d**	**u**	**a**	**Final Decision**
29-Mar-99	0.1949	0.4654	0.3397	0.5000	No Attack
30-Mar-99	0.2986	0.3652	0.3362	0.5000	No Attack
31-Mar-99	0.3788	0.2873	0.3339	0.5000	Attack
01-Apr-99	0.4115	0.2558	0.3328	0.5000	Attack
02-Apr-99	0.3624	0.3027	0.3349	0.5000	Attack
05-Apr-99	0.2469	0.4147	0.3383	0.5000	No Attack
06-Apr-99	0.3729	0.2930	0.3341	0.5000	Attack
07-Apr-99	0.3990	0.2673	0.3337	0.5000	Attack
08-Apr-99	0.4152	0.2524	0.3324	0.5000	Attack
09-Apr-99	0.4393	0.2282	0.3325	0.5000	Attack

J.7 Multilevel correlation for Snort

In the following table, alerts detected by Snort sensor from DARPA dataset 1999 are given beside the total number of remaining alerts after multilevel correlation process. Total alerts are reduced from 43267 to 8614 with almost 80 % reduction

Table 38 Multilevel Correlation result - Snort sensor

Date	Darapa attacks detected by Snort				Alerts after Correlation			
Time Slot	icmp	udp	tcp	Total	icmp	udp	tcp	Total
29-Mar-99	916	0	188	1104	138	0	107	245
30-Mar-99	944	0	459	1403	50	0	228	278
31-Mar-99	3054	0	572	3626	218	0	314	532
01-Apr-99	2720	0	1109	3829	292	0	455	747
02-Apr-99	4364	0	605	4969	363	0	294	657
05-Apr-99	10720	0	289	11009	3577	0	182	3759
06-Apr-99	2453	0	448	2901	228	0	310	538
07-Apr-99	2272	0	697	2969	202	0	358	560
08-Apr-99	7059	0	909	7968	236	0	427	663
09-Apr-99	2769	0	720	3489	297	0	338	635
Total	**37271**	**0**	**5996**	**43267**	**5601**	**0**	**3013**	**8614**

Following table represent the reduction rate per each day for Snort sensor. The huge number of false positive alerts reduction shows the effectiveness of our various proposed correlation rules and methodology. Please note that, Snort reports no alert for UDP traffic data unlike Bro which reports little amount of alerts for UDP traffic.

Table 39 Reduction rate - Snort sensor

Date	Total reduction and reduction rate				
Time Slot	icmp	udp	tcp	Total	rate %
29-Mar-99	778	0	81	859	77.81
30-Mar-99	894	0	231	1125	80.19
31-Mar-99	2836	0	258	3094	85.33
01-Apr-99	2428	0	654	3082	80.49
02-Apr-99	4001	0	311	4312	86.78
05-Apr-99	7143	0	107	7250	65.86
06-Apr-99	2225	0	138	2363	81.45
07-Apr-99	2070	0	339	2409	81.14
08-Apr-99	6823	0	482	7305	91.68
09-Apr-99	2472	0	382	2854	81.80
Total	**31670**	**0**	**2983**	**34653**	**80.09**

J.8 Multilevel correlation for Bro

In the following table, alerts detected by Bro sensor are given beside the total number of remaining alerts after multilevel correlation process. Total alerts are reduced from 762336 to 78612 with almost 89% reduction.

Table 40 Multilevel Correlation result - Bro sensor

Date	DARPA attacks detected by Bro				Alerts after Correlation			
Time Slot	icmp	udp	tcp	Total	icmp	udp	tcp	Total
29-Mar-99	398	4	15843	16245	398	4	1048	1450
30-Mar-99	0	0	27282	27282	0	0	3396	3396
31-Mar-99	253	12	72089	72354	253	12	7059	7324
01-Apr-99	0	2	77647	77649	0	2	8275	8277
02-Apr-99	253	12	47302	47567	253	12	4539	4804
05-Apr-99	2172	14	41908	44094	2172	14	1276	3462
06-Apr-99	0	2	61748	61750	0	2	7168	7170
07-Apr-99	0	8	69729	69737	0	8	7963	7971
08-Apr-99	0	18	127444	127462	0	15	13820	13835
09-Apr-99	0	13	218183	218196	0	11	20912	20923
Total	**3076**	**85**	**759175**	**762336**	**3076**	**80**	**75456**	**78612**

Following table represent the reduction rate per each day for Bro sensor.

