COST MINIMIZATION MODEL FOR AN ADAPTIVE INTRUSION RESPONSE SYSTEM

Enikuomehin A. Toyin[*1], Idowu A. Gbolahan[2]*, Akerele C. Bunmi[3]*, Owate A. Patrick .*,

Aina Bidemi Aina**,

Department of Computer Science[*],
Department of Geography[**],
Lagos State University,
Lagos, Lagos State 234/01, Nigeria
toyin@lasunigeria.org[1] ,gbolahan.idowu@lasunigeria.org[2],chriscob2000@yahoo.com[3*] powate@yahoo.com[4*]*
http://www.lasunigeria.org

Abstract
As attacks on computer systems are becoming increasingly numerous and sophisticated, there is a growing need for intrusion detection and response systems to dynamically adapt to better detect and respond to attacks. Annual reports from the Computer Emergency Response Team (CERT) indicated a significant increase in the number of computer security incidents each year. In this paper, we investigate the intrusion detection process, its technical cost implication, and its divergent nature and further proposes a system that is platform independent for an appropriate impact sensitive intrusion response system with an embedded database. We carry out several experiments on attack and present our result in form of graphs and data analysis. The paper compares several response systems and discusses it challenges in a continuous growing networking society as it is today. We finally propose an adaptive model for intrusion response system that saves technical cost for network administrators.
Keywords: Intrusion Detection, Response System, Network Administration, Java APIs

1. INTRODUCTION

Intrusion detection refers to a variety of techniques for detecting attacks in the form of malicious and unauthorized activities while Intrusion response is the counter- measure evasive and or ensure safety of the corrective actions to thwart attacks in computing environment. Ignoring security threats can result in grievous consequences. Recently, Intrusion Detection Systems (IDS) have been used in monitoring attempts to break security, which has serious consequences[1]. An impact sensitive Intrusion Response System (ISIRS) is one that provides automatic and immediate response to the intrusion through automated decision making process based upon the current state of the system. If it determines that the system it protects is in a calm state, it deploys minimum resource usage to it. But if it detects that the system to be protected is under attack that is the panic state, a more aggressive posture is maintained by deploying more resources to stop the intrusion. The impact of a response on the system is evaluated based of the defined system goals and their importance. The impact assessment process for a specific response includes three steps:

(1) Identifying the system resources affected by each response
(2) For each resource determining the priority of responses based on their effect on the resource
(3) Computing the negative impact of the responses on the associated resource using the ordering obtained in step 2

Eventually, the impact of a response on the system as a whole is an aggregation of the response's impact on the system resources. The response action in majority of the existing response systems is conservatively invoked once the existence of intrusion is confirmed. Though this strategy reduces false-positive response, delayed response action can potentially expose systems to higher level of risk from intrusions, specifically in cases where it becomes impossible to restore systems to its pre-attacked state.

Intrusion response system is a broad topic which covers a larger scope than is being discussed in this paper. The scope of this paper is to develop an intrusion response system that responds to any form of intrusion alarm from network Intrusion Detection System (NIDS). Intrusion detection and response requires constant attention of system administrators. In order to remain secure, the network environment must continually be monitored for new attacks and prompt responses must be elicited. Though most intrusions are done over a network, all of these are directed towards a system and its resources. This justifies the need for an Adaptive Intrusion Response

System which detects intrusion attempts and automatically responds to it promptly. The primary aim of this paper is to realize a balance between intrusion damage and response cost to ensure adequate response without sacrificing the normal functionality of the system under attack.
In section two of this paper we reviewed existing intrusion response systems and a justification of our approach to the design and implementation. Section three highlights our approach to the design, development and construction of each module of the IRS while section four explains the implementation of each module using some specific implementation tools and the results obtained in testing each of the modules of the IRS.

2. DESIGN METHODOLOGY

A critical analysis of the existing systems and its problems was carried out in order to design a system that will elicit a prompt response to detected or suspected intrusion. Relevant literature on concepts, techniques and works in the field of Intrusion Detection and Response were reviewed. This allows us to propose an Intrusion Response Model based on the literature surveyed. We designed and implement an Intrusion Response System using Java programming language while Java databases a RDBMS (Relational Database Management System) will provide the back end. Furthermore, a laboratorial test for the newly developed system. The study is meant to determine whether the proposed solution in the new system will be desirable within the existing managerial and organizational framework. As we have seen in the preliminary survey, intrusion response is carried out manually and this can lead to delayed or inefficient responses and several other disadvantages discussed earlier. The requirements needed for establishing this project includes Java DB for the database and also Java the programming language. The reasons for adopting these technologies include their open source nature hence affordability, OOP (Object Oriented programming) design, ease of use, robust documentation and platform independence. These attributes make the project technically feasible since it can easily be developed and deployed on both legacy and modern intranet within a short time frame with limited expertise.

