International Journal of Innovative Research in Computerand Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | Impact Factor: 7.488|

||Volume 9, Issue 5, May 2020||

Internal Intrusion Detection and Protection System

Shankaraya Shastri [1], Amir Hamza Sheikh[2], Amol Khanolkar[3], Umme Aiman Palegar [4], Salveera Belawadi[5]

Assistant Professor, Dept. of Computer Science Engineering, S.G Balekundri Institute of Technology, Belagavi, Karnataka, India [1]

Students, Dept. of Computer Science Engineering, S.G Balekundri Institute of Technology, Belagavi, Karnataka, India[2,3,4,5]

ABSTRACT: Currently, most computer systems use user IDs and passwords as the login patterns to authenticate users. However, many people share their login pattern switch coworkers and request these coworkers to assist co-tasks, thereby making the pattern as one of the weakest points of computer security. Insider attackers, the valid users of a system who attack the system internally, are hard to detect since most intrusion detection systems and firewalls identify and isolate malicious behaviours launched from the outside world of the system only. In addition, some studies claimed that analysing system calls (SCs) generated by commands can identify these commands, with which to accurately detect attacks, and attack patterns are the features of an attack. Therefore, in this paper, a security system, named the Internal Intrusion Detection and Protection System (IIDPS), is proposed to detect insider attacks at SC level by using data mining and forensic techniques. The IIDPS creates users' personal profiles to keep track of users' usage habits as their forensic features and determines whether a valid login user is the account holder or not by comparing his/her current computer usage behaviours with the patterns collected in the account holder's personal profile. The experimental results demonstrate that the IIDPS's user identification accuracy is 94.29%, whereas the response time is less than 0.45 s, implying that it can prevent a protected system from insider attacks effectively and efficiently.

KEYWORDS: Data mining, insider attack, intrusion detection and protection, system call (SC), users' behaviours

I. INTRODUCTION

As network-based computer systems have important roles in modern society, they have become the targets of intruders. Therefore, we need to find the best possible ways to protect our systems. The security of a computer system is compromised when an intrusion takes place. An intrusion can be defined as any action done to hamper the integrity, confidentiality or availability of the system. There are some intrusion prevention techniques which can be used to protect computer systems as a first line of defense. But only intrusion prevention is not enough. As systems become more complex, there are always exploitable weaknesses in the systems due to design and programming errors, or various penetration techniques. Therefore Intrusion detection is required as another measure to protect our computer systems from such type of vulnerabilities. An intrusion detection system dynamically monitors the events taking place in a monitored system, and decides whether these events are symptomatic of an attack or constitute a legitimate use of the system.

II. RELATED WORK

Computer forensics science, which views computer systems as crime scenes, aims to identify, preserve, recover, analyze, and present facts and opinions on information collected for a security event. It analyzes what attackers have done such as spreading computer viruses, malwares, and malicious codes and conducting DDoS attacks. Most intrusion detection techniques focus on how to find malicious network behaviors and acquire the characteristics of attack packets, i.e., attack patterns, based on the histories recorded in log files used self-developed packet sniffer to collect network packets with which to discriminate network attacks with the help of network states and packet distribution. These files contain traces of computer misuse. It means that, from synthetically generated log files, these traces or patterns of misuse can be more accurately reproduced. The authors systematically summarized and compared different intrusion detection methods, thus allowing us to clearly view those existing research challenges.

Theseaforementionedtechniquesandapplicationstrulycontributetonetworksecurity.However, they cannot easily authenticate remote-login users and detect specific types of intrusions, e.g., when an unauthorized user logs in to a system with a valid user ID and password. In our previous work, a security system, which collects forensic features for users at command level rather than at SC level, by invoking data mining and forensic techniques, was developed. Moreover, if attackers use many sessions to issue attacks, e.g., multistage attacks, then it is not easy for that system to identify attack patterns. IIDPS that utilizes a forensic technique to profile user behaviors and a data mining technique to carry out cooperative attacks. The authors claimed that the system could detect intrusions effectively and efficiently in real time. However, they did not mention the SC filter. Provided another example of integrating computer forensics with a knowledge-based system. The system adopts a predefined model, which, allowing SC-sequences to be normally executed, is employed by a detection system to restrict program execution to ensure the security of the protected system.

