Journal of Computing and Information Technology - CIT 15, 2007, 3, 227–235
doi:10.2498/cit.1000961

An Expert System-based Site Security Officer

Adesina Simon Sodiya[1], Olusola Adeniran[1] and Ronke Ikuomola[2]

[1]University of Agriculture, Abeokuta, Nigeria
[2]Federal College of Education, Akoka, Lagos, Nigeria

A Site Security Officer (SSO) who is a network security staff that responds to alarms from an Intrusion Detection System (IDS), is always faced with the critical problem of low response time when the network becomes big. Even a skilled SSO is hard-pressed and less productive when collecting and analyzing IDS output manually as the frequency of intrusion increases. In this work, an Expert System-based SSO (ExSSO) is designed to correct this problem. The design presents an architecture that encodes associated expert rules for responding to different categories of intrusions into its rule-based component. The Intrusion Index (II), which determines the extent of intrusion, is calculated to classify intrusions into three categories namely low, high and very high. The inference engine component utilizes the encoded rules to interpret and respond to intrusions based on the Intrusion Index. Visual Basic 6.0 is used to implement the design because of its interactiveness and high ability to support database. Testing the new design with data from three different network environments, the result shows a system that can investigate and respond to an average of 57 intrusions per minute as against the maximum response time of 2 per three minutes in human-based SSO.

Keywords: intrusion, IDS, network security, SSO, expert system, intrusion index

1. Introduction

Systems and networks are subject to electronic intrusions. Computer intrusion has been a major concern in the computer security field for over two decades and it is certain that the problem is going to be on the increase. Since the problem is everywhere, government agencies and commercial organizations are now implementing various intrusion detection systems that can monitor this computer security breaches.

Intrusion detection systems (IDSs) are security tools that are used to detect traces of malicious activities which are targeted against networks and their resources (Toth and Kruegel, 2002). According to Sodiya et al. (2004), intrusion detection systems are systems that detect internal and external attack on computer systems and undertake some measures to eliminate them. An Intrusion Detection System (IDS) does not only helps the administrators to detect intrusions and limit damages, but also helps to identify the source of attacks, which sometimes acts as a deterrent, especially in case of insider attacks (Wang et al., 2006). Once intrusion detection systems have obtained information about the event, they analyzed it to find signs of intrusion. Intrusion Response Systems (IRSs) take over after signs of an intrusion are identified and either record the attack or attempt to actively counter it. Although IRSs are tightly coupled with the ID systems themselves and also important in defending against threats, not much research effort has been put into their study. Therefore, intrusion response, in most cases, remains a manual process, which has to be performed by the Site Security Officer (SSO).

Every intrusion detection system needs to communicate with the outside world. This may be done passively by notifying the SSO or actively by trying to hinder the intruder or striking back. It is often not possible to use active response since the IDSs generate too many false alarms and the response would hit innocent users or hosts. The only possible action is then to report what has happened to the SSO and perhaps provide him with the means to investigate the event further.

Toth and Kruegel (2002) mentioned that the current intrusion response systems can be divided into notification, manual response and

automatic response systems. Majority of IRSs operate as notification systems, which means that they simply display or forward output delivered by the IDS (e.g. incident data) to the SSO. Usually, urgent notification is realized via e-mail or text message services over a mobile phone. Manual IRS allows the SSO to manually launch countermeasures against a detected intrusion by choosing from a predetermined set of response mechanisms.

With automated response, SSO gives the system power to take action upon detecting an intrusion attempt or an intrusion. The techniques range from increasing response delays in TCP handshakes, to blocking IPs, to resetting connections, and removing routes. Ability of an IDS to automatically stop intrusions seems too good to be true, and, in fact, many experts see it as a sure way to shoot oneself in the foot. A typical scenario might be a spoofed attack with the source address pointing at the DNS server. An automated system responding to an attack may block access to the DNS server, hence disabling the entire network (Dobruki, 2003).