2.1. *THE PROPOSED SYSTEM*

The proposed system is a platform independent impact sensitive intrusion response system with an embedded database. The proposed design is based on the popular 2-tier architecture which has the following features:

- Simple and intuitive user interface which helps in decision making.
- Lightweight application. The designed System is lenient on System resources and the file size is under 5Mb.
- Operating System independent due to the use of the Java platform.
- Visual profiling of attack history.
- Automated and centralized defense mechanism.

2.2. Users of the Proposed System

The primary Users of the System are:

- System and Network Administrators.
- The System can be deployed within the University's intranet and/or Cyber Cafe to make it immune from malicious users. The Use of this System will also enable the relevant authorities profile the different types of intrusion attempts with a view towards eliminating or mitigating its effects.

The proposed system can be deployed over a network or on a standalone system which is the current technology in use in most places and this technology is readily available and easily accessible. it will help meet the requirements of the network where users can have uninterrupted access to network resources and share these resources in a secure manner. It will allow for authorized staff to be able to monitor, review and act on all intrusion attempts in an intranet. The system is designed to be easy to use, easily accessible and understandable to end users that have authorized access to the system and designed to handle a large number of users.

2.3. *Architecture of the Proposed System*

When an intrusion is detected (in form of an anomalous activity) from the IDS and it is sent as Input into the Response Logic Manager, it is analyzed and sent to the Alert manager, the alert manager sounds a warning alert and evokes a response immediately. This is shown in the figure below:

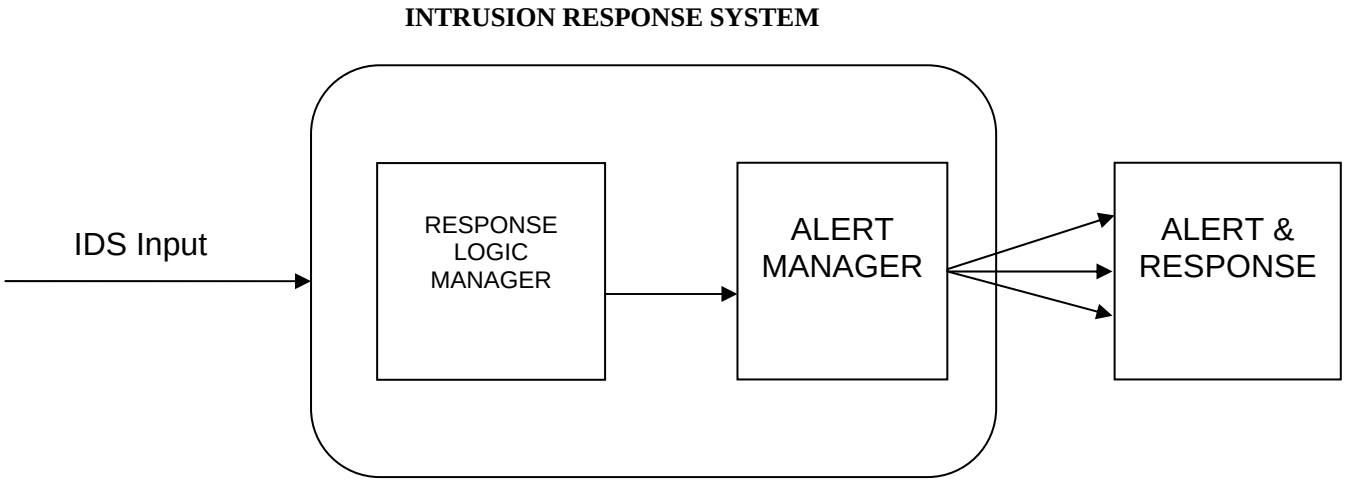


Fig 1 Architecture of Proposed System

The model used for this project is the **Iterative or Evolutionary Model** which is based on the idea of developing an initial implementation, exposing this to user comment and refining it through many versions until an adequate system has been developed. There are two basic types of evolutionary development;

