A Testbed for Quantitative Assessment of Intrusion Detection Systems using Fuzzy Logic

Gautam Singaraju[1] Lawrence Teo[1,2] Yuliang Zheng[1,2]

[1] *Laboratory of Information Integration, Security and Privacy (LIISP), University of North Carolina at Charlotte, 9201 University City Blvd, Charlotte, NC 28223, USA.*

[2] *Calyptix Security Corporation P.O. Box 561508, Charlotte, NC 28256, USA. http://www.calyptix.com/*

{*gsingara,lcteo,yzheng*}*@uncc.edu*

Abstract

The current Intrusion Detection System (IDS) technology is a major investment for a firm and its evaluation is desired prior to a commitment. A testbed compares different IDSs on a common platform. A major challenge in evaluating IDSs stems from the fact that they are generally tested in specific environments. A real-world environment could be different from the environment designed for a testbed. The results obtained, from such testbeds, may not be accurate and reliable. Hence, a quantitative and metrics based evaluation of IDSs is desired.

We propose Testbed for evaluating Intrusion Detection Systems (TIDeS), that allows a user to select the best IDS for a specific customized environment. A quantitative analysis is provided by TIDeS, using fuzzy logic, under varying network loads. We also propose robust metrics to evaluate an IDS. We follow up with recommendations, based on our experience, on the general practices in the field of IDSs.

Keywords: Testbed, TIDeS, Intrusion Detection, Environment Profile, Evaluation Framework, Scalability, Fuzzy Logic

1. Introduction

The evolution of security has brought forth systems that alert the user about possible intrusions into their network. Intrusion Detection Systems (IDSs) [32], as they are more popularly referred to, have become common components in the corporate and home networks and are still growing in popularity. Yet, a commercial IDS remains costly. Although there are few that are available for free, a user would ideally like to check their effectiveness before committing to one.

IDSs employ different technologies [7] and claim to effectively detect an intrusion. These various technologies evoke questions about their effectiveness and their comparative performance within a spectrum of network conditions. IDSs have been tested in specific environments, but their effectiveness under different environments have yet to be measured. Under scrutiny are network parameters; for example, the network bandwidth conditions and out-of-order packet sequences.

Careful evaluations of IDSs to determine their effectiveness by varying network parameters have always been desired [19]. Hence, generic evaluations to suit various network conditions along with the ability to scale the network allowing a user to add legitimate and illegitimate traffic are important factors in building a testbed.

The Air Force and The Defense Advanced Research Projects Agency (DARPA) [23, 24] in association with Lincoln Labs have been developing testbeds. Some of these systems are not available to the public to evaluate their own networks. The testbed by Lincoln Labs, Lincoln Adaptable Real-Time Information Assurance Testbed (LARIAT) [33], is one such system. These systems were based on a specific network implementation. For example, the initial testbed for Air Force evaluation was based on the Air Force environment.

Apart from a strong testing scenario, the need for a testbed that provides a robust and a reliable metrics to quantify an IDS has been suggested by National Institute for Standards and Technology (NIST) [26]. The metrics should be based on a quantitative analysis of the IDS by varying the network parameters. Legitimate protocols and illegitimate traffic should be easily included for testing by the system. Apart from these, a user should be able to customize the testbed to include the users' own traffic profiles. In other words, a testbed should be built with the plug and play ar-

Proceedings of the Second IEEE International Information Assurance Workshop (IWIA'04)
0-7695-2117-7/04 $ 20.00 © 2004 IEEE

chitecture and be scalable. The present state of the testbeds was the driving reason to put forth a generic and reliable means for evaluating an IDS.

We propose TIDeS, a testbed that evaluates IDSs on a common platform. The key to its strong evaluation framework is the reliable rating mechanism based on Fuzzy Logic [22], apart from the various testing mechanism. The testing mechanism includes environment profiles, as recommended by NIST [26], that are customizable by varying network load and using different protocols available with the testbed. TIDeS allows the testing of a set of IDSs to determine the best IDS amongst them in a specific environment and/or the users' environment.

The paper first discusses in detail the design specifications of the TIDeS framework. The rest of the paper has been organized as follows: Section 2 highlights the related work. Section 3 illustrates the TIDeS framework and discusses in depth its design ideas. Section 4 explains the various ideas of the Fuzzy logic that have been used in the design of the metrics. Section 5 demonstrates the various procedures that are involved in the testing and the evaluation of the framework. Section 6 discusses the results of test performance on an IDS. Section 7 discusses the future advancement to the system.

2. Related work

2.1. DARPA

DARPA [23, 24] perceived the need and problems in evaluating IDSs. Working with Lincoln Laboratory at MIT, on a testbed for IDSs from 1998-2001, DARPA has proposed an evaluation schema for the evaluation of IDSs.

Designed to evaluate an IDS depending upon both known and unknown attacks embedded in background legitimate traffic, the DARPA evaluation [24] schema used the Air Force network as the basis for their evaluation. The Air Force network was modelled by creating virtual machines, more specifically, IP addresses. Background traffic was launched and attacks were introduced in the network. The testbed measured the probability of detection of attacks and probability of false-alarm rates for the IDS under test. The false alarm rates were used for the comparative assessment of IDSs. However, the evaluations and corresponding data were not made public, due to confidentiality. Also, the suggestions were made that the testing was reportedly flawed as it could not accurately model the Air Force network [25]. The information was classified [25] and hence the design was not made public.

In the 1998 DARPA evaluation [23], there were about 300 instances of 38 different attacks that were launched during the testing phase. However, in the following year [24], 200 instances of 58 attacks were used. These tests lasted over a period of five days to a week; each day based on a 22-hour format.

In the 1998 DARPA evaluation, the network consisted of the simulated Internet and the internal network connected through a router. The attack database only consisted of the outsider attacks [20]. The internal network, however, was not tested. The 1999 framework included the insider attacks apart from Windows based attacks [21]. The framework used two different subnets [18]; the first behaved as a traffic generation system generating both legitimate and illegitimate traffic. The other subnet consisted of the targeted victim systems. An internal traffic generator generated the internal traffic, both legitimate and illegitimate, testing an IDS for the internal attacks. The 1999 attack database also included various other attacks that are classified into 'user to root access', 'remote to local', and Denial Of Service DOS probe attacks.

Real-time data collected from the Air Force network was used to simulate the background traffic that was launched according to a time-line. The IDS output was used to ascertain the false positives and the false negatives which are the measure of the effectiveness of an IDS.

Lincoln Laboratory, as a part of DARPA Framework has come up with enhancements to the testbed, which has been renamed as Lincoln Adaptive Real-time Information Assurance Testbed (LARIAT) [33]. Unfortunately, LARIAT has not been publicly available.

The corpus generated by the Lincoln Labs consisted of the background data. Additionally, the attacks towards the system can be replayed for different IDSs. Hence, having the same testing scenario for all the IDSs. Though this is used to the right effect, there are fundamental flaws in the traffic data itself as mentioned by the DARPA evaluation critique [25]. They are briefly mentioned below.

The number of attacks that were launched during the testing over 10 weeks (5 days/week) was 300 [20]. This would mean that the average number of attacks on these machines were 5-6 per day. This is significantly lower compared to the attacks on a network of utmost importance [25]. Also, the traffic characterization and the launching process needs to be improved as the network architecture requires to be scalable. The simulated traffic that was based on the Air Force Network, was not accredited to be correct.

An IDS has a break point in terms of the bandwidth: as the amount of data is increased, the false alarm rate decreases. This does not imply that the IDS detection capability has increased. As the false rate is inversely proportional to the total data that has been monitored, keeping the number of attacks constant and increasing the total data monitored, gives a false impression of the capabilities of the IDS.

