Signature-based Multi-Layer Distributed Intrusion Detection System using Mobile Agents

Mueen Uddin[1], Azizah Abdul Rehman[2], Naeem Uddin[2], Jamshed Memon[2], Raed Alsaqour[3], and Suhail Kazi[4]
(Corresponding Author: Mueen Uddin)

Faculty of Computer, Engineering and Technology, Asia Pacific University of Technology & Innovation[1]
Bukit Jalil, 57000, Kuala Lumpur, Malaysia
Department of Information System, Faculty of Computer Science and Information Systems[1,2]
Universiti Technologi Malaysia, 81310, Skudai, Johor, Malaysia.
School of Computer Science, Faculty of Information Science and Technology[3]
University Kebangsaan Malaysia, 43600 Bangi, Selangor, Malaysia
Faculty of Mechanical Engineering, Universiti Teknologi Malaysia, 81310, Skudai, Johor, Malaysia[4]
(Email: mueenmalik9516@gmail.com)

(Received Apr. 20, 2012; revised and accepted July 5, 2012)

Abstract

The Internet and computer networks are exposed to an increasing number of security threats. With new types of attacks appearing continually, developing flexible and adaptive security oriented approaches is a severe challenge. Intrusions detection systems (IDSs) are systems that try to detect attacks as they occur or after the attacks took place. IDSs collect network traffic information from some point on the network or computer system and then use this information to secure the network. In this context, signature-based network intrusion detection techniques are a valuable technology to protect target systems and networks against malicious activities. Signature-based detection is the most extensively used threat detection technique for (IDSs). One of the foremost challenges for signature-based IDSs is how to keep up with large volume of incoming traffic when each packet needs to be compared with every signature in the database. When an IDS cannot keep up with the traffic flood, all it can do is to drop packets, therefore, may miss potential attacks. This paper proposes a new model called Signature-based Multi-Layer IDS using mobile agents, which can detect imminent threats with extremely high success rate by dynamically and automatically creating and using small and efficient multiple databases, and at the same time, provide mechanism to update these small signature databases at regular intervals using mobile agents.

Keywords: Anomaly-based IDS, intrusion detection systems, mobile agent, signature-based IDS, snort

1 Introduction

Nowadays, with the spreading of the Internet and online procedures requesting a secure channel, it has become an inevitable requirement to provide the network security [29]. There are various threat sources including software bugs mostly as the operating systems and software used becomes more functional and larger in size. Intruders who do not have rights to access these data can steal valuable and private information belonging to network users.

Firewalls are hardware or software systems placed in between two or more computer networks to stop the committed attacks, by isolating these networks using the rules and policies determined for them. It is exceedingly clear that firewalls are not enough to secure a network completely because the attacks committed from outside of the networks are stopped whereas inside attacks are not. This is the situation where IDSs are in charge. IDSs are used in order to stop attacks, recover from them with the minimum loss or analyze the security problems so that they are not repeated [3]. Three types of data are used by IDSs. These are network traffic data, system level test data and system status files [2, 25].

The rest of the paper is organized as follows: Section two presents the IDSs and discusses its types; signature-based IDS and anomaly-based IDS. Section three explains the research problem statement. Section four presents the related work. Section five introduces the proposed multi-layer signature-based IDS model. Section six presents the implementation of the proposed model. Section seven describes and discusses the experimental setup and results analysis, and section eight presents the conclusions and possible direction of future work in this area.

2 Intrusion Detection Systems (IDSs)

IDSs are hardware and software systems that monitor events occurred on computers and computer networks in order to analyze security problems. IDS and firewalls have become key components in ensuring the safety of network systems.

Intrusions and invasions inside computer networks are called as “attacks” and these attacks threaten the security of networks by violating privacy, integrity and accessibility mechanisms. Attacks can be originated from users who login to the computer using Internet trying to gain administrator rights and other users who misuse the rights they have. IDSs automate monitoring and analyzing the attacks [3, 4, 25].

The current Internet faces escalating threats form more sophisticated, intelligent and automated malicious codes [17]. In the past, we have seen computer worms spread themselves without any human interaction and launched the most destructive attacks against computer networks [8]. As an example, in January 2003, the SQL Slammer worm, also known as sapphire, was released into the Internet exploiting a weakness into Microsoft SQL servers. In only 10 minutes the worm spread worldwide consuming massive amount of bandwidth and bringing down 5 of the 13 root DNS servers [28]. Amongst worm defensive mechanisms, IDS are the most widely deployed techniques that utilize the self-duplicating repetitive and recurring nature of computer worms to detect the patterns and signatures of theses malicious codes in the network traffic. Consequently IDSs have become an integral part of the security infrastructure of organizations. These systems based on the parameters used for detection, are broadly divided to signature-based and anomaly-based IDSs. IDSs are used in order to stop and prevent attacks and recover from them with the minimum loss or analyze the security problems so that they are not repeated [3].

