Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

COMBINING HOST-BASED AND NETWORK-BASED INTRUSION DETECTION SYSTEM: A COST EFFECTIVE TOOL FOR MANAGING INTRUSION DETECTION

[*] I. C. Kalu[1] and L. N. Onyejegbu[2]

[1,2]Department of Computer Science,
University of Port Harcourt, Port Harcourt,
Rivers State, Nigeria

***** *Corresponding author*

[1] edumaije@gmail.com, [2]nneka2k@yahoo.com

Received: 16-04-14
Accepted: 19-05-14

ABSTRACT

Intrusion detection has emerged as an important approach to data security. Current researches in areas of intrusion detection have tended towards network-based systems and how to improve on their intrusion detection. Only a little attention is given to host-based systems. For a thorough check on intrusion, both host-based and network-based systems should be involved to effectively detect attacks from insider as well as outsider sources. Installing separate systems, however, could be expensive. Some Intrusion detection System (IDS) vendors prefer to market them separately for some commercial gains. This study aims to integrate the advantages of both types of IDSs in a simple design that is cost effective. The proposed system uses its knowledge-based approach in the security log of the event log file in the Windows operating system to detect failed logins and unauthorized logins for the host-based IDS module while the network-based IDS module uses a hybrid approach to detect attacks like land attack, syn flood, smurf, ping of death, and dictionary attacks. These attacks were simulated using hping. The proposed system is implemented in Java. The results show that the proposed system is able to detect attacks both from within (host-based) and outside sources (network-based).

Key Words: Intrusion Detection System (IDS), Host-based, Network-based, Signature, Security log.

INTRODUCTION

Nowadays, the use of the Internet has increased considerably. Its use has spread to the core of most of the businesses. The connectivity it provides allows corporations to extend their activity and increase productivity. Since the Internet developed recently at such a fast rate, its availability increased greatly. Connections to the Internet are now available everywhere, and at relatively low prices. This means that virtually anybody can access it and access any network. This ease of accessibility introduced a new kind of criminality: cyber-crime (NIST, 2006). This type of crime developed exponentially during the past decade, mainly due to the democratization of the Internet (Grossklags and Chuang, 2008).

Data is the most important asset in an organization (Labib, 2004). This highlights the crucial need for network security in order to keep data secure. Computer network security is often deployed in two ways. The first security application tries to establish a strong outside barrier in order to prevent unauthorized users gaining access to a network (Ingham and Forrest, 2002). Since internal users still need to access resources outside the local network, this barrier has to let some communications go through. Intruders usually take advantage of these characteristics to carry out exploits. In order to address this security issue, the second type of security applied is monitoring the network for traces of exploits (Bace and Mell, 2001).

Before the IDS came to the surface, there was the concept of audit. Audit is defined as the process of generating, recording and reviewing a chronological record of system event (Bace, 2000). People use to audit their system to achieve a variety of goals. These goals include the following: Implement and maintain personal accountability for system activities, rebuild events, check damages, monitor affected areas, allow efficient damage recovery, and restrict improper use of the system. This audit process, as shown in figure 1, could be done by either computer or manual process. It includes the audit trail generator, logger, analyzer and reporting mechanism.