- **Exploratory Development:** The objective of this process is to work with the user to explore their requirements and deliver a final system. The development starts with the parts of the system that are understood. The system evolves by adding new features proposed by the user.
- **Throwaway Prototyping:** The objective of the evolutionary development process is to understand the customer's requirements and hence develop a better requirements definition for the system. The prototype concentrates on experimenting with the user's requirements that are poorly understood. The evolutionary approach to software development is often more effective than the waterfalls approach in producing systems that meet the immediate needs of customers. The advantage of a software process that is based on an evolutionary approach is that the specification can be developed incrementally and also the approach gives room for changes to be carried out. [2].This process was chosen to meet the basic requirements of designing an IRS for an intranet.

2.4. PROCESSES

This looks at the activities that take place at the background such as what happens when a response is evoked, how it works with a reliable Intrusion Detection System (IDS) webpage comes up with its peculiar information without mix up of information and the list continues. We will therefore look at the different segments that make up the Intrusion response system.

2.5. DATABASE

The database is where all the records that are entered into the system arc stored. The database model used for this research work is the relational database model. A relational database can be seen as the data handling part of another application. The application instructs the database to perform searches, as well as add data via the Structured Query Language or SQL. The SQL standard is supported by all major database vendors, but the implementation of the full standard in all cases is not a guarantee. The common workhorse functions are the same in most cases. SQL is a very human readable language. It does have syntax rules, but it is not as hard as learning a programming language. The most basic types of queries are SELECT, INSERT and UPDPTE. Select searches for data, INSERT adds data and UPDATE changes existing data. DELETE is also fairly common. The database for this research work is designed using **Java DB** (Java Database). The database contains a table which handles different functions and data items. The database was designed to test the efficacy of the Intrusion Response System by simulating record adding functions e.g. inserting random names and addresses from different clients for a particular duration.

IRS_DATA TABLE

This is the only table in the database. It handles the names and addresses inserted into the database. It is used to demonstrate This table is purely for the purpose of simulating inserts into the database to demonstrate intrusion response. The primary key is the RID

COLUMN NAME	DATA TYPE	SIZE
RID	INT	10
FULL NAME	VARCHAR	100
ADDRESS	VARCHAR	100

Table 1 Database Table of the System

2.5.1. Log File Design

The IRS Server application maintains and organizes logged data through the use of an XML file. This file is named Logs.xml and contains the IP Address of the suspicious Client and the detection time. The Extensible Markup Language (XML) is a general-purpose markup language. It is classified as an extensible language because it allows its users to define their own elements. Its primary purpose is to facilitate the sharing of structured data across different information systems, particularly via the Internet. It is used both to encode documents and to serialize data. The XML File structure was designed suing a Java Library known as *jdom* (Java Document Object Model). JDOM is a Java based document object model for XML that was designed specifically for the Java platform so that it can take advantage if its language features. JDOM integrate with Document Object Model (DOM) and Simple API for XML (SAX), supports Xpath and XSLT.

<?xml version= "1.0" encoding="UTF-8">

<Details>Admin Logs<Details IP_41.211.228.40="—15-Mar-2009 00:18:37" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:56" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:57" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:58" /><Details IP_41.211.228.40="—15-Mar-2009 00:19:59" /><Details IP_41.211.228.40="—15-Mar-2009 00:20:52" /><Details IP_41.211.228.40="—15-Mar-2009 00:27:09" /><Details IP_41.211.228.40="—15-Mar-2009 00:29:51" /><Details

Fig 2 XML Structure of the Log file**.**

3. ALARM DESIGN

The alarm of the System was designed using the Java Sound API (Application Programming Interface).The Java Sound API is a library for effecting and controlling the input and output of sound media, including both audio and Musical Instrument Digital Interface (MIDI) data. This API is supported by an efficient sound engine which guarantees high-quality audio mixing and MIDI synthesis capabilities for the platform. The provided reference implementation of this API supports the following features: Audio file formats: AIFF, AU and WAV, Music file formats: MIDI Type 0, MIDI Type 1, and Rich Music Format (RMF) and Sound formats: 8-bit and 16-bit audio data, in mono and stereo, with sample rates from 8 kHz to 48 kHz. The location for the audio alarm is C:\irsounds this folder is created by the server on its first run, the User can choose to place an audio alarm file in this directory. This audio file is activated and played to alert the administrator whenever an intrusion is detected by the System.