Development of an accurate evaluation framework is essential, since false alarm rates for positive and negative ratio would not be the right basis for measurement for a perfor-

mance of an IDS. This is mentioned as a hidden fallacy [25].

2.2. Air Force architecture

The Air Force Research Laboratory (AFRL) [8] which participated with the DARPA in the 1998-1999 evaluations was inclined towards the testing of the IDSs under more complex hierarchial network conditions. The Air Force came up with an architecture consisting of five layers; namely, the scheduler, the master, the controller, the slave layer, the automata layer and the virtual networking layer.

The AFRL evaluations were also based on the Air Force network. In these evaluations, an IDS was tested with a 4-hours corpus that was generated. The DARPA and the AFRL developed and used the same software in the testbeds.

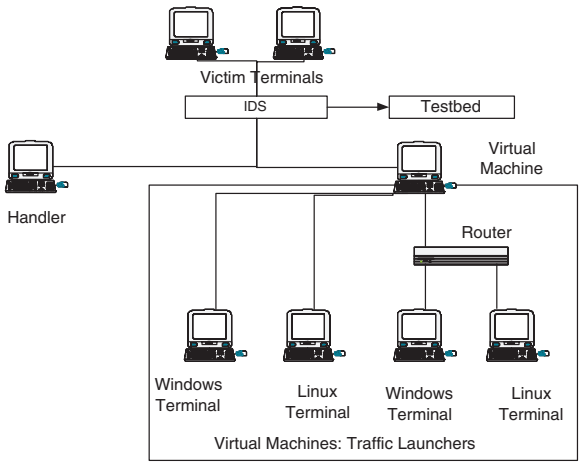

Figure 1. The Testbed architecture

3. Testbed for evaluating Intrusion Detection Systems (TIDeS)

There is a need for a scalable and rigorous testing platform for an IDS under different network conditions [1]. New attacks, each with a unique method of exploiting and compromising a system, are rolled out everyday. However, the present testbed technology limits the user's capability by not allowing the testbed to be scalable to the user's environment. A plug and play based architecture would allow new traffic scripts to be easily incorporated into the testbed.

The pros and cons of the earlier testbeds and their recommendations [8, 25] formed the basis for the development of Testbed for Intrusion Detection System (TIDeS). A scalable architecture, along with rigid metrics for evaluation, forms the foundation for the TIDeS framework.

With its strong mathematical background in Fuzzy Logic concepts, TIDeS provides the capability to evaluate an IDS on the user's network. The user can customize the testing scenarios by being able to add to or remove attacks from the attack database. Additional flexibility is because of being able to map an entire network onto a few computers.

The TIDeS is a scalable suite for testing and evaluating an IDS in various commercial and non-commercial environments. The addition of new traffic scripts requires minimal effort from the user. The framework consists of testing scenarios, classified on the basis of network bandwidth conditions and/or the real-time captured data for testing. These testing scenarios are classified into:

1. Non-environmental based testing scenario
2. Environmental based testing scenario

Non-environmental based testing scenario does not depend on data that has been collected on the network. The tests that are conducted in this scenario are described below. These tests are performed until the IDS, under scrutiny, or the network breaks down. These testing scenarios allows an IDS to be tested in a wide operating range.

The All-Legitimate traffic testing scenario launches only legitimate traffic during the testing phase. The network traffic is increased till the IDS or network breaks down, whichever occurs first. The number of false alarms are determined and would be classified as false positives as there are no attacks that were launched during this testing phase.

The All-Illegitimate traffic testing scenario launches only attacks from the attack database. During the testing, if an IDS does not detect an attack, it could be classified as a false negative. Just as in the All-Legitimate traffic scenario, the traffic is slowly increased to the break point of the IDS or of the network, whichever occurs first.

A Mixed traffic testing scenario launches both the legitimate and illegitimate traffic, generated randomly, but the launched traffic is logged. The IDS output and the logged launch traffic profile are used to determine the false alarms. Similar to the testing scenarios discussed above, the network load is increased to the break point of the IDS or the network.

Environmental based testing depends upon the traffic that has been captured from the user's network. The real-time network data allows a background undetected noise to be always present [19] and such an evaluation is desired. This testing scenario is important, as the evaluation of IDS is being performed under the actual network conditions; that is, the real time network parameters [26]. Hence, monitoring a user's network would create a common matrix, which we refer to as Environment Profile.

Such a testing of the entire spectrum of conditions leads to the effective evaluation of IDSs. The results from the testing is provided to the Fuzzy Logic Evaluation Framework.

An IDS is given a ranking depending upon its performance in the testing scenarios, viz., Non-Environmental and Environmental based testing scenarios.

The flexibility in TIDeS' architecture is due to its main components, the Handler, the Virtual Machine Emulator, the Launcher, the Environment Profile Generator, the Scripts and the Evaluation Framework.

A short description of the different parts of the framework has been defined in this section. The next subsections describe the components in a greater detail. The Handler interfaces with other components of the testbed. The Virtual Machine emulator maps the entire network onto a single computer. The Launcher launches the traffic by controlling the modules, namely agents, that generate the traffic on the network. Environment Profile Generator is used to generate the environmental traffic patterns of the user's network. Scripts are used to generate legitimate and illegitimate traffic on the network. The Evaluation Framework consists of the mathematical model that evaluates the effectiveness of an IDS using Fuzzy Logic.

3.1. Handler

The handler is the main controller and is an interface to the testbed. The handler has the capability of monitoring the tests, apart form interfacing with the other components of the testbed. Figure 2 shows the handler launching legitimate traffic, with the duration of test being 30 minutes and a network load of 100 kilobytes.

Handler, interfacing with the launcher which in turn interfaces with agents, provides a scalable system in terms of the testing conditions. Agents are the software modules that perform the task of launching the traffic after receiving control signals from a launcher. As shown in the Figure 2, Agents shown can be modified to add or remove different scripts.

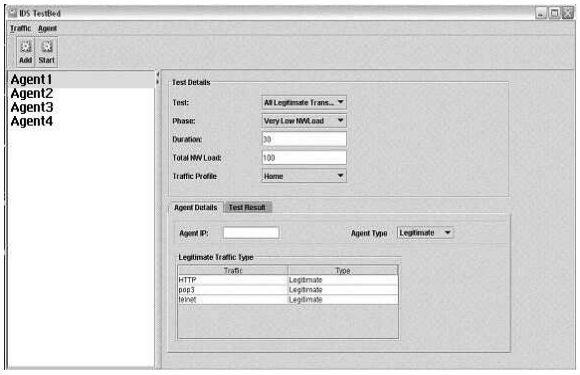

Figure 2. The Handler

The interface shown is also capable of selecting the environment profile and sending control signals to agents. Apart from these, the handler is portable to various operating systems, thus increasing scalability.

3.2. Virtual Machine Emulator

For testing an IDS, a testbed should appropriately create traffic sessions originating from different IP addresses. However, creating such an architecture would require enormous requirements in terms of computers and is a physical constraint to implement an entire network. A compromise on the number of computers would have to be made because of the cost constraints. A solution to this problem would be to create virtual machines on a single physical computer.

The Virtual Machine Emulator emulates numerous virtual machines with unique IP addresses on a single physical machine, thus creating an effect of a virtual network. This component also has the capability to emulating routers and each Virtual Machine can have a different Operating System.

