Creating a Policy Based Network Intrusion Detection System using Java Platform

Samuel N. John¹, Charles U. Ndujiuba², Robert E. Okonigene³, Chinonso E. Okereke⁴, Miebaka E. Wakama⁵

¹, ², ⁴, ⁵ Department of Electrical and Information Engineering, Covenant University, Ota, Ogun State, Nigeria.
³ Department of Electrical and Electronics Engineering, Ambrose Alli University, Ekpoma, Edo State, Nigeria.

Abstract – Computer network attacks are rapidly increasing on a global scale. Security mechanisms are established and used extensively to counter these security threats knowing their ubiquitous nature and severity. Like in most Financial institutions and big Organizations worldwide in most Nigeria Universities network security threats have become a major challenge and of great concern to the University authorities. Most Nigeria Universities rely solely on their established networks for revenue generation. The Universities have adopted some policies that are peculiar to its needs. Several reported breaches in these Universities network by intruders and sometimes insiders threat have become a major challenge to the authorities. Hence, this paper addressed the security policy based apparatus to stop the havoc caused by these threats. Among the deployable security arsenals is Intrusion Detection System (IDS). There was a need to customize IDS to suit the security policies of specific University network and these security policies vary from network to network. In order to meet this need, a signature-based Network Intrusion Detection System was design to suit the policies of each university network and to accommodate policy changes. This was achieved through Java platform, that is, Jnetpcap Library and Java Expert System Shell (JESS). The results obtained shows that Java’s Jnetpcap Library and Expert System Shell provided a way to accommodate dynamic network demands as well as stop specific intrusion. The created Network Intrusion Detection System (NIDS) improved the overall robustness and security of the Network.

Keywords:- Intrusion Detection System, Signature-based, JESS, Jnetpcap Library, Policy Protocols, Security.

1 INTRODUCTION

Most Nigeria Universities have adopted the internet to disseminate information about the authorities policies to the immediate communities and the world. The policies involved in online: payment of school fees, registration, display of student results, student payment status, application for admission, verification of basic admission requirements, admission into various degree programmes, obtaining matriculation number, registration of courses, qualification to write examinations, information about each student status, may defer slightly depending on the University. During our study we observed that due to security breaches in most of the Universities network there are cases of identity theft, use of unauthorized pins for payment, unauthorized access to vital information, unauthorized change of students data. There are several reported cases of increased amount of attacks and sophisticated intrusions developed by intruders to compromise security of these networks. Universities have lost huge amount of revenues as a result of some of these security breaches. The function of an Intrusion Detection System (IDS) is to monitor events occurring in a computer system or network and analyze for signs of possible intrusion. An intrusion can be defined as any set of activities that attempt to compromise the integrity, confidentiality or availability of the network [1, 2]. It can also be seen as actions or traffic not legally allowed on a system or network [3]. Some examples are denial of service, masqueraders, malicious activities such as trojan and viruses. Another definition of intrusion is network policy violation. Intrusion Detection System is divided into Host-based and Network-based. The former refers to monitoring of a single computer system unit (HIDS) while the latter refers to the monitoring and analysis of the traffic of network segments or the whole network (NIDS) [4]. NIDS attempts to detect unauthorized access or intrusion from the analyzed data or information pulled together from the network traffic [5]. It is usually positioned behind the firewall for effectiveness as shown in Figure 1 as it provides a deep inspection of the payload [6]. The Intrusion Detection is usually implemented using the anomaly-based or signature-based methods. In anomaly-based intrusion detection, the baseline for normal behavior and characteristic of the network is gotten and an alert generated when a deviation from the specified baseline occurs [8]. However signature-based intrusion detection method defines patterns from known malicious codes then seeks out such patterns using algorithms to compare packets from the analyzed data and then generates alerts [8].
Signature-based detection has its advantages in that it is more accurate in identifying an intrusion attempt, provides an easy way of tracking down the cause of alarm due to detailed log files and also, reduced false positives alerts [1]. The term ‘intrusion’ which is also policy violation depends on the policy setup by the organization using the NIDS. It is therefore important that the NIDS be setup to suit the policy of the organization.

In Covenant University, Otto Nigeria minimum guidelines (policies) were set out for proper, efficient and effective use of ICT (internet) in order to regulate and ensure delivery of qualitative education [9]. Although not every policy is implementable in the NIDS, one of the ICT guidelines (policy) is Web filtering to ensure efficiency and high availability of internet services to all used. The policy requires that MP3 traffic and other high bandwidth intensive services that may not have direct educational or research value are adequately filtered. This is apart from the generic forms of intrusion that will disturb the efficiency of the network such viruses and general malware [9]. NIDS has the ability to perform a deep inspection of the payload of the traffic in the network. This helps to achieve thorough filtering. The thoroughness of the filtering depends on the specified rules or signatures which can be changed from time to time. This paper shows the creation of a signature-based NIDS that suits the specified policy and caters for frequency occurring threats and further support the overall security and integrity of the University Network.