The two categories listed above are not proactive in countering an intrusion. Even when signs of an intrusion have been detected, countermeasures are not triggered automatically and defending the network remains the task of the SSO. This opens a time window of vulnerability between the time when the intrusion has been detected and the time when the first countermeasure is launched. Also, it is sometimes difficult for the SSO to collect each day the output from the detection system for signs of intrusion and respond to them quickly in a case where the number and frequency of intrusion increases rapidly. This makes the function of an SSO important and complex in intrusion detection. This leads to the objective of this work, which is to remove the human involvement in IDS response and reporting by building an Expert system-based Site Security Officer (ExSSO). It is hoped that this will help the organization in adopting a kind of site security policy that will protect their database fast enough without any downtime delay.

2. Literature Review

Many attempts have been made over the past ten years to improve on intrusion detection and response system. These efforts have only made ways into the effectiveness of intrusion detection and have not eliminated the requirement for human expert intervention.

A classification of response functions in other IDSs is given in Carver et al. (2000). The response function in a detection system 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 Carver (2001), an Adaptive Agent-based Intrusion Response System (AAIRS) was proposed for intrusion response. This was the first response system implementing a notion of learning. In this work, the interface relies on human action to update its IDS confidence metric.

An adaptive intrusion detection system is described in Ragsdale et al. (2000). 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 it is a new attack. Alarms from different IDS agents in the system have different confidence metrics according to previous detection results. The confidence in a suspected incident and the nature of the incident affects the course of action taken.

A study in Toth et al. (2002) proposed yet another promising model for automating intrusion response. 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 IRS includes a thorough consideration of some intrusion detection and response cost modeling aspect by Lee et al. (2002). They provided a good introduction to modeling costs in intrusion detection and responses.

Carver (2000) did a comprehensive and thorough survey to investigate 56 IDSs. From his findings, there were no deducted solutions for intrusion response. There were, however, some responses implemented in a variety of IDSs. Most of the IDSs were notification and manual response systems, which are not preferable solutions. There were, however, some automatic response systems as well, but these were rather

immature. There is this possibility of having a delay between an alert and human reaction when manual system responds to attack and also an automated system responding to an attack may likely block access to the DNS server, hence disabling the entire network.

In his survey of ten different IDSs, Fisch asserts that most of the existing IDSs had the ability to generate daily or weekly reports of suspicion events or users (Fisch, 1996). The focus of his research work is not on the user interface for reporting and notifying the Site Security Officer, but, instead, relies on automatic response. It probably had some notification capabilities, although these are not described in detail. From the literature, so far no work has fully implemented completely automated intrusions analysis and response.

3. Architecture for Expert System-based Site Security Officer (ExSSO)

The main task of intrusion detection system (IDS) is to monitor the events occurring in the network and then analyze them for signs of intrusion. Once an intrusion has been detected, IDS issues alert notifying the SSO, who then responds to the IDS alarm to make appropriate measure. The components of ExSSO is presented in Figure 1.

3.1. Inference Engine

Inference Engine (IE) is the central processing unit of an expert system. It uses knowledge base to draw conclusions for each situation. Inference engine can be thought of as the system software which stimulates a situation and actually makes the decision.

ExSSO inference engine uses two major components in making decision. These are intrusion index and interpreter.

Intrusion Index: It is used to measure the extent or gravity of attack/intrusion so as to determine the action to be taken when an intrusion is detected.

The intrusion index is then calculated as follows:

$$\text{SSO Intrusion Index (II)} = \frac{\sum_{i=1}^{n} X_i}{\sum_{i=1}^{n} X_{i\ \max()}},$$

where: $n = 3$ (because there are three variables considered).