Table 41 Reduction rate - Bro sensor

Date	Total reduction and reduction rate				
Time Slot	icmp	udp	tcp	Total	rate %
Time Slot	0	0	14795	14795	91.07
29-Mar-99	0	0	23886	23886	87.55
30-Mar-99	0	0	65030	65030	89.88
31-Mar-99	0	0	69372	69372	89.34
01-Apr-99	0	0	42763	42763	89.90
02-Apr-99	0	0	40632	40632	92.15
05-Apr-99	0	0	54580	54580	88.39
06-Apr-99	0	0	61766	61766	88.57
07-Apr-99	0	3	113624	113627	89.15
08-Apr-99	0	2	197271	197273	90.41
Total	**0**	**5**	**683719**	**683724**	89.69

APPENDIX K

List of Abbreviation

ASCII: American Standard Code for Information Interchange

bpa: Basic Probability Assignment

CISSP: Certified Information Systems Security Professional

CVE: Common Vulnerabilities and Exposures

DARPA: Defense Advanced Research Projects Agency

DDoS attack: Distributed Denial of Service attack

DNS: Domain Name System

DST: Dempster Shafer Theory

FDR: Final Decision Rule

FoD: Frame of Discernment

FTP: File Transfer Protocol

FPR: False Positive Rate

HIDS: Host-based Intrusion Detection System

HTTP: The Hypertext Transfer Protocol

ICMP: Internet Control Message Protocol

IDS: Intrusion Detection System

IDMEF: Intrusion Detection Message Exchange Format

IDWG: Intrusion Detection Working Group

IPS: Intrusion Prevention System

LAN: Local Area Network

MANET: Mobile Ad hoc Network

MySQL: My Structured Query Language

NSF: National Science Foundation

NIDS: Network-based Intrusion Detection System

NOA: Number of Alerts

NOTA: Number of True Alerts

NOFA: Number of False Alert

R2L: Remote to Local

RA: Real Attacks

RDBMS: Relational Database Management System

SNMP: Simple Network Management Protocol

SSL: Secure Sockets Layer

TCP: Transmission Control Protocol

TDR: True Detection Rate

U2R: User to Root

URI: Uniform Resource Identifier

UDP: User Datagram Protocol

WAN: Wide Area Network

XML: Extensible Markup Language

REFERENCES

Almgren. M., Debar. H., and Dacier. M. 2000 A Lightweight Tool for Detecting Web Server Attacks. Proceedings of the Network and Distributed System Security Symposium (NDSS 2000). 157 - 170

Almgren. M., and Lindqvist., U. 2001 Application-Integrated Data Collection for Security Monitoring. Proceedings of the 4th Workshop on recent Advances in Intrusion Detection (RAID), LNCS Springer Verlag. 22 – 36

Alsubhi. K. 2008 A Fuzzy-Logic Based Alert Prioritization Engine for IDS: Architecture and Configuration. Master's thesis, University of Waterloo. Ontario, Canada

Anastasios. T., Seraphin. B., and Allan. J. 1994 Alarm Correlation and Fault Identification in Communication Networks. Proceedings of the IEEE Transactions on Communications. 523 – 533

Axelsson. A. 2000 The Base-Rate Fallacy and the Difficulty of Intrusion Detection. Proceedings of the ACM Transactions on Information and System Security (TISSEC). 186 – 205

Bass. T. 1999 Sensor Data Fusion for Next Generation Distributed Intrusion Detection Systems. Proceedings of 1999 IRIS National Symposium on Sensor and Data Fusion, 24-27

Bass. T. 2000 Intrusion Detection Systems and Multisensory Data Fusion. Communication of the ACM. 99 – 105

Bro Intrusion Detection System Documentation. <www. *http://www.Bro-ids.org/download/index.html*>

Burroughs. J., Wilson. F., and Cybenko, V. 2002 Analysis of Distributed Intrusion Detection Systems using Bayesian Methods. Performance, Computing, and Communications Conference. 329 – 334

Chatzigiannakis. V., Androulidakis. G., Pelechrinis. K., Papavassiliou. S., and Maglaris. V. 2007. Data Fusion Algorithms for Network Anomaly Detection: Classification and Evaluation. Proceedings of IEEE Third International Conference on Networking and Services ICNS, Page 50

Cuppens. F., and Miège. A. 2002 Alert Correlation in a Cooperative Intrusion Detection Framework. Proceedings of the IEEE Symposium on Security and Privacy. 202 - 215

Cuppens, F. 2002 Managing Alerts in a Multi Intrusion Detection Environment. Proceedings of IEEE 17th annual Computer Security Applications Conference. 22 – 31

DARPA Intrusion Detection Data Sets – MIT Lincoln Laboratory < *http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html*>

