NETWORK NODE INTRUSION DETECTION SYSTEM

Abeer M. Yousif, Suleiman S. Fawzy
Department of Computer, College of Science, University of Al-Nahrain. Baghdad- Iraq.

Abstract

Computer network technologies have grown rapidly in the last few decades. With the increased use of networked computers for critical applications, computer intrusions have been increased and became a significant threat to these systems, and, thus Intrusion Detection Systems (IDS) have become essential addition to security infrastructure of most organizations. This paper presents the design and implementation of a Network Node Intrusion Detection System (NNIDS) that support IPv4 protocol. It detects a variety of attacks which are directed to the resources of filing system. The implied detection rules are based on matching the predefined normal behavior of the system with the characteristics of the detected user’s events.

Several simulated attacks have been sent to the proposed system to test it. Test shows that most of the attacks can be detected with acceptable ratios of false positive and false negative values.

1. Introduction

Intrusion detection (ID) is the process of monitoring the events occurring in a computer system or network and analyzing them for signs of intrusions. Intrusion Detection System (IDS) is a software or hardware product that automates this monitoring and analysis process [1]. Intrusion Detection Systems can be classified into three categories with respect to: 1) where data is collected; 2) where and how data is
processed; and 3) the way that analyze the collected data (analyze strategy). Regarding the place of the data collection, IDSs can be divided into four types: Host-Based Intrusion Detection System, Network-Based Intrusion Detection System, Hybrid Intrusion Detection, and Network-Node Intrusion Detection System [2].

IDSs can be classified according to where and how data is processed into Distributed-Based Intrusion Detection System and Centralized-Based Intrusion Detection System [3]. With respect to the method of analysis the collected data by IDS, IDSs can be classified into two types: Misuse-Intrusion Detection System and Anomaly-Intrusion Detection System [1].

Much research has been devoted to intrusion detection in recent years. Two enormously popular open source tools, Snort [6] and Bro [7], have shown that static signature based IDS’s can be quite successful in the face of known attacks. Combined with automatic monitoring and incident response, system administrators have a powerful tool against network attacks. In [8], the authors present the case for collaborative intrusion detection system where intrusion detection nodes cooperate to determine if network attack is taking place and take corrective actions if it does.

The proposed system is applied by explicitly looking for the filing system attacks within a network due to some triggering events of suspicious behavior. All normal behaviors are predefined to the system as a result of previous analysis. Also, it proposes a scheme of escalating levels of alertness so that the system administrator can take preemptive actions against the attack, such as warning the suspicious users for their misuse actions.

The rest of the paper is organized as follows: Section 2 introduces the proposed system and presents its architecture with explanation of the different stages of it. Detailed study of the effectiveness and performance of our system are presented in section 3. Section 4 summarizes the derived conclusions. Finally, we discuss some future work that we intend to explore in section 6.

2. PROPOSED SYSTEM

The name of the proposed system is chosen to be FMS (the acronym for File Monitoring System). FMS should monitor all the packets coming to each host in the network and detect the intrusion cases by reporting the Administrator with alarm messages. To achieve the above aim, different features were considered, they are:

1. FMS will be designed as a Distributed-based Intrusion Detection System, i.e. the data will be collected and analyzed on each host in the network individually. Since, the shared resources of each host will be protected independently.

2. The analysis strategy of FMS was chosen to be Anomaly, where the profiles that represent the normal behavior of each user per each host will be initialized first by Administrator, and then stored on each host lately.

3. The intrusion that will be detected by FMS is concerned with checking illegal user's actions that are associated with shared files and folders, such as open file, delete folder … etc.

FMS consists of two main subsystems; Client subsystem and Administrator subsystem as illustrated in figure (1).

Client Subsystem

Administrator Subsystem

Figure 1: FMS Architecture

Client subsystem modules are:

1. Initialization and reconfiguration module: it initializes Client subsystem information. This module performs the following functions:
   a. Identify the important user shared resources which must be protected with a particular level of protection.
   b. Determine the Administrator's PC.
   c. Assemble the local shared resources into a list $L$. $L$ is formatted as shown in figure (2).