3.3. Launcher

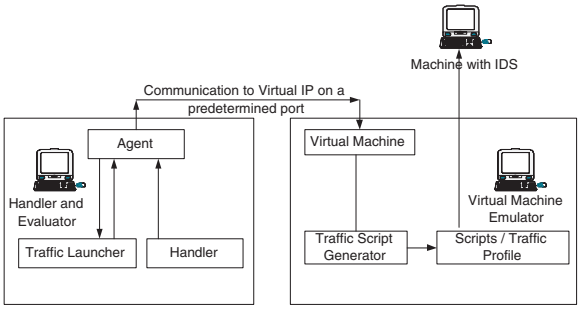

Figure 3. The communication for architecture for launching traffic in the Testbed

The traffic is generated by the Launcher when a control signal is received from the handler through the Agent and then the Virtual Machine Emulator as shown in Figure 3. The Launcher in turn activates the scripts that generate traffic. The scripts interface with the various services with the computer on which the IDS installed and tested.

The Launcher can launch an environment profile as described in subsection 3.4. The handler activates the launcher which in turn activates the environment profile. By accessing the different services, the scripts create the traffic on the network.

Figure 3 shows how the communication architecture of TIDeS is organized. The Handler as indicated in subsection 3.1 activates the agents. These Agents load the environ-

mental profiles or the traffic scripts. These control signals activate the Virtual Machine Emulator subsection 3.2. The control is then passed onto the scripts. The scripts generate the network traffic.

3.4. Environment Profile generator

To test an IDS, real-time traffic analysis is required. The environment where a system is installed determines the performance of the system [25]. As suggested, testing an IDS with a standard benchmark would not be an ideal solution. For instance, an IDS working in a home environment would not be an ideal IDS working on a university network.

TIDeS has an important feature of letting the user configure the amount of legitimate and illegitimate traffic that is to flow through the network. This flexibility allows mapping the traffic on the users' network onto the TIDeS testbed for traffic simulation.

The environment profile is generated from the real-time conditions by analyzing networks. The Environment Profile is exported to the machine that hosts the Virtual Machine Emulator. These requests are used to create traffic on the network. The traffic generator generates different environment profiles for each of the IP address. Hence, effectively all the sub-profiles constitute the entire Environment profile.

Figure 4 and Figure 5 illustrate the mechanism to create an environment profile. Figure 4 shows the process of separating the various IP addresses from a particular server dump. Given server dumps from n servers, the parser program parsers these dumps and creates virtual profiles for individual IP addresses. Figure 5 shows the process of combining the various generated individual files for a virtual IP address.

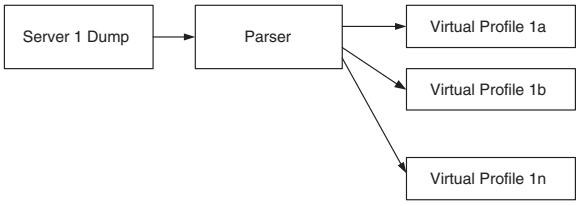

Figure 4. Creation of Environment Profile - separating data logs of a server

Listed below are the Environment Profiles that are present with the TIDeS framework.

1. University Environment Profile
2. Stand-alone Environment Profile
3. Home Environment Profile

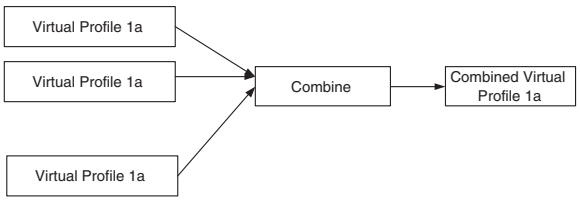

Figure 5. Creation an Environment Profile - combining various server logs

3.4.1. Protocols monitored. IDS detection capabilities vary depending upon the amount of background traffic present as shown in by [19]. There are a variety of protocols on the Internet and more are added by the day. Some of the most popular ones have been used to formulate the Environment profile of the system, as these are the backbone protocols of the Internet.

The TIDeS testbed has the capability to add new protocols. The default protocols with TIDeS are HTTP, SMTP, POP3, TELNET, FTP and SSH. The subsection 3.5.1 discusses the use of default protocols that were used in the evaluation, in more detail.

Similar work has been performed for HTTP [9, 34] and also for mail server protocols, SMTP and POP3 [6].

3.4.2. Capturing schema and Environment Profile. The data was captured for a period of 7 days each over a period of 24 hours. The data captured was the connection requests by the machines.

University Environment Profile The details of the four servers are given below:
Server 1 is a server that accepts HTTP connections. Server

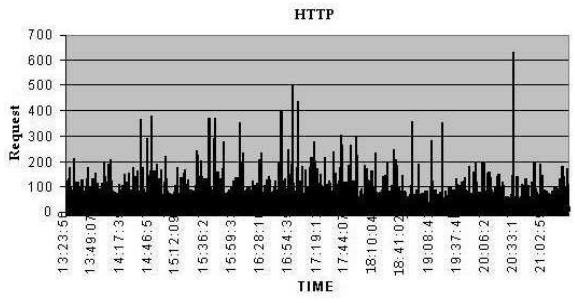

Figure 6. HTTP Connections to server 1 Working Day

2 is an interactive server that accepts SSH, TELNET and FTP connections. Server 3 is one of the two mail servers. It accepts SMTP connections. Server 4 is the other mail

servers, and accepts POP and IMAP connections. Both the mail servers also accept SSH connections from the management staff, but not from the general users. These four servers are used in a university's network. The amount of connections and the timing of the connections were studied over considerable time. The servers had been inoperative for a few minutes everyday, early in the morning, for maintenance. The servers were working for 'working day' period in a day, which includes the few minutes of maintenance. These servers run on Sun Solaris OS which has Snoop as a packet capturing application developed by Sun Microsystems.

Home Environment Profile The Home Environment Profile is generated by monitoring a Home system. Typically, these systems are exposed to many attacks from the Internet for the short duration they are exposed to the Internet. Home users typically connect using their modems and are connected to the Internet with a slow connection, usually 56kbps. At these rates, the systems cannot handle high throughput and hence a different scenario for evaluation.

This profile need not be monitored for a longer period of time as these systems connect to the Internet for approximately an hour or two at the maximum and connect during different times of the day depending upon usage.

The connections and the data throughput was measured for a 3-hour period at a home environment computer that was connected to the Internet.

Stand-alone Environment Profile The Stand-alone Environment Profile is generated by monitoring a Stand-alone system, that is connected to the system and is not disconnected from the system for long periods of time and is not normally shut down.

Connected to broadband, these machines are not only vulnerable to attacks from the Internet, because of their high speed connectivity, but also from insider attacks the organization.

This Environment profile has also been monitored for over 24 hours a day for 7 days a week and a profile is drafted.

3.4.3. Virtual Environment Profile. The traffic to a computer system comes from many sources of the Internet. To effectively evaluate the IDS, the TIDeS testbed has to generate an accurate model of the various network sources. This requires a vast infrastructure. An alternate idea is to have a Virtual Machine Emulator that emulates different machines running different operating systems on the same machine.

The virtual network setup has to be created including routers to other network components. This would enable testing the IDS with traffic emanating from different IP addresses. To achieve this, a popular system called as Honeyd [30, 35] has been used.

Honeyd is a popular Honeypot [35]. Honeypots create virtual network to study the process of hacking by the hacker community. All traffic coming for various non-existent computers in network are created using Honeyd. Honeyd interacts with the hacker as though there is a real physical computer with an IP addresses and an Operating System.

3.4.4. Honeyd used in TIDeS. Honeyd is typically a system that is used to monitor hacker activities. Honeyd has been used in a novel use in the TIDeS setup. With the help of Honeyd IP addresses are generated.

