ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

Implementation of IDS in Windows

Sunil Dutt, Mahesh Singh

M.Tech (pursuing), Dept. of CSE, AITM, Palwal, Haryana, India

Associate Professor, Dept. of CSE, AITM, Palwal, Haryana, India

ABSTRACT: IDS is new terminology which provides network security. In this paper we describe about the Host Based Intrusion Detection System using SNORT IDS. The SNORT sensor is listening to the flow of data within the segments, without impeding network performance. SNORT informs of an attack to the administrator. It can be connected with a secure point Firewall & VPN Server to give a high-end security system.

.

KEYWORDS: Intrusion Detection System, Virtual Private Network, Firewall, Network Security.

I. INTRODUCTION

An intrusion detection system (IDS) monitors electronic network congestion and monitors for wary activity and warning the system or network administrator. In some events the IDS may also react to unnatural or malicious traffic by taking legal action such as barring the user or root IP address from bringing at the electronic network. IDS come in a kind of “looks” and approach the goal of finding wary traffic in different ways. There are electronic network based (NIDS) and host based (HIDS) intrusion detection systems. There are IDS that search out based on looking for particular signatures of known threats- alike to the route antivirus software typically search out and defends against malware- and there are IDS that search out based on comparing traffic forms against a baseline and looking for discrepancy. There are IDS that simply monitor and alert and there are IDS that perform an action or actions in response to a detected threat. We’ll cover each of these briefly.

NIDS: Network Intrusion Detection Systems are placed at a essential stage or stages within the electronic network to supervise traffic to and from all devices on the electronic network. Ideally you would scan all inward and outward traffic, however doing so might create a bottleneck that would spoil the overall speed of the network.

HIDS: Host Intrusion Detection Systems are run on individual hosts or devices on the electronic network. A HIDS monitors the inward and outward packets from the device only and will warn the user or administrator of wary activity is founded.

Signature Based: A signature based IDS will monitor packets on the electronic network and equate them versus a database of signatures or attributes from known malicious risks. This is similar to the way most antivirus software finds out malware. The issue is that there will be a lag between a new risk being discovered in the wild and the signature for noticing that risk being applied to your IDS. During that lag time your IDS would be unable to find out the new threat.

Anomaly Based: An IDS which is anomaly based will monitor electronic network congestion and equate it against an existing baseline. The baseline will distinguish what is “normal” for that network- what sort of bandwidth is generally used, what rules are used, what ports and devices generally attach to each other- and alert the administrator or user when congestion is founded which is anomalous, or significantly different, than the baseline.

Web Attack: According to Gartner’s research report, in more than 300 websites they have analyzed, 97% are with security weakness. And 75% attacks are on application level. In the OWASP 2007 report, there are 10 major security weaknesses in the Web attack procedures, and here are those related to program coding.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

Cross Site Scripting: By using unchecked input attributes, cause of hackers could embed executable malicious code. It may be embedded with Trojan or aimed to a phishing site.

Injection Flaw: By the weakness of unchecked input attributes, hackers could modify unauthorized access command or database. On some special environment, then even may obtain system administrator perquisite. It is often used Drive-by Download method.

Malicious File Execution: It is an imperfection from web system design and may cause remote malicious codes to be executed remote malicious code. It usually happens on PHP based Webpages.

Insecure Direct Object Reference: Because of the defect in the system function design of file reading, hackers could arbitrarily access files under any paths. This threat may cause the exposure of major system files. For example, the password file would lead to accounts and passwords hacked, and hackers would get the system administrator perquisite.

Cross Site Request Forgery: It is based on the Cross Site Scripting extends attack method. Hackers inject the malicious code, causing unauthorized code to automatically get executed on an authorization user id. Why is the detection of web attack so difficulty ? When the weaknesses exist in web applications, attacks from hackers are normal http requests, and they could penetrate firewalls and evade Intrusion Detection System without making alerts. Besides attack methods today usually use multi-level or multi-type encoding to evades Intrusion Detection Systems.