4. DESIGN TOOLS

Primarily the tools used for the development of this work are JDK (Java Developers Kit) 1.6 and the Netbeans IDE (Integrated Development Environment) 6.O.The JDK contains a lot of important libraries used for the development of this work such as Java Mail API (Application Programming Interface), JDOM, Java.nct e.t.c. Applet Viewer: A viewer to test applets embedded in HTML documents include The Java compiler (javac): This compiles Java source code to Java byte code binary files (known as class files). The class files produced by javac can be run by a Java interpreter on any platform. The Java interpreter: used to execute Java class files under Linux, the Java Debugger (JDB): This is a command-line debugger which is in alpha development. It is important to need to fully understand what an IDE is before discussing NetBeans. An Integrated Developing Environment is computer software to help computer programmers develop software. It normally consists of a source code editor, a compiler and/or interpreter, build-automation tools, and (usually) a debugger. Sometimes a version control system and various tools to simplify the construction of a GUI are integrated as well. Although some multiple-language IDEs arc in use, such as the Eclipse IDE, NetBeans or Microsoft Visual Studio, typically an IDE is devoted to a specific programming language, as in the Visual Basic IDE. These have many features.

5. DEPLOYMENT PLATFORM

The designed application is operating system independent and is expected to work well on all platforms that implement the TCP/IP architecture. The Server is started by double clicking a file named runServer.bat. This file is a batch file, DOS (Disk Operating System) batch files have the filename extension **"bat"**. A batch file is a text file containing a series of commands intended to be executed by the command interpreter. When it is run, the shell program (usually COMMAND.COM or cmd.exe) reads the file and executes its commands, normally line-by-line. The Batch file was used to run the sequence of commands necessary to start the application automatically, it is intended to automate a tedious processes.

6. TESTING AND RESULT DOCUMENTATION

This is the default interface of the Client (attacker).It consists of a field to specify the address of the Server and the number of statements to be inserted into the database. The interface also displays a list that contains the different types of attacks. The User has to click the Connect button to communicate with the Server, the Send and Disconnect button are used to initiate transfer of packets and disconnect the Client respectively. The Client selects a random port to communicate through each time it is run. The selected Port No can range between 1024 and 65535, but it can only select an available Port No usually above 1024. Port Nos below 1024 are usually used by System applications and services such as Web Servers, FTP, SMTP e.t.c.. The **Test** interface was created to test the efficacy of the response system. It is used to connect to the server, initiate an intrusion and also disconnect from the server. The I.P address text box allows any authorized user connecting to the database specify the address of his system on the network. The **NoS** textbox allows the user to specify the number of inserts he/she wants to make into the database (this is automatically generated for the purpose of simulation). The **connect** button allows the user connect to the server. The **send** button allows the user send the type of attack he is initiating. The **disconnect** button disconnects the user from the server. The **attack type** drop down list specifies the attacks A, B & C. This is done in order to demonstrate the type response the server would initiate. For response type **A**, if the number of database inserts is exceeded, the server responds by disabling the send and connect buttons for a few seconds (i.e. the user cannot interact with the server for a while) this is done in order to prevent the user from overloading the server with multiple database inserts at once. This is the interface of the Server. On the interface the Port no and the IP Address of the Server is displayed. It is the administrator that has access to this interface for monitoring. The Server has been assigned a default **Port No 2009**.It is assumed that this Port will be available for the Client-Server communication. On this interface are two tables that respectively represent the network status and a view of the database. The administrator also has the option of clicking the Purge button to clear the database of all inserted records. The name and address indicates the names that have been inserted into the database.

7. Experiment

A real time simulation of the effect and responsiveness of the Network intrusion response system was obtained in a laboratory setting network environment as shown in figure bellow.
The simulated environment consist of a three network zone with each segment connected to a Cisco ASA firewall serving as an IDS with the responsibility of routing packet between the three zones and alerting the NIRS of intrusion.

Zone A is an external network on the internet as a control environment for real time evaluation; the components of this network are unknown except the targeted web server. The zone contains a server that is not protected with an NIRS. Zone B is a DMZ hosting an array of network infrastructure servers and shared resources accessible by clients from zone A and C. This zone is bordered by the developed NIRS. Zone C consist of hosts running the client interface of the NIRS system.