Handler sends a setup signal to the virtual IP address on a predetermined port. Honeyd receives this information and initializes of a particular Environment profile. The environment profile would then provides requests to the computer or network that is to be tested. The traffic is generated until the environment profile is exhausted or a stop request from the handler.

Figure 1 shows a configuration from the handler with Honeyd on the Virtual Machine Emulator (VME). An environment profile server runs. As and when there is a connection to it for a particular IP address, the VME starts by launching the requests. Each of the different IP address run a different environment profile script. Virtual Machine Emulator module has been an important module to the TIDeS as it allows a mapping to a large number of IP address, from a single physical computer, for testing purposes.

3.5. Scripts

The detection capabilities of an IDS vary differently in different traffic conditions. Different statistical throughput, delay or even the order of requests for traffic introduces changes in the detection capabilities of the IDS. Hence to test IDS for its true capability, there is always a requirement for generating authentic "real-time" scenarios.

The real-time scenarios are created, by automated scripts, by the different environment profiles. These scripts interact with different servers having services installed on them. TIDeS, with its plug-and-play architecture, allows traffic protocols to be incorporated into the evaluation framework.

The Scripts are operating system independent and are activated by a launcher as shown in figure 3. The scripts connect to the server and interact with these services on the server. There are 6 legitimate traffic scripts and approximately 40 attack scripts.

An IDS's capabilities are dependent on the number of times the IDS correctly distinguish an illegitimate traffic

Attack	*Service*
Passguess	POP3 protocol
Apache2	Apache webserver
Back	Apache webserver
CrashIIS	IIS webserver
DOSnuke	NetBIOS - Windows machine
Imapd	IMAP server
Land	Older TCP/IP implementations
Mailbomb	SMTP protocol
Processtable	SMTP protocol
SSHProcesstable	SSH protocol
Teardrop	Older TCP/IP implementations
Crashpentium	Pentium hardware
Ctrld	FTP protocol
Stacheldraht toolkit	DDOS - 4 different types
Trinoo Toolkit	DDOS
Vanilla portscan	Scans for ports
Stealth portscan	Scans for ports
Half-open portscan	Scans for ports
UDP portscan	Scans for ports
FTP bounce attack	FTP attack
PingSweep	ICMP protocol
Fingerprinting	ICMP protocol
Xdestroy	X-server
Xkey	X-server
Xscan	X-server
Xsnoop	X-server
Xtester	X-server
Xwatchwindow	X-server

Table 1. Default attack scripts present with TIDeS

from the background traffic. Many ratios to calculate the capabilities of the system have been proposed in the framework in section 3.6.

The subsections below indicate the different legitimate and illegitimate traffic that are incorporated into the TIDeS framework.

3.5.1. Legitimate traffic. Legitimate traffic is used to generate the different background traffic to test the IDS. The default protocols that were monitored in subsection 3.4.1 were used to create the traffic on the network. TIDeS has incorporated about 6 legitimate protocols into it, namely, HTTP protocol, SMTP protocol, POP protocol, SSH protocol, FTP protocol and TELNET protocol.

TIDeS has been built as a plug-and-play system and hence new traffic can be incorporated into the system. All that needs to be done is to create script for the protocol and have the service present on the system to be tested upon.

Network Condition	*% of the Network load/Error rate*
Very Low	0-20
Low	20-40
Medium	40-60
High	60-80
Very High	80-100

Table 2. Network conditions and their range setting.

3.5.2. Illegitimate traffic. Illegitimate traffic scripts create attacks on the network. TIDeS has about 40 attack scripts incorporated into it. And again, because of a plug and play architecture, new attack scripts can be easily included into the architecture.

The illegitimate traffic scripts that are with the TIDeS architecture is shown in Table 1. Some of the attacks that are presented are directed towards the mail servers. Also some of Distributed Denial Of Service attacks have been included [14]. Some of the novel ways have been suggested [10, 31] others are Windows based attacks [21]. Trinoo [15], the popular toolkit has been included. Trinoo performs 4 types of Distributed Denial of Service attacks(DDOS). Stacheldraht [16], another DDOS tool, has been included too.

3.6. Evaluation framework

NFR security systems [32], Air Force Evaluations [8] and, other organizations [3], have mentioned problems with the IDS benchmarks and have laid out guidelines to effectively test an IDS. Most recently, NIST [26] has proposed new standards for creating a testbed and has recommendations for the design of testbeds.

The TIDeS evaluation framework has many parameters that are used to evaluate IDS. Depth, defined as number of attacks detected by the system to the total number of known attacks; breadth, defined as the number of unknown attacks to the attacks detected that fall outside the framework of the system's attack database; false alarms, performance under stress, reliability and accuracy of detecting individual attacks are few of the parameters.

The evaluation is based on error rate and network load parameters. The decision making process is based on the concepts of fuzzy logic and fuzzy rules. The fuzzy logic sets are shown in Table 2. The evaluations involved in the performance evaluation are performed with the help of percentages of false positives, false negatives, and cumulative false alarms. Figure 8 explains the fuzzy rules and the mapping of the fuzzy set.

Apart from this, the evaluation depend upon Stress test, Consistency and Reliability test and Accuracy test. Table 3

	Very Low	*Low*	*Medium*	*High*	*Very High*
Very Low	Good	Good	Good	Excellent	Excellent
Low	Average	Average	Good	Excellent	Excellent
Medium	Poor	Poor	Average	Average	Good
High	Very Poor	Very Poor	Poor	Average	Poor
Very High	Very Poor	Very Poor	Very Poor	Poor	Poor

Table 3. The Fuzzy evaluation rules

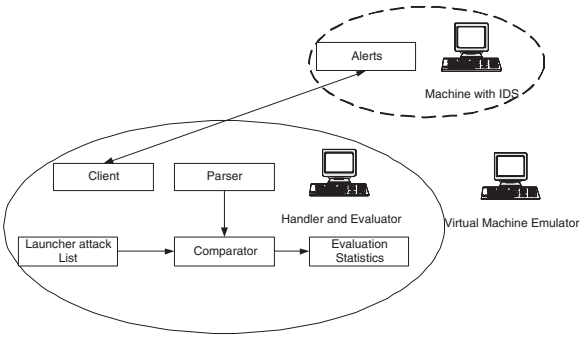

Figure 7. Communication architecture for finding false positives and false negatives

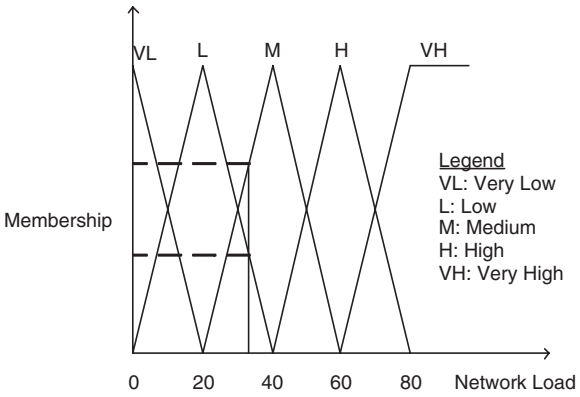

Figure 8. The Fuzzy Rules and the mapping of the fuzzy set

describes the rules for the evaluations.

