Host-based Intrusion Detection System using Sequence of System Calls

P. Ramprakash¹, M. Sakhivadivel², N. Krishnaraj³, J. Ramprasath⁴

¹²³⁴Assistant Professor, Department of Information Technology, Dr. Mahalingam College of Engineering and Technology, Pollachi, INDIA

ABSTRACT

Intrusion Detection is the powerful tool to protect the networks and system from attacks. Intrusion detection system is divided into host based intrusion detection system and network based intrusion detection system. Network based intrusion systems are primarily used to detect the unwanted and abnormal behavior activates in the network. Host based intrusion detection systems are used for monitoring and detecting attacks in the system side. In this paper we use host based intrusion detection methods and data mining techniques to detect the misbehavior and unknown attacks based on the system calls. Here we use sendmail processor’s system call dataset for detecting the abnormal behaviour of system calls. Normal and intrusive behaviour are collected and gather the system calls sequences from sendmail process. Data mining play the important role to detect the intrusion. We use classification rule mining techniques for detecting the intrusion in system calls. From the data set using RIPPER rule the normal behaviour characterises of the system call are extracted. Using normal behaviour constructs the profile and compares the sequence of system call to detect the anomaly sequence system calls. The proposed approach gives good performance and reduces the time complexity and false alarm rates.

Keywords: System Call, Anomaly Detection, HIDS, data mining, RIPPER

I. INTRODUCTION

Apart from anti viruses, firewalls and additional security devices are used to give the complete security, Intrusion Detection Systems are also essential. Some of the software defensiveness such as buffer over flow in UNIX, bugs in Microsoft internet explorer and operating system still are inconveniencing the users of the present software. When an intruder penetrates the firewall or the preventing security facilities, IDS can come in handy. Hence, Intrusion Detection Systems (IDS) can continuously improve the security in our operating system and other software.

Intrusion Detection System (IDS) constitute a fundamental element in Communication Company’s infrastructure. Along with the firewalls, the IDS represent the main security tools of the network: firewalls for the external threats and IDS as much for the external ones as for the internal ones to the organization. Although during a period of time the firewalls only reduces the exhibition of the system, reason why is important to have a monitoring and detecting system.

Intrusion Detection system’s main securities functions are monitor, detects, and responds to abnormal activity by organisations internal and external intrusion. Intrusion Detection systems use some rules to define the normal behaviour and alarm will be raised if that event is intruder. Certain Intrusion Detection systems have the ability of sending out alerts, so that the administrator of the IDS will get a warning of a possible security happening in the form of a page, email, or SNMP trap. Many intrusion detection systems recognize extracting incident as well as issue proper alert, they also react automatically to the event. Such a reaction might take in logging off a user, disabling a user account, and launching of scripts.

Intrusion Detection is the insistent active challenges in understanding or detecting the occurrence of intrusive activities. It refers to all processes used in discovering illegal uses of network or computer devices. This is accomplished through exclusively designed software with a sole principle of detecting unusual or abnormal activity. Such software is called Intrusion Detection System.

Types of intrusion detection system

There are two type of Intrusion Detection systems that employ one or both of the intrusion methods outlined above. Host-based systems base their results on information attained from a single host (generally audit trails), even as network-based intrusion detection systems gain the data by supervising the traffic in the network to which the hosts are connected.

1) Host-based intrusion detection

A generic intrusion detection model proposed by Denning (1987) is a rule based pattern matching system in which the intrusion detection tasks are conducted by checking the similarity between the current audit record and the corresponding profiles. If the current audit record deviates from the normal patterns, it will be considered an anomaly. Several IDSs were developed using profile and rule based approaches to identify intrusion activity.

2) Network-based intrusion detection

With the proliferation of the computer networks, more and more individual hosts are connected into local area networks, and/or wide area networks. However, the hosts, as well as the networks are exposed to the intrusion due to the vulnerabilities of network devices and network protocols. The TCP/IP protocol can be also exploited by the network intrusions such as IP spoofing, port scanning, and so on. Therefore, network-based intrusion detection has become
important and is designed to protect a computer network as well as of its hosts. The installation of a network-based intrusion detection system can also decrease the burden of the intrusion detection task on every individual host.

B. Intrusion Detection techniques

The signatures of some attacks are known, where as other attacks only reflect some deviation from normal patterns. Consequently, two main approaches have been devised to detect intruders.