This is helpful in detecting applications that issue a series of malicious SCs and identifying attack sequences having been collected in knowledge bases. When an undetected attack is presented, the system frequently finds the attack sequence in 2 s as its computation overhead. a grid based platform, named the dynamic grid based intrusion detection environment, which exploits grid's abundant computing resources to detect a massive amount of intrusion packets and to manage a dynamic environment.

The advanced metering infrastructure, which is one of the most crucial components of smart card, serves as a bridge for providing bidirectional information flow between the user domain and the utility domain. The authors treat an IDS as a second-line security measure after the first line of primary advanced metering infrastructure security techniques such as encryption, authorization, and authentication.

III. METHODOLOGY

In this section, we first introduce the IIDPS framework and describe components of the IIDPS in detail. It can identify a user's forensic features by analyzing the corresponding SCs to enhance the accuracy of attack detection It can be able to port the IIDPS to a parallel system to further shorten its detection response time It effectively resist insider attack. The IIDPS can detect those malicious behaviors issued by them and then prevent the protected system from being attacked. The IIDPS consists of an SC monitor and filter, a mining server, a detection server, a local computational grid, and three repositories, including user log files, user profiles, and an attacker profile. The SC monitor and filter, as a loadable module embedded in the kernel of the system being considered, collects those SCs submitted to the kernel and stores these SCs in the format of (uid,pid,SC)in the protected system where uid, pid, and SC respectively represent the user ID, the process ID, and the SC c submitted by the underlying user, i.e., c ∈ SCs. It also stores the user inputs in the user's log file, which is a file keeping the SCs submitted by the user following their submitted sequence. The mining server analyzes the log data with data mining techniques to identify the user's computer usage habits as his/her behavior patterns, which are then recorded in the user's user profile.

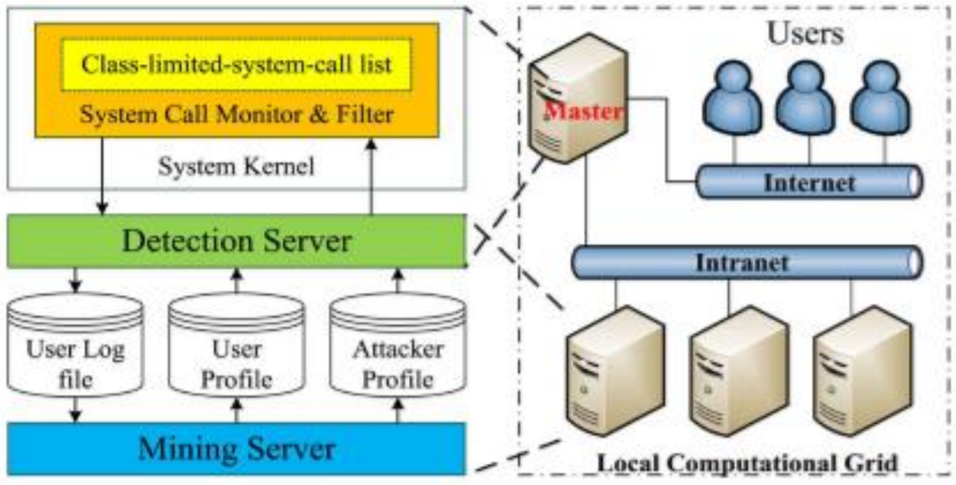


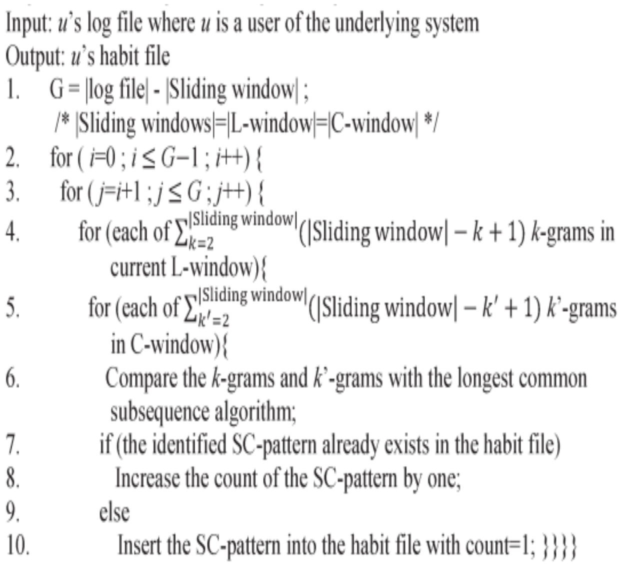
Fig. 1. IIDPS system framework.