Figure 9 shows the TIDeS with the evaluator. It can be seen from the figure that the testbed, through of Agents, Handler generates network traffic. The traffic generated from the launcher and the IDS outputs are compared by the evaluator. The evaluator helps determine the performance of an IDS. Other taxonomies [2, 13] give suggestions for evaluation metrics for IDS along the guidelines suggested in [17]. These classes are:

- Managerial and Architectural Metrics
- Performance Metrics
- Analytical Metrics
- Interactivity Metrics

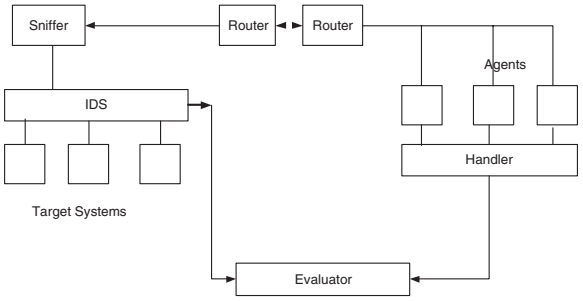

Figure 9. The Testbed with Evaluator

3.6.1. Managerial and Architectural Metrics. These metrics evaluate the architectural efficiency of an IDS. These are subjective measurement criterion requiring user evaluation. These metrics are:

- *Distributed Management*: Determines the distribution capabilities among different analyzers. It is used to determine the extent an IDS supports distributed management.
- *Configuration Difficulty*: The ease with which the IDS can be installed and configured by the user. How well a user understands the deployment of an IDS would enable a correct deployment of the IDS.
- *Ease of Policy and License Management*: The ease of setting security and intrusion detection policies as well as the difficulty in obtaining, updating and extending licences.
- *Availability of Updates*: The availability and cost of updates of signature and/or behavior profiles as well as the availability and cost of product upgrades.

- *Adjustable Sensitivity*: The ease of altering the sensitivity of IDS at various times and for different environments in order to achieve a balance between false positive and false negative error rates.

- *Data Storage Capacity Needed*: The amount of disk space consumed for storing the signature profiles, logs and other application data.

- *Scalable Load Balancing*: The performance of load-balancers and its effects. It measures the ability of an IDS to partition traffic into independent, balanced sensor loads, and the ability of load-balancing subprocess to scale upwards and downwards.

- *Multiple Sensor Support*: The cardinality of sensors supported.

- *IP Fragment vs Stream Reassembly*: According to [32], it is necessary of IDS today to do *reassembly* and defines three important reassembly-related activities that an IDS can perform, namely *defragmenting, reordering* and *stream reassembly*[1]. An IDS that does not perform reassembly can easily miss an attack that has been artificially fragmented and transmitted out of order. This parameter, however, is difficult to evaluate as it is difficult to extract details from the vendors pertaining to the criteria used in reassembling the TCP streams.

- *State Tracking*: This is yet another critical subjective metric as it works effectively to reduce false positives. A network IDS that performs state tracking will know what sessions the target sees and will not raise alerts on traffic that the target would discard as invalid. This is useful in hardening the NIDS against storms of random traffic used to confuse it. It also means that the IDS will be able to keep accurate information of TCP session start-up times, the client-server relationship, and amounts of data transferred in either direction.

3.6.2. Performance Metrics. These metrics measure and evaluate the parameters that impact the performance of the IDS. They measure the ability of an IDS to perform a particular job and to fit within the performance constraints of the monitored system. We use the Fuzzy-Logic based evaluation framework for these metrics. The performance metrics considered are:

- *Observed False Positive Ratio*: This is the ratio of alarms wrongly raised by the IDS to the total number of transactions. Considering 'D' to be the set of IDS detected intrusions and 'A' to be the set of actual intrusions, with 'T' being the total transactions, the *False Positive Ratio* is given by

$$\frac{|D - (D \cap A)|}{|T|} \tag{1}$$

The mathematical equation for this is:

$$\frac{Num.ofFP \times 100}{TotalTransactions} \tag{2}$$

- *False Negative Ratio*: This is the ratio of actual attacks that are not detected by the IDS to the total number of transactions. This is given by

$$\frac{|A - (|D - (A \cap D|))|}{|A|} \tag{3}$$

where the terms *A, D* and *T* are the same as in 1.

Mathematically:

$$\frac{Num.ofFN \times 100}{TotalAttackTransactionsLaunched} \tag{4}$$

where Num. of FN is calculated as

$$noFN = TotalAttacks - TotalTruePositives \tag{5}$$

[2]

- *Cumulative False Alarm Rate*: The weighted average of False Positive and False Negative ratios.

- *Induced Traffic Latency*: Given by the delay measured in the arrival of the packets at the target network in the presence and absence of an IDS.

- *Stress Handling and Point of Breakdown*: The point of breakdown of an IDS is defined as the level of network or host traffic that results in a shutdown or malfunction of IDS. It is measured as packets/sec or number of simultaneous TCP streams.

- *IDS Throughput*: This is defined as the observed level of traffic up to which the IDS performs without dropping any packets.

3.6.3. Analytical Metrics. The metrics considered in this class are as follows:

- *Depth and Breadth of System's Detection Capability*: The *depth* of the IDS' detection capability is defined as the number of attack signature patterns and/or behavior models known to it. The *breadth* of the system's detection capability is given by the number of attacks and intrusions recognized by the IDS that lie outside its knowledge domain.

[1] The process of combining multiple TCP segments so that they represent a complete stream of data as the target system received it.

[2] A *true positive* is when an attack is successfully detected by the IDS given by total alarms raised by the IDS - num. of FP

- *Reliability of Attack Detection*: This is defined as the ratio of false positives to total alarms raised. Reliability of attack detection is given by:

$$\frac{|D - (A \cap D)|}{D} \tag{6}$$

Mathematically:

$$\frac{Num.ofFP \times 100}{TotalAlarmsRaised} \tag{7}$$

- *Possibility of Attack*: This is defined as the ratio of false negatives to true negatives [3]. Possibility of attack is given by

$$\frac{|A - (|D - (A \cap D|))|}{|T - D|} \tag{8}$$

Mathematically:

$$\frac{Num.ofFN \times 100}{TotalTransactionsPassedbytheIDS} \tag{9}$$

where Total Transaction Passes by the IDS is given by Total Trans. Passed =
Total Transactions Launched - Total Alarms Raised

- *Consistency*: This is given by the variation in the performance (false positive and false negative measurement) of an IDS under varying network load and traffic environments.

- *Error Reporting and Recovery*: The extent of event notification and logging. This is again a subjective criteria requiring user discretion.

3.6.4. Interactivity Metrics. These are again a set of subjective metrics demanding user analysis. These metrics are:

- *Firewall Interaction*: Ability to interact with the Firewall systems.

- *Router Interaction*: Degree to which an IDS interacts with the router and redirects attacker's traffic to a Honeypot.

- *SNMP interaction*: Ability of an IDS to send an SNMP trap to one or more network devices in response to a detected attack.

- *User friendliness*: The ease to set up and configure an IDS in users' environment.

[3] A *true negative* is when a non-attack packet is allowed to pass through by the IDS

4. Fuzzy Logic and its use in IDS evaluation

The evaluation of Intrusion Detection Systems is inherently complex and non-linear in nature. The dependency of the IDS on the operating environment makes it important for us to have a flexible and yet unbiased evaluation framework. Fuzzy systems come in handy in designing such applications as they combine the high level flexibility and knowledge representation of conventional decision support and expert systems with the power and analytical depth of natural computing paradigms [11]. Their proven ability as universal approximators coupled with their ability to handle complex, non-linear, and often noisy systems with a minimum set of rules makes them a powerful tool in design and construction of intelligent, information decision support systems.