2. PROPOSED SYSTEM

The name of the proposed system is chosen to be FMS (the acronym for File Monitoring System). FMS should monitor all the packets coming to each host in the network and detect the intrusion cases by reporting the Administrator with alarm messages. To achieve
d. Send the list \( L \) to the Administrator subsystem to set/update permissions depending on the type of shared resource and the class of the user. The implemented classes of users in FMS are student, employee and visitor within the environment of university.

e. Get list of shared/permissions (or ACL) from the Administrator subsystem and organize it in Binary Tree structure. Each record in the tree is formatted as shown in figure (3).

f. Get list of users from the Administrator subsystem. This list contain full information of each user such PC name, PC’s IP, PC user account name, username, and user Type. These information are aggregated during registrations processes of each user.

g. Determine Time threshold value (\( t \)) that will be used in Detection Module to compute the threat. This value represents the maximum acceptable period of time to compute the repetitions of each action.

![Figure 2: local shared resources](image)

![Figure 3: record of ACL tree](image)

2. Authentication Module: it identifies users to the system. This module consists of two units, as follow:

a. Registration Unit: it registers the allowed user into the system depending on user’s information.

b. Login Unit: it log-in any user into the system depends on user's registration

3. Sniffing Module: sniffer function is listening to the network segment or sniffs packets on a specific network segment. FMS sniffs packets on a 3-layer segment (Application, Transport, and Internet layers) of TCP/IP model. Sniffer module will receive all incoming window messages, and then filters them to reduce the amount of processing by ignoring unimportant packets to its purpose. Filtering process depends on the type of incoming messages; meaningful messages are from the type of Window Socket Message (WinSockMsg), which are specified to network packets. In addition, WinSockMsg will be filtered to keep only the messages that are related to file system operations.

After finishing filtering process, Sniffer module will generate a sequence of events that represent the executed operations by the sniffed network segment.

4. Detection Module: it detects malicious events by applying specific rules during traffic analysis, and then generates an alarm code which represents the level of threat that occurred, and a notify message that describes the threat. The primitive values of the alarm level are estimated according to the problem environment from 1 to 9 as illustrated in table (1). These values are depending on three factors: user class, object class, and action type. The primitive values will be increased depending on the number of occurrences of the same event within period \( t \).

![Table 1: Alarm level values](image)

5. Alarm Module: it generates alarm messages which represent user’s suspicious activities and then sending these messages to the Administrator. The alarm message depends on alarm code and notify message that was generated in the Detection Module.

On the other hand, Administrator subsystem consists of the following modules:

1. Check Administrator Authority Module: it checks the authenticity of Administrator identity.

2. Initialization Module: it initializes pre-system information. This module consists of three units, as follows:

a. Permission Unit: it Sets/Updates the permissions for all shared resources of each user, and then sends them to the Client subsystem.

b. Users Information Unit: it creates Database contains list of users with some required information to the system after registration process. These information are PC name, PC local IP, PC account name, user name, and user type.

c. Registration Database (RegDB) Unit: it creates Database contains all identification
information of all users whom allowed using the system.
3. Monitoring Module This module consists of two units, as follow:
   a. Alarm Messages Unit: it selects the case of monitoring process either to be online or offline modes.
   b. Users Profiles Unit: it creates profile for any misbehavior user to be utilized in the future detection.
4. Response Module: it generates a proper response (Warning messages) in the case of detecting an intrusion.

3. PERFORMANCE ANALYSIS

The evaluation of FMS has two prospective; Accuracy, Time and memory Consuming. The accuracy evaluation of such intrusion detection system depends on two factors: false positive and false negative measures. Both measures depend on (1) number of detected attacks, (2) number of alerts. While the time consumption of FMS is computed according to the execution time of system initialization stage and detection stage. Memory consumption is the size of memory, which is used by FMS to construct the ACL into binary tree.

3.1 Accuracy

To measure the accuracy of FMS, a simulation was conducted by applying different types of file attacks on the network resources to compute false positive and false negative values and alarms. These attacks are associated with the file access types: Open, Modify, Delete and Copy. Important criteria are taken into consideration during test cases, these are: (1) number of occurrences of each event. (2) Number of executed action for specific subject. (3) Variety of user authentication cases. (4) Time of event occurrence.

**Test Case One**: - in this case, false positive and false negative values have been calculated such that each of the previous four types of actions has been applied for ten times. Tables (2&3) show the results respectively.

From Table (2) several points were noticed, which are:
1. The cases of attacks that are sent by unauthorized users didn't present any false positive values because false positive depends on the number of authorized accesses which are described as attacks.
2. When the class of the object is **System class** and the type of action are either **Modify** or **Delete** then some false positive hits have been registered because detection rule decision is made according to the number of the occurrences of those actions by the same user.
3. Other cases of false positives alarms are obtained when **Open** and **Copy** actions are performed for all resources classes. Also, false positive alarms are obtained when **Modify** and **Delete** actions are performed for **User** and **Application** classes. Those cases are registered because the complete path of the accessed object isn't received by detection engine correctly due to the network traffic.