The detection server compares users' behavior patterns with those SC-patterns collected in the attacker profile, called attack patterns, and those in user profiles to respectively detect malicious behaviors and identify who the attacker is in real time. When an intrusion is discovered, the detection server notifies the SC monitor and filter to isolate the user from the protected system. The purpose is to prevent him/her from continuously attacking the system. Both the detection server and the mining server are run on the local computational grid to accelerate the IIDPS's online detection and mining speeds and enhance its detection and mining capability.

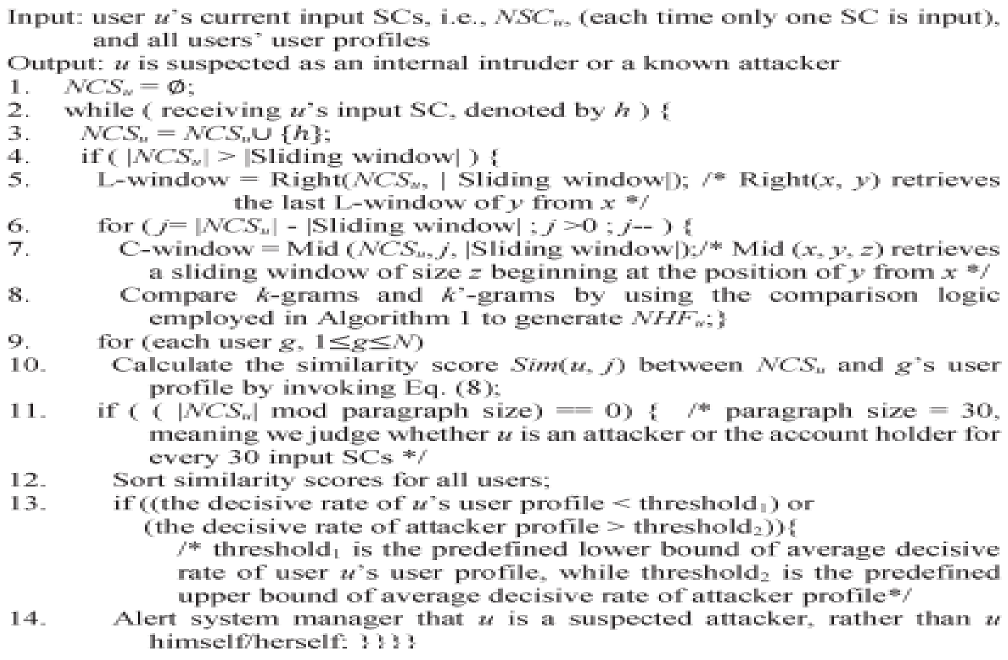
If a user logs in to the system by using another person's login pattern, the IIDPS identifies who the underlying user is by computing the similarity scores between the user's current inputs, i.e., SCs, and the behavior patterns stored in different users' user profiles. In the IIDPS, the SCs collected in the class-limited-SC list, as a key component of the SC monitor and filter, are the SCs prohibited to be used by different groups/classes of users in the underlying system, e.g., a secretary cannot submit some specific privileged SCs. Therefore, commands that generate these SCs will be prohibited to be used by all secretaries.