Among the basic concepts that underlie human understanding, three stand out in importance: granularization (partitioning of whole into parts), organization (integration of parts into a whole) and causation (association of causes with effects). A granule may be viewed as a clump of points (objects) drawn together by indistinguishability, similarity, or functionality. Modes of Information Granulation (IG), in which granules are crisp, play an important role in many theories, methods and techniques like interval analysis, quantization, rough set theory and qualitative process theory. These theories do not reflect is that human reasoning and concept formation granules are fuzzy, as are their attributes and their attribute values. Fuzzy IG, on the other hand, plays a pivotal role in the remarkable human ability to make rational decisions in an environment of partial knowledge, partial certainty, and partial truth. Fuzzy logic provides machinery for dealing with fuzzy information granulation in ways that parallel human reasoning and decision-making process.

The generalization of two-valued logic leads to multi-valued logic and parts of fuzzy logic. Any theory, method, technique or problem may be fuzzified by replacing the concept of a crisp set with that of a fuzzy set. Similarly, any theory, method, technique or problem can be granulated by partitioning variables, functions and relations into granules.

4.1. Fuzzy Logic basics

A fuzzy set has been defined as a collection of objects with membership values between 0, a complete exclusion, and 1, a complete membership. The membership values express the degrees to which each object is compatible with the properties or features distinctive to the collection [29].

Fuzzy sets and fuzzy logic have become one of the emerging areas in contemporary technologies of information processing. Fuzzy sets involve capturing, representing and working with linguistic notations - objects with unclear

boundaries.

A fuzzy set is characterized by a membership function mapping the elements of a domain, space, or universe of discourse *X* to the unit interval [0, 1] [22]. That is,

$A : X \rightarrow [0, 1]$

Thus , a fuzzy set *A* in *X* may be represented as a set of ordered pairs of a generic element $x \in X$ and its grade of membership:

$A = (x, \mu_A(x))|x \in X)$.

μ_A is the membership function associated with the fuzzy set *A*. The value of the function denoted as $\mu_A(x)$ describes the degree of membership of *x* in *A*.

Fuzzy systems are knowledge-based or rule-based systems [36] at the heart of which is a knowledge-base system consisting of the so-called fuzzy IF-THEN rules. A fuzzy IF-THEN rule is an IF-THEN statement in which some words are characterized by continuous membership functions. For example, the following is a fuzzy IF-THEN rule:

IF the false alarm rate of the IDS is high,

$$THEN lesser score is awarded to the IDS. \quad (10)$$

where the words 'high' and 'less' are characterized by the membership functions shown in figure 10.

Figure 10 illustrates membership function for 'high', where the horizontal axis represents the false alarm rate of an IDS and the vertical axis represents the membership value for high and Membership function for 'less', where the horizontal axis represents the points awarded to IDS performance and the vertical axis represents the membership value for less.

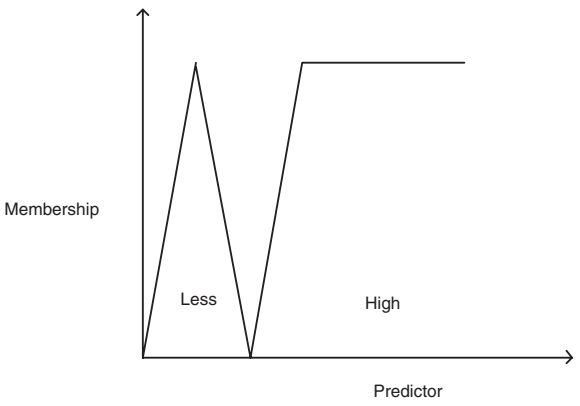

Figure 10. Membership function for 'High' and 'Less'

The starting point of constructing a fuzzy system is to obtain a collection of fuzzy IF-THEN rules from human experts or based on domain knowledge. The next step is to combine these rules into a single system. Different fuzzy systems use different principles for this combination.

4.2. Fuzzy Logic with IDS

In the measurement of the IDS capabilities, numerous parameters and evaluation metrics to be considered. The impact of an IDS' accuracy is organization specific. Some organizations may have high tolerance of false positives because staff and time availability to investigate them, while other organizations would rather have a system that misses some attacks as long as it doesn't raise many false alarms.

Consequently, the user would like to have a test combine the false positive/false negative information into a single synthetic value and grade the performance over a rigid evaluation framework. A flexible and unbiased framework in evaluating the performance of various IDS. A static linear mathematical formula-based model may not provide the required flexibility and may not encompass the organization requirement criteria. Fuzzy logic, on the other hand, provides a simple non-linear logical solution to such problems that is flexible and that easily encompasses the organizational IDS constraints. It provides engineers the decision making tool that parallels human reasoning.

The Fuzzy set approach starts off by encapsulating all available domain knowledge and organizing it into a manageable format. A collection of 'IF-THEN' rules forms a suitable control and decision making protocol. Most importantly, these rules include linguistic terms as in equation 10 that are inherently associated with the generalization aspect. Because of the assumed generality, the same protocol (underlying control and decision making philosophy) can be used successfully in a broad spectrum of IDS evaluation situations, once the linguistic terms have been sensibly calibrated. For example, the term *heavy* network load means something different during the peak hours of work in a lab in a university than at a highly protected private government networks. The context is of paramount importance, and fuzzy sets are ready to cope with this conceptual challenge.

4.3. Fuzzy Sets and its Membership values

Selection of linguistic variables and its constraints and calibrating it to perfection is the key for building a good fuzzy-logic based decision making framework. An advantage of using fuzzy logic is that we can use the same linguistic terms and sets for different metrics and recalibrate the set constraints for different traffic profiles and organizations. This flexibility is extremely important in the evaluation of systems as complex as the Intrusion Detection Systems. It allows us to use a single framework for a variety of IDS evaluation situations.

Keeping this reusability in mind, we have defined generic fuzzy linguistic terms and have calibrated the set constraints on the scale of 0 to 100. The linguistic variables defined for the input in Figure 11 variables are defined in Table 2.

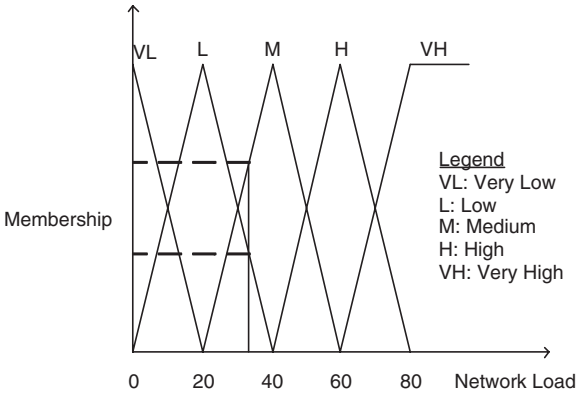

Figure 11. Fuzzy sets for input variables like network load

The set variables can be used to represent various parameters like network load, false positives, false negatives, cumulative false alarm rates etc,. The set constraints can be calibrated differently for different organizations and traffic requirements. The output of the Fuzzy system can also be modelled on the same Fuzzy sets or can also be change as per the user requirements.

5. IDS testing and evaluation

The testing of IDS proceeds in TIDeS is through various different grading strategies. The metrics developed in subsection 3.6 included different metrics including the Managerial and Architectural Metrics, Performance Metrics, Analytical Metrics and Interactivity Metrics. Some of these Metrics that are defined are subjective and depend on the user inputs. The non-subjective metrics require testing and hence the various tests needs to be incorporated into the system.

5.1. The Evaluator

The evaluator performs the grading of an IDS depending on the results of the tests that are performed on the IDS. The results from the test program [4] and the IDS are fed to the evaluator which then grades the performance of the system. There are two crucial elements that the evaluator considers for grading:

- Error Rate (False Positives and False Negatives)
- Network Load

The grading is based on various comparisons between the error rates and network load. The basic decision making process is based on the concepts of fuzzy logic and fuzzy rules. The network load and error rates are classified into various fuzzy category sets as described in Figure 11.

