ENHANCED NETWORK INTRUSION DETECTION SYSTEM

Benson Y. Baha and S. Isaac

Department of Mathematics and Computer Science, Taraba State University, Jalingo.

Corresponding author: bybaha@yahoo.com, +234(0)8032311462

ABSTRACT

The internet is constantly evolving and new vulnerabilities and exploits are found regularly, which motivates the need of an enhanced Network Intrusion Detection system (eNIDS) to be integral part of the network. An eNIDS provide an additional level of protection to detect the presence of an intruder, and help to provide accountability for the attacker's action. Some systems may attempt to stop an intrusion but this is neither required nor expected of a monitoring system. An eNIDS is primarily focused on identifying possible incidents, logging information about them, and reporting attempts. It knows how long the input buffer for an application is and capable of detecting all overflow attacks aimed at this service and have a complete coverage property. Existing Network Intrusion Detection Systems (NIDS) were surveyed to find the present extent of protection and actions on attackers. The design of the eNIDS was detailed using Use-case diagram and corresponding Data flow diagram. Java was used to implement the design of the eNIDS.

Keywords: Attack, Detection, Enhanced, Intrusion, Network

INTRODUCTION

In today's internet age, having a sound firewall is the first line of defence against external threats, a second layer of protection to detect the presence of attacks within traffic that flows in through the holes punched into the firewall is very important. With the increased use of the internet in businesses, it is very necessary to restrict unauthorized access to the networks from intruders of other networks. Organizations share data from their database servers in order to facilitate quick business transactions. In this type of business, database servers and organization networks are vulnerable to intruders who would want to corrupt the database servers. Theoretically, the basic solution is to disconnect the network from internet, which is not practicable since the organization would want to interact with other organization's networks. In many cases, organizations disconnect unneeded applications to improve security. This is done by disconnecting applications like e-mail services to reduce the ways hackers can get into the networks (Anderson, 2001). On the other hand, organizations may build up a security system in order to restrict the intrusion on the networks like a NIDS.

A NIDS is a security system that monitors all inbound and outbound network activity and identifies any suspicious patterns that may indicate a network or system attack from someone attempting to break into or compromise a system (Denning, 1987). NIDS is considered to be a passive-monitoring system, since the main function of an NIDS product is to warn suspicious activity taking place - not prevent them. A NIDS essentially reviews the network traffic and data and then identify probes, attacks, exploits and other vulnerabilities (Vaccaro and Liepins, 1998). NIDS can respond to the suspicious event in one of several ways, which includes displaying an alert, logging the event or even paging an administrator. In some cases, the NIDS may be prompted to reconfigure the network to reduce the effects of the suspicious intrusion. A NIDS specifically looks for suspicious activity and events that might be the result of a virus, worm or hacker. This is done by looking for known intrusion signatures or attack signatures that characterize different worms or viruses and by tracking general variances which differ from regular system activity (Sebring, 1988; Smaha,1988; Paxson, 1998).

A specialized device is required to maintain tight bounds on worst case performance and update new rules without interrupting operations with standard network transaction speed. In this research, we have developed an approach that relies on a special purpose architecture that executes packet matching to detect intrusion attacks on the host system and

display warnings to the administrator and also store information regarding the IP addresses, the source, the destination, date and allow the traffic based on that information. We demonstrated how the problem can be solved by converting the large database of strings into many tiny state, each of which searches for a portion of the rules and a portion of the bits of each rule.

BRIEF BACKGROUND OF NETWORK INTRUSION DETECTION SYSTEMS

Intrusion detection is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of attempts to compromise the confidentiality, integrity, availability, or to byepass the security mechanisms of a computer or network. Vinchurkar and Reshamwala (2012) defined an intrusion detection system as an active process or device that analyzes system and network for unauthorized and nasty activity. The goal of any IDS is to catch perpetrators in the act before they do real damage to resources. An IDS protects a system from attack, misuse, and compromise. Bace and Mell (2013) outlined several compelling reasons to acquire and use IDS as to prevent problem behaviors by increasing the perceived risk of discovery and punishment for those who would attack or otherwise abuse the system,

i. To detect attacks and other security violations that are not prevented by other security measures,
ii. To detect and deal with the preambles to attack,
iii. To document the existing threat to an organization
iv. To act as quality control for security design and administration, especially of large and complex enterprises
v. To provide useful information about intrusions that do take place, allowing improved diagnosis, recovery, and correction of causative factors.