IV. PROPOSED ALGORITHM

Two algorithms are also presented for generating a user habit file and detecting an internal intruder.



```
Input: u's log file where u is a user of the underlying system
Output: u's habit file
1.   G = |log file| - |Sliding window| ;
      /* |Sliding windows|=|L-window|=|C-window| */
2.   for ( i=0 ; i ≤ G−1 ; i++) {
3.     for ( j=i+1 ; j ≤ G ; j++) {
4.       for (each of $\sum_{k=2}^{|\text{Sliding window}|}$(|Sliding window| − k + 1) k-grams in
            current L-window){
5.         for (each of $\sum_{k'=2}^{|\text{Sliding window}|}$(|Sliding window| − k' + 1) k'-grams
            in C-window){
6.           Compare the k-grams and k'-grams with the longest common
             subsequence algorithm;
7.           if (the identified SC-pattern already exists in the habit file)
8.             Increase the count of the SC-pattern by one;
9.           else
10.            Insert the SC-pattern into the habit file with count=1; }}}}
```



```
Input: user u's current input SCs, i.e., $NSC_u$, (each time only one SC is input),
        and all users' user profiles
Output: u is suspected as an internal intruder or a known attacker
1.    $NCS_u = \emptyset$;
2.    while ( receiving u's input SC, denoted by h ) {
3.      $NCS_u = NCS_u \cup \{h\}$;
4.      if ( $|NCS_u|$ > |Sliding window| ) {
5.        L-window = Right($NCS_u$, | Sliding window|); /* Right(x, y) retrieves
                    the last L-window of y from x */
6.        for ( j= $|NCS_u|$ - |Sliding window| ; j >0 ; j-- ) {
7.          C-window = Mid ($NCS_u$, j, |Sliding window|);/* Mid (x, y, z) retrieves
              a sliding window of size z beginning at the position of y from x */
8.          Compare k-grams and k'-grams by using the comparison logic
              employed in Algorithm 1 to generate $NHF_u$;}
9.        for (each user g, $1 \le g \le N$)
10.         Calculate the similarity score Sim(u, j) between $NCS_u$ and g's user
              profile by invoking Eq. (8);
11.       if ( ( $|NCS_u|$ mod paragraph size) == 0) {   /* paragraph size = 30,
              meaning we judge whether u is an attacker or the account holder for
              every 30 input SCs */
12.         Sort similarity scores for all users;
13.         if ((the decisive rate of u's user profile < $\text{threshold}_1$) or
             (the decisive rate of attacker profile > $\text{threshold}_2$)){
                /* $\text{threshold}_1$ is the predefined lower bound of average decisive
                rate of user u's user profile, while $\text{threshold}_2$ is the predefined
                upper bound of average decisive rate of attacker profile*/
14.         Alert system manager that u is a suspected attacker, rather than u
              himself/herself; }}}}
```

V. IMPLEMENTATION

Implementation of software refers to the final installation of the package in its real environment, to the satisfaction of the intended users and the operation of the system. The people are not sure that the software is meant to make their job easier.

- The active user must be aware of the benefits of using the system
- Their confidence in the software built up
- Proper guidance is impaired to the user so that he is comfortable in using the application

Before going ahead and viewing the system, the user must know that for viewing the result, the server program should be running in the server. If the server object is not running on the server, the actual processes will not take place.

System Flow:

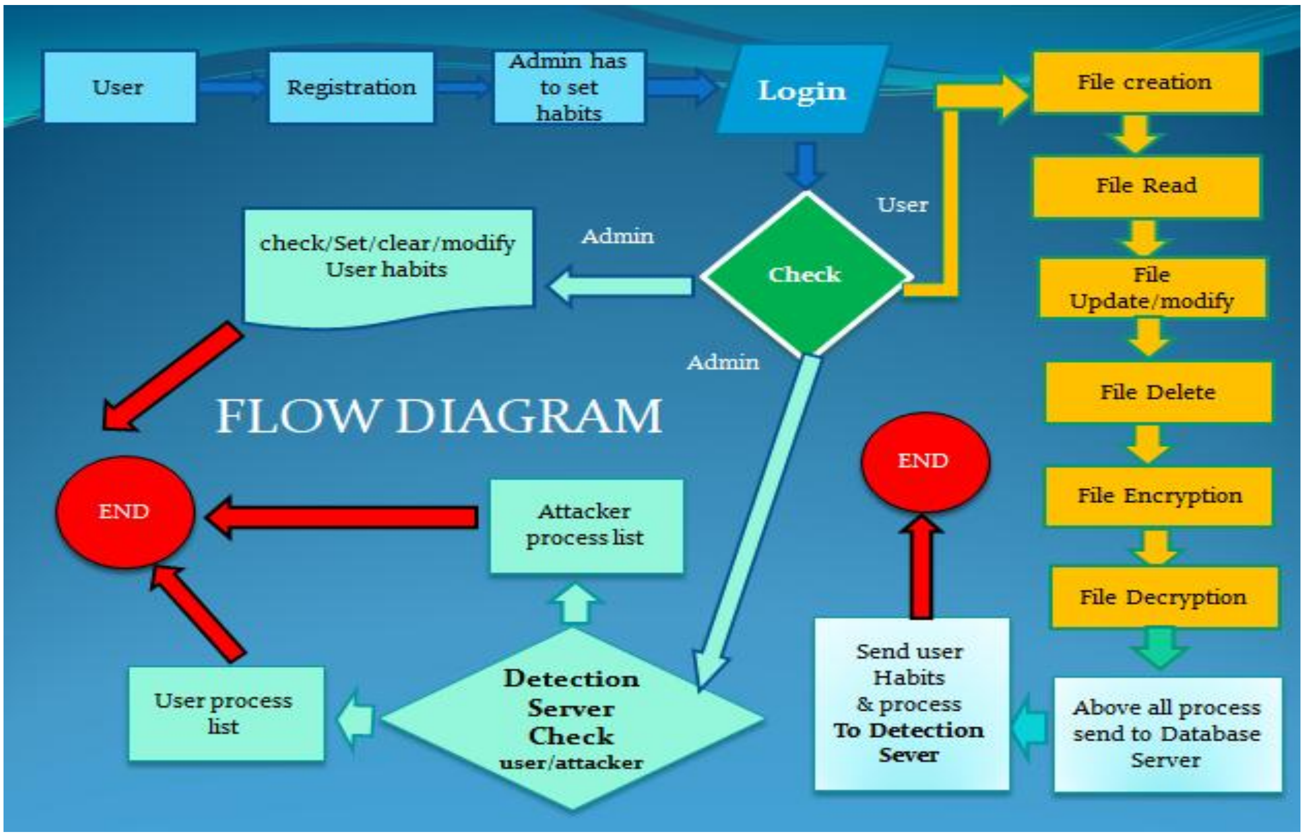
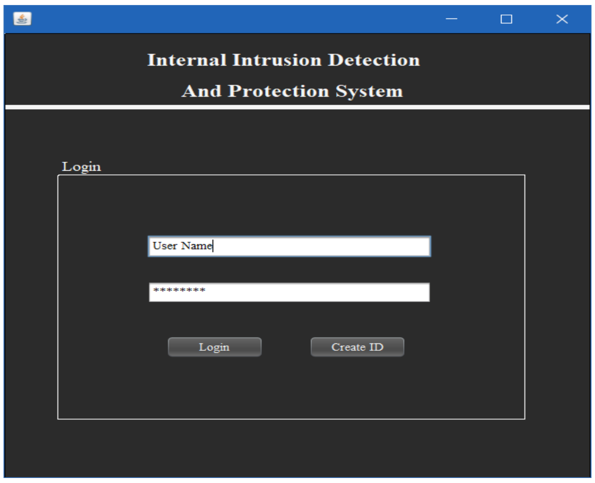


Figure 2: Flowchart of the project

Modules:

The first page is the login page. Through login page if the user don’t have an account can register by creating an id as shown in the below figure 5.2



User Name
****

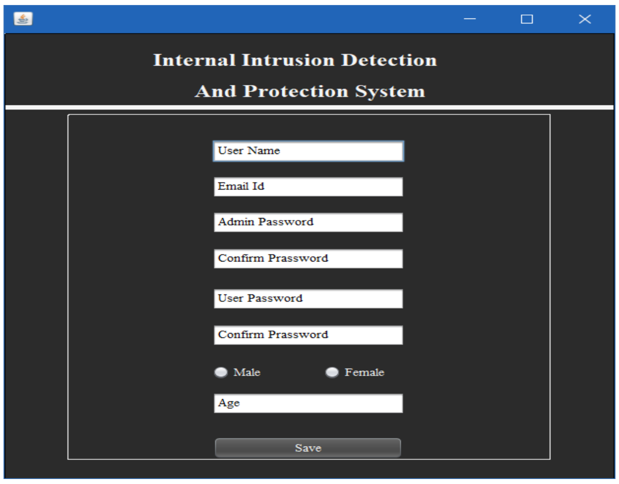
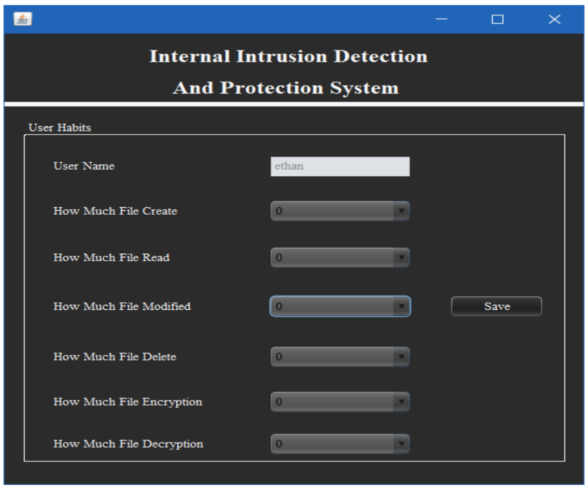