The tests are carried at various levels, testing the different aspects of the IDS performance. The tests start at a very low network load and proceed to the overflowing point and the system is thereby tested for its performance at varying levels of network load.

5.2. Basic tests

Below is outlined a brief sketch of the possible tests and its corresponding evaluation scheme.

- *Test 1*: Testing for False Alarms
  - *Case 1*: False Positives
    Only the attack traffic is launched and is directed towards the system in an isolated test network starting at very low network load range. The Network load is measured as the percent of total network bandwidth occupied by the traffic. The % false positive alarms are measured as per Equation 1. Mapping the %FP and average network loads during the testing phase, onto their respective fuzzy sets and obtain their membership in the corresponding set. During the period of test, the network load would be fluctuating between the minimum and maximum range values. The network load input is therefore the average network load experienced during the test period. The testing is carried on until the system breaks down and falls into the poor or the very poor category. The value of the test determines the false positive membership of the system.
  - *Case 2*: False Negatives
    A similar process process is repeated for false negatives with only legitimate traffic launched at the IDS. The amount of traffic predicted as attacks now become the false negatives. Similar calculations are made for false negatives giving us the output false negative performance set.
  - *Case 3*: Cumulative False Alarms
    The output sets obtained in the above tests are fed back to the fuzzy evaluator to obtain a cumulative performance report for the system. This process is known as forward chaining, where the fuzzy result of one test is forwarded for further evaluation. The evaluation process would be similar to

[4] indicating the traffic type launched onto the network and the number of such transactions

the above discussed method, giving us a precise grade for the system's error rate performance on a fuzzy scale.

- *Test 2*: Testing Performance Under Stress
  From the tests performed in the previous scenario, a plot of graph of % error v.s. network load is made. The deterioration in the performance of the system yields the stress analysis.

- *Test 3*: Consistency and Reliability

  – *Case 1*: Error consistency test: The test is similar to test 1. However, the network traffic is a mixture of legitimate as well as attack traffic. The %error in this case is measured as follows:

$$\%Error = \frac{(\%FP + \%FN) \times 100}{totaltransactions} \quad (11)$$

  Similar to test 1, we have the performance of the IDS tested at various network loads and its consistency checked against the results of test 1. A reliable IDS would be the one with consistent performance. The extent of reliability would be obtained by forward chaining the results of test 1 and test 3 to the evaluator.
  Besides error consistency, we also measure the ratio of %FP to %FN and the possibility of attack given by
  Percentage possibility of Attack =

$$\frac{\%FN \times 100}{totaltransactionspassed} \quad (12)$$

  – *Case 2*: Consistency over period of time: Launching a mixed traffic mix at various times over the period of a day, week or month and an analysis of the consistency in the IDS result is measured.

- *Test 4*: Testing with Environment Traffic Profiles In this test, an organizational environment is simulated using an environment profile. Capturing schema is explained in the section. During the test, the traffic launching is modelled in a way to simulate the exact pattern of traffic as was observed during the data gathering phase. The purpose of including this provision is to have the IDS tested in an almost real-time environment with real-time traffic generated by the testbed. The IDS is evaluated for all the metrics and comparison is made between the real-time environment results and the simulated environment result.

5.3. Other tests and final assessment

- *Depth test*:
  Theoretical Depth: This is the number of attacks known to the IDS opposed to total number of known attacks.
  Practical Depth: This is the number of attacks actually detected by the IDS.

- *Breadth test*: This is given by the number of attacks detected by the IDS that are unknown to it.

Final assessment is made based upon the results of all the tests performed. Either taking the average of all the results obtained or using the forward-chaining mechanism, an final assessment of an IDS can be made.

6. Results

Various quantitative analysis is performed on the IDS during the testing phase with the TIDeS framework. These quantitative values are fed to the evaluation framework that performs the fuzzy evaluations of the systems. However currently, IDS do not have a standard method of displaying alerts. This remains a serious flaw in the IDS output format as they do not conform to a similar alert messages.

There are efforts to get a common signature for IDS output formats into XML format to determine various attacks and their time stamp. IDMEF, a common messaging format being proposed by Intrusion Detection Working Group [12], is an effort towards this direction. However, the effort is in very primitive stages. Many IDSs have not adopted these standards. Some IDSs were taking steps towards this direction, at the time of writing of this paper. However, the process of creating a common platform for all the alerts for all IDSs needs to accelerated.

A preliminary analysis has however been performed under these constraints, by approximating the alert time and the time of launching the attack on the network.

The preliminary results shown below, are performed where there are alerts generated by an IDS when there was no illegitimate traffic launched on the network. The evaluations were performed on the working of an well-known IDS. The testing launched 897 legitimate traffic transactions. The number of attacks that were detected were 170 under a network load of 10% of a T1 LAN conditions. This indicates a 18.5% error in the detection capabilities.

Figure 12 shows the various fuzzy set evaluation for %FP (False Positives) for the test phase 1 for All-legitimate traffic. The evaluator also performs evaluations for %FN (False Negatives), %Network Load, % Error and % attack possibility. The same is performed for all the different testing

phases. The overall performance will give the complete performance of the system. For the fuzzy sets indicated in Figure 11, the evaluations indicate a 41.8% False positive error and 58.1% as the error score.

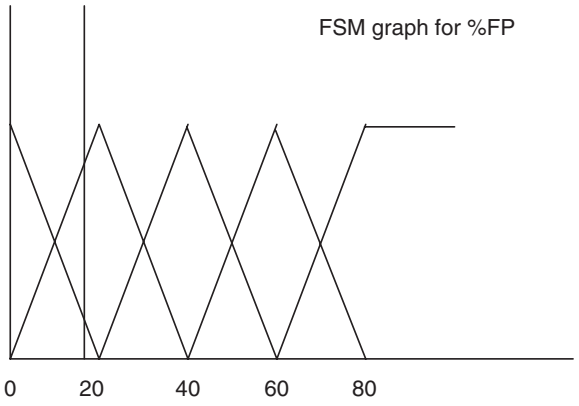

Figure 12. The Fuzzy Sets evaluations for Phase 1

However, as mentioned before, IDSs available need to have a common IDS output format. Further testing of various IDSs depend upon a step towards the IDMEF by these IDS. Even if some of the IDSs do not conform to such a methodology, a complete testing of the IDSs in the market would not be possible, unless the IDS outputs are evaluated manually by a large group of personnel. Provisions with the evaluator allow the manual feed of the false alarms and evaluate the IDS.

7. Conclusion and future work

The development of traffic profiles and the evaluation framework allows TIDeS to be built to evaluate systems in the users environment. Moreover, since TIDeS has been built in a plug and play architecture, the user can easily introduce or remove scripts depending on users' environment. The TIDeS system consists of approximately 40 illegitimate scripts and has 6 legitimate traffic scripts. As more protocols and attacks are being discovered, more legitimate and illegitimate scripts will be added to the system.

Fuzzy Logic has been used to evaluate an IDS. These technique of evaluation allows a set of values to be mapped onto the performance of a system. Lingual terms are mapped onto values in the system with the help of this fuzzy logic evaluation schema.

However, the testing of the systems have been impeded by the fact that the output of the IDS are not conforming to a standard format. There is a move towards a standard output using IDMEF. The process of checking the signature using IDMEF, which converts the output of a system into XML format, has to be tested with TIDeS.