From the test interface on the client system from network zone C, the IP address of server 1 was inserted into the application. Attack type A was selected, clicked on the connect button to communicate with the server in the DMZ. When disconnected, the result of the attack is stated bellow in table 1.1

The same process was repeated for server 2 and server 3 which are also behind the NIDS with attack B and C respectively. The processes for all the attacks are carried out for fifteen times result are also on the same table 1.1

A similar process of attack was tried on a web server in Zone A that is not behind a NIRS, the result obtained was described in table 1.2.

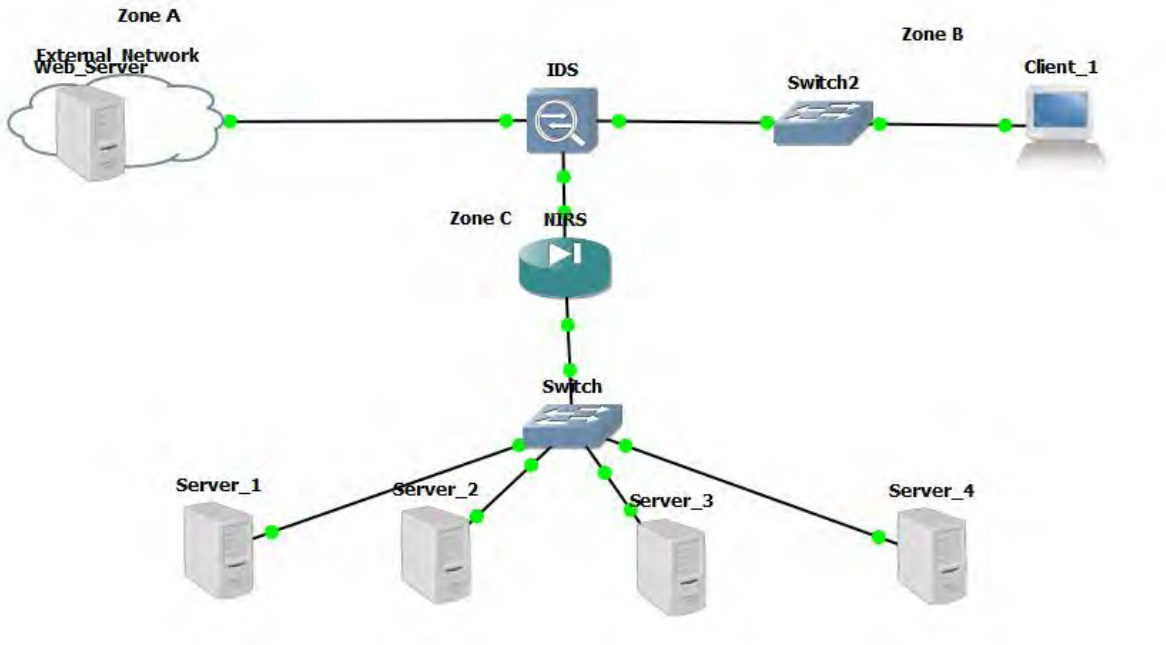


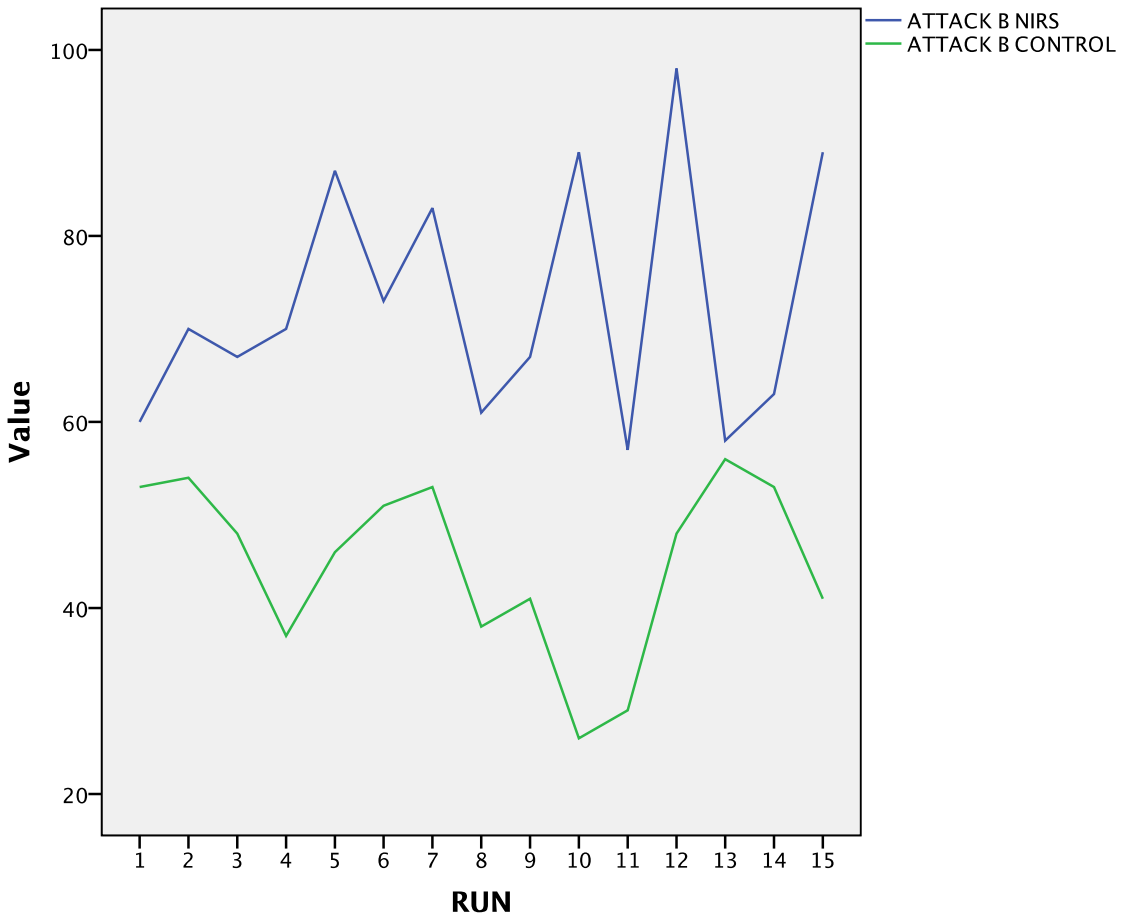
Fig 4: A schema of the simulation lab

8. Further Results:

We provide below further results in graph formats. The result is an output of the dataset provided below when run. The experiment was carried out over 15 fifteen runs.

RUN	ATTACK A		ATTACK B		ATTACK C	
	NIRS	CNTRL	NIRS	CNTRL	NIRS	CNTRL
1	90	45	60	53	73	45
2	78	52	70	54	82	24
3	75	27	67	48	73	51
4	70	47	70	37	86	34
5	69	32	87	46	70	43
6	92	31	73	51	60	49
7	58	50	83	53	88	34
8	66	27	61	38	90	52
9	65	26	67	41	68	40
10	66	31	89	26	88	31
11	52	28	57	29	97	41
12	92	25	98	48	81	45
13	50	27	58	56	65	24
14	61	39	63	53	62	46
15	96	28	89	41	57	47