1) Anomaly Detection

Anomaly Detection assumes that an intrusion will always reflect some deviations from normal patterns. Anomaly Detection may be divided into static and dynamic detection. A static anomaly detector depends on the statement that the part of the system which is monitored will not change. Generally speaking, static detectors will deal only with the software part of a system that is being watched. A system component, however, may change its behavior as normal. Therefore static anomaly detectors focus on integrity checking. Dynamic detection typically operates on audit records or on monitored networked traffic data. Audit records of operating systems do not record all events that is recorded in the audit will be observed and these events may occur in sequence. Partial ordering of events is adequate for detection in distributed system. When we consider with other cases, the order is not straightly represented. It is represented only in the form of cumulative information. Cumulative information is nothing but the cumulative processor resources are being utilized during a time interval, is retained. In such cases, dividing the nominal resource consumption from anomalous resource consumption can be handled by thresholds.

2) Misuse Detection

The facts of system susceptibilities and identified attack patterns determine the misuse detection. Misuse detection is mainly concerned with detecting intruders who are trying to enter into a system by utilizing little notorious vulnerability. If possible, a system security administrator should take care of the system by taking care with known vulnerabilities and tries to remove it. The word intrusion scenario explains the known type of intrusion. That is, this intrusion is a series of events which causes an intrusion without some external defensive involvement. Repeatedly, this intrusion detection system compares current action to identified intrusion scenarios which make sure that one or more attackers are not trying to make use of known vulnerabilities. To do this, modelled intrusion scenarios must be described. The major differentiation among misuse systems are the way of modelling they used for intrusion. The original misuse detection systems used rules to describe events indicative or intrusion activities that a security administrator looked for within the system. Large number of rules can be difficult to analyse. Otherwise rules are not grouped by intrusion scenarios and therefore making modification to the rule set can be difficulty as the affected rules are spread out across the rule set. To overcome these difficulties, new rule organization use state transition diagram. Misuse detection system utilizes the rule to monitor for events that probably fit in to an intrusion detection scenario. Monitoring system calls are being used for monitoring live events or later using audit records.

3) Advantages and Disadvantages of anomaly detection and misuse detection

The main disadvantage of misuse detection approaches is that they will detect only the attacks for which they are trained to detect. Novel attacks or unknown attacks or even variants of common attacks often go undetected. At a time when new security vulnerabilities in software are discovered and exploited every day, the reactive approach embodied by misuse detection methods is not feasible for defeating malicious attacks. The main advantage of anomaly detection approaches is the ability to detect novel attacks or unknown attacks against software systems, variants of known attacks, and deviations of normal usage of programs regardless of whether the source is a privileged internal user or an unauthorized external user. The disadvantage of the anomaly detection approach is that well-known attacks may not be detected, particularly if they fit the established profile of the user. Once detected, it is then difficult to characterize the nature of the attack for the forensic purposes. Another drawback of many anomaly detection approaches is that a malicious user who knows that he or she is being profiled can change the profile slowly over time to essentially train the anomaly detection system to learn the attacker’s malicious behavior as normal. Finally a high false positive rate may result for a narrowly trained detection algorithm, or conversely, a high false negative rate may result for a broadly trained anomaly detection approach.

II. RELATED WORK

The Computer Immunology Project at the University of New Mexico (Warrender et al., 1999; Forrest et al., 1997, 1996) explored designs of IDS based on immunology. Small, individual agents would roam on a distributed system, identify intrusions, and resolve the intrusions. One portion of the project developed a sense of self for security related computer programs by observing the normal sets of system calls executed by the programs. This sense of self can be used to detect intrusions by discovering when a program executes an unusual set of system calls. The Computer Immunology Project differs from our system in their focus on individual agents rather than an integrated system of cooperating Multi-agents.

1996, Forrest et al [1] is proposed a model for detecting intrusion at sequence of system calls given by privilege process. She traced sendmail process of UNIX for collecting sequence of system calls in normal mode. She also collected instructive behavior of instructive process in unsafe mode.
Forrest introduces a method for constructing the normal behavior by collecting shot sequence of system calls in a running process. Shot sequence may be 5, 7, 9 and 11. Once normal behavior database constructed then scan the new traces for abnormal behaviors and also looking for sequence system calls that are not presented in the normal data base. After normal and abnormal behavior collected it can be analyzed to detect the intrusion.