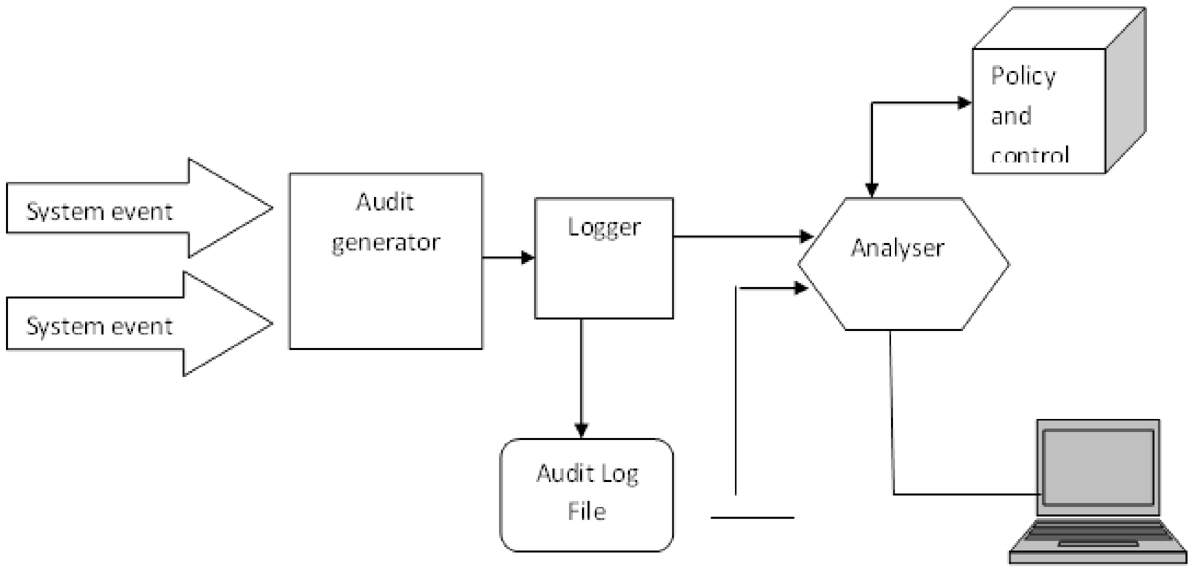


Fig 1: Basic audit system

Intrusion detection is the process of monitoring and analysing the events occurring in a computer system in order to detect signs of security problems (Bace, 2000). Intrusion detection systems (IDSs) are software applications which automate these monitoring and analysis processes (Bace and Mell, 2001). IDSs are typically used to detect attacks or violations not detected by other security means; to detect reconnaissance attempts preceding attacks such as with probes and scans. They can also be used to control the quality of an existing security design and administration, or to help diagnosis, recovery and correction of breaches in case of a current attack occurred (Bace and Mell, 2001). Figure 2 shows different component of IDS:

Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

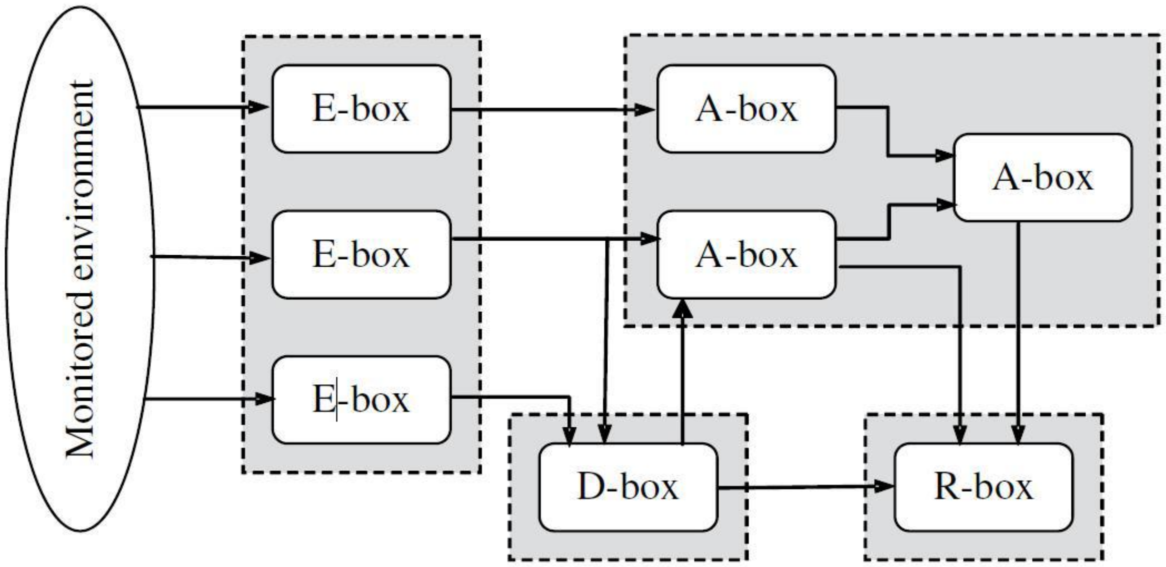


Fig 2: Components of IDS

An ID captures monitored data through its sensors ("E-box"). It then compares this data in the analysis module ("A-box") and stores it ("D-box"). It can finally react to a detected intrusion via a reaction component ("R-box") (Garcia-Teodoro et al, 2008).

Over the past ten years, intrusion detection and other security technologies such as cryptography, authentication, and firewalls have increasingly gained in importance. Additionally, intrusion detection systems (IDSs) are categorized according to the kind of input information they analyse. This leads to the distinction between host-based and network-based IDSs. Host-based IDSs analyse host-bound audit sources such as operating system audit trails, system logs, or application logs. Network-based IDSs analyse network packets that are captured on a network (Bace, 2000).

Most IDSs are based on hand-crafted signatures that are developed by manual encoding of expert knowledge. These systems match activity on the system being monitored to known signatures of attacks. The major problem with this approach is that these IDSs fail to generalize to detect new attacks or attacks without known signatures. Recently, there has been an increased interest in data mining-based approaches to building detection models for IDSs. These models generalize from both known attacks and normal behaviour in order to detect unknown attacks. They can also be generated in a quicker and more automated method than manually encoded models that require difficult analysis of audit data by domain experts (Lee, 1998). Debar et al., (1999) summarized the characteristics of IDS in figure 3 below:

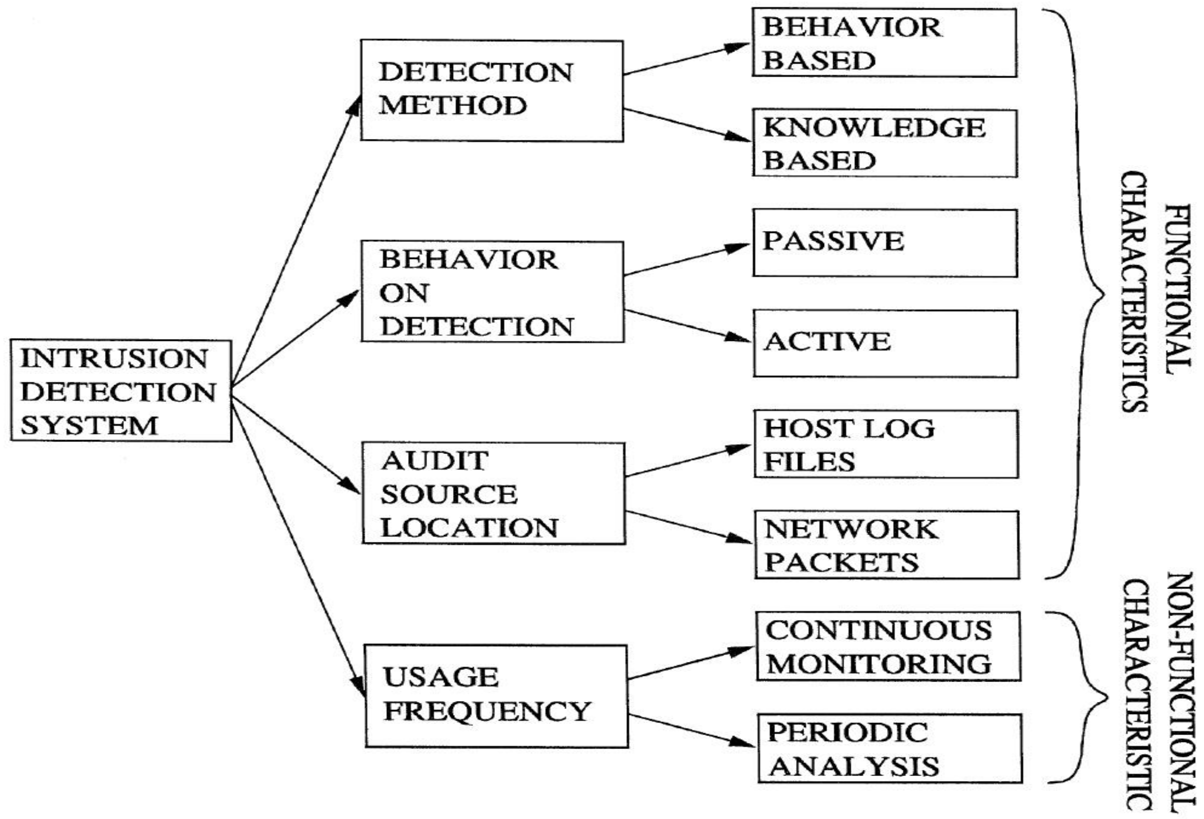


Fig 3: Characteristics of IDS

Network-oriented intrusion detection system can be roughly divided into distributed IDSs and network-based IDS (Vigna and Kemmerer, 2000). Chen (2001) listed some of the available distributed IDS: Autonomous Agent for Intrusion Detection (AAFID), Event Monitoring Enabling Responses to Anomalous Live Disturbances (EMERALD), Graph-based IDS (GrIDS) and Generic Intrusion Detection Model (GIDEM). Examples of network-based IDS are Snort, Bro, Fragroute and Sguil while Tripwire is an example of Host-based IDS.

MATERIALS AND METHODS

The proposed IDS is developed in JAVA programming language. Java is a full-featured, general-purpose programming language that is capable of developing robust mission critical applications. Many open source tools were used to implement the proposed intrusion detection system. Among them are WinPcap, JpCap, JavaMail, MySQL, and JNA. The following subsections give an overiew on each of them.

- **WinPcap**: this is a programming interface that allows packet capture over networks. Under UNIX/Linux PCAP is implemented through the library libcap. The library WinPcap is the Windows version of the library libcap. Supervision tools can use pcap (or WinPcap) to capture packets over the network; and to record captured packets in a file and to read saved file.

- **JpCap**: Jpcap is an open source library for capturing and sending network packets from Java applications. It provides facilities to: capture raw packets live from

Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

the wire, save captured packets to an offline file, and read captured packets from an offline file, automatically identify packet types and generate corresponding Java objects (for Ethernet, IPv4, IPv6, ARP/RARP, TCP, UDP, and ICMPv4 packets), filter the packets according to user-specified rules before dispatching them to the application, send raw packets to the network. Jpcap is based on libpcap/winpcap, and is implemented in C and Java programming languages. Jpcap can be used to develop many kinds of network applications, including network and protocol analyzers, network monitors, traffic loggers, traffic generators, user-level bridges and routers, network intrusion detection systems, network scanners, security tools (Fugii, 2007)

- **JavaMail**: The JavaMail API provides classes that model a mail system. JavaMail classes and interfaces are set within four packages namely javax.mail, java.mail.internet, javax.mail.event, and javax.mail.search. The javax.mail package defines classes that are common to all mail systems. The javax.mail.internet package defines classes that are specific to mail systems based on Internet standards such as MIME, SMTP, POP3, and IMAP. The JavaMail API includes the javax.mail package and subpackages.