Table 1: The dataset used for the runs



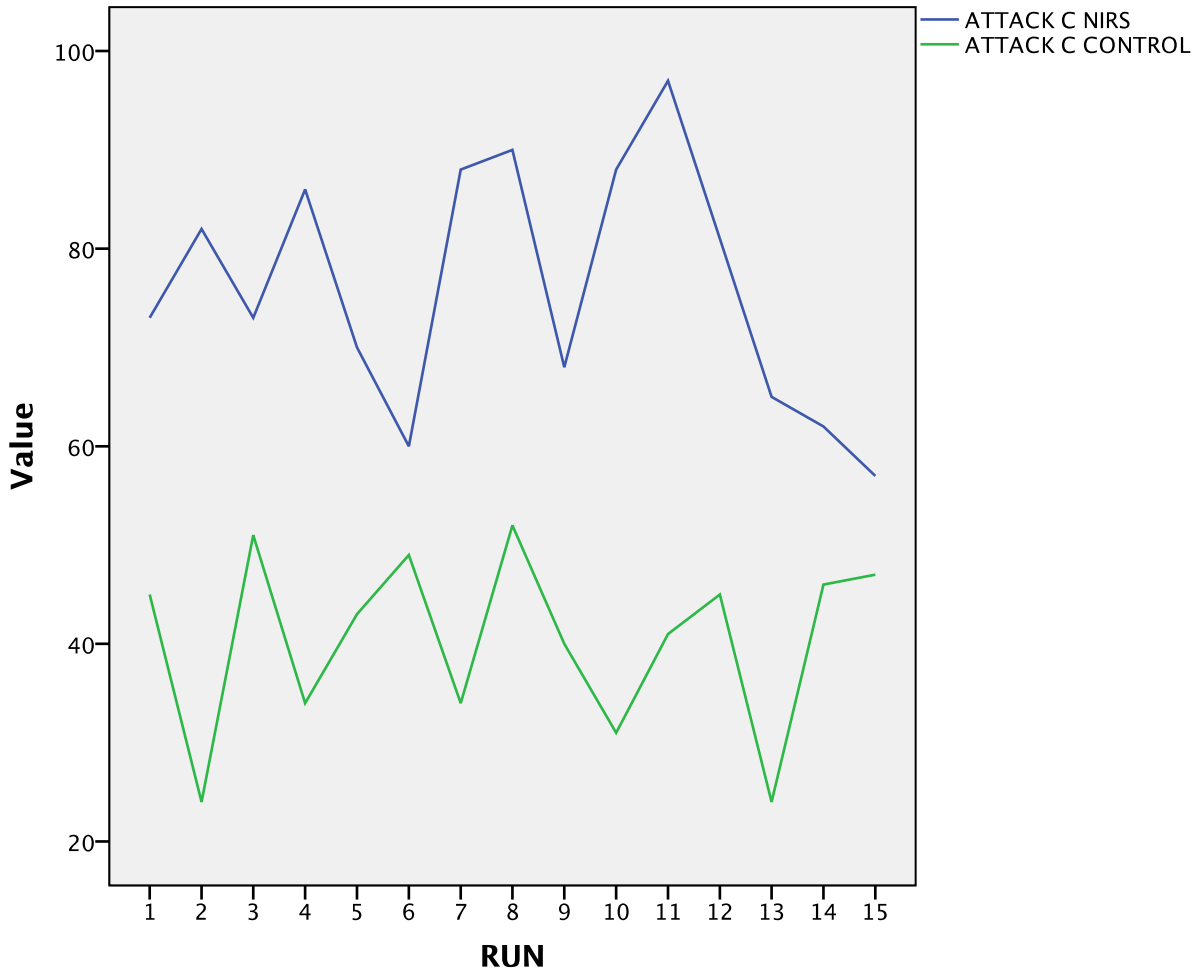


Fig 4. The graph

9. OVERVIEW OF REPORTING AND RESPONSE SYSTEMS

Many attempts have been made to improve on intrusion detection and response system. These efforts have only made ways into the effectiveness of intrusion detection and response system. These efforts have only made ways into the effectiveness of intrusion detection and have not eliminated the requirement for an automated response system. A classification of response functions in other response systems is given in [4]. The response function in detection systems can be categorized as a notification system, manual response system, or automatic response system. According to the authors, most systems today are notification systems. In [5], an Adaptive Agent-Based Intrusion Response System (AAIRS) was proposed. This was the first response system implementing a notion of learning. In his work, the interface relies on human action to update its intrusion detection systems confidence metric. An adaptive intrusion detection system is described in [6]. This system is used together with AAIRS to provide both adaptive detection and response. The response system is relatively advanced. It keeps track of previous alarms and classifies attacks on the basis of whether they are a continuation of an existing incident or whether it is a new attack. Alarms from different intrusion detection systems in the system have different confidence metrics according to previous detection results. The confidence in a suspected incident and nature of the incident affects the course of action taken. In [7], another promising model for automating intrusion response was proposed. The authors suggested a way of approaching the problem of response to network intrusions by constructing dependency trees that model configuration of the network. Another significant work in the area of intrusion response system includes a thorough consideration of some intrusion detection and response cost modeling aspect by [8] They provided a good introduction to modeling costs of intrusion detection and responses. Comprehensive and thorough surveys of 56 intrusion detection systems were carried out in [4]. From his findings, there were no deducted solutions for intrusion response. There were, however, some responses implemented in a variety of intrusion detection systems. Most of the intrusion detection systems were notification and manual response systems, which were not preferable solutions. There were however, some automatic response systems as well, but these were rather insignificant. There is this possibility of having a delay between an alert and human reaction when manual system responds to attack.