Later work by Warrender et al [3] extended this technique in sequence time-delay embedding (stide), which memorized all contiguous sequences of predetermined, fixed lengths during training. An anomaly count was defined as the number of mismatches in a temporally local region. A threshold was set for the anomaly score above which a sequence is flagged anomalous, indicating a possible attack. A.Hafmer , Martin and Lee continued the Forrest work

Y.Li [4] and others are proposed a model by using Hidden markov model (HMM). They are traces the shot sequence of system call collected and analysis the normal and abnormal behavior by Hidden markov model. From the past research work on the system call area normal behavior is gathering from the normal data set and they will be able to detect the new sequence of system calls of the new process that may be normal or abnormal.

III. PROPOSED SYSTEM

We propose an anomaly intrusion detection system that is implemented via RIPPER rule, data sets are collected from university of new Mexico. These data sets have the normal and abnormal sequence of system calls. In the dat a set all the sequence of system calls have been traced with s-trace command in Linux environment. From the sequence of system call, we build the normal behavioral database and to detect abnormal behavioral from the normal database. Normal sequence of system calls data sets is an input for our system. The following steps show how to detect the abnormal behaviors in system calls.

1. Collecting the normal sequence of system calls from sendmail process in safe mode.
2. Collecting abnormal behavior sequence of system calls while the instructive process is running.
3. From these normal sequence of system calls, preprocessing the data and build the normal database.
4. Analyze the normal database; extract the rule using data mining concepts.
5. Compare the new sequence of system calls with normal data set and detect the mismatch sequence.
6. Analysis of the mismatch sequence is carried out if the mismatch is included, then an alarm will be produced. Otherwise it will be of normal sequence.

IV. SYSTEM ARCHITECTURE AND IMPLEMENTATION

Following steps should be carried out to detect intrusion.

1. In the normal trace and abnormal trace step all of the normal system calls and abnormal system calls are collected. We have used UNM data set for this step.
2. In the preprocessing step using window size to reduce the system call sequence into small size sequence.
3. In the data mining step use the RIPPER rule method of data mining and with all of the normal rules are extracted.
4. Compare the new sequence with normal data base and find the number of mismatches.
5. Using fuzzy system analysis the mismatch veris normal or abnormal.

![System Architecture](image)

4.1. Normal trace and abnormal trace

![Sequence of system call form sendmail processor](image)

System can be monitored at various levels. Various things including ability, cost, and accuracy to make distinguish from normal and abnormal behaviour . Typically, intrusion detection systems monitor either user behaviour or privileged processes.

The privileged process is observed through ‘system calls’ that the Unix process uses to access system resources.
Hofmeyr et al. (1998) find out that short sequences of system calls are a good discriminator for several types of intrusion.

In the normal trace step all the normal system calls are collected. For abnormal trace instructive system call sequence are collected. Here we used UNM data set for normal and abnormal trace. UNM data set sequence of system calls are collected in the following processes, stide named, xlock, ps, login in Unix. All these processes are executed several times and collected the system call sequence stored in a separate file. The each trace system call sequence size widely vary from ten to thousand calls. Normal behavior data sets are login normal, ps normal, MIT live ipr UNM live ipr send mail. During the period of collecting these traces, there are no intrusions or any suspicious activities happening. Instructive processes are iprcp, buffer overflow sun sendmailcp, sm5x, sm565a. The abnormal traces contain several traces including intrusions that exploit well-known problems in Unix systems. For example, Sunsendmailcp (SSCP) is scripts that sendmail uses to append an email message to a file, but when used on a file such as /rhosts, a local user may obtain root access. Syslog attack uses the syslog interface to overflow a buffer in send mail. Forwarding loops occur in send mail when a set of files in $home/.forward form a logical circle. In our study, intrusion traces include five error conditions of forwarding loops, three sunsendmailcp (sscp) attacks, two traces of syslog-remote attacks, and two traces of unsuccessful intrusion attempts-sm565a. Detailed descriptions of these intrusions can be found in Hofmeyr et al. (1998). Each trace has two attributes: the first one is the process ID, indicating the process the system call belongs to; and the second one is the system call value. Instructive process does not follow the normal sequence. We can compare the new sequence of system calls with the normal database and to detect the mismatches[5][6].