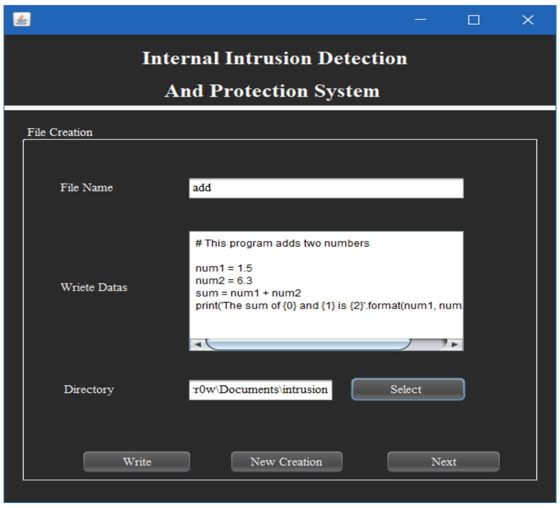
Fig 5.1 View of user login page

Fig 5.2 View of user registration page

After the user is authorized by creating an ID, the admin assigns the habits to the user to perform operations. The Fig.5.3 shows the View of setting user's user habit by admin. Fig. 5.4 View of file creation by the user



User Name: ethan
How Much File Create: 0
How Much File Read: 0
How Much File Modified: 0
How Much File Delete: 0
How Much File Encryption: 0
How Much File Decryption: 0



File Name: add

Wriete Datas:

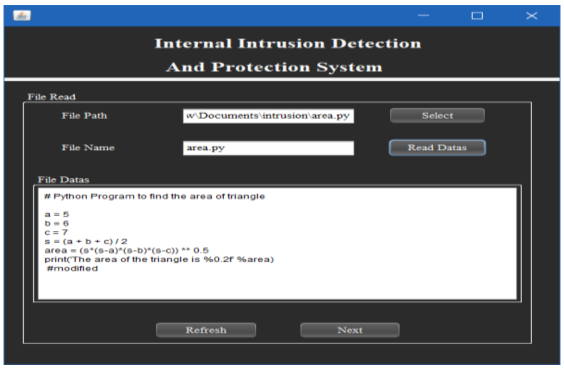
```
# This program adds two numbers

num1 = 1.5
num2 = 6.3
sum = num1 + num2
print('The sum of {0} and {1} is {2}'.format(num1, num
```

Directory: r0w\Documents\intrusion

Fig. 5.3 View of setting user's user habit by admin

Fig. 5.4 View of file creation by the user

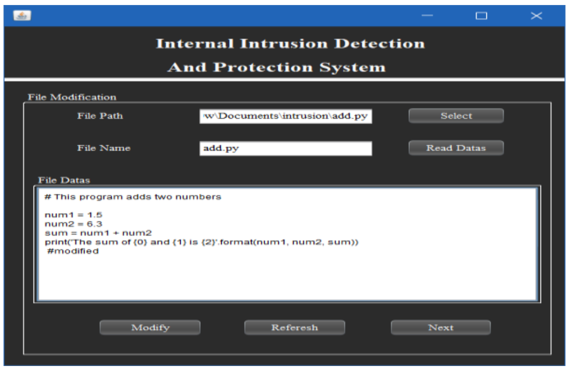
The Fig.5.5 and 5.6 gives the view of the file read operation and file modification operations done by the user respectively. The user can read the file and even modify the file.