Snort: Snort is a lightweight Intrusion Detection System developed by Marty Roesch in 1998. It is open source and has good processing effect, and is also the most popular Intrusion Detection System in Open platform. The most importance issue in the Signature-based Intrusion Detection System fields is the requirement of frequently-updated signature database. Snort has a customized rule set language. Users could construct the signature database they demand once familiar with the language. The system architecture is illustrated in Figure 1.

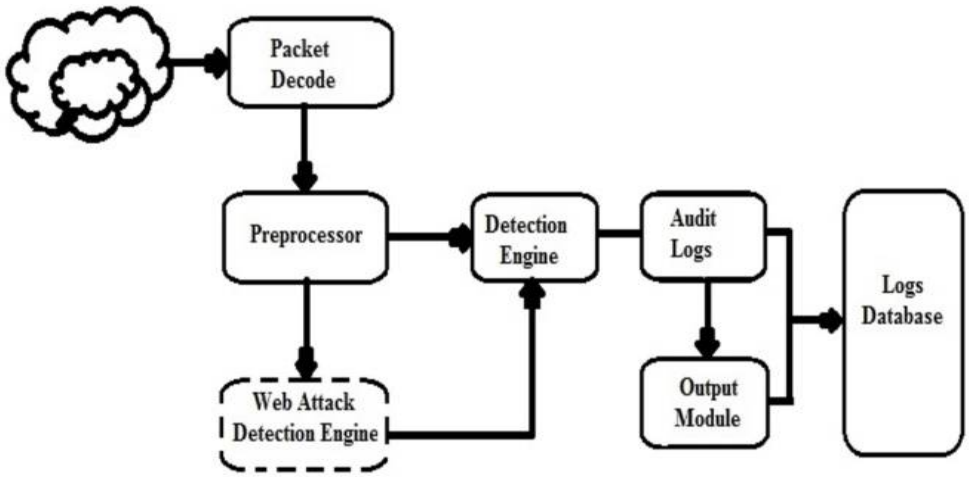

Figure 1. System architecture

Packet Decode: Snort uses PCAP library to capture the packets transferred/received in the LAN in which contain the captured time, packet length, and link type (for example: Ethernet, FDDI etc.). It also creates a pointer pointing each packer for efficient analysis. With the inline mode, it has additional function of firewall such as packet transfer, packet modification, rejecting specific packets or dropping them.

Pre-processor: After packets are captured, Snort will transfer them to Pre-processor for packet repacking and

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

normalizing based on the format of each protocol. The pre-processor also analyses statistics of the network traffic and detects unregulated attacks such as deny of service and worms.

Detection engine: The detection engine is the core of Intrusion Detection System. Users could download the signature database from the official website. With suitable settings detection of network attacks can be effective. If the system the packet captured matches any signature pattern, the system will alert an attack alarm to audit logs.

Audit Logs: When the system determine attacked, it will generate logs and make an alert containing related information of the attack for the administrator to remove the attack. Snort has 2 mechanisms for alerts, Event Queue and Thresholds. When an attack happens and violates multiple rules, it would generate alerts in the order of the priorities defined for each rule in advance. Therefore the administrator gets to exclude less urgent items with Event Queue. The Threshold mechanism is to generate only one alert when a huge amount of the same attack behaviors happen in a short time. It is helpful when Deny of Service or the worm attack happens, which often causes lots of alerts, to decrease the same alerts and to simplify the complexity of tracking the source of attacks.

Output Module : Snort supports various output modules for users to choice from under different environments and objects.

a) Default Logging
b) SNMP traps
c) XML Logging
d) Syslog
e) SMB Alerting
f) PCAP logging
g) SnortDb
h) Unified Log

ModSecurity: The open source Web Application Firewall "ModSecurity" is developed by Ivan Ristic. It adopts rule-based detection engine which is more flexible than regular expression detection engine. Therefore ModSecurity is effective on detecting more kinds of Web attacks, such as SQL Injection, Cross Site Scripting, Insecure Direct Object Reference and Cross Site request Forgery.