4.2. Preprocessing the data set

Once we collected the sequence of system calls from the running privilege process the next step to be data preprocessing. Collected system call information are raw data type, applying some preprocessing techniques on raw information and change the data set into processing dataset. Fig 3 show the sendmail system call trace, in the first column Represent the currently traced processor number and the processor number followed by system call with have list of the arguments and return values. All these system call names are replaced by unique number. For example 6 is open, 7 is close 72 is mmap and so on. This is done for reduce the complexity of data, easy to access it and format to sequence of numbers. Since the system call sequence numbers are very large for any particular processor, we cannot compare those sequences In order to do that we use sliding window size.

Fig 3 Normal system call traced

To construction of normal data base the slide window size is 3. From the following system call sequence we can build the normal behavior data base

Open, read, mmap, open, read, close

For this sequence, we will record the possible sequence of system call in position 1, position 2 and position 3 shown in Table 1. The pairs are generated based on a window size that can be chosen by the environment constraints.
TABLE 1

<table>
<thead>
<tr>
<th>Current</th>
<th>Position 1</th>
<th>Position 2</th>
<th>Position 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>read</td>
<td>mmap</td>
<td>mmap</td>
</tr>
<tr>
<td>Read</td>
<td>mmap</td>
<td>mmap</td>
<td></td>
</tr>
<tr>
<td>Mmap</td>
<td>mmap</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>mmap</td>
<td>mmap</td>
<td>open</td>
</tr>
<tr>
<td>Mmap</td>
<td>mmap</td>
<td>open</td>
<td></td>
</tr>
<tr>
<td>Open</td>
<td>read</td>
<td>close</td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>close</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Analysis the data set we founded that some system calls occur more than once and followed by different system call. In the sequence read system call is occur two times and followed by different system calls are mmap and close. So we record all the system calls, then expanded database for different sequence after the sliding windows complete. Table 2 show is the expanded format.

TABLE 2

<table>
<thead>
<tr>
<th>Current</th>
<th>Position 0</th>
<th>Position 1</th>
<th>Position 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Open</td>
<td>read</td>
<td>mmap, close</td>
<td>mmap</td>
</tr>
<tr>
<td>Read</td>
<td>mmap, close</td>
<td>mmap</td>
<td>Open</td>
</tr>
<tr>
<td>Mmap</td>
<td>mmap, open</td>
<td>open, read</td>
<td>close, read</td>
</tr>
</tbody>
</table>

From this sliding window we can produce many system call sequence and stored in database file. Once we build the preprocessed data base from raw data we can easily extract the normal behavior rule form these data set. After finishing this step missing system call number is replaced with -1 and stored in the data base.

4.3. RIPPER Rule

RIPPER is a rule-based learner that builds a set of rules that identify the classes while minimizing the amount of error. The error is defined by the number of training examples misclassified by the rules[7][8]. Here RIPPER rule is used for extract the rule from pre processed data base and build the normal behavior database.

Ripper rule is If then rule format. Here we define how to build the set of rules for normal and abnormal sequence of system calls. Ripper rule describe the set of objects, then the objects should be classified by if then rule from the classification group in to classes. The Ripper rule is:

In the class A IF the third position attribute is M and seventh position attributes is N then tenth position attributes P, it is true
In the class B fifth position attributes is Q. Then it is true
In the example class A is chosen only attitude 3 and 7 value are M and N respectively. Class B is chosen when the attribute5 value is Q. In the Ripper rule we also define the condition the attribute not equal to some value. Sometimes we can take multiple conditions and define the normal behaviours

RIPPER outputs a set of if-then rules for the “minority” classes, and a default “true” rule for the remaining class. The following exemplar RIPPER rules were generated from the system call data:

normal:- p2 = 104, p7 = 112. [meaning: if p2 is 104 (vtimes) and p7 is 112 (vtrace) then the sequence is “normal”]

normal:- p6 = 19 , p7 = 105 . [meaning: if p6 is 19 (lseek) and p7 is 105 (sigvec) then the sequence is “normal”]

... abnormal:- true. [meaning: if none of the above, the sequence is “mismatch”]

These RIPPER rules used for to estimate whether a sequence is “mismatch” or “normal”. But what the intrusion detection system needs to know is whether the trace being analysed is an intrusion or not.