10. Reporting and Response

Human beings are incapable of dealing with the speed and amount of information which computers generate. They are also prone to error, misinterpretation, and it is very difficult to accurately predict their capabilities. Computers on the other hand precisely execute what they have been instructed to, deterministic in that the execution of the same sequence of instructions will always produce the same result, and their processing capabilities can be estimated in advance. Reporting is a key phase of intrusion detection, as it is the main point of interaction with computers and humans. The immense amount of data gathered, analyzed, sorted and classified by the intrusion detection system must now be presented to the human administrator. The impact of overwhelming amount of information is delivered to the automated response system with proper analysis and classification. If not handled properly, humans have the tendency to eventually switch off features that are too loud [10]. Many experts choose to do away with automated response and concentrate efforts on optimizing manual response instead. We can view manual response as having somewhat different properties and goals than automated one. We cannot expect the reaction time to be near real-time and we must have an operator who is trained in incident response. To our benefit though, the response, which is tailored to the specific incident, can be followed by in-depth analysis and recovery, and lead to problem eradication. The main limitation of manual response is inevitable delay between an alert and human reaction. An automated exploit script performs its tasks in about 30 seconds. It is simply impossible to achieve this kind of response time from humans regardless of available resources.

11. CONCLUSION AND FURTHER WORK

A model for active response system has been proposed. This follows the implementation of some result based model for the generalization of NIDs in such a way that that the cost of such on the users is minimized. An efficient and secure system or network can be achieved by using appropriate policies and tools like firewalls, anti-virus, intrusion detection and response mechanisms. Nothing can replace the benefits achieved by monitoring and enforcement of restriction policies. Active network monitoring reduces network intrusions. Despite the awareness and availability of various tools, the inability to properly implement the policies has resulted in a number of avoidable casualties. An effective implementation policy based on constant feedback, compliance checks and intricate expertise is a must for effective Network Security. Protection against network intrusions and attacks is a topic that's on every IT professional's mind today. Before you can effectively secure your network, it's important to know and understand the types of attacks to which the network may be venerable.

REFERENCES

[1] B. Foo, Y. S. Wu, Y. C. Mao, S. Bagchi and E. H Spafford (2005). ADEPTS: "Adaptive intrusion response using attack graphs in an e-commerce environment". In proceedings of the International Conference on Dependable Systems and Networks (DSN), pages. 508-5 17.

[2] N. Stakhanova, S. Basu, J. Wong (2005). *"A Taxonomy of Intrusion Response Systems."* Department of Computer Science, Iowa State University Ames, IA 500011 U. S. A

[3] K. Ilgun, R.A. Kemmerer, and P.A. Porras (March 1995). *"State Transition Analysis:* A Rule-Based Intrusion Detection System." IEEE Transactions on Software Engineering, 21(3).

[4] Carver, C. A. and Pooch, U. W. (2000). *"An Intrusion Response Taxonomy and Its Role in Automatic Intrusion Response"*. IEEE Systems, Man and Cybernetics Information Assurance and Security Workshop, 129- 135.

[5] Carver, C. A. (2001). *"Adaptive Agent-Based Intrusion Response."* Ph.D Dissertation, Department of computer science, Texas A & M University, College Station, TX

[6] Ragsdale, D. J., Carver, C. A., Humphries, J.W. and Pooch, U. W. (2000). *"Adaptation Techniques for Intrusion Detection and Response Systems"* IEEE International Conference on Systems, Man and Cybernetics, 4 (2344-2349). http: / /www.itoc.usmaethjgjçale/pubs/adapt.pdf

[7] Y.-S. Wu, B. Foo, Y.-C. Mao, S. Bagchi and E. Spafford (2005). *"Automated Adaptive Intrusion Containment in Systems of Interacting Services."* Elsevier Journal on Computer Networks, Special Issue on "Security through Self-Protecting and Self-Healing Systems", vol. (Inpress), to appear in Spring 2007; available as Purdue ECE TRO5-14

[8] T. Ryutov, C. Neuman, K. Dongho, and Z. Li (2003). *"Integrated access control and intrusion detection for Web Servers."* In the 23rd International Conference on Distributed Computing Systems (ICDCS), pages 394-401.

[9] Wu Thomas (2004). *"An Introduction to Object Oriented Programming with JAVA"S (3rd Ed.)* (pp. 112-115). McGraw-Hill International