Debar. H., and Wespi A. 2001 Aggregation and Correlation in Intrusion-Detection Alerts. Proceedings of the 4th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), LNCS 2516. 85 – 103

Feng. C., Peng. J., Qiao. H., and Rozenblit. W. 2007 Alert Fusion for A Computer Host Based Intrusion Detection System. Proceedings of 14th annual IEEE International Conference and Workshop on the Engineering of Computer-Based System. 433 - 440

Fan. G., JiHua. Y., and Min. Yu. 2009 Design and Implementation of A Distributed Alert Aggregation Model. Proceedings of IEEE 4th International Conference on Computer Science & Education. 975 - 980

Frincke. D. 2000 Balancing Cooperation and Risk in Intrusion Detection. ACM Transactions on information and System Security (TISSEC). 1 – 29

Giacinto. G., Roli. F., and Didaci. L. 2003 Fusion of multip;e classification for intrusion detection in computer networks. Pattern Recognition Letters 24. 1795 - 1803

Gu. G., Alvaro. A., and Lee. W. 2008 Principal Reasoning and Practical Application of Alert Fusion in Intrusion Detection Systems. P*roceedings of the 2008 ACM symposium on Information, computer and communications security.* 136 - 147

H. Wu., M. Siegel., R. Stiefelhagen., and J. Yang. 2002 Sensor fusion using Dempster – Shafer Theory. Proceedings of 19th IEEE Instrumentation and Measurement Technology Conference. 7 – 12

Jøsang. A. 1997 Artificial Reasoning with Subjective Logic. Proceedings of the 2nd Austrian Workshop on Commonsense Reasoning.

Jøsang. A. 2001 A Logic for Uncertain Probability. International Journal of Uncertainty, Fuzziness and knowledge-Based Systems. Vol. 9, No. 3

Jøsang. A. 2002 The Consensus Operator for Combining Beliefs. Artificial Intelligence Journal 142(1-2). 157 - 170

Jøsang. A. 2009 Fission of Opinion in Subjective Logic. Proceedings of 12th International Conference on Information Fusion Seattle, WA, USA.

Jøsang. A. 2011 Subjective Logic Draft. <*http://folk.uio.no/josang/*>

Julisch. K. 2003 Using Root Cause Analysis to Handle Intrusion Detection Alarms. PhD's thesis, University of Dortmund, Department of Computer Science, Germany.

Julisch. K., and Alarm. M. 2001 Clusters to Improve Alarm Handling Efficiency. Proceedings of the 17th Annual Computer Security Applications Conference (ACSAC). 12 – 21

Julisch. K., and Dacier. M. 2002 Mining intrusion detection alarms for actionable knowledge. Proceedings of the eight ACM SIGKDD International Conference on Knowledge Discovery and Data mining. 366 – 375

K. Julisch. 2003 Clustering Intrusion Detection Alarms to Support Root Cause Analysis. Proceedings of ACM Transaction on Information and System Security. 443 – 471

Kabiri. P., and Ghorbani. A. 2007 A Rule-Based Temporal Alert Correlation System. International journal of network Security. 66 – 72

Kuhl. M.E., Kistner. J., Costantini. K., and Sudit. M. 2007. Cyber attack modeling and simulation for network security analysis. Proceedings of IEEE Winter Simulation Conference. 1180 - 1188

Li. Z., Chen. Y., and Beach. A. 2006. Towards scalable and robust distributed intrusion alert fusion with good load balancing. Proceeding of ACM SIGCOMM workshop on Large-scale attack defense. 115 - 122

Maggi. F., and Zanero. S. 2007 On the Use of Different Statistical Tests for Alert Correlation – Short Paper. Lecture Notes in Computer Science LNCS, Springer Verlag. 4637/2007. 167 – 177

Mansmann. F., Fischer. F., Keim. D., and North. S. 2009. Visual support for analyzing network traffic and intrusion detection events using Treemap and graph representations. Proceedings of the ACM Symposium on Computer human Interaction for the Management of Information Technology. Article 03.