Intrusion detection and prevention systems are primarily focused on identifying possible incidents, logging information about them, attempting to stop them, and reporting them to security administrators. According to Scarfone and Mell (2007) there are four types of intrusion detection and prevention systems technologies: the network-based, wireless, Network Behavior analysis and host-based.

Denning and Neumann (1986) published a model of IDS that formed the basis for many systems today. The model used statistics for anomaly detection and resulted in early IDS at SRI International named the Intrusion Detection Expert System (IDES), which ran on Sun workstations and could consider both user and network level data. IDES had a dual approach with a rule-based Expert System to detect known types of intrusions plus a statistical anomaly detection component based on profiles of users, host systems, and target systems. Lunt (1990) proposed adding an artificial neural network as a third component such that all three components could then report to a resolver. The Time-based Inductive Machine (TIM) did anomaly detection using inductive learning of sequential user patterns in Common Lisp on a VAX 3500 computer (Lunt, 1993). The Network Security Monitor (NSM) performed masking on access matrices for anomaly detection on a Sun-3/50 workstation.

Roesch (1998) developed a free and open source network intrusion detection and prevention system that detects thousands of worms, vulnerability exploit attempts, port scans, and other suspicious behavior. However, the components of the intrusion detection system are implemented as separate programs, they are susceptible to tampering. An intruder can potentially disable or modify the programs running on a system, rendering the intrusion detection system useless or unreliable.

SecTool (2006) developed an open source host-based intrusion detection system that performs log analysis, integrity checking, root kit detection, time-based alerting and active response. However, it is characterized of a file hashing that requires taking the system offline and hashing from an alternative operating system.

Teng *et al.* (1990) developed an Unix-based network intrusion detection system detects intrusions by first parsing network traffic to extract its application-level semantics and then executing event-oriented analyzers that compare the activity with patterns deemed troublesome.

MATERIALS AND METHODS

Materials used in the research include use-case diagram and a corresponding data flow diagram, level 0 and 1. The system specifications used during the design of the software are:

- At least 4GB of AMD RAM and any sufficient Hard Disk space.

- Keyboard and Mouse.
- A LAN network (Wireless router or Modem for testing).
- A Network cable.

The non physical component used for the system are:

OPERATING SYSTEM : Window's 7
FRONT END : Java
TOOL USED :
JFrameBuilder

JFrame

The JFrame defines the Java interfaces and classes that programmers use, which act as the main window. The project contain some library class like Guava-11.0.2.jar that contains several of Google's core libraries that we rely on in our Java- based projects: Collections caching, primitives support, concurrency libraries, common annotations, string processing, I/O and AbsoluteLayout.jar that contain the layout for the JFrame. The software consists of a set of interfaces and classes written in the Java programming language. Using these standard interfaces and classes, programmers can write applications that connect to the MainGUI, and process the results. The software is consistent with the style of the core Java interfaces and classes, such as java.lang and java.awt.

Jpcap

Jpcap is an open source which is used to capture the packets and sending the network packets. Jpcap captures the packets from a network interface and analyze packets in java. Raw packets are also captured. Packets are also saved as offline file and we can also read those offline files. Packets are filtered by the Jpcap according to the users specific rule before being dispatching to application.

NetBeans or Jcreator

This is the compiler needed to run the software, which can either be any of the two.

DETAILED eNIDS STEPS USING USE-CASE DIAGRAM

1. First the user receives the incoming packets from the internet.
2. Then the received packets are examined by matching with the manual virus database with the help of JPCAP software.
3. If the packets contain the viruses which are in the manual database then these packets are called as infected packets.
4. The infected packets are shown with a red color warning and the detailed reports of these infected packets are sent to the user.
5. Authentication of static IP addresses can also be done by the user by adding the IP
6. addresses to the database of authenticated IP addresses.

Overview of Design using Data flow Diagram (DFD)

This section provides an overview of the entire design document. This document describes all data; architectural, interface and component-level design for the software. Fig 2 is the general level 0 DFD and Fig 2 is an extension of the DFD level 0 diagram.

1. User requests to download packets from the internet.
2. When the packets are downloaded the security system examines the packets by checking with manual virus database.
3. Then if the packets are infected the detailed report of the infected packets are send to the user.
4. The IP authentication is also done by the user by adding the IP addresses that need to be authenticated to the database of authorized IP addresses database.