As many attacks are being discovered everyday, more illegitimate traffic scripts need to be added into the system. The second direction towards the future work is to incorporate more scripts into the system. As the system is plug and play format, the executable scripts can be simply plugged into the system. Attacks mentioned in [27, 28] can be implemented to increase the attack set. Apart from these, attacks that can be launched as a process of integrating the SMTP protocol with MTA agents [5] and some ICMP protocol based legitimate and illegitimate scripts can be implemented [4]. These are presently very important as ICMP is emerging as a widely used protocol.

Another important direction of future work, is the environment profiles wherein the simulation is performed by capturing data from the users' network. More data is to be collected to get an average over time to average the effect of background noise. Adding more Environment Profiles to the system into the standard traffic profiles will allow the user to test the IDS with the system that might very closely resemble users' environment, in case, the user does not wish to capture data on the network. The TIDeS can be scalable to more accurately model the environment profiles by setting up victim and normal machines.

References

[1] N. Athanasiades, R. Abler, J. Levine, H. Owen, and G. Riley. Intrusion detection testing and benchmarking methodologies. *IEEE International Information Assurance Workshop, Darmstadt, Germany*, pages 63–72, 2003.

[2] S. Axelsson. Intrusion detection systems: A survey and taxonomy. Technical Report 99-15, Chalmers Univ., 2000.

[3] R. Bace. An introduction to intrusion detection assessment for system and network security management, 1999. http://www.infidel.net/Articles/ICSA_Whitepaper.pdf.

[4] M. Baltatu, A. Lioy, F. Maino, and D. Mazzocchi. Security issues in control , management and routing protocols. *Elsevier Computer Networks*, 34:881–894, 2000.

[5] T. Bass, A. Freyre, D. Gruber, and G. Watt. E-mail bombs and countermeasures: Cyber attacks on availability and brand integrity, 1998.

[6] L. Bertolotti and M. C. Carlzarossa. Models of mail server workloads. *Performance Evaluation an International Journal*, (46):65–76, 2001.

[7] E. Biermann, E.Cloete, and L. Venter. A comparison of intrusion detection systems. *Computers and Security*, pages 676–683, 2001.

[8] T. G. Champion and R. S. Durst. Air force intrusion detection system evaluation environment. *RAID Symposium*, 1999.

[9] J. Chazinski. Http/tcp connection and flow characteristics. *Performance Evaluation an international journal*, (42):149–162, 2000.

[10] F. Cohen. 50 ways to defeat your intrusion detection system. http://hackersplayground.org/ papers/50_Ways_to_Defeate_Your_IDS.txt.

[11] E. Cox. *Fuzzy logic for business and industry*. Charles River Media, 2000.

[12] D. Curry and H. Debar. Intrusion detection message exchange format data model and extensible markup language (xml) document type definition, January 2003. http://www.ietf.org/internet-drafts/draftietfidwgidmefxml-10.txt.

[13] H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion detection systems. *Computer Networks vol 31*, pages 802–822, 1999.

[14] S. Dietrich, N. Long, and D. Dittrich. Analyzing distributed denial of service attack tools: The shaft case. *Proceedings of the 14th Systems Administration Conference, LISA XIV*, pages 329–329, December 2000.

[15] D. Dittirich. The dos projects's "trinoo" distributed denial of service attack tool. http://staff.washington.edu/dittrich/misc/trinoo.analysis.txt.

[16] D. Dittrich. The stacheldraht distributed denial of service attack tool. http://staff.washington.edu/dittrich/misc/ stacheldraht.analysis.txt.

[17] G. A. Fink, B. L. Chappell, T. Turner, K. F. ODonoghue, and N. S. W. Center. A metrics-based approach to intrusion detection system evaluation for distributed real-time systems. *International Parallel and Distributed Processing Symposium: IPDPS 2002 Workshops*, April 2002.

[18] J. Haines, R. Lippmann, D. Fried, M. A. Zissman, E. Tran, and S. Boswell. 1999 darpa intrusion detection evaluation: Design and proceedures. Technical report, MIT Lincoln Laboratory, 2001. http://www.ll.mit.edu/IST/ideval/pubs/2001/TR-1062.pdf.

[19] C. Iheagwara and A. Blyth. Evaluation of the performance of id systems in a switched and distributed environment: the realsecure case study. *Computer Networks: The International Journal of Computer and Telecommunications Networking*, 39(2):93–112, June 2002.

[20] K. Kendall. A database of computer attacks for the evaluation of intrusion detection systems. Master's thesis, MIT, June 1999.

[21] J. Korba. Windows nt attacks for the evaluation of intrusion detection systems. Master's thesis, Massachusetts Institute Of Technology, June 2000.

[22] Z. L.A. Fuzzy sets and systems. *Proceedings of Symposium on System Theory, Polytechnic Institute of Brooklyn*, pages 29–37, 1965.

[23] R. Lippmann, D. Fried, I. Graf, J. Haines, K. Kendall, D. McClung, D. Weber, S. Webster, D. Wyschogrod, R. Cunningham, and M. Zissman. Evaluating intrusion detection systems: The 1998 DARPA off-line intrusion detection evaluation. In *Proceedings of the DARPA Information Survivability Conference and Exposition*, Los Alamitos, CA, 2000. IEEE Computer Society Press.

[24] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das. The 1999 darpa off-line intrusion detection evaluation. http://www.ll.mit.edu/IST/ideval/pubs/2000/1999Eval-ComputerNetworks2000.pdf.

[25] J. McHugh. Testing intrusion detection systems: A critique of the 1998 and 1999 darpa intrusion detection system evaluations as performed by lincoln laboratory. *ACM transactions on Information and system Security*, 3(4):262–294, November 2000.

[26] P. Mell, V. Hu, R. Lippmann, J. Haines, and M. Zissman. An overview of issues in testing intrusion detection systems. NIST Interagency Report NIST IR 7007, NIST, http://csrc.nist.gov/publications/nistir/nistir-7007.pdf, June 2003.

[27] S. Northcutt, M. Cooper, M. Fearnow, and K. Frederick. *Intrusion Signatures and Analysis*. New Riders, 2001.

[28] S. Northcutt and J. Novak. *Network Intrusion Detection: An Analyst's Handbook*. New Riders, 2nd edition, 2001.

[29] W. Pedrycz and F. Gomide. *An introduction to fuzzy sets: Analysis and Design*. MIT Press, 1998.

[30] N. Provos. Honeyd - a virtual honeypot daemon (extended abstract). *10th DFN-CERT Workshop, Hamburg, Germany*, February 2003. www.citi.umich.edu/u/provos/ papers/honeyd-eabstract.pdf.

[31] T. H. Ptacek and T. N. Newsham. Insertion, evasion, and denial of service: Eluding network intrusion detection. Technical report, Secure Networks, Inc., Suite 330, 1201 5th Street S.W, Calgary, Alberta, Canada, T2R-0Y6, 1998.

[32] M. J. Ranum. Experiences benchmarking intrusion detection systems. Technical report, NFR Security, Inc., December 2001.

[33] L. M. Rossey., R. K. Cunningham, D. J. Fried, J. C. Rabek, R. P. Lippmann, and J. W.Haines. Lariat: Lincoln adaptable real-time information assurance testbed. *Fourth International Workshop on Recent Advances in Intrusion Detection (RAID2001)*, 2001. http://www.raid-symposium.org/raid2001/program.html.

[34] F. D. Smith, F. H. Campos, K. Jeffay, and D. On. What tcp/ip protocol headers can tell us about the web. *ACM Sigmetrics*, pages 245–256, June 2001.

[35] L. Spitzner. *Honeypots: Tracking Hackers*. Addison-Wesley Pub Co., September 10, 2002.

[36] L.-X. Wang. *A course in fuzzy systems and control*. Prentice Hall, New Jersey, 1997.