- **MySQL**: MySQL is one of the most used database management system over the world. It is used in this work to implement a relational database that stores information about captured packets and generated alarms once an intrusion is detected over the network (Thibaud, 2006)

- **JNA**: Java Native Access JNA library uses native functions allowing code to load a library by name and retrieve a pointer to a function within that library, and uses libffi library to invoke it, all without static bindings, header files, or any compile phase. The developer uses a Java interface to describe functions and structures in the target native library. This makes it quite easy to take advantage of native platform features without incurring the high development overhead of configuring and building JNI code. It contains classes for retrieving information from the operating system's event log.

Proposed System

The Proposed system is divided into two modules "Network Module" and "Host Module". In Host based IDS, A log analyser analyses security log files and its attribute. There are so many attributes in the security log files but two were chosen: first, is login failed and second, is un-authorization accessing. Both attributes are suitable to analyse security of the system. The detection engine applies intrusion analysis technique to find intrusions in these files. Finally, alarm generator generates alarm, system administrator is alerted and the intrusion is recorded in the database.

The network module of the proposed IDS captures traffic packets which are travelling over the network. After capturing packets, it decodes the packets and matches them against the stored patterns in the database to detect any attack that may exist in the packets. The network intrusion detection module is intended to detect numerous attacks. Since it is not possible to design an intrusion detection system for every type of attack, the proposed system is designed to detect land attack, syn flood, smurf, ping of death, and dictionary attacks. Intrusions detected by the detection engine are flagged

as attacks. Finally, alarm generator generates alarm, system administrator is alerted and the malicious packet is recorded in the database for future analysis.
Figure 4 presents the global architecture of the proposed intrusion detection system. It is made up of four levels.

- The first level corresponds to analysis of inputs from the audit log and network packet sniffing from the audit analyser and packet sniffer respectively.
  At the second level, log analysis is done by the analyser while the packet decoding is done to transmit extracted information to the third level.
- At the third, each packet is analyzed to detect a pattern for specific attacks while the analysis from the log is checked for traces of intrusions. Patterns are stored in the database. An alarm is triggered when an intrusion pattern is observed.
- The fourth level is dedicated to the alarms' management and their output mode. Reports are generated while an email alert is sent to the administrator. A table of the database records different generated alerts to help an administrator to keep record of the attacks.