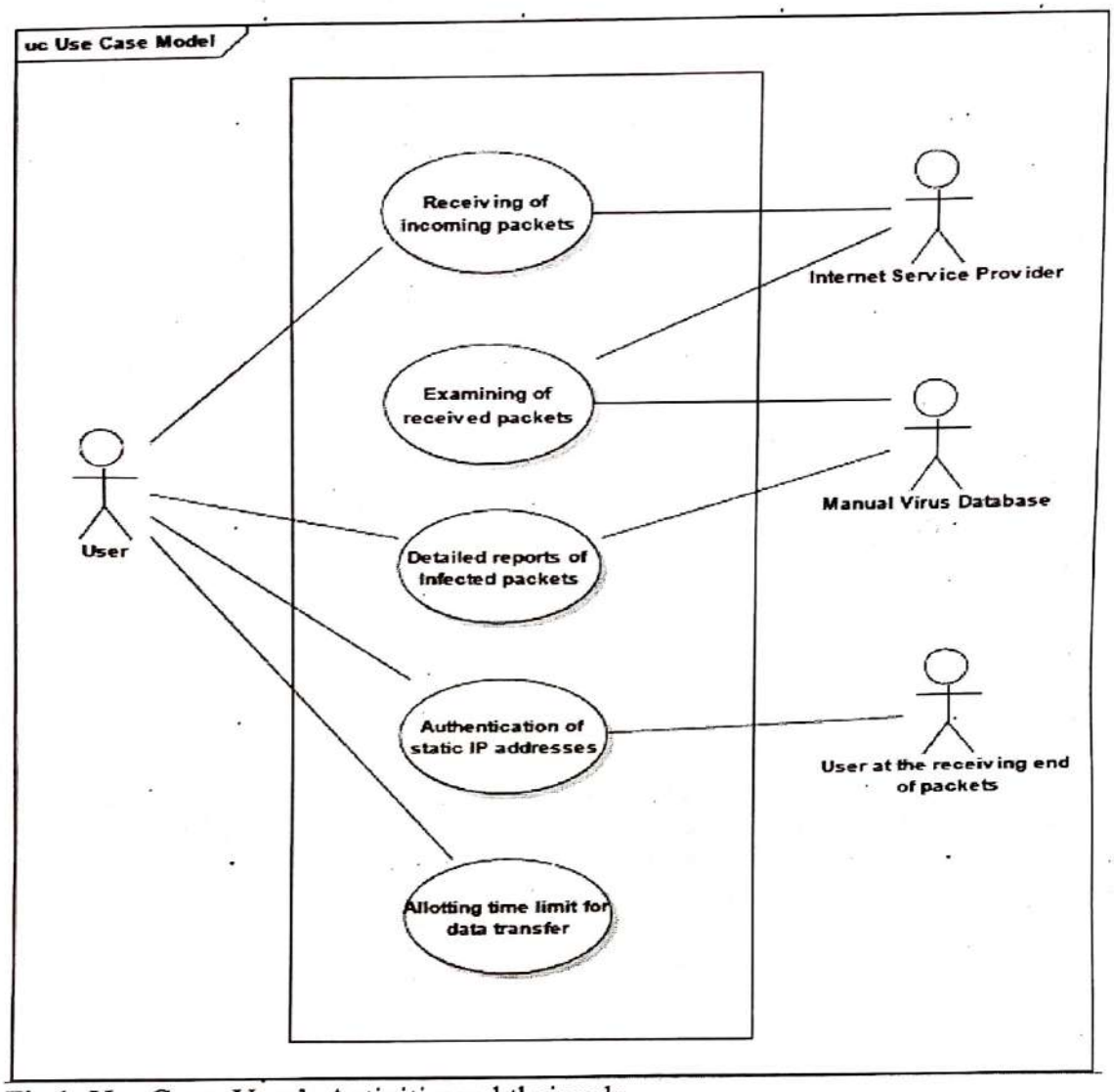

Fig 1: Use Case: User's Activitiesand their role