Aijaz Ahmed [10], presented the design and implementation of a Signature-based Network Intrusion Detection System using JESS (Java Expert Shell System) (SNIDJ) which used Snort as a packet-sniffing tool for capturing network traffic and this was manually converted into facts as represented in JESS. Vamshi K Kankanala [11] showed the design and implementation of a Web-based Network Intrusion Detection Expert System (WNIDES) which was implemented using (JESS). The system used Snort program to capture network packets and has a Graphic User interface (GUI) to input new rules (i.e. signature). Chaitanya Chinthireddy [12] presented the design and implementation of an online IDS based on Snort rules as signatures using JESS and tested the system with manually generated packets. The design and implementation of a signature-based NIDS system with Java programmed sniffer using Jnetpcap Library and detection engine written in Java Expert System Shell (JESS) with Snort rules as signature was used . The signature-based NIDS has a GUI that allows users to input signatures apart from the already existing signatures in the detection system. The system was tested on the Covenant University Network. The paper is organized as follows: section 2 shows the overview of the implementation tools used in building the system. Section 3 is the design and methodology of the NIDS, section 4, indicate the implementation of the JNETPCAP library and section 5, shows the system testing and results.

2 OVERVIEW OF THE IMPLEMENTATION TOOLS IN NIDS

In this project the following implementation tools were used to build the NIDS system:

- Jnetpcap Library
- Java Expert System Shell (JESS)

2.1 JNetPcap Library

JnetPcap library is an open source java wrapper around libpcap and WinPcap native libraries. Libpcap is an application programming interface (API) for packet capturing from a live network interface. The Windows port of libpcap is called Winpcap. Libpcap and Winpcap libraries are used for making the packet capture and filtering engines among other network tools.

Jnetpcap expose the functionalities found in the libpcap and Winpcap library which provides enough tools necessary to carry-out thorough analysis on packets captured from the network.. Jnetpcap has an extensive list of protocols which is able to decode and these includes the following:

- Ethernet: 802.3, 802.2, SNAP, SLL, VLAN
Network: IP4, IP6, ICMP, ARP, RIP1, RIP2, TCP, UDP, HTTP, HTML, WEBIMAGE
- VOIP: RTP, SDP, SIP
- VPN: L2TP
- WAN: PPP [13]

Jnet pcap’s ability to decode such protocol and provide the appropriate header structures for these protocols avails the ability to carry-out adequate analysis and intrusion detective schemes on the protocols as the needs arises.

2.2 JESS (Java Expert System Shell)

Java Expert System Shell is a rule engine and scripting environment written entirely in Oracle's Java™ language with a fully developed API used to create rule-based expert system [14,15]. A Rule Based Expert System Architecture is made up of a set of rules called the Rule Base, a memory that analyzes program data known as the working memory and the inference engine which does the decision making and implementation of the decisions made. The Inference Engine is made up of components such as a pattern matcher, an agenda and an execution engine. A pattern matcher refers to the component that scans through the program data and also the rule base to see which rule or rules can be executed on the program data. The agenda on the other hand orders the implementation of the rules in the case where multiple rules must be implemented. The execution engine then ‘fires’ or executes the selected rules in their specified order [15].

JESS’s rule engine uses the Rete matching algorithm which is known to speed the matching of patterns rules against the working memory or program data also known as facts [16,17]. Figure 2 shows the architectural diagram of a Rule Based Expert System with the components that link the system together.

3 SYSTEM DESIGN & METHODOLOGY

The overall design model of our Network Intrusion Detection System is based on the working model of Rule Based Expert System Architectural Diagram. The design is as shown in Figure 3.

Figure 3. Overall Design of signature-based NIDS

It comprises of a sniffer or sensor and the rule based expert system which is the detection engine. Components of the rule based expert system are modified to suit the intrusion detection system i.e. the Rule Base in the developed NIDS is the Signature Model or set of signatures, the Working Memory represents the program data or traffic captured by the sniffer, the Pattern Matcher and the Agenda are represented by the Decision Engine while the Execution Engine does the implementation of the specified signatures. The NIDS also shows details of the network packets and alerts generated in its GUI. The flowchart of the System is shown in the Figure 4.