II. RELATED WORK

Classification of Intrusion Detection Systems : Intrusions can be divided into 6 main types [11]

1. Attempted break-ins, which are detected by atypical behaviour profiles or violations of security constraints.
2. Masquerade attacks, which are detected by atypical behaviour profiles or violations of security constraints.
3. Penetration of the security control system, which are detected by monitoring for specific patterns of activity.
4. Leakage, which is detected by atypical use of system resources.
5. Denial of service, which is detected by atypical use of system resources.
6. Malicious use, which is detected by atypical behaviour profiles, violations of security constraints, or use of special privileges.

However, we can divide the techniques of intrusion detection into two main types.

Anomaly Detection : Anomaly detection techniques assume that all intrusive activities are necessarily anomalous. This means that if we could establish a "normal activity profile" for a system, we could, in theory, flag all system states varying from the established profile by statistically significant amounts as intrusion attempts. However, if we consider that the set of intrusive activities only intersects the set of anomalous activities instead of being exactly the same, we find a couple of interesting possibilities: (1) Anomalous activities that are not intrusive are flagged as

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

intrusive. (2) Intrusive activities that are not anomalous result in false negatives (events are not flagged intrusive, though they actually are). This is a dangerous problem, and is far more serious than the problem of false positives.

The main issues in anomaly detection systems thus become the selection of threshold levels so that neither of the above 2 problems is unreasonably magnified, and the selection of features to monitor. Anomaly detection systems are also computationally expensive because of the overhead of keeping track of, and possibly updating several system profile metrics.

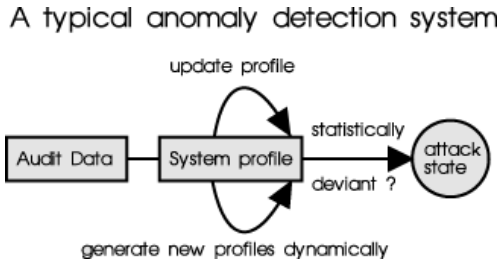

Figure 2 : Typical Anomaly Detection System

Misuse Detection: The concept behind misuse detection schemes is that there are ways to represent attacks in the form of a pattern or a signature so that even variations of the same attack can be detected. This means that these systems are not unlike virus detection systems -- they can detect many or all known attack patterns, but they are of little use for as yet unknown attack methods. An interesting point to note is that anomaly detection systems try to detect the complement of "bad" behavior. Misuse detection systems try to recognize known "bad" behavior. The main issues in misuse detection systems are how to write a signature that encompasses all possible variations of the pertinent attack, and how to write signatures that do not also match non-intrusive activity. Several methods of misuse detection, including a new pattern matching model are discussed later. A block diagram of a typical misuse detection system is shown in Figure 3 below.

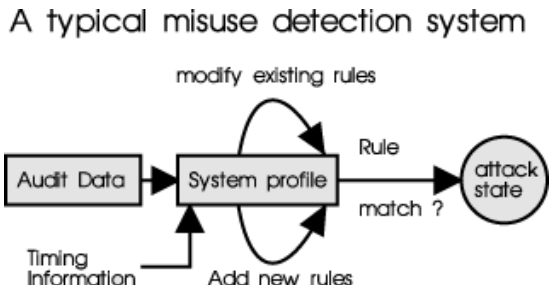

Figure 3 : Typical Misuse Detection System

Anomaly Detection Systems : There have been a few major approaches to anomaly intrusion detection systems, some of which are described below.