2.1 IDSs Types

IDSs are classified as either signature-based or anomaly-based [9]. Signature and anomaly-based systems are similar in terms of conceptual operation and composition. The main differences between these methodologies are inherent in the concepts of “attack” and “anomaly”. An attack can be defined as a sequence of operations that put security of a system at risk. An anomaly is an event that is suspicious from the security perspective. Based on this distinction, the main advantages and disadvantages of each IDS type can be pointed out [7, 15, 31].

2.1.1 Signature-based IDS

Signature-based IDSs aim to distinguish events that violate system network policy. Signature-based schemes (also denoted as misuse-based) seek defined patterns, or signatures, within the analyzed data. For this purpose, a signature database with correspondence to known attacks is stated as priority. Signature-based IDS schemes provide worthy detection results for specified, well-known attacks. However, they are not capable of detecting new, unfamiliar intrusions, even if they are built as minimum variants of already known threats. There are different definitions of attack signatures. In this paper, the main discussion will focus on content signatures, which represent a string of characters that appear in the payload of attack packets. No knowledge of normal traffic is required, but a signature database is needed for this category of detection systems. For worm detection, this type of system does not care how a worm finds the target, how it propagates itself or what transmission scheme it uses. The system takes a look at the payload and identify whether or not it contain a worm.

One of the main challenges faced by signature-based IDS is that every signature requires an entry in the database, and so a complete database might contain hundreds or even thousands of entries. Each packet is to be compared with all the entries in the database. This can be highly resource- consuming and doing so will slow down the throughput and making the IDS vulnerable to Distributed Denial of Service (DDoS) attacks. Some of the IDS evasion tools use this vulnerability and flood the signature signature-based IDSs with too many packets to the point that the IDS cannot keep up with the traffic, thus making the IDS time out and drop packets and as a result, possibly miss attacks [6]. Further, this type of IDS is still vulnerable against unknown attacks as it relies on the signatures currently in the database to detect attacks.

2.1.1 Anomaly-based IDS

Anomaly-based IDSs attempt to analyze abnormal activities and flag these activities as attacks. Anomaly detectors detect behaviors on a computer or computer network that are not normal [7]. According to this approach, behaviors deviating from behaviors assumed as normal are thought to be attacks & anomaly detectors compute the deviation in order to detect these attacks [32]. Anomaly detectors construct profiles of users, servers and network connections using their normal behaviors. These profiles are produced using the data that is accepted as normal. After the profile construction, detectors monitor new event data, compare the new data with obtained profile and try to detect deviations. These deviations from normal behaviors are flagged as attacks [3, 18, 31].

Anomaly-based IDSs detect abnormal behaviors and generate alarms based on the abnormal patterns in network traffic or application behaviors. Typical anomalous behaviors captured include [32]:

(1) Misuse of network protocols such as overlapped IP fragments and running a standard protocol on a stealthy port.

(2) Uncharacteristic traffic patterns, such as more User Datagram Protocol (UDP) packets compared to Transmission Control Protocol (TCP).

(3) Suspicious patterns in application payload.

The main benefit of anomaly-based detection techniques is their potential to detect previously unseen intrusion events. However, and despite the likely inaccuracy in formal signature specifications, the rate of false positives (or FP), events erroneously classified as attacks; in anomaly-based IDSs is usually higher than in signature-based ones [10, 14].

The other problems with anomaly-based IDSs are: defining what normal network behavior is, deciding the threshold to trigger the alarm, and preventing false alarms. The users of the network systems are normally human, and people are hard to predict. If the normal model is not defined carefully, there will be lots of false alarms and detection system suffers from degraded performance.

3 Problem Statement

The number of security breaking attempts originated inside computer networks is increasing steadily. Attacks made in this way are assuredly done by the authorized users of the system and cannot be immediately located. Signature-based IDS (SIDS); the most widely used IDS technique provides solutions to detect these attacks. One of the major problems with this technique is to espouse large volume of incoming traffic when each packet needs to be compared with every signature in the database. When IDS does not keep up with the traffic flood, all it can do is to drop packets; therefore, may miss potential attack. There are a number of challenges being faced by SIDS, such as:

(1) Signature creation.

(2) Part of signature to be stored in a database.

(3) Maintenance of large complementary signature database.

(4) Updating small signature databases.

(5) Transfer signatures from a large database to small databases.

(6) Avoid traffic flooding on the network.

4 Related Work

SIDSs require that their databases need to be updated regularly at different time intervals so as to detect the imminent threads generated on the network. This process is a quite time consuming and requires a quick underlying system to update the database. Two-layer signature-based model was proposed to address signature-based detections with unequal databases. But this model does not have any mechanism for adding, removing or updating signatures in the large signature-based database [13, 24]. If the signature database is not updated timely, then new threats will not be detected using this model.