Mathew. S., Shah. C., and S. Upadhyaya. 2005 An Alert Fusion Framework for Situation Awareness of Coordinated Multistage Attacks. Proceedings of Third IEEE International Workshop on Information Assurance. 95 – 104

Mathew. S., Britt. D., Giomundo. R., and Upadhyaya. S. 2006. Real-Time Multistage Attack Awareness Through Enhanced Intrusion Alert Clustering. Proceeding of the 3rd international ACM workshop on Visualization for computer security. 1 – 6

McHugh. J. 2000 Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory. Proceedings of the ACM Transactions on Information and System Security (TISSEC). 262 – 294

P. Ning., Y. Cui., and D. Reeves. 2002 Analyzing Intensive Intrusion Alerts Via Correlation. Proceedings of the 5th International Symposium on Recent Advances in Intrusion Detection (RAID 2002), LNCS 2516. 74 – 94

P. Ning., Y. Cui., and D. Reeves. 2002 Constructing Attacl Scenarios Through Correlation of Intrusion alerts. Proceedings of 9th ACM Conference on Computer and Communications Security. 245 – 254

P. Ning., Y. Cui., D. Reeves., and D. Xu. 2004 Techniques and Tools for Analyzing Intrusion Alerts. Proceedings of ACM Transactions on Information and System Security (TISSEC). 274 – 318

Ruta. D., and Gabrys. B. 2000. An Overview of Classifier Fusion Methods. Computing and Information Systems, 7, 1-10.

Sadoddin. R., and Ghorbani. A. 2006 Alert Correlation Survey: Framework and Techniques. Proceedings of the 2006 International Conference on Privacy, Security and Trust: Bridge the Gap Between PST Technologies and Business Services, published by ACM. 380(37)

S. Ambareen., and V. Rayford. 2007 Alert Correlation with Abstract Incident Modeling in a Multi Sensor Environment. IJCSNJ International Journal of computer Science and Network Security. 7(8)

Siaterlis. C., and Maglaris. V. 2005. One Step Ahead to Multisensor Data Fusion for DDoS Detection. Journal of Computer Security, 13. 779–806

Siaterlis. C., and Maglaris. V. 2004 Towards Multisensor Data Fusion for DoS Detection. Proceedings of ACM Symposium on Applied Computing. 439 - 446

Siraj. A., and Vaughn. B. 2005 Multi-level Alert Clustering for Intrusion Detection Sensor Data. Proceedings of IEEE International Conference on Intelligence and Security Informatics. 748 – 753

Siraj. A., and Vaughn. R. 2005 A Cognitive Model for Alert Correlation in a Distributed Environment. Monitoring and Surveillance. LNCS 3495

Snort Intrusion Detection System Documentation. <http://www.Snort.org/docs>

Valeur. F., Vigna. G., Kruegal. C., and Kemmerer. C. 2004. Comprehensive approach to Intrusion Detection Alert Coorelation. Proceedings of IEEE Transaction on Dependable and Secure Computing. 146 - 169

Viinikka. J., Debar. H., Me. L., and Seguier. R. 2006 Time series modeling for IDS alert management. Proceedings of Asian ACM Symposium on Information, Computer and Communications security. 102 – 113

Wang. Y., Yang. H., Wang. X., and Zhang. R. 2004 Distributed Intrusion Detection System Based on Data Fusion Method. Proceedings of IEEE 5th World Congress on Intelligent Control and Automation. 4331 - 4334

Xiao. S., Zhang. Y., Liu. X., and Gao. J. 2008 Alert Fusion Based on Cluster and Correlation Analysis. IEEE International Conference on Convergence and Hybrid Information Technology ICHIT. 163 - 168

Xu. M., Wu. T., and Tang. J. 2008 An IDS Alert Fusion Approach Based on Happened Before Relation. IEEE Fourth International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM. 1 - 4

Xu. M., and Han W. 2007 Distributed Intrusion Alert Fusion Based on Multi Keyword. The First International IEEE Symposium on Data, Privacy, and E-commerce. 469 – 471

Yu. D., and Frincke. D. 2005 Alert Confidence Fusion in Intrusion Detection Systems with extended Dempster-Shafer Theory. *Proceedings of the 43rd annual ACM Southeast regional conference.* 142 - 147

Yusof. R., Selamat. S., and Sahib. S. 2008 Intrusion Alert Correlation Technique Analysis for Heterogeneous Log. Proceedings of International Journal of Computer Science and Network Security IJCSNS. 8(9)

Zamboni. D. 2001 Using Internal Sensors for Computer Intrusion Detection. PhD Thesis, Purdue University.

Zang. T., Yun. X., and Zhang. Y. 2008 A Survey of Alert Fusion Techniques for Security Incident. Web-Age Information Management WAIM '08. The Ninth International IEEE Conference, 475 - 481

VITA AUCTORIS

Faisal Mahmood was born in 1978 in Islamabad, Pakistan. He went on to the International Islamic University, Islamabad where he obtained B.Sc (Hons) in Computer Sciences in 1999 and M.Sc Computer Sciences in 2001. He is currently a candidate for the Master of Science degree in Computer Science at the University of Windsor and will be graduating in Fall 2012.