Statistical approaches: In this method, initially, behavior profiles for subjects are generated. As the system continues running, the anomaly detector constantly generates the variance of the present profile from the original one. We note that, in this case, there may be several measures that affect the behavior profile, like activity measures, CPU time used, number of network connections in a time period, etc. In some systems, the current profile and the previous profile are merged at intervals, but in some other systems profile generation is a one time activity. The main advantage

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

to statistical systems is that they adaptively learn the behavior of users; they are thus potentially more sensitivte than human experts. However there are a few problems with statistical approaches: they can gradually be trained by intruders so that eventually, intrusive events are considered normal, false positives and false negatives are generated depending on whether the threshold is set too low or too high, and relationships between events are missed because of the insensitivity of statistical measures to the order of events. An open issue with statistical approaches in particular, and anomaly detection systems in general, is the selection of measures to monitor. It is not known exactly what the subset of all possible measures that accurately predicts intrusive activities is. Static methods of determining these measures are sometimes misleading because of the unique features of a particular system. Thus, it seems that a combination of static and dynamic determination of the set of measures should be done. Some problems associated with this technique have been remedied by other methods, including the method involving Predictive Pattern Generation, which takes past events into account when analyzing the data.

Predictive pattern generation: This method of intrusion detection tries to predict future events based on the events that have already occurred [14]. Therefore, we could have a rule

E1 - E2 --> (E3 = 80%, E4 = 15%, E5 = 5%)

This would mean that given that events E1 and E2 have occurred, with E2 occurring after E1, there is an 80% probability that event E3 will follow, a 15% chance that event E4 will follow and a 5% probability that event E5 will follow. The problem with this is that some intrusion scenarios that are not described by the rules will not be flagged intrusive. Thus, if an event sequence A - B - C exists that is intrusive, but not listed in the rulebase, it will be classified as unrecognized. This problem can be partially solved by flagging any unknown events as intrusions (increasing the probability of false positives), or by flagging them as non-intrusive (thus increasing the probability of false negatives). In the normal case, however, an event is flagged intrusive if the left hand side of a rule is matched, but the right hand side is statistically very deviant from the prediction. There are several advantages to this approach. First, rule based sequential patterns can detect anomalous activities that were difficult with traditional methods. Second, systems built using this model are highly adaptive to changes. This is because low quality patterns are continuously eliminated, finally leaving the higher quality patterns behind. Third, it is easier to detect users who try to train the system during its learning period. And fourth, anomalous activities can be detected and reported within seconds of receiving audit events. Another approach taken in intrusion detection systems is the use of neural networks. The idea here is to train the neural network to predict a user's next action or command, given the window of n previous actions or commands. The network is trained on a set of representative user commands. After the training period, the network tries to match actual commands with the actual user profile already present in the net. Any incorrectly predicted events (events and commands are used interchangeably in this discussion) actually measure the deviation of the user from the established profile. Some advantages of using neural networks are: [8] they cope well with noisy data, their success does not depend on any statistical assumption about the nature of the underlying data, and they are easier to modify for new user communities. However, they have some problems. First, a small window will result in false positives while a large window will result in irrelevant data as well as increase the chance of false negatives. Second, the net topology is only determined after considerable trial and error. And third, the intruder can train the net during its learning phase.

Misuse Detection Systems: There has been significant research in misuse detection systems in the recent past, including attempts at SRI, Purdue University and the University of California-Davis. Some of these systems are explained in depth in this section.

Expert systems are modeled in such a way as to separate the rule matching phase from the action phase. The matching is done according to audit trail events. The Next Generation Intrusion Detection Expert System (NIDES) developed by SRI is an interesting case study for the expert system approach. NIDES follows a hybrid intrusion detection technique consisting of a misuse detection component as well as an anomaly detection component. The anomaly detector is based on the statistical approach, and it flags events as intrusive if they are largely deviant from the expected behavior. To do this, it builds user profiles based on many different criteria (more than 30 criteria, including