File Path: w\Documents\intrusion\area.py
File Name: area.py

File Datas:

```
# Python Program to find the area of triangle

a = 5
b = 6
c = 7
s = (a + b + c) / 2
area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
print('The area of the triangle is %0.2f' %area)
 #modified
```



File Path: w\Documents\intrusion\add.py
File Name: add.py

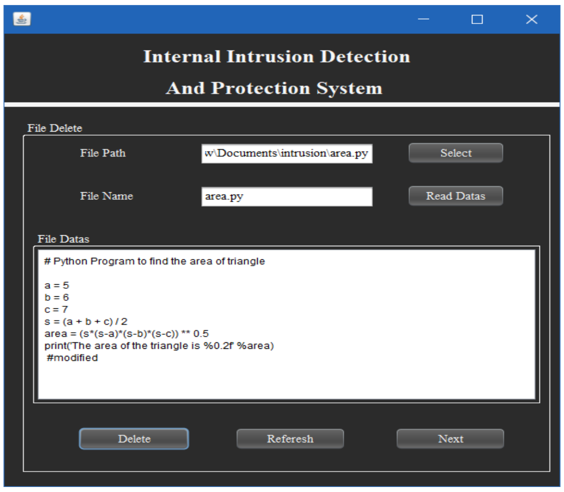
File Datas:

```
# This program adds two numbers

num1 = 1.5
num2 = 6.3
sum = num1 + num2
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
 #modified
```

Fig. 5.5.View of file read operation by the user

Fig. 5.6.View of file modification by the user

The user can delete the file, encrypt and even decrypt any file as shown in Fig 5.7, 5.8 and 5.9 respectively.

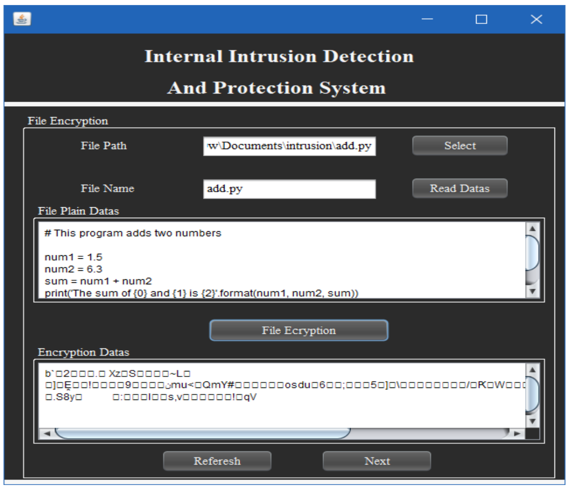


File Path: w\Documents\intrusion\area.py

File Name: area.py

```
# Python Program to find the area of triangle

a = 5
b = 6
c = 7
s = (a + b + c) / 2
area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
print('The area of the triangle is %0.2f' %area)
 #modified
```

Fig. 5.7 View of file deletion operation by the user



File Path: w\Documents\intrusion\add.py

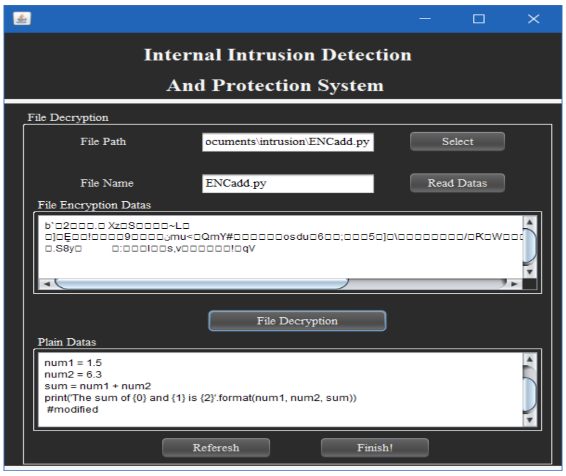
File Name: add.py

```
# This program adds two numbers

num1 = 1.5
num2 = 6.3
sum = num1 + num2
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
```

```
b`□2□□□.□ Xz□S□□□□~L□
□]□Ȩ□□!□□□□9□□□□ǫmu<□QmY#□□□□□□osdu□6□□;□□□5□]□\□□□□□□□□/□R□W□□□
□.S8y□        □:□□□l□□s,v□□□□□□!□qV
```

Fig. 5.8 View of file encryption operation by the user

After performing the operations, the user clicks the finish button and is automatically logged out. The operations performed by the user, goes to the database and they are visible in the admin panel as shown in Fig 5.10