The variables considered for the calculation of II are:

a)

Attack Category	Score
Confidentiality	1
Integrity	2
Availability	3

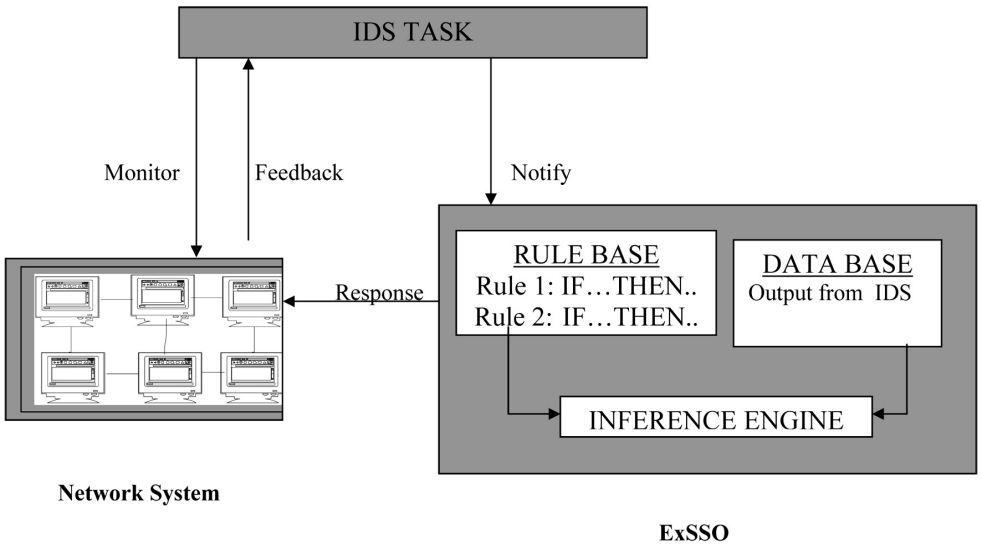

Figure 1. Architecture of an Expert system-based Site Security Officer (ExSSO).

b) **Attack Implication**: this has to do with the possible consequence of the attack. It is divided into:

Low	1
High	2
Very High	3

c) **Security Violation Level**: this has to do with the depth of the security violation. It can also be seen as the extent of circumvention.

Security Violation Level	Score
Low	1
High	2
Very High	3

X = variables score
$X_{i\ \max()}$ = represents maximum score obtainable for variable i

The maximum value of SSO Intrusion is 1 (SSO Intrusion Index (II) <= 1). This is then divided into 3 ratings as follows:

Intrusion Index is low when	$0 \leq$ II <0.3
Intrusion Index is high when	$0.3 \leq$ II <0.7
Intrusion Index is very high when	$0.7 \leq$ II <1

This means that if the calculated index is between 0 and 0.3, then II is low, and so on.

Interpreter: The inference engine contains interpreter that decides how to apply rules to infer new knowledge. The interpreter scans the condition parts of each rule until one is found that can be fired. Then the next cycle starts, and the interpreter tries to find another rule that can be fired. The execution ends if no rules are applicable.

3.2. Rule Base

The major actions that are invoked when an intrusion is found are classified into three:

Deflect: Send a warning message to the suspected intruder and then store information of the suspected intrusion in the database. If the same kind of suspicion is found in that terminal again, the action will turn to prevent.

Prevent: Prevent the user from the network resources.

Preempt: Strikes offensively against the likelihood of a particular intrusion occurring later.

The rule base consists of the following set of rules:

R1: IF II is low THEN Deflect
R2: IF II is high THEN Prevent
R3: IF II is very high THEN Preempt

Deflect Algorithm

```
Attack_flag = .F.
Check S  /* S represents the network system */
If Attack_flag = .T.
   Scan database
If Terminal = N    /* N represents the terminal
where the attack is coming from */
Check Sup_dat
If found()
   Prevent()
else
   enable audit()
   Notify user    /* by sending an e-mail warning message */
 Endif
Endif
```

Function audit()

```
Scan audit database
For Terminal = N
Enable reporting-window    /* Display all the
activities occurring in the user's terminal */
Add to Sup_dat    /* where Sup_dat is the
database keeping records of intrusion */
End
```