Working Principle

The sniffer/sensor of the policy-based NIDS captures live network traffic in the form of packets. Packets are then interpreted by the sniffer and facts are generated which are sent to the Expert System (JESS). Packets from the network traffic which are now facts are sent to the detection engine were they are compared with existing signature/rules in the Signature Model. The JESS program matches facts in the working memory to rules in the rule base or signature model of our intrusion detection System.

Figure 2. Architecture Diagram of Rule Based Expert System [15]
Figure 4. Flowchart of NIDS Working Principle

The rules contain function calls that manipulate the fact base. The output is then sent to the decision engine to the execution engine where decision taken is implemented either an alert and log or log and pass on.

4 IMPLEMENTATION OF JNETPCAP LIBRARY

Basically, NIDS was programmed using the Java Platform and JESS. The Jess language can be extended in java and also incorporated into a Java program making it possible to build a single java program, in our case the signature-based Intrusion Detection System and robust GUI. The implementation can be categorized into the following subheadings.

4.1 Packet Capturing, decoding and analysis

The Jnetcap library allows the sniffer to capture and decode the packets which is to be analyzed by the NIDS in the network. The NIDS analysis the captured packets with their various protocols such as ARP, IP, TCP, UDP and HTTP. Reassembly of fragmented packets will be carried out as detection of signatures strictly relies on complete payload for matching with the rules or else threats or attacks may be missed or not detected.

4.2 Converting Packets to Facts

The properties or details of the packet form the facts that are to be matched with the rules which mostly lie in the payload where most attack-signatures are found. However, certain clients or server addresses, ports may not be allowed and so will also be checked including some other facts that may be of concern. The template of our rules tells what packet data properties are to be checked because these templates form the various properties of the packet which are to be analyzed. This include the Source and Destination addresses and ports, payload of packets, Type of service (TOS), Time to live (TTL), Identification number (ID) and Ack flag. These data are extracted from the packet and converted to their respective valid variable types in the program for comparison with the rules. Some cases may arise when certain information may not be available, for instance, when the last header in the packet is the ARP header, which will leave out the IP, TCP and HTTP header fields. In such a case, the MAC address and payload will be checked as they are most concern in this scenario and other similar cases. Other unavailable field will be set to null. The bottom line in the packet facts are generated from every packet header fields that follows the packet in question.

These facts, after been, generated are then passed on to the rule engine for matching with the rules of the engine which will trigger upon a positive match between the rule and facts that are generated and vice versa.

5. SYSTEM TESTING AND RESULTS

The NIDS system was executed and tested on a Covenant University network. The results are explained below.

At the start of the NIDS a Graphic User Interface pops up giving the details of the packets that move along the network. The details include the source, destination IP address and protocol among others as shown in Figure 5. This also shows the content of the payload message with the time tag (as shown in Figure 6) which is needed for a more in-depth analysis of their content. When patterns matching rules are detected, alerts are shown in the alert window as shown in Figure 7. Apart from been shown in the GUI, details concerning the
packets, payload and alerts are stored and saved in appropriate text file for record purposes and further analysis as the case arises.

![Figure 5. GUI Showing Packet Details](image1)

Figure 5. GUI Showing Packet Details

![Figure 6. GUI Showing Payload](image2)

Figure 6. GUI Showing Payload

![Figure 7. GUI Showing Alerts Detected](image3)

Figure 7. GUI Showing Alerts Detected

The NIDS program also provides an interface that includes custom rules/signatures which are not listed in the generic signature base. In the Figure 8, the window allows the user to include their own rules for the rule engine depending on what are to be detected/tested by the NIDS system, as well as how detection should be handle. Each field allows flexibility in choices such as: any source or destination address or ports, the direction of flow and also choice of to log or alert the packet or both. This interface enables the policies that guide an organization to be included and thereby making the NIDS system relevant and useful to the network environment of such organization. Covenant University is an example of such organization where this was implemented.

![Figure 8. GUI Showing Interface to input new rules](image4)

Figure 8. GUI Showing Interface to input new rules

6 CONCLUSION

This IDS provides a more user friendly interface to the public which makes it very easy for users to get acquainted with, especially when including rules into the rule engine. Not to mention of its platform-independent execution environment as it is being developed as a java program. Also the use of the JnetPcap library exposes us to a lot of already decoded protocols which we could exploit for various attacks as it may concern the network in question. This library also makes it possible for us to analyze customized protocols for privately owned user networks enabling us to build an IDS for such privately owned networks.

7 REFERENCES