**Table 2: False Positive values of the Test Case one**

<table>
<thead>
<tr>
<th>User Class</th>
<th>Action Type</th>
<th>System Class</th>
<th>Application Class</th>
<th>User Class</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
</tr>
<tr>
<td>Student</td>
<td>Open</td>
<td>1 0</td>
<td>2 0</td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>Modify</td>
<td>0 0</td>
<td>0 0</td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>Delete</td>
<td>1 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>Copy</td>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>Open</td>
<td>0 1</td>
<td>0 0</td>
<td>0 1</td>
</tr>
<tr>
<td>Employee</td>
<td>Modify</td>
<td>1 0</td>
<td>2 0</td>
<td>1 0</td>
</tr>
<tr>
<td></td>
<td>Delete</td>
<td>1 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>Copy</td>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td>Visitor</td>
<td>Open</td>
<td>2 0</td>
<td>0 0</td>
<td>2 0</td>
</tr>
<tr>
<td></td>
<td>Modify</td>
<td>1 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>Delete</td>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
<tr>
<td></td>
<td>Copy</td>
<td>0 0</td>
<td>0 0</td>
<td>0 0</td>
</tr>
</tbody>
</table>

From Table (3) several points have been noticed, they are:
1. The cases of attacks which are sent by authorized users haven't caused any false negative alarm, because false negative depends on the number of unauthorized accesses (attacks) which are undetected.
2. There are few attacks cases that FMS may fail to detect them. These cases may occur due to one of the following reasons:
   a. The Client subsystem didn't generate events for these actions,

399
b. The Administrator subsystem didn't receive the alarm messages under heavy traffic, or
c. Detection rules passed the actions with a particular permission due to the rules follow a
target path similar to the path of other target in
ACL.

All of the above-mentioned reasons due to the network traffic. As result, the accuracy of FMS
is measured in percent as follow:

1. The number of executed actions in Table (2) is 360 actions; they imply 26 cases of false
   positive. Therefore, the accuracy of FMS in the term of false positive is %92.7. The numbers of
   executed actions in Table (3) are 360 actions; they imply 14 cases of false negative. Therefore,
   the accuracy of FMS in the term of false negative is %96.1.

Test Case Two: - like test case one this case applies four types of actions but this case is to
evaluate the value of alarm level.
Each arrow symbol (i.e. \( \rightarrow \)) in the following tables represents the increase of the alarm level
for its primitive value. The bold numbers represent the false positive cases.

<table>
<thead>
<tr>
<th>System Class</th>
<th>Application Class</th>
<th>User Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
</tr>
<tr>
<td>Open</td>
<td>3 3→6</td>
<td>0 2→5</td>
</tr>
<tr>
<td>Modify</td>
<td>0→5 9→9</td>
<td>0 8→9</td>
</tr>
<tr>
<td>Delete</td>
<td>0→5 9→9</td>
<td>8 8→9</td>
</tr>
<tr>
<td>Copy</td>
<td>0 3→6</td>
<td>0 2→5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>System Class</th>
<th>Application Class</th>
<th>User Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
<td>Auth Unauth</td>
</tr>
<tr>
<td>Open</td>
<td>3 3→6</td>
<td>0 2→5</td>
</tr>
<tr>
<td>Modify</td>
<td>0→5 9→9</td>
<td>8 8→9</td>
</tr>
<tr>
<td>Delete</td>
<td>0→5 9→9</td>
<td>0 8→9</td>
</tr>
<tr>
<td>Copy</td>
<td>0 3→6</td>
<td>0 2→5</td>
</tr>
</tbody>
</table>

The results of test case two can be summarized as follows: The attack cases that sent by
authorized users, except Modification and Deletion of the system resources, didn't present
threat, nevertheless, some false positive cases present. The generated alarm for the false
positive case is equal to the primitive value because the error in detection didn't occur again
in the next occurrence and the rule decided these actions as authorized actions.

3.2 Time and Memory Consumption

In FMS, there is a time required to construct
the ACL during the initialization stage, and
additional time is required to search the ACL for
permission of an accessed object during the
detection stage.

In addition to the consumed time, the used
memory to construct the ACL is an important
efficiency factor it depends on the number of
shared files and folders.