P. Wheeler in [30], in his thesis, divides all efforts to address this issue in the following categories:

(1) Improving content matching algorithms- signature matching is one of the most computationally intensive tasks of IDS. Methods such as Landmark Segment Method (LSM) [1] and modified Aho-Corasick algorithm [23] have been proposed to reduce the execution time of signature matching.

(2) Parallel processing – parallelism can be achieved at different levels: a) at the node level by running a subset of rules on different nodes to keep the database size small; b) at the component level by running different components of the IDS, such as preprocessors and rules engine, on different nodes to distribute the load; and c) at sub-component level by dividing a particular component of the IDS into *n* parallel instances, with each instance responsible for *1/n* of the incoming packets.

While all the above efforts are geared toward improving the IDS throughput, they either require significant modifications to existing IDS by including new content matching algorithms, or require dedicated hardware resources to achieve acceleration, or need to coordinate tasks and aggregate output from parallel processing units.

The proposed model is extremely straightforward, less resource demanding, yet can be used in combination with the above schemes. This model updates the large complementary signature-based database regularly in different time intervals using mobile agents. Mobile agents automatically add, remove and update large signature database without consuming time and resources. Hence, the proposed model does not add any additional overhead over the system performance and cost.

5 Proposed Model

There are many models proposed in this scenario to avoid these problems but still there are couples of deficiencies needs to be addressed. From the above discussions it is particularly obvious that anomaly-based IDS has enormous risk in generating high false positives, while in contrast, signature-based IDS are less susceptible to generate false alarms as the decision to generate an alarm is based on the signatures detected and does not require any knowledge of the normal traffic. However, signature-based IDS have their own limitations.

(1) Every signature requires an entry in the database, and each packet needs to be compared with all the entries in the database. This may potentially slow down the throughput of the systems.

(2) Signature-based IDS is vulnerable against newly emerging attacks.

For the second issue, there are already proposals and systems such as in [21] use a complementary payload-based anomaly detection system to detect new attacks, create signatures, and provide the new signatures to the signature-based IDS for the detection of the new threats. The research highlighted in this paper addresses the first limitation of signature-based IDS. We proposed a new

Multi-layer model for signature-based IDS with mobile agents being used to transfer signatures from large complementary database to small signature based databases.

The proposed Multi- layer signature-based IDS model consists of multiple IDSs deployed in different layers, and each is contained with small signature database. There is also a large complementary signature database containing all the entries of signatures detected during the training period. We present a straightforward and automatic way to decide the set of rules and signatures to be deployed in the different IDS and continuously update the rules based on the usage pattern. Mobile Agents are used to perform the updating process of small signature databases. Using this model, we expect to optimize the detection rate of frequent threats to the network as well as providing means to detect uncommon attacks to the network.

5.1 Multi-Layer Signature-based IDS Model

The work proposed in this paper is motivated by the fact that it is easy and a less time consuming to update small signature databases compare to large complementary signature database continuously from time to time. By doing this, we can also improve the throughput of signature-based IDS, since a packet needs to be matched with less number of signatures in small signature database compare to one with enormous number of signatures. This idea is not new, and in fact, it has been suggested by an installation note to system administrators of *Snort* [8]. To turn this idea into an effective one, we need to address three key issues.

(1) How to decide whether a given signature is likely to be helpful for possible attacks? We need systematic guidelines whether to remove a given signature. Currently, configuring the signature database is still a manual trial and error process of disabling some signatures and adding them back in after missing some attacks.

(2) What to do if we make the wrong choice and classify a useful signature as unlikely and remove it from the database? How to protect the network in this case?

(3) What to do once a new service or protocol is added to the network? We cannot entirely rely on the administrators to remember to manually add the corresponding signatures to the database. This process is labor-intensive and can be error prone.

This paper addresses all of the above issues by proposing a new model based on small signature databases called multi-layer signature-based model. As shown in Figure 1, the proposed model consists of multiple smaller databases containing the most frequent attack signatures on the IDS (SSD) and a bigger complementary signature data-base containing thousands of signatures used to update smaller databases from time to time using mobile agents (CSD). By distributing the signature database between