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

CPU and I/O usage, commands used, local network activity, system errors etc.) [8]. These profiles are updated at periodic intervals. The expert system misuse detection component encodes known intrusion scenarios and attack patterns (bugs in old versions of sendmail could be one vulnerability). The rule database can be changed for different systems. One advantage of the NIDES approach is that it has a statistical component as well as an expert system component. This means that the chances of one system catching intrusions missed by the other increase. Another advantage is the problem's control reasoning is cleanly separated from the formulation of the solution. There are some draw backs to the expert system approach too. For example, the expert system has to be formulated by a security professional and thus the system is only as strong as the security personnel who programs it [7]. This means that there is a real chance that expert systems can fail to flag intrusions. It is for this reason that NIDES has an anomaly as well as a misuse detection component. These two components are loosely coupled in the sense that they perform their operations independently for the most part. The NIDES system runs on a machine different from the machine(s) to be monitored, which could be unreasonable overhead. Furthermore, additions and deletions of rules from the rule-base must take into account the inter-dependencies between different rules in the rule-base. And there is no recognition of the sequential ordering of data, because the various conditions that make up a rule are not recognized to be ordered.

Keystroke monitoring is a very simple technique that monitors keystrokes for attack patterns. Unfortunately the system has several defects -- features of shells like bash, ksh, and tcsh in which user definable aliases are present defeat the technique unless alias expansion and semantic analysis of the commands is taken up. The method also does not analyze the running of a program, only the keystrokes. This means that a malicious program cannot be flagged for intrusive activities. Operating systems do not offer much support for keystroke capturing, so the keystroke monitor should have a hook that analyses keystrokes before sending them on to their intended receiver. An improvement to this would be to monitor system calls by application programs as well, so that an analysis of the program's execution is possible.

Model Based Intrusion Detection states that certain scenarios are inferred by certain other observable activities. If these activities are monitored, it is possible to find intrusion attempts by looking at activities that infer a certain intrusion scenario. The model based scheme consists of three important modules[4]. The anticipator uses the active models and the scenario models to try to predict the next step in the scenario that is expected to occur. A scenario model is a knowledge base with specifications of intrusion scenarios. The planner then translates this hypothesis into a format that shows the behavior as it would occur in the audit trail. It uses the predicted information to plan what to search for next. The interpreter then searches for this data in the audit trail. The system proceeds this way, accumulating more and more evidence for an intrusion attempt until a threshold is crossed; at this point, it signals an intrusion attempt. This is a very clean approach. Because the planner and the interpreter know what they are searching for at each step, the large amounts of noise present in audit data can be filtered, leading to excellent performance improvements. In addition, the system can predict the attacker's next move based on the intrusion model. These predictions can be used to verify an intrusion hypothesis, to take preventive measures, or to determine what data to look for next. However, there are some critical issues related to this system. First, patterns for intrusion scenarios must be easily recognized. Second, patterns must always occur in the behaviour being looked for. And finally, patterns must be distinguishing; they must not be associated with any other normal behaviour.

III. PROPOSED SOLUTION

Web attack detection engine implementation

The signature database of Snort is constructed by protocols and keywords. Then, it uses PCRE library for regular expression pattern matching which has better effect on specific signature attacks than on elastic Web attacks. Therefore, the purpose of this paper is to construct a Web attack detection engine by modifying the preprocessor of Snort “Http Inspect” to be able cooperate with the Core Sets of ModSecurity. The development environment is under Windows 7 using GCC 3.4.5 (built-in MinGW 5.1) for implementation of the following functions. The functions are illustrated in Figure 4.