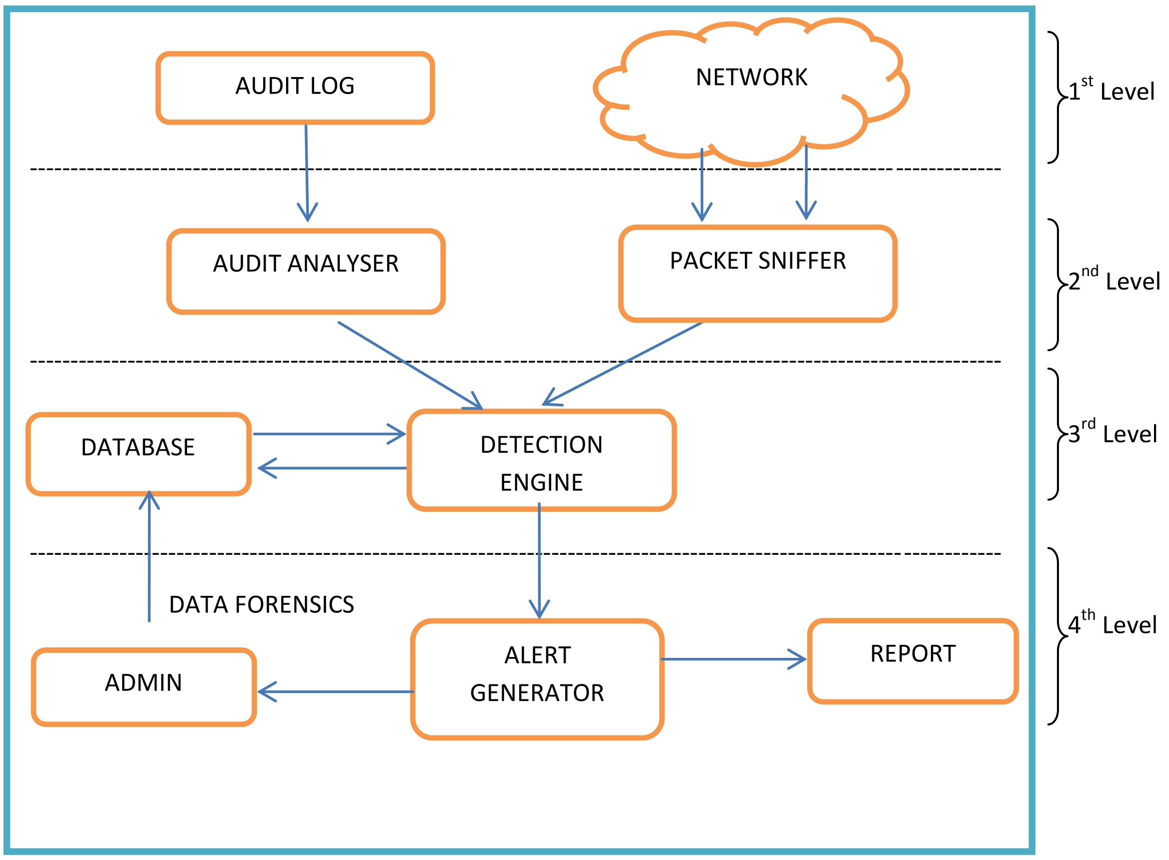


Fig 4: Global architecture of the proposed system

Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

RESULTS

The proposed network module of intrusion detection system is implemented according to the following four steps, namely listening to the network and capturing the packets, decoding the packets, detecting specific attacks, and printing the output module.

1) **Listening to the network and capturing the packets**: At this first step, a sniffer is developed using Jpcap library (jpcap v0.016). In an Ethernet network, each system has a network card which has its own physical address. When a network card is configured in the promiscuous mode, thanks to the Jpcap library, all packets are captured without being out from the traffic. The sniffer is therefore implemented using the Jpcap library through the following steps:

- seeking and printing all network interfaces available on the host machine, thanks to the method capture.findDevice(),
- selecting of the network interface to be used by the sniffer,
- activating of the network interface onto the promiscuous mode, thanks to capture.open(device, true) ,
- starting the packets capturing process through the interface PacketReceiver

2) **Decoding the packets**: Packet decoding process also is based on the Jpcap library. The decoder receives one after another all the packets from the sniffer and finds their category (TCP, UDP, ICMP, etc.) by comparing them to different available classes in the Jpcap library namely IPPacket, TCPPacket, UDPPacket, ICMPPacket, etc. For instance, if the concerned packet is TCP, the decoder collects its source and destination addresses, source and destination ports, data field and TCP flag.

3) **Detecting specific attacks**: In the proposed architecture, intrusion detection is done at this level. At level 3, a first search of intrusion is done based on the patterns. Patterns are stored in a database and scanned for intrusion detection. The database contains signatures of known attacks.

4) **Output module**: This module is executed once an attack is detected. It has three distinct modes. The first one is an alarm that informs about intrusion detection. The second mode uses one table in the database for recording attacks. The third mode is an alarm through an electronic mail sent to the system administrator. This last mode uses the Javamail library.

In Host-based module, a log analyser analysed security log files and its attribute. There are so many attributes in the security log files, we have chosen two attributes: first, is failed login and second, is un-authorization login. Both attributes are suitable to analyse security of the system. Depending on knowledge-based successful login analysis, the chance to detect unauthorized login is enhanced. If the same user logs in from two separate sites at the same time or in a short time, that account will considered to be infected. Again, if a user logs in at unusual times, that account will be flagged as being intruded. Alarm generator generates alarm at the time the abnormality is detected in the security log file.

Testing Methodology

The testing methodology is based on simulating computer users - intruders as well as normal users while the intrusion detection system is running. hping tool is used to simulate users in the experiment. Five experiments were carried out to test the proposed intrusion detection system installed on a server. Figure 5 shows how the testing was done. The user is simulated by using hping that generates and analyses TCP/IP packets and supports protocols such as TCP, UDP, ICMP, RAW-IP with traceroute mode and many other features (Cheswick and Bellion, 2003). The tool hping is installed on one host of the network to simulate different attacks towards other machines in the same network.