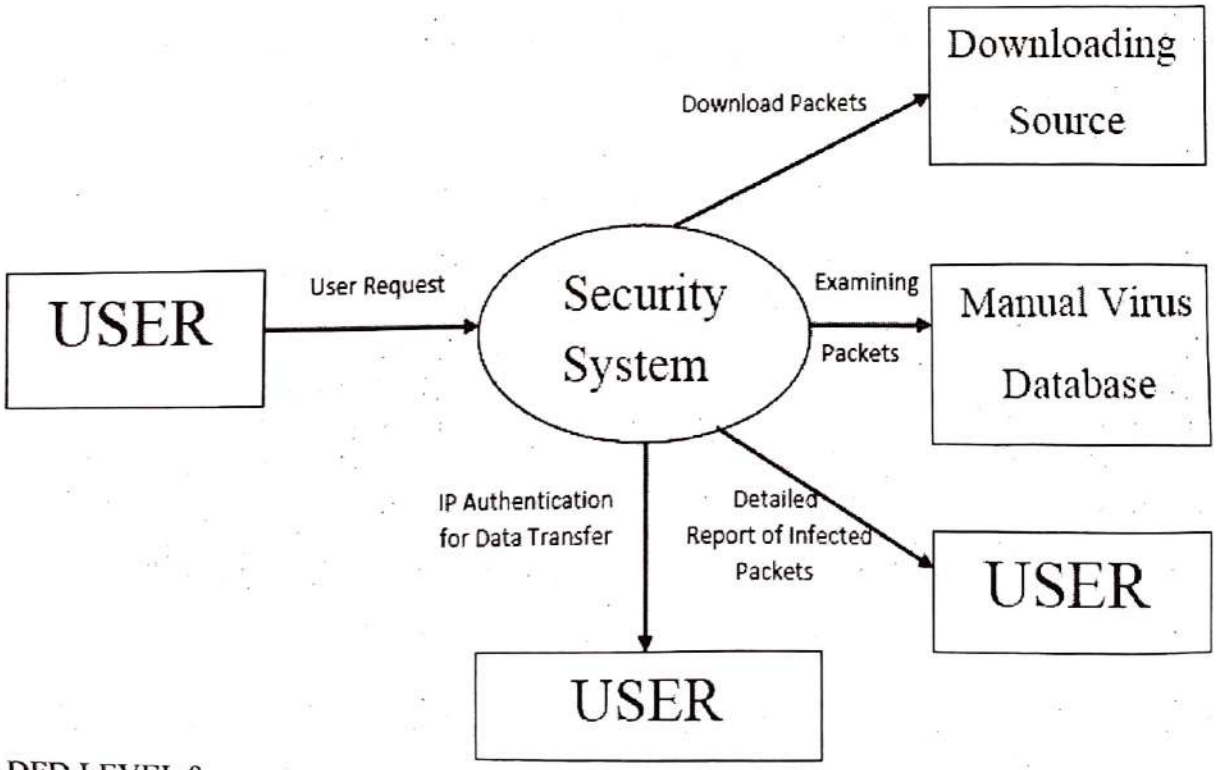

Fig 2: DFD LEVEL 0

DFD – LEVEL 1:

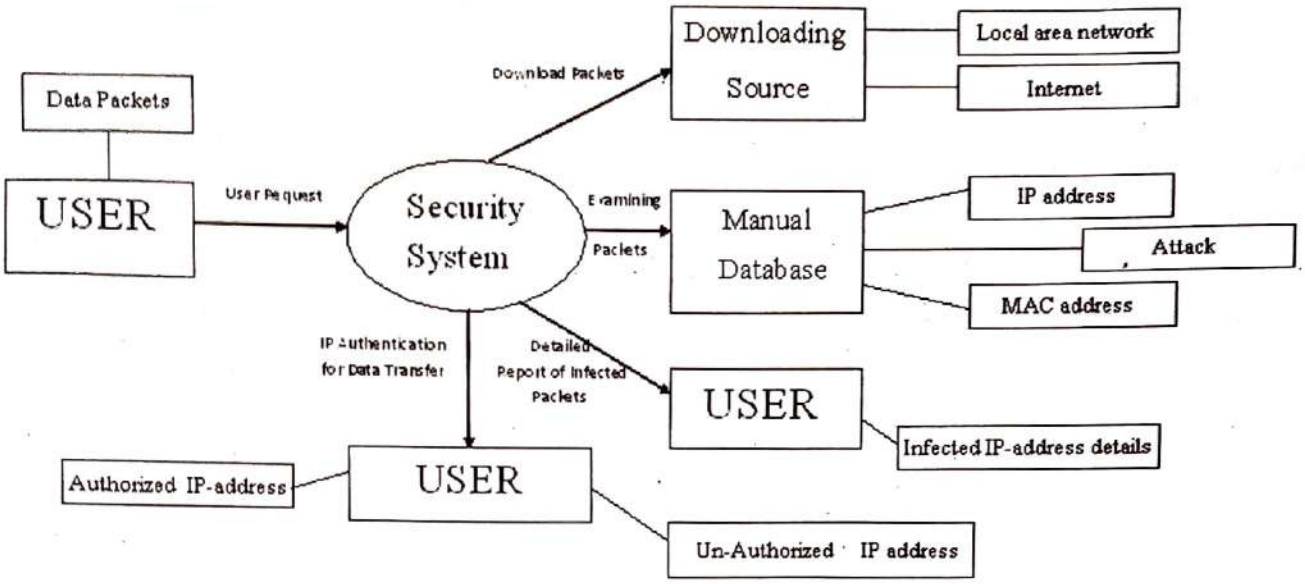

Fig 3: DFD LEVEL 1

SUMMARY AND CONCLUSION

The eNIDS was designed to provide the basic detection techniques so as to secure the systems that are directly or indirectly connected to the Internet or Intranet. Some of the facilities it provides include monitoring and analysis of system events and user behaviors, testing the security states of system configurations, recognizing patterns of system events that correspond to known attacks and recognizing patterns of activity that statistically vary from normal activity. It knows how long the input buffer for an application is and capable of detecting all overflow attacks aimed at this service and have a complete coverage property. Unlike other conventional Intrusion Detection Systems, eNIDS provides additional facilities for Intrusion Protection. This facilitates blocking or allowing particular IP, range of IPs or a subnet IPs by applying relevant rule on the operating system.

In conclusion, the eNIDS does its job but it is up to the Network Administrator to make sure that his network is out of danger. It helps the Network Administrator to track down bad guys on the Internet whose very purpose is to bring the network to a breach point and make it vulnerable to attacks. This eNIDS employs a log that is valid only for the current session and it does store the information about the past sessions. This feature can be extended by enhancing the log capability to store the information about the past sessions.

REFERENCES

Anderson, R. (2001). *Security Engineering: A Guide to Building Dependable Distributed Systems*. New York: John Wiley & Sons. Pp. 387388. ISBN 978-0-471-38922-4.

Bace, R. and Mell, P. (n.d). NIST Special Publication on Intrusion Detection Systems. Retrieved from www.21cfrpart11.com/files/library/government/intrusion_detection_systems_0201_draft.pdf#page31 on 11/02/2014.

Denning, D. E. (1987). "An intrusion-detection model". IEEE Transactions on Software Engineering, Vol.SE-13 (No. 2):222-232.

Denning, D. E. (1986), "An Intrusion Detection Model", Proceedings of the Seven IEEE Symposium on Security and Privacy, pages 119131

Lunt, T. F. (1990). "IDES:An Intelligent System for Detecting Intruders", Proceedings of the Symposium on Computer Security; Threats, and Countermeasures; Rome, Italy, pages 110121.

Lunt, T. F. (1993), "Detecting Intruders in Computer Systems", Conference on Auditing and Computer Technology, SRI International.

Paxson, V. (1998). A System for Detecting Network Intruders in Real-Time", Proceedings of the (7USENIX) Security Symposium, San Antonio, TX.

Roesch, M. (1998). Snort Software: http://en.wikipedia.org/wiki/snort_%28 software%29.

Sebring, M. M. and Whitehurst, R. A. (1988). "Expert Systems in Intrusion Detection: A Case Study", the (11th) National Computer Security Conference.

SecTools.Org: (2006) Results; http://sectools.org/tools2006.html

Scarfone and Mell (2007). Guide to Intrusion Detection and Prevention Systems: Recommendations to the National Institute of Standards and Technology. Special 4Publication 800-94. Retrieved from
http://csrc.nist.gov/publications/nistpubs/800-94.pdfSmaha, S. E. (1988). "Haystack: An Intrusion Detection System" , the Fourth Aerospace Computer Security Applications Conference, Orlando, FL.

Teng, H. S., Chen, K., and Lu, S. C.Y. (1990) "Adaptive Real-time Anomaly Detection Using Inductively Generated Sequential Patterns", IEEE Symposium on Security and Privacy.

Vinchurkar, D. P. and Reshamwala, A. (2012). A Review of Intrusion Detection System Using Neural Network and Machine Learning Technique. International Journal of Engineering Science and Innovative Technology (IJESITA), Vol. 1, issue 2, Pp54-63.

Vaccaro, H.S. and Liepins, G.E. (1989), "Detection of Anomalous Computer Session Activity",the (1989) IEEE Symposium on Security and Privacy.