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

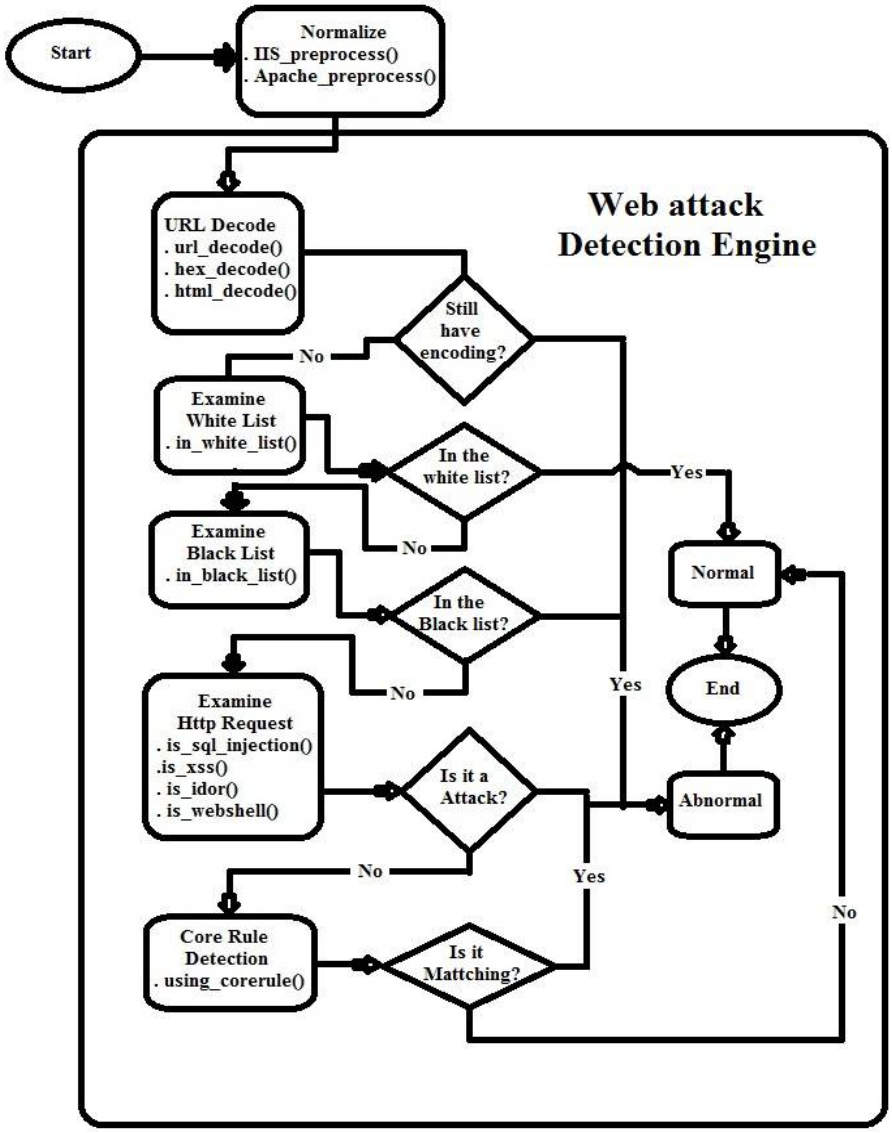

Figure 4. Diagram of Implementation Steps

Normalization: Since the web server log is taken as input data, which Snort could not analyse directly this paper uses off-line mode to analyse collected web server logs. First, pre-process the web server based on Apache and IIS format to get full URLs, and then put them in the Web attack engine for analysis.

URL decode: Before the browser sends a HTTP Request, it would first encode some symbols and Chinese words. Therefore, it should be decoded before the analysis, which would lower false alert rate of the system. At the meanwhile, hackers usually use multi-type encoding methods to encode the malicious codes evades Intrusion Detection Systems. This function would decode URLs and then determine those that still contain encoded URLs to be Web attacks.

Examining White List: Many kinds of web servers have special http request pattern and different search engines

ISSN(Online): 2320-9801
ISSN (Print): 2320-9798

International Journal of Innovative Research in Computer and Communication Engineering

(An ISO 3297: 2007 Certified Organization)

Website: www.ijircce.com

Vol. 5, Issue 4, April 2017

such as Google, Yahoo, Msn and Baidu spiders. They can by add into the white list in the first place to speed up the analysis them, as well as those that are similar to malicious attacks but normal Http request.