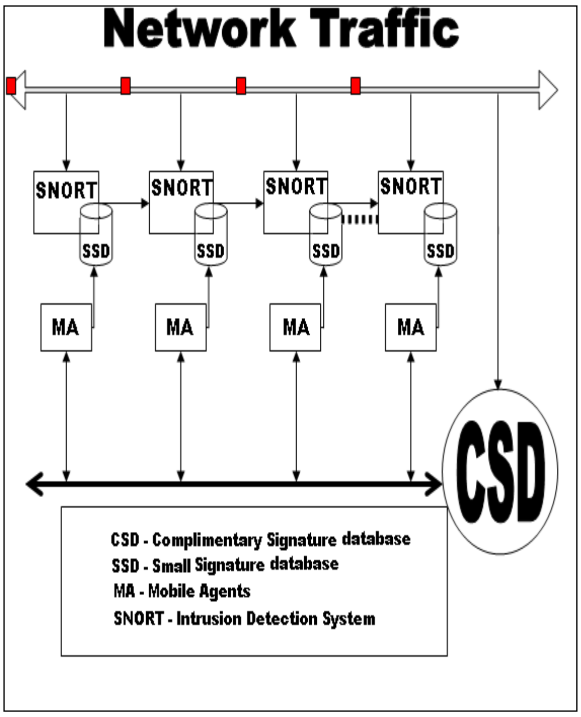
Network Traffic
SNORT
SNORT
SNORT
SNORT
SSD
SSD
SSD
SSD
MA
MA
MA
MA
CSD
CSD - Complimentary Signature database
SSD - Small Signature database
MA - Mobile Agents
SNORT - Intrusion Detection System

Figure1: Multi-layer signature-based IDS using mobile agents

multiple nodes, it is essential to consider two main aspects of the proposed model.

(1) The signature database size on the nodes should not always be equal; it depends on the algorithm to update the small signature databases and also to keep the database size varying all the time.

(2) The size of the databases on nodes dynamically changes to optimize the detection rate on more likely imminent threats.

6 Implementation of the Proposed Model

The implementation of the proposed model comprises of two main phases:

(1) **Training Period:** during the training period the system gathers information about the current threats to the network.

(2) **Running Phase:** during this phase the signature databases are updated based on their usage patterns.

The duration of the training period depends on the network conditions and can vary from few hours to several days. The longer the training period, it is more likely to have a better baseline of the most frequent attacks on the network.

Once the training period is done, we look at the alert logs in the database. Depending on the criteria and parameters that can be defined by the user, we identify the most frequent attacks based on the 1) minimum number of occurrences of a signature, 2) the age of the alert in the database, 3) and the maximum number of signatures that we would like to keep in the all the IDS.

After identifying the most frequent attacks, we create a signature database for all IDS containing signatures for the most frequent attacks and create a complementary signature database containing the remaining signatures recognizable by the system. After the initial training period, mobile agents are used to update the small signature database if some new signature is updated in large complimentary database.

This work is done periodically at a certain time intervals defined in the implementation of mobile agents. It is extremely much necessary to emphasize that the proposed model will not miss attacks frequently as compare to previous models proposed. This is because small signature databases are updated constantly with most updated and latest signatures using mobile agents. The complementary IDS help to detect less frequent attacks that may arise by comparing the signatures with packets travelling on the network.

6.1 Snort

Martin Roesch, a software engineer working on the computer security topics, has developed *Snort* in 1990 in order to detect attacks targeting his home network. *Snort* is a fast, most commonly used network intrusion detection system that uses signature-based and open-source mechanism and runs over IP networks analyzing real-time traffic for detection of misuses or intrusions [21]. It depends on a template-matching scheme and makes content analysis. It has the ability to flag alerts depending on predefined misuse rules and saves packets in *tcpdump* files or in ordinary text files. It produces alarms using misuse rules defined previously. *Tcpdump* is a software program that captures network packets from computer networks and stores them in *tcpdump*-formatted files. *Snort* is preferred to be used in academic research projects as it is an open-source tool and for this reason we have also chosen Snort as the SIDS in our research work. *Snort* is an open-source project and it has a language to define new rules and its architecture makes it possible to integrate new functionalities at the time of compilation [16, 21, 22, 27].

Snort consists of the following four components [20, 27]:

(1) **Packet capture/decode Engine:** *Snort*'s packet-capturing engine uses the *libpcap* packet-capturing library written in Lawrence Berkeley National Laboratories. Captured packets are processed by decoding engine and decoded packets become compliant with the network-level protocols. Resultant packets are re-decoded for upper-level protocols that are TCP and UDP.

(2) **Preprocess or Plug-ins:** Packets are passed through a number of preprocessors. This step aims investigating and processing packets before they are passed tothe detection engine. Every other preprocessor examine the packets for a different attribute and make a decision to pass the packet to the detection engine without making any modification, modifying and then passing it to detection engine or not passing and generating an alert for the packet.

(3) **Detection Engine:** Detection engine tests packets for a number of attributes stated in *Snort* rules definition file. Detection plug-ins provides extra detection functions.

(4) **Output plug-ins:** This plug-ins accepts alarms generated from detection engine, preprocessors or decoding engine.

For prototype implementation, we chose *Snort* [11] as the signature-based IDS platform. We configured *Snort*'s output plug-ins to log the alerts in the MySQL database for easy access and queries. *Snort* stores its signatures in rule files referenced in the *Snort*'s configuration file.