Prevent Algorithm

```
Attack_flag = .F.
      Check S  /* S represents the network
system */
If Attack_flag = .T.
   Scan database
If Terminal = N    /* N represents the terminal
where the attack is coming from */
Prevention ()    /* Prevent the users from certain resources */
Endif
Endif Function Prevent()
Disconnect N    /* Disconnect terminal N from
Network resources */
End
```

Preempt Algorithm

```
Attack_flag = .F.
```

```
   Check S  /* S represents the network system
*/
   If Attack_flag = .T.
      Scan database
If Terminal = N     /* N represents the terminal
where the attack is coming from */
Close terminal     /* Disable and shut down the
user's terminal */
      Endif
Endif
```

3.3. ExSSO Organization

ExSSO is not an independent IDS component and it is designed to process IDS output immediately. Since intrusion detection activities are real-time events, ExSSO collects data (IDS alarms) continuously from IDS and responds to these alarms. The Intrusion Index calculated determines the nature of the response (Deflect, Prevent and Preempt).

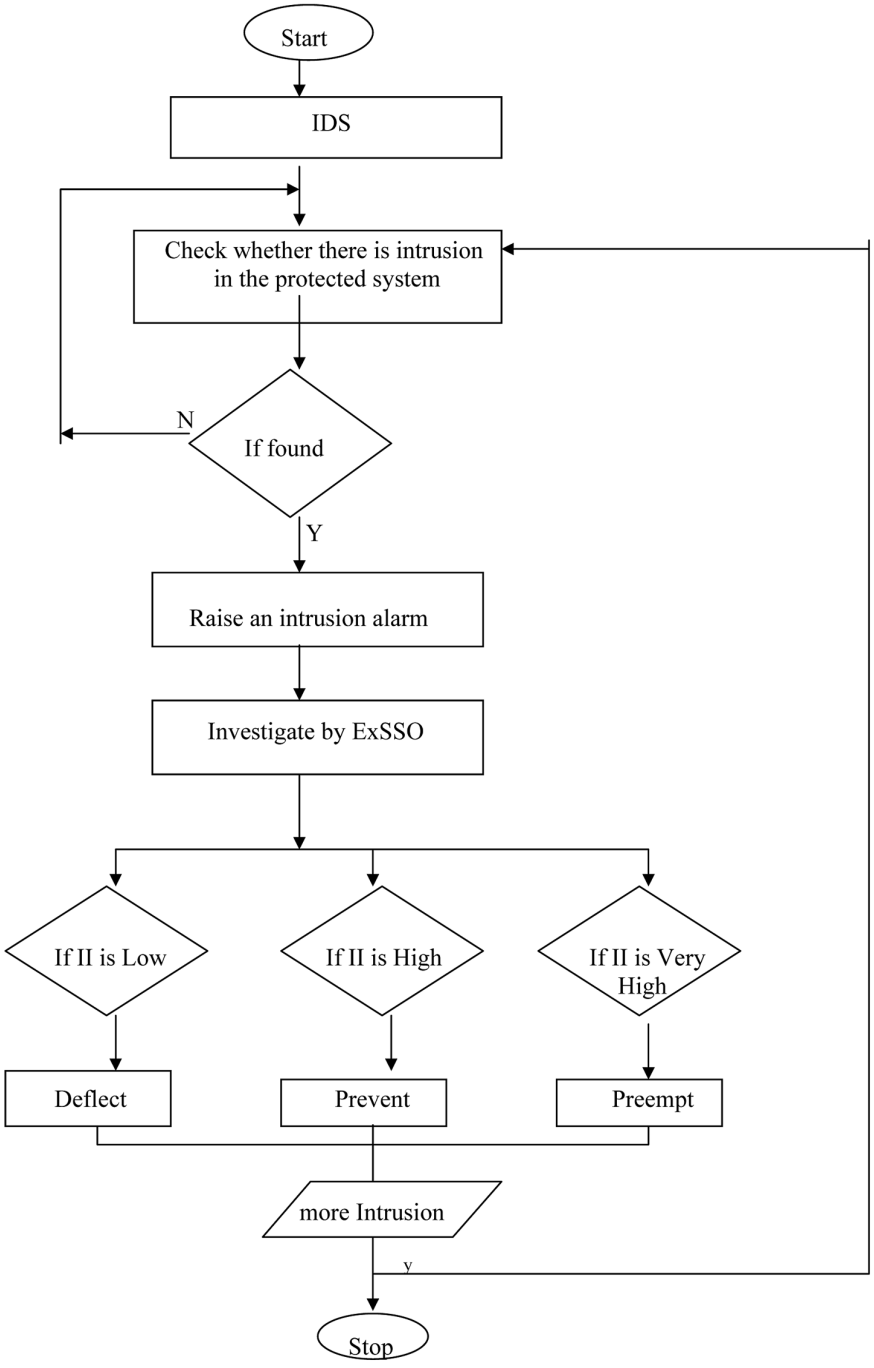

Figure 2. A representation of ExSSO inference engine.

4. Implementation Procedure

4.1. Programming Language

Visual Basic 6.0 was used as the programming language for the implementation of the design. The design can run under the Window XP and uses the output (alarm) generated by the intrusion detection system as its input.

4.2. Data Source

The data source to our design is the output in form of alarm from an IDS. We got the data used for the implementation from a network organization in Lagos State, Nigeria. The output of the IDS contains the following information: time & date of attack, source and destination IP address and port number, security violation level, attack implication, attack types, and attack categories.

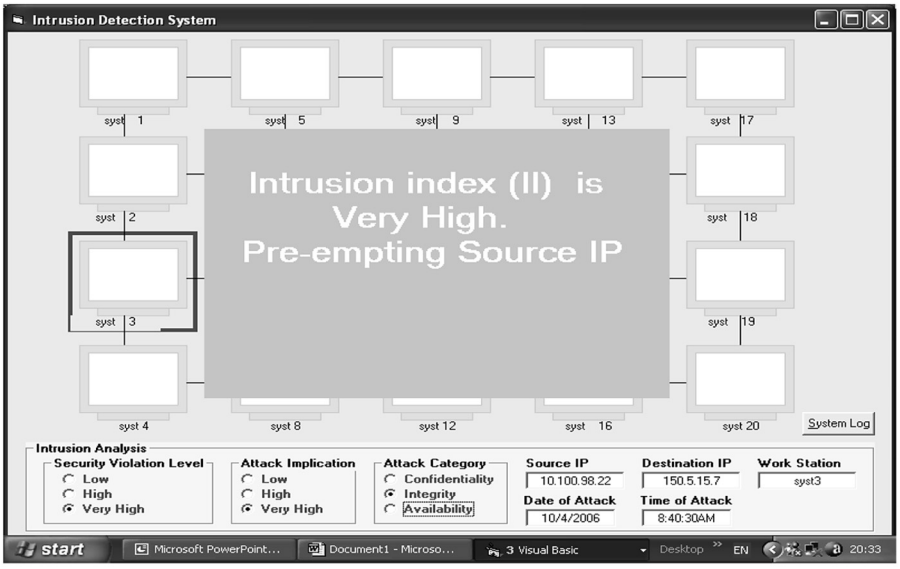

Figure 3a.