Ripper rule output is following for fifth position system calls:

38: p3 = 40, p4 = 4. that is if p is 40 (lstat) and p is 4 (write) then 7 th system call is 38 (stat)

5: true. That is none of the above, then the 7 th system call is open.

From the Ripper rule we can extract the some rules and store in the data base. Each ripper rule has some confidence information. From these information we build the normal rule and that should be stored in the database. Once we can created the normal database then compare the new sequence of system calls with normal database and produce the result. The result may be normal or mismatch. Note that all the mismatch sequence are not instructive behaviours. It may be part of error code legal sequence. So we can analyse the mismatch sequence by Fuzzy system and produce sequences normal or abnormal.

4.4 Fuzzy System

Fuzzy system is used to detect the intrusion. The fuzzy system is also a rule based system. Here we should use if-then rule. For these if-then rules we use four fuzzy parameter. Each of these parameters increases the system flexibility and reduces the false alarm rate. The four parameters are the following:

1. Confidence average
2. Rate of mismatch sequence
3. Dangerous score of mismatch sequence
4. Rule count in the process

All these four parameters only have three values. The values are small, medium and high. Fuzzy system output format is normal or abnormal. Fuzzy system if-then rules are
1. IF Mismatch=M and Confidence_avg=M and Rule_count_dif=M and Score_mismatch= L THEN decision= Normal 
2. IF mismatch= M and confidence_avg=M and Rule_count_dif= M and Score_mismatch= L THEN decision= Normal 
3. IF mismatch= L and confidence_avg= L and Rule_count_dif= S and Score_mismatch= L THEN decision= Normal 
4. IF mismatch= L and confidence_avg= S and Rule_count_dif= L and Score_mismatch= L THEN decision= Abnormal

From these rules the systems detects the instructive behaviour system calls and produce the alarm.

V. RESULTS

We have simulated the experiment several times by exploiting various processes such as Login, PS, Xlock, Named and Stide. Here the sample dataset is taken as shown in Table-3 for pre-processing and the ripper rule is used to construct the normal behaviour of the database. The result of our system produced by the fuzzy decision as the sequence of the system call is normal or abnormal. In the table 3 shows used the data set for leaning the system

<table>
<thead>
<tr>
<th>Linux Processor</th>
<th>UNM Dataset</th>
<th>Normal File</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Intrusive File</td>
<td>Number of System call</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Number of System call</td>
</tr>
<tr>
<td>stide</td>
<td>99514</td>
<td>105</td>
</tr>
<tr>
<td>named</td>
<td>1800</td>
<td>6</td>
</tr>
<tr>
<td>ps</td>
<td>3231</td>
<td>2</td>
</tr>
<tr>
<td>Login</td>
<td>612</td>
<td>31</td>
</tr>
<tr>
<td>Xlock</td>
<td>462</td>
<td>37</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Linux Processor</th>
<th>RIPPER Rule</th>
</tr>
</thead>
<tbody>
<tr>
<td>stide</td>
<td>191</td>
</tr>
<tr>
<td>named</td>
<td>446</td>
</tr>
<tr>
<td>ps</td>
<td>188</td>
</tr>
<tr>
<td>login</td>
<td>142</td>
</tr>
<tr>
<td>xlock</td>
<td>312</td>
</tr>
</tbody>
</table>

In the table 5 shows the final system of our system. From mismatch sequence detect the intrusion. The out can be normal and abnormal.

Fig 4 and Fig 5 are shows that number of of sequences of system call in xlock and stide was increasing, number of rules also gradually was growing and it was not remained steady ever.
VI. CONCLUSION

Thus in the proposed model we have done the novel way of detecting the intrusion by using the fuzzy system and RIPPER rule approach in the system call. The Rate of false alarm has been decreased and Visibility is high and All of the intruder activities are visible. In the proposed method Rate of information assurance is higher than other methods. It is also operating system independent. The proposed method can detect the intrusions which other IDSs cannot detect (such as buffer overflow, lprcp, etc.). Pattern and rule creation in this method is easier than other methods, because number of system calls are specific and low.

REFERENCES

[7] Cheryl Martin, Alexander Liu "A Comparison of System Call Feature Representations for Insider Threat Detection“ , University of Texas at Austin, United States Military Academy IEEE 2005