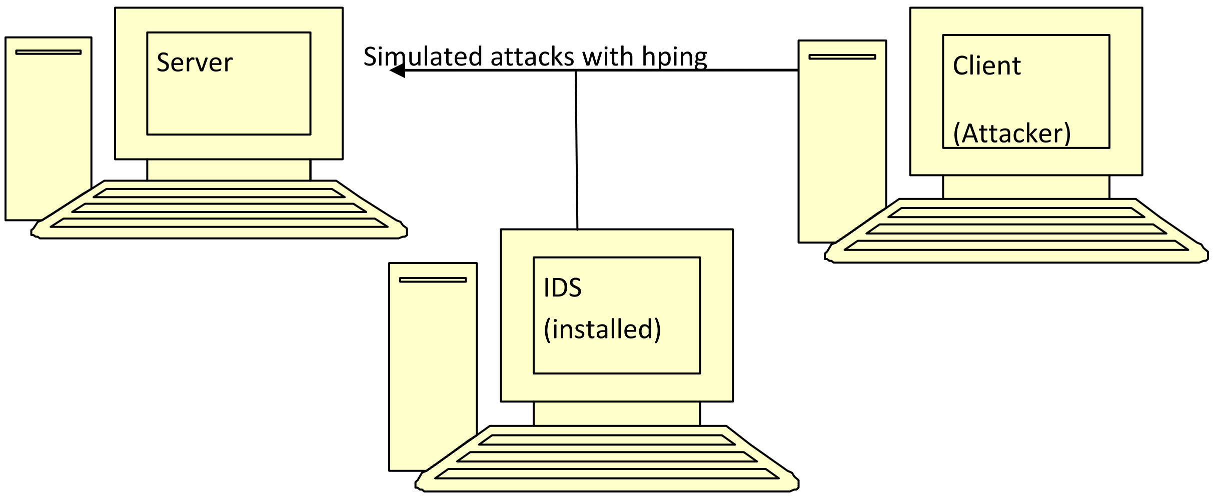
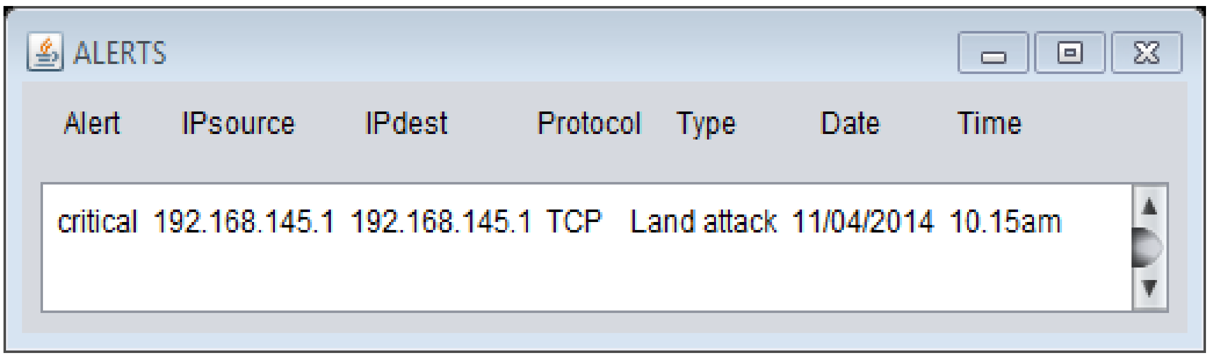


Fig 5: Proposed IDS testing

1) The LAND attack: TCP packets with the same source and destination IP address are sent over the network to simulate the LAND attack through the command: # hping3 -n -c 1 -a 192.168.145.1 192.168.145.1

Figure 6 presents the behaviour of the implemented system.



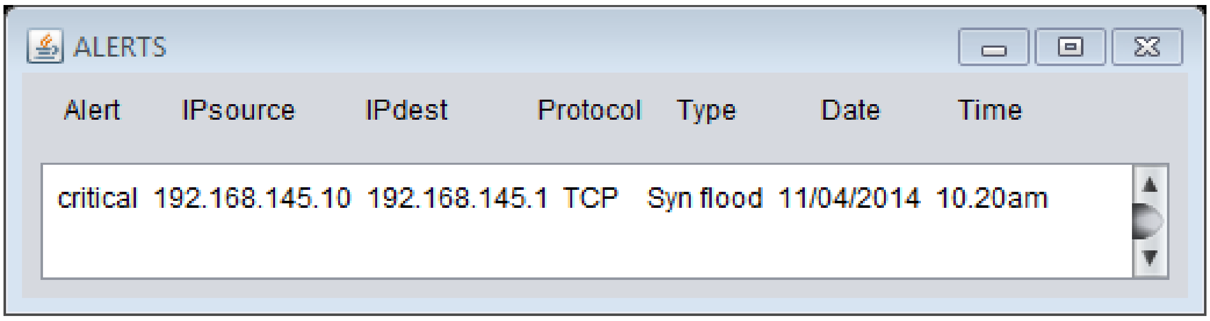
```
Alert     IPsource      IPdest        Protocol  Type         Date        Time
critical  192.168.145.1 192.168.145.1 TCP       Land attack  11/04/2014  10.15am
```

Fig 6: Land attack detection

Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

2) Syn flood attack: Flood attacks are simulated towards the host machine with 192:168:145:1 as victim through the command:

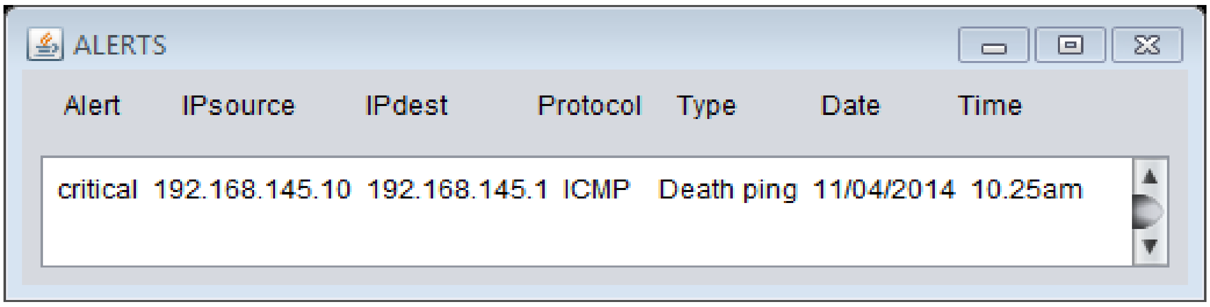
# hping3 –rand-source 192.168.145.1 –flood –S -p80
Figure 7 presents the behaviour of the implemented system.



```
Alert     IPsource        IPdest          Protocol   Type       Date         Time
critical  192.168.145.10  192.168.145.1   TCP        Syn flood  11/04/2014   10.20am
```

Fig 7: Syn flood attack

3) Death's ping attack: Death ping attacks are simulated towards the host machine with 192:168:145:1 as victim through the command:
# hping3 -icmp -c 20 192.168.145.1
Figure 8 presents the behaviour of the implemented system.