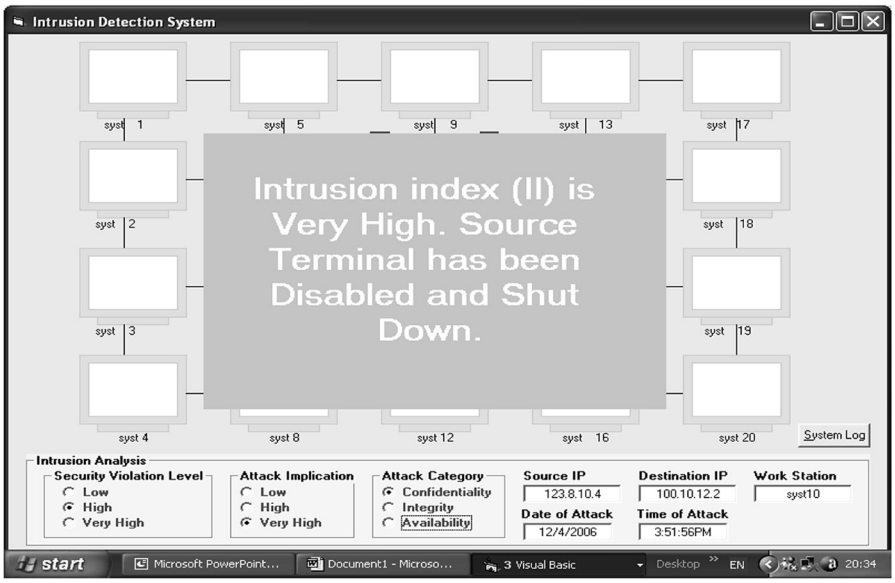

Figure 3b.

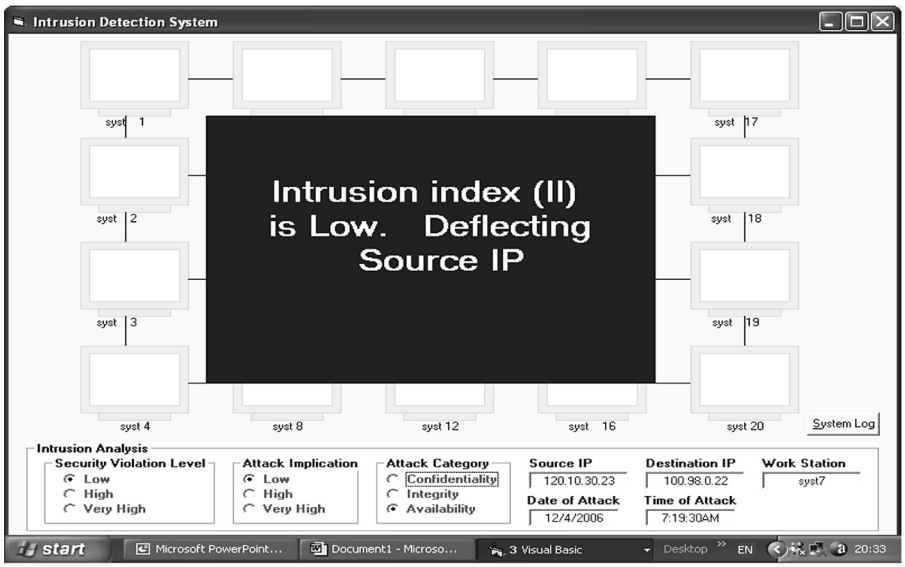

Figure 3c.