Further, we developed an algorithm using mobile agents as an engine to create the most frequent signature databases for the multiple IDS as well as generating the complementary signature database for the secondary IDS. The same algorithm runs on all IDS systems for certain intervals to keep the primary signature database constantly updated. Removing signatures that are no longer occurring frequently can do this, and also adding any signature detected as a frequent alert by the secondary IDS.

The pseudo code for the proposed algorithm is illustrated in Algorithm 1 below. The algorithm accepts three input parameters: *MinFreq* specifying the minimum number of attack occurrences to be considered as frequent, *ValidTime* setting the time beyond which the attacks seen are considered as valid and threatening, and *MaxNum* representing the maximum number of the signatures acceptable in all IDSs.

Algorithm 1: Generate and update signature databases

```
1: Begin
2: N = 0  # number of current signatures
3: Query the MySQL database to retrieve the set of
    signatures detected, S.
4: for every signature f in S do
5:   Freq = number of occurrences of f
6:   LTime = last detection time of f
7:   if N <= MaxNum and Freq >= MinFreq
              and Ltime >= ValidTime
     then
8:     Remove the signature from the secondary
       database
9:     Add the signature in multiple IDS
10:    N = N+1
11:  end if
12: end for
13: Restart multiple and complementary IDS
14: End
```

7 Experimental Setup

This section of the paper describes and discusses in detail the experimental setup made for performing experiments and then and analyze the results generated after performing different experiments at different time intervals with different parameters.

The experiments were performed choosing two different hardware platforms to simulate attacks and run IDS, one more powerful than the other. The objective was to simulate DDoS attacks on IDSs by running IDS on the slower machine and attacking tools on the faster machine. The aggregate computational and networking resources of attackers usually overwhelm the resources on the IDS machine. In this case, we would like to evaluate the effects of having multiple smaller signature databases and how effectively it helps in improving the throughput and decreasing packet loss rate. A small packet loss rate directly leads to small possibility to miss real attacks that might be hidden in false positive storms.

The detailed hardware and software configurations of systems used for performing experiments are as follows.

Attacking System:

- 512MB memory / Windows XP
- CPU: AMD Geode NX running at 1.4GHz
- 10/100Mbps NIC

IDS System:

- 256MB memory / Windows XP
- CPU: Pentium III running at 500 MHz
- 10 Mbps NIC

7.1 Attacking Tools

During the training period, following tools were used: *IDSwakeup* [12], *Stick* [9], *Sneeze* [5] and *Nikto* [27] to trigger alerts by the IDS and create a baseline of the most frequent attacks on the network. Here, we briefly describe each tool.

IDSwakeup is designed to test the functionality of the IDS by generating some common attack signatures to trigger IDS alarms. *Stick* is another tool fed with *Snort* configuration files to reverse engineer threats and create packets with signatures in the same way as those detected by *Snort* as attacks. *Stick* can be used to test the functionality of IDS as well as be deployed as a stress tester. It can be also used as an IDS evasion tool by generating a lot of traffic, and camouflaging the real attacks in a flood of false positives. *Sneeze* is a Perl-based tool highly similar to *Stick* in terms of functionalities. It distinguishes itself from Stick by the fact that it can accept *Snort*'s rules at runtime and dynamically generates attack packets, whereas *Stick* needs to be configured with *Snort*'s rules at compilation time. *Nikto* focuses on web application attacks by scanning and testing web servers and their associated CGI scripts for thousands of potential vulnerabilities [20].

7.2 Analysis of Results

For performing experiment, we set the value of *MinFreq* to 1, i.e., we considered any attack that appeared at least once as a frequent attack. In addition, we set *ValidTime* to be a negative number so that all the threats detected in the training period are to be included in the database. Our program scanned all the rule files of *Snort* and created a new rule file called "signature.rule" containing the most frequently signatures detected during the training period to be referenced by the *snort.conf* file as the only signature rule file. In addition, our program created complementary rule files taking out the most frequently used signatures for the secondary IDS. *Snort* was restarted on both systems pointing to the new signature files.

To test the performance of all IDS, we conducted our experiment using two tests in two different scenarios. In the first scenario, we manually enabled 3211 signatures and attacked the network-using *Sneeze* [27]. In the second scenario, we let our algorithm do its job by enabling only the most frequent signatures (in this case 33). Table 2, 3 and Figures 2, 3 shown the results of our tests in regards of the effects on packet drop rate.

Our test results clearly show the difference in the performance of the IDS using small signature database comparing to just enabling all signatures. In our environment with our specific configuration, by reducing the size of the database almost 97 times (3211/33), on average, we were able to decrease the percentage of the dropped packets by 6.77 times. This is a significant improvement reducing the possibility of a real worm attack sneaking in the midst of dropped packets by the IDS while ensuring all imminent threats can be detected by the IDS.

The results of the signatures detected during the training period are shown in Table 1.