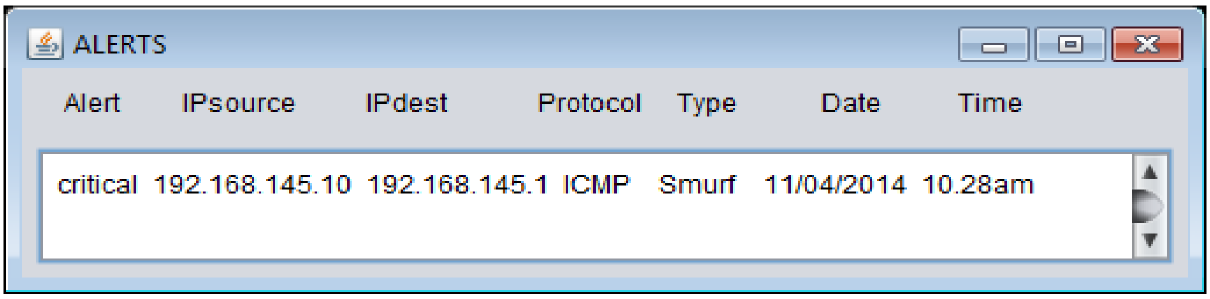


```
Alert     IPsource        IPdest          Protocol   Type        Date         Time
critical  192.168.145.10  192.168.145.1   ICMP       Death ping  11/04/2014   10.25am
```

Fig 8: Ping of Death attack

4) Smurf attack: Smurf attacks are simulated towards the host machine with 192.168.145.1 as victim through the command:

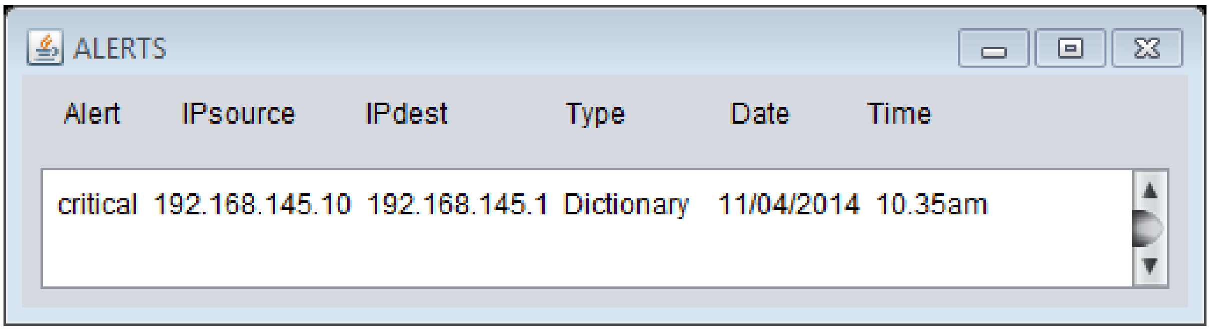
#hping3 -1 –flood –a 192.168.145.1
figure 9 shows the behaviour of the implemented network intrusion detection system



```
Alert     IPsource        IPdest          Protocol   Type     Date         Time
critical  192.168.145.10  192.168.145.1   ICMP       Smurf    11/04/2014   10.28am
```

Fig. 9: Smurf attack

5) Dictionary attack: an attempt by an intruder to gain access to the victim machine 192.168.145.1 after three attempts is captured by the host-based module of the proposed system. The output is shown in figure 10.



```
Alert     IPsource       IPdest         Type         Date        Time

critical  192.168.145.10  192.168.145.1  Dictionary   11/04/2014  10.35am
```

Fig 10: Dictionary attack

DISCUSSION

In this work, we simulated five different attacks: LAND, SYN-flood, death ping, smurf and dictionary attacks.

1. **First experiment**: Local Area Network Denial (LAND) is a Denial of Service (DoS) attack which occurs when a specially crafted TCP SYN packet is created such that the source IP address and port are set to be the same as the destination address and port, which in turn is set to point to an open port on a victim's machine. A vulnerable machine would receive such a message and reply to the destination address effectively sending the packet for reprocessing in an infinite loop. This machine CPU is consumed indifinitely freezing the vulnerable machine, causing a lock up, or even crashing it. TCP packets with the same source and destination IP address are sent over the network to simulate the LAND attack through the command # hping3 -n -c 1 -a 192.168.145.1 192.168.145.1

2. **Second experiment**: The idea behind SYN Flood attack is to abuse the TCP connection establishment process. Normal TCP connection establishment involves a three-way handshake process. Firstly, the client sends request packets with SYN bit to the server. Secondly, the server allocates a TCP control block to prepare to receive the next packet and sends a reply packet with SYN/ACK bits to the client. At this point, the server remains in half-open state. Thirdly, the client sends packets with ACK bit to the server. After the server receives packets from the client, a normal TCP connection is established. SYN Flood attack generates a large number of half-open TCP connections to fill up the TCP control block in order to exhaust the available resources. As such, the target host becomes unable to accept any more incoming TCP connections. This creates a denial of service for legitimate requests. This attack was simulated through the command # hping3 –rand-source 192.168.145.1 –flood –S -p80. Rand-source in the command is for the hping tool to randomly generate different source IPs.