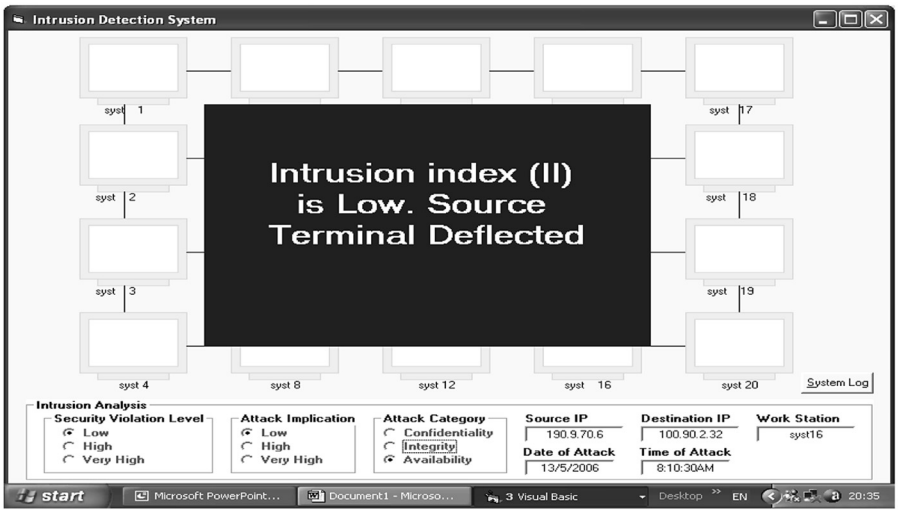

Figure 3d.

Figure 3 (a – d): Interface design of an ExSSO.

4.3. Interface design

The boxes on the ExSSO interface screen represent the 20 systems on the network (Figure 3a). The flashing red signal on system 3 is an alert signal indicating that system 3 has been attacked by an intruder.

The design gives detailed information about the intruder, the level of attack and the attack system. The ExSSO design output contains the following intrusion analysis :

— **Security Violation Level**: is the attack level given the strength of suspicion (the attack level can fall within the range of low, high and very high).

— **Attack Implication**: is the type of loss experienced as a result of the attack which is graded as low, high and very high.

— **Attack Categories**: are attacks under the three computer security properties which are: attack due to confidentiality, integrity and availability.

— **Source IP address**: the IP address of the system where the attack is coming from.
— **Destination IP address**: the IP address of the attacked system.
— **Time**: identification of time of operation.
— **Date**: identification of date of operation.

5. Evaluation of ExSSO

5.1. Summary of the Intrusive Event

In this work, the running and testing of the design was carried out and the summary of the intrusive event is shown in the table below.

Attack Level	Incident Level_N1	Incident Level_N2	Incident Level_N3
Very High	27	10	20
High	64	55	50
Low	33	18	30

Table 1. Summary of the intrusive event level.

It was discovered that the intrusive event that occurred on the network during the experiment, categorized as very high, high and low has an incident level of 27, 64 and 33 on network 1, an incident level of 20, 50 and 30 on network 2 and an incident level of 10, 55 and 15 on network 3.

Therefore, we can say that a high level attack has the highest incident level on the network. As a result, most of the users are prevented from certain network resources.

5.2. Summary of the Attack Categories

Attack Categories	Incident Level_N1	Incident Level_N2	Incident Level_N3
Availability	14	10	16
Confidentiality	35	28	20
Integrity	75	45	64

Table 2. Attack categories.

Attack categories under integrity violation allowing the attacker to change the system state of data residing on or passing through a system have the highest incident level of 75 on network 1, an incident level of 45 on network 2 and an incident level of 64 on network 3. The attacks that fall under these categories are determent to the state of the network system.

5.3. Summary of the Attack Types

Attack Types	Incident Level_N1	Incident Level_N2	Incident Level_N3
Penetration	76	60	70
Scanning	34	21	20
Dos	14	2	10

Table 3. Attack types.

It was discovered that most of the attacks launched on the network are penetration attacks, whereby intruders gain control of the system by exploiting a variety of software flaws. The penetration attacks have an incident level of 74 out of the 124 attacks on network 1, an incident level of 60 out of 83 on network 2 and an incident level of 70 out of 100 attacks on network 3.

5.4. Summary of the Response Time

Network	No. of Attacks	Time taken for response (minutes)	Response time per attack (seconds)
1	124	2.18	1.055
2	83	1.50	1.080
3	100	1.70	1.020
	Average response time		**1.050**

Table 4. Average response time.

Consequently, the number of intrusions that the systems can respond to in a minute is equal to $60/1.05 = 57.14$, approximately 57 intrusions.

6. Conclusion and Future Work

Since human beings are incapable of dealing with the speed and amount of information generated by intrusion detection systems, this has been successfully carried out to eliminate the problems with human intervention in IDS. We designed an ExSSO algorithm for responding to attacks in the intrusion database.