8 Conclusions

This paper has focused on the efficiency and performance of the new IDS: called signature-based multi-layer IDS using mobile agents. We had discussed the development of a new signature based ID using mobile agents. The proposed system uses mobile agents to transfer rule-based signatures from large complementary database to small signature database and then regularly update those databases with new signatures detected.

It then describes an experimental setup used to perform the analysis for the verification and validation of proposed SIDS, so that it can be implemented in a large network environment. The results clearly indicate that proposed IDS model performs much better than normal systems with only one database for string signatures. Our experiments proved a significant decrease in the packet drop rate, and as a result, a significant improvement in detecting threats to the network. The paper also highlights the foundations of IDS s and their advantages over other technologies, together with their general operational architecture, and provides a classification for them according to the type of processing related to the behavioral model for the target system. Further, the proposed model can be improved by providing a more comprehensive and automated system that can distribute, add and remove the signatures across databases of multiple IDS systems based on the frequency of their appearance and their level of threat to the network.

Finally, we believe more research needs to be done to determine the criteria to choose the optimal training period for a network.

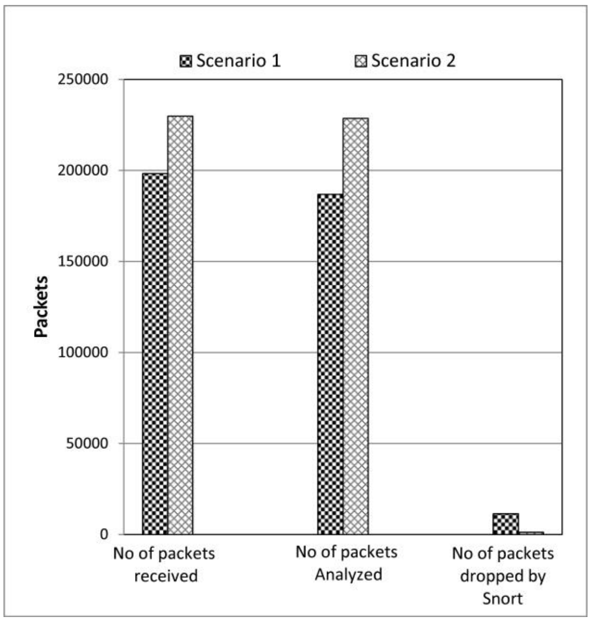

Figure 2: Performance measurement (Test 1)

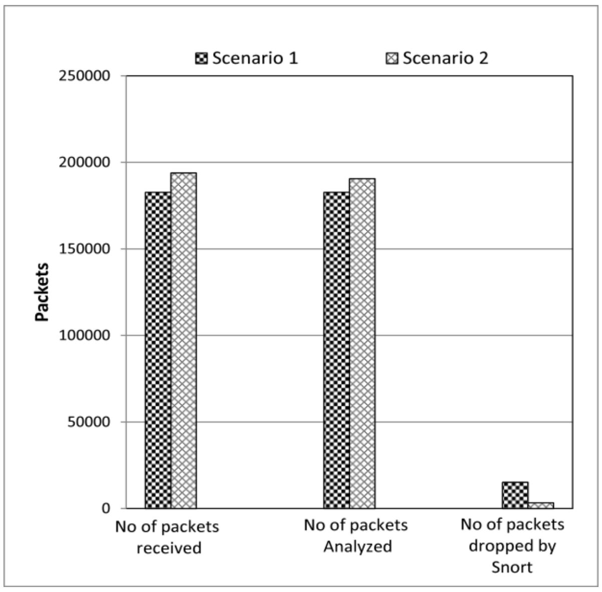

Figure 3: Performance measurement (Test 2)

Table 1: Signatures detected during training period

Signature name	Number of occurrences
(spp_frag2) TTL Limit Exceeded (reassemble) detection	2
(portscan) UDP Portscan	3
ICMP Destination Unreachable Communication	3
Administratively Prohibited	3
(spp_frag2) Teardrop attack	4
MISC gopher proxy	4
(portscan) TCP Portscan	5
WEB-MISC Compaq Insight directory traversal	11
DDOS tfn2k icmp possible communication	14
ICMP Large ICMP Packet	14
WEB-CGI search.cgi access	17
WEB-MISC http directory traversal	18
TELNET SGI telnetd format bug	19
(http_inspect) DOUBLE DECODING ATTACK	20
FINGER query	21
BACKDOOR Q access	26
BACKDOOR SIGNATURE – Q ICMP	28
DDOS mstream client to handler	29
BAD-TRAFFIC udp port 0 traffic	35
SNMP request udp	40
DOS arkiea backup	48
(http_inspect) WEBROOT DIRECTORY TRAVERSAL	49
BAD-TRAFFIC tcp port 0 traffic	84
(http_inspect) OVERSIZE REQUEST-URI DIRECTORY	96
NETBIOS RFParalyze Attempt	105
(spp_rpc_decode) Incomplete RPC segment	129
FTP command overflow attempt	163
WEB-CGI Allaire Pro Web Shell attempt	981
WEB-CGI Armada Style Master Index directory traversal	998
WEB-IIS index server file source code attempt	1525
BAD-TRAFFIC same SRC/DST	1754
ICMP Echo Reply	3409
BAD-TRAFFIC loopback traffic	21886
(snort_decoder): Invalid UDP header, length field	<86006