Scientia Africana, Vol. 13 (No.2), December 2014. Pp69-80
© College of Natural and Applied Sciences, University of Port Harcourt, Printed in Nigeria *ISSN 1118 - 1931*

3. **Third experiment (death ping)**: An attacker sends an ICMP echo request packet containing a very large amount of information, such as oversized data to a target host. The kernel buffer of the target host will then be overflowed if it attempts to respond to this ICMP request. As a result, the host will crash following this attack. Ping of death belongs to the category of fragment packet attack that usually sends packets with bad length or offsets. This attack was simulated through the command # hping3 -icmp -c 20 192.168.145.1

4. **Fourth experiment (smurf)**: A smurf attack is carried out by using ICMP protocol. In normal conditions, source host *A* sends an echo request message to destination host *B* and then *B* makes an echo reply message to *A*. When a smurf attack occurs, host *A* spoofs the IP address by using host *C* and then sends an echo request message to a broadcast address so that the victim host *C* will receive all echo reply messages. As a result, links and routers to host *C* may get clogged by flooding traffic, and *C* cannot receive requests from other users. The smurf attack can be easily detected by observing a traffic behaviour in which a large number of echo replies is sent to a certain system with no echo request originating from the victim system. Smurf attack was simulated through #hping3 -1 –flood –a 192.168.145.1

5. **Fifth experiment (dictionary):** it is a method of breaking into a password-protected computer or server by systematically entering every word in a dictionary or word list as a password. Dictionary attacks work because many computer users and businesses insist on using ordinary words as passwords. The proposed system ensures that users input their username and password before they can have access to the system. It flags an attack when three unsuccessful attempts are made to have access to the system.

In this work, the proposed combined host-based and network-based intrusion detection system is implemented in Java. This system has been tested by simulating five different attacks: land, syn flood, death ping, dictionary and smurf attacks. The proposed system detects all these attacks correctly. The proposed is able to detect failed logins and unauthorized logins from the windows security log. The proposed Intrusion detection system is extensible and portable and many other functionalities can be implemented. The proposed system is able to detect insider threats (host-based) as well as outsider threats (network-based).

REFERENCES

Bace, R. G. (2000). *Intrusion detection*. Indianapolis: Macmillan Technical, pp 23-35

Bace, R., and Mell, P. (2001). *Nist special publication on intrusion detection systems*, National Institute of Standards and Technology: Tech. Rep

Chen, J. (2001). *Design and Implementation of a Host-based and Event-based Detector*. Msc thesis University of Florida, USA

Cheswick, N. and Bellovin, S. (2003). *Firewalls and Internet Security*: Repelling the Willy Hacker. New Jersey: Pearson Education Inc.

Debar, H., Dacier, M., and Wespi, A. (1999). *Towards a taxonomy of intrusion detection systems.* Computer Network, vol. 31, no. 9, pp. 805–822.

Fujii, F. (2007) Jpcap tutorial. [Online]. Available: http://netresearch.ics.uci.edu/kfujii/Jpcap/doc/tutorial /index.html

Garcia-Teodoro, P., Diaz-Verdejo, J., Macia-Fernandez, G., and Vazquez, E. (2008). *Anomaly-based network intrusion detection: Techniques, systems and Challenges.* Computers & Security, vol. 30, pp. 1 – 11.

Grossklags, J., Christin, N., and Chuang, J.(2008). *Security and insurance managementin networks with heterogeneous agents.* 9th ACM conference on Electronic commerce. New York, NY, USA: ACM, pp. 160–169.

Ingham, K., and Forrest, S. (2002). *A history and survey of network firewalls* University of New Mexico: Tech. Rep.

Labib, K. (2004.) *Computer security and intrusion detection*, Crossroads, vol. 11, no. 1, pp. 2–2

Lee, W., and Stolfo, S. J., (1998). *Data mining approaches for intrusion detection,* 7th USENIX Security Symposium

N. I. of Standards & Technology (2006). *An Introduction to Computer Security. The NIST Handbook*, NIST, Ed. U.S. Department of Commerce.

Thibaud, C. (2006*) MySQL 5: installation, mise en oeuvre, administration et programmation.* Edition Eyrolles,

Vigna G. and Kermmerer R. A. (2000). NetSTAT: *A Network-based Intrusion Detection Approach.* NY: Reliable Computer Group.