The direct application of this problem in the network environment warrants the significance of this research effort. Experimental evaluations bring promises to this area. Our algorithms successfully respond to malicious attacks on the network. It might also be necessary to design an agent-based SSO that will move around the network and respond to intrusion alarm.

References

[1] C. A. CARVER, Intrusion Response Systems: A survey. Department of Computer Science, Texas A & M University, College Station, TX, (2000).

[2] C. A. CARVER, U. W. POOCH, An Intrusion Response Taxonomy and Its Role in Automatic Intrusion Response. *IEEE Systems*, Man and Cybernetics Information Assurance and Security Workshop, (2000) pp. 129–135.

[3] C. A. CARVER, Adaptive Agent-based Intrusion Response. *Ph. D. Dissertation*, (2001), Department of Computer Science, Texas A & M University, College Station, TX.

[4] M. DOBRUCKI, Priorities in the deployment of network intrusion detection systems. *Master Thesis*, (2002), Helsinki University of Technology, Department of Computer Science and Engineering.

[5] E. A. FISCH, A Taxonomy and Implementation of Automated Responses to Intrusive Behavior. *Ph. D. Thesis*, (1996), Texas A & M University, College Station, Texas.

[6] W. LEE, W. FAN, M. MILLER, S. STOLFO, E. ZADOK, Toward Cost-sensitive Modeling for Intrusion. it Journal of Computer Security, Vol. 10, No. 1 – 2, (2002).

[7] D. J. RAGSDALE, C. A. CARVER, J. W. HUMPHRIES, U. W. POOCH, Adaptation Techniques for Intrusion Detection and Intrusion Response Systems. *IEEE International Conference on Systems, Man, and Cybernetics*, (2000), 4, pp. 2344–2349. `http://www.itoc.usma.edu/ragsdale/pubs/adapt.pdf`

[8] A. S. SODIYA, H. O. D. LONGE, A. T. AKINWALE, A New Two-tiered Strategy to Intrusion Detection. *Information Management and Computer Security*, 12(1), (2004).

[9] T. TOTH, C. KRUEGEL, Evaluating the Impact of Automated Intrusion Response Mechanisms. *Proc. of the 18th Annual Computer Security Applications Conference*, (2002).

[10] Y. WANG, S. R. BEHERA, J. WONG, G. HELMER, V. HONAVAR, L. MILLER, R. LUTZ, M. SLAGELL, Towards the Automatic Generation of Mobile Agents for Distributed Intrusion Detection System. *The Journal of Systems and Software*, (2006), Department of Computer Science, Iowa State University, United States. `www.elsevier.com/locate/jss`

Received: September, 2006
Revised: February, 2007
Accepted: May, 2007

Contact addresses:
Adesina Simon Sodiya
University of Agriculture
Abeokuta, Nigeria
e-mail: `sinaronke@yahoo.co.uk`

Olusola Adeniran
University of Agriculture
Abeokuta, Nigeria
e-mail: `ojadeniran@yahoo.com`

Ronke Ikuomola
Federal College of Education
Akoka, Lagos, Nigeria
e-mail: `deronikng@yahoo.com`

DR. A. S. SODIYA is a lecturer of computer science, University of Agriculture, Abeokuta, Nigeria. His research interests are computer security, artificial intelligence, network management, and information systems. He has published his papers in both local and international journals.

OLUSOLA ADENIRAN is currently a Senior Lecturer at the Department of Mathematics, University of Agriculture, Abeokuta, Nigeria. His area of specialization is Loop Theory – an area which has been found to have a lot of applications in coding and cryptology. He has been involved in many projects in computer science.

RONKE IKUOMOLA is a lecturer of computer science at Federal College of Education, Akoka, Lagos, Nigeria. She has just been awarded M. Sc. degree in computer science. Her areas of interest are network security and data mining.