Table 2: Performance measurements (Test 1)

	Scenario 1: All rules were enabled	**Scenario 2: Most frequent rules were enabled**
No of packets received	198375	229873
No of packets analyzed	186923	228654
No of packets dropped by *Snort*	11452	1219

Table 3: Performance measurements (Test 2)

	Scenario 1: All rules were enabled	**Scenario 2: Most frequent rules were enabled**
No of packets received	182688	193890
No of packets analyzed	182688	190581
No of packets dropped by Snort	15155	3309
Percentage of packets dropped by *Snort*	8.296%	1.707%

Acknowledgments

This research was supported in part by Ministry of Higher Education (MOHE), University Kebangsaan Malaysia (UKM) Malaysia, under Fundamental Research Grant Scheme (FRGS). Grant: FRGS/1/2012/SG05/UKM/02/7

References

[1] M. Aldwairi, T. Conte, and P. Franzon, "Configurable string matching hardware for speeding up intrusion detection," *ACM SIGARCH Computer Architecture News*, vol. 33, no. 1, pp. 99-107, 2005.

[2] R. Bace, *An Introduction to Intrusion Detection & Assessment*, Technical White Paper, ICSA, 1999.

[3] R. G. Bace, *Intrusion detection: Sams*, 2000.

[4] R. Base and P. Mell, *Intrusion Detection Systems*, National Institute of Standards and Technology (NIST), Special Publication, vol. 51, pp. 800-831, 2001.

[5] D. Bailey, Sneeze. (http://archives.neohapsis.com/archives/snort/2001-08/0180.html)

[6] J. Beale, A. R. Baker, B. Caswell, and M. Poor, Snort 2.1 Intrusion Detection: Syngress Media Inc, 2004.

[7] T. Bhaskar, B. Narasimha Kamath, and S.D. Moitra, "A hybrid model for network security systems: Integrating intrusion detection system with survivability," *International Journal of Network Security*, vol. 7, no. 2, pp. 249-260, 2008.

[8] S. Chen and Y. Tang, "Slowing down internet worms," in *Proceedings of 24th International Conference on Distributed Computing Systems*, pp. 312-319, 2004.

[9] G. Coretez, *Fun with Packets: Designing a Stick*, Endeavor Systems, Inc, 2002.

[10] D. Dagon, X. Qin, G. Gu, W. Lee, J. Grizzard, J. Levine, and H. Owen, "Honeystat: Local worm detection using honeypots," *RAID 2004*, LNCS 3224, pp. 39-58, 2004.

[11] S. Dharmapurikar and J. Lockwood, "Fast and scalable pattern matching for content filtering," in *Symposium on Architecture for networking and communications systems*, pp. 183-192, 2005.

[12] IDSwakeup. (http://www.hsc.fr/ressources/outils/idswakeup/index.html.en)

[13] P. Garcia-Teodoro, J. Diaz-Verdejo, G. Macia-Fernandez, and E. Vazquez, "Anomaly-based network intrusion detection: Techniques, systems and challenges," *Computers and Security*, vol. 28, no. 1-2, pp. 18-28, 2009.

[14] F. Gong, *Deciphering Detection Techniques: Part II Anomaly-based Intrusion Detection*, White Paper, McAfee Security, 2003.

[15] P. Kabiri and A. A. Ghorbani, "Research on intrusion detection and response: A survey," *International Journal of Network Security*, vol. 1, no. 2, pp. 84-102, 2005.

[16] K. Kendall, *A Database of Computer Attacks for the Evaluation of Intrusion Detection Systems*, S. M. Thesis, Massachusetts Institute of Technology, 1999.

[17] D. Moore, C. Shannon, G. M. Voelker, and S. Savage, "Internet quarantine: Requirements for containing self-propagating code," in *Twenty-Second Annual Joint Conference of the IEEE Computer and Communications*, pp. 1901-1910, 2003.

[18] B. Mukherjee, L. T. Heberlein, and K. N. Levitt, "Network intrusion detection," *IEEE Network*, vol. 8, no. 3, pp. 26-41, 1994.

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

[20] R. U. Rehman, *Intrusion Detection Systems with Snort*, Upper Saddle River, New Jersey: Publishing as Prentice Hall PTR, 2003.