Both, the consumed time and memory depend
on some of the H/W specifications of the
computer that runs FMS. FMS was installed in
computer which has the following specification
according to CPU and RAM:

1. CPU – 1.7 GHz.
2. RAM – 512 MB.

Table (7) shows the time and memory
consumption for different cases of numbers of
shared resources.

The discussion of results can be summarized as
follows:

1. The numbers of shared resources depends on
the contents of the shared resources of applied
environment. For example, the number 41756
represents files and folders all have size
around 10 Gigabytes. The size of shared
resources is ineffective on the size of ACL;
because ACL deals with object's name only.
2. The consumed time for constructing the ACL in some cases may be too long (e.g. 300 seconds \( \approx 5 \) minutes), but this time will be spend for once, only during initialization stage, and hence it is reasonable time for a system tries to index shared resources size about 90 Gigabytes or more.

3. The consumed time to search ACL is short time when the number of objects is not relatively big. And this is the advantage of using the Binary Tree structure to index the shared resources.

4. The size of used memory in some cases may be considered big, but this is, today, not a problem with available computer's memory size is more than 512 MB.

### Table 7: Time and Memory consumption

<table>
<thead>
<tr>
<th>Size of shared resources in GB</th>
<th>Number of shared resources</th>
<th>Time to construct ACL in second</th>
<th>Time to search ACL in MB</th>
<th>Size of used RAM in MB</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \approx 4.21 )</td>
<td>8,579</td>
<td>16</td>
<td>0.009</td>
<td>4.936</td>
</tr>
<tr>
<td>( \approx 7.69 )</td>
<td>13,317</td>
<td>32</td>
<td>0.011</td>
<td>7.352</td>
</tr>
<tr>
<td>( \approx 11.62 )</td>
<td>19,855</td>
<td>42</td>
<td>0.015</td>
<td>13.328</td>
</tr>
<tr>
<td>( \approx 48.10 )</td>
<td>21,900</td>
<td>42</td>
<td>0.021</td>
<td>16.188</td>
</tr>
<tr>
<td>( \approx 10.72 )</td>
<td>41,756</td>
<td>94</td>
<td>0.081</td>
<td>31.348</td>
</tr>
<tr>
<td>( \approx 39.09 )</td>
<td>62,254</td>
<td>118</td>
<td>0.200</td>
<td>59.104</td>
</tr>
<tr>
<td>( \approx 32.01 )</td>
<td>78,481</td>
<td>155</td>
<td>0.406</td>
<td>82.413</td>
</tr>
<tr>
<td>( \approx 89.77 )</td>
<td>140,735</td>
<td>300</td>
<td>0.912</td>
<td>112.932</td>
</tr>
</tbody>
</table>

### 3.3 System Restrictions

During system implementation, several restrictions have been raised, the main ones are:

1. In some cases, the system can detect the threat but couldn’t generate the appropriate relative event. For example, the intruder deletes more than 100 files and folders in one click, the system will detect the threat during sniffing process but it can’t generate the relative event due to the speed of deleting action.

2. During sniffing process, some packets couldn’t be sniffed correctly due to the problems and restrictions of network environment. Like heavy load on the client PC.

3. Any changes to the existing shared resources from the user will not be protected until the system makes its periodical update checking and modification.

### 4. CONCLUSIONS

1. The type of network messages that is relevant to the file system are deal with port numbers equal 139 and 445 without any differences. Both of these ports refer to SMB working over NetBIOS, and then NetBIOS over TCP/IP.

2. It is necessary to index disk's resources to be capable applying structural searching technique in order to achieve high-speed searching performance.

3. The power of the network intrusion detection system depends on which network segment is sniffed and, thereby, on what is the extracted information from this segment.

4. Heavy traffic causes failing of receiving the events from the clients.

### 5. SUGGESTIONS FOR FUTURE WORK

1. Utilization of users profiles in detection engine.

2. Applying another response type instead of Warning Message, such as prevention the attacks by ending user network session, or blocking user's IP.

3. Running FMS on a Wide Area Network (WAN) or Metropolitan Area Network (MAN).

Taken into considerations all the necessary scalable factors to establish such a system, examples: distributed databases, local and global Administrator subsystems and their administrators' privileges.

### References


5. Oryspayev D. Oryspayuli, August. 2006. *What intrusion detection approaches work well if only TCP/IP packet header information is available?*. Master Thesis, Faculty of Electrical Engineering, Mathematics and Computer Science, University of Twente, Netherlands.