Examining Black List: Collect known malicious web sites for black list . They are usually presented as normal files likes as js, css, and other text files, or malicious binary files that contains shell codes such as gif, ppt and pdf etc. The malicious files could not be analyzed by simply examining the HTTP Requests. They need to be examined by further analysis of their behaviors. Therefore, this research refers to the black list released from web security forums. If an http request contains web sites in the black lists, it will be determined a Web attack. The advantage is low false alert rate, but it could not detect an unknown malicious web site automatically.

Examining Http Request: This function mainly implements the ability to recognize the attacks toward the four major web application vulnerabilities from OWASP 2007, which include SQL Injection, Cross Site Scripting, Insecure Direct Object Reference and Cross Site Request Forgery. For example, the Web Shell detection function. When a user uploads data with web programs, such as ASP, PHP or JSP, etc., this can be determined a Web Shell attack.

Core Rule Detection: This paper implements a detection engine to adopt the Core Rule Sets of ModSecurity to detect Webattacks. The advantage is: Snort could make use of another attack database to increase the detection rate.

V. CONCLUSION AND FUTURE WORK

This scheme is to improve the inadequacy of Web attack detection for Snort by implementing a Web attack detection engine on the basis of the Http Inspect pre-processor. In addition to detect SQL Injection, Cross Site Scripting, Insecure Direct Object Reference and Cross Site Request Forgery, this detection engine also provides the ability to detect Web Shell, encoded attacks and malicious web sites on black lists. Besides, this paper implements the detection engine with the Core Rule Sets of ModSecurity to provide Snort with another attack database in order to detect various attack methods.

REFERENCES

1. J.P Anderson. Computer Security Threat Monitoring and Surveillance. Technical report, James P Anderson Co., Fort Washington, Pennsylvania, April 1980.
2. Mark Crosbie and Eugene Spafford. Defending a Computer System Using Autonomous Agents. Technical Report CSD-TR-95-022, Department of Computer Sciences, Purdue University, 1995.
3. Dorothy E Denning. An Intrusion Detection Model. In *IEEE Transactions on Software Engineering*, Number 2, page 222, February 1987.
4. T D Garvey and Teresa F Lunt. Model based intrusion detection. In *Proceedings of the 14th National Computer Security Conference*, pages 372-385, October 1991.
5. Koral Ilgun. USTAT - A Real-time Intrusion Detection System for UNIX. Master's Thesis, University of California at Santa Barbara, November 1992.
6. Sandeep Kumar. Classification and Detection of Computer Intrusions. Ph.D. Dissertation, August 1995.
7. Teresa F Lunt. Detecting Intruders in Computer Systems. Conference on Auditing and Computer Technology, 1993.
8. Teresa F Lunt. A survey of intrusion detection techniques. In *Computers and Security*, 12(1993), pages 405-418.
9. Biswanath Mukherjee, L Todd Heberlein and Karl N Levitt. Network Intrusion Detection, IEEE Network, May/June 1994, pages 26-41.
10. Barton P Miller, David Koski, Cjin Pheow Lee, Vivekananda Maganty, Ravi Murthy, Ajitkumar Natarajan, Jeff Steidl. Fuzz Revisited: A Re-examination of the Reliability of UNIX Utilities and Services. Computer Sciences Department, University of Wisconsin, 1995.
11. Steven E Smaha. Haystack: An Intrusion Detection System. In *Fourth Aerospace Computer Security Applications Conference*, pages 37-44, Tracor Applied Science Inc., Austin, Texas, December 1988.
12. Eugene H Spafford. The Internet Worm Program: An Analysis. In *ACM Computer Communication Review*; 19(1), pages 17-57, Jan 1989.
13. Eugene H Spafford. Security Seminar, Department of Computer Sciences, Purdue University, Jan 1996.