[21] M. Roesch, "Snort-lightweight intrusion detection for networks," in *Proceedings of LISA '99: 13th Systems Administration Conference*, pp. 229-238, 1999.

[22] R. Russell, *Snort Intrusion Detection 2.0*, Syngress Pub., 2003.

[23] B. C. S. Ryu and J Kim, "design of packet detection system for high-speed network environment," in *The 6th International Conference Advanced Communication Technology*, pp. 496-498, 2004.

[24] M. Salour and X. Su, "Dynamic two-layer signature-based ids with unequal databases," in *Fourth International Conference on Information Technology*, pp. 77-82, 2007.

[25] K. Scarfone and P. Mell, *Guide to Intrusion Detection and Prevention Systems (Idps)*, NIST Special Publication, vol. 8, pp. 800-894, 2007.

[26] Snort Users Manual 2.6.1. (www.snort.org/assets/166/snort_manual.pdf)

[27] C. Sullo and D. Lodge, Nikto2. (http://cirt.net/nikto2)

[28] M. Uddin, K. Khowaja, and A.A. Rehman, "Dynamic multi-layer signature based intrusion detection system using mobile agents," *International Journal of Network Security & Its Applications (IJNSA)*, vol. 2, no. 4, pp. 129-141, 2010.

[29] C. Wong, C. Wang, D. Song, S. Bielski, and G. R. Ganger, "Dynamic quarantine of Internet worms," in *Proceedings of the 2004 International Conference on Dependable Systems and Networks*, pp. 73-82, 2004.

[30] P. S. Wheeler, *Techniques for Improving the Performance of Signature-based Network Intrusion Detection Systems*, University of California, 2006.

[31] J. Zhang and M. Zulkernine, "Anomaly based network intrusion detection with unsupervised outlier detection," in *IEEE International Conference on Communications*, pp. 2388-2393, 2006.

[32] Z. Zhang, H. Shen, and Y. Sang, "An observation-centric analysis on the modeling of anomaly-based intrusion detection," *International Journal of Network Security*, vol. 4, no. 3, pp. 292-305, 2007.

Mueen Uddin is a Senior Lecturer at Faculty of Computing and Technology, Asia Pacific University of Technology & Innovation. He completed his PhD in Information Systems from UTM Malaysia in 2012, BS & MS in Computer Science from Isra University Hyderabad Pakistan in 2008 with specialty in Information networks. His research interests include Green IT, energy efficient data centers and Virtualization technologies, digital content protection and deep packet inspection, intrusion detection and prevention systems, MANET routing protocols and their analysis. Dr. Mueen has over 22 international Journal publications and many Conference papers.

Azizah Abdul Rahman is an Associate Professor at Faculty of Computer Science and Information Systems, University Teknologi Malaysia. She completed her B.Sc and M.Sc from USA, and PhD in information systems from University Technology Malaysia. Her research interests include designing and implementing techniques for information systems in an organizational perspective, knowledge management, designing networking systems in reconfigurable hardware and software, and implementing security protocols needed for E-businesses. Dr Azizah has more than 40 publications in international journals and Conferences in the field of Green IT, Knowledge management, information systems design and implementation.

Jamshed Memon: Mr. Memon is a PhD fellow at Universiti Teknologi Malaysia; his research interests include network security, Intrusion detection and prevention systems, and Green IT, virtualization and eco-friendly technologies. He has done M.Sc from UK in information security.

Naeem Uddin: Mr. Naeem is a Master Student in Universiti Teknologi Malaysia; his research interests include network security, software design and implementation, system modeling programming languages and soft system implementations. He has done B.Sc from Isra University Pakistan.

Raed Alsaqour is an Assistant Professor in the School of Computer Science, Faculty of Information Science and Technology, Universiti Kebangsaan Malaysia, Malaysia. He received his B.Sc degree in computer science from Mu'tah University, Jordan, in 1997. M.Sc degree in distributed system from University Putra Malaysia, Malaysia, in 2003 and his PhD degree in wireless communication system from Universiti Kebangsaan Malaysia, Malaysia, in 2008. His research interests include wireless network, ad hoc network, vehicular network, routing protocols, simulation, network performance evaluation, and Green IT. He also has a keen interest in computational intelligence algorithms (fuzzy logic and genetic) applications and security issues (intrusion detection and prevention) over network.

Suhail Kazi is a Senior Lecturer at Faculty of Mechanical Engineering, University Teknologi Malaysia. He obtained his B.Sc in Mechanical Engineering from Mehran University of Engineering and Technology, Jamshoro, Pakistan in 1999. He then pursed his M.Sc in Computer Science from Isra University, Hyderabad, Pakistan in 2005. He completed his PhD at Universiti Teknologi Malaysia in 2011. His major field of Study is active tremor control and mechatronics. His research interests include intelligent active force control, Energy harvesting, Vibration harvesting, system modeling and simulation, vibration control and mechatronics.