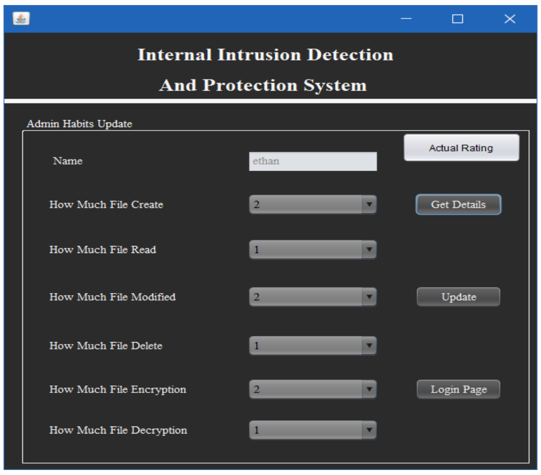
File Path: ocuments\intrusion\ENCadd.py

File Name: ENCadd.py

```
b`□2□□□.□ Xz□S□□□□~L□
□]□Ȩ□□!□□□□9□□□□ǫmu<□QmY#□□□□□□osdu□6□□;□□□5□]□\□□□□□□□□/□R□W□□□
□.S8y□        □:□□□l□□s,v□□□□□□!□qV
```

```
num1 = 1.5
num2 = 6.3
sum = num1 + num2
print('The sum of {0} and {1} is {2}'.format(num1, num2, sum))
 #modified
```

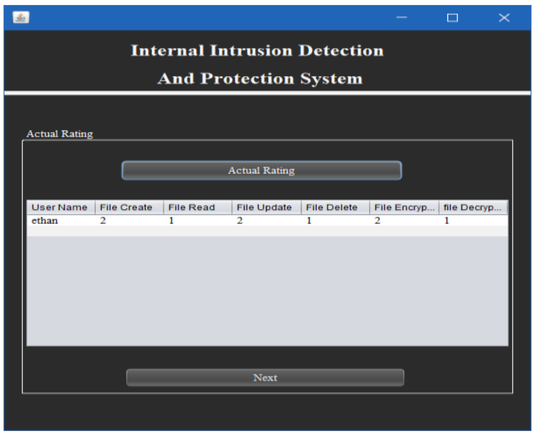
Fig. 5.9 View of file encryption operation by user



Name	ethan
How Much File Create	2
How Much File Read	1
How Much File Modified	2
How Much File Delete	1
How Much File Encryption	2
How Much File Decryption	1

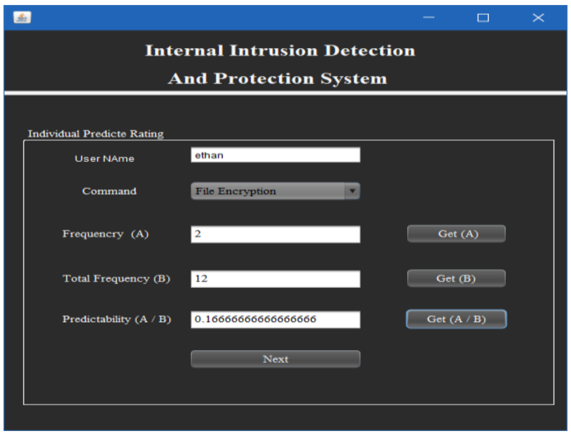
Fig. 5.10 View of habits update in admin panel

The admin can get the actual rating set by him to the user by clicking the actual rating as in Fig.5.11. And next the admin can know the number operations performed by the user in each operation(Fig.5.12) that is the predicate rating.



User Name	File Create	File Read	File Update	File Delete	File Encryp...	file Decryp...
ethan	2	1	2	1	2	1

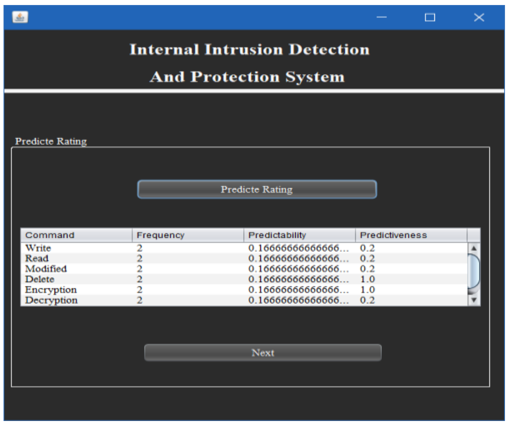
Fig. 5.11 View of actual rating



User NAme	ethan
Command	File Encryption
Frequencry (A)	2
Total Frequency (B)	12
Predictability (A / B)	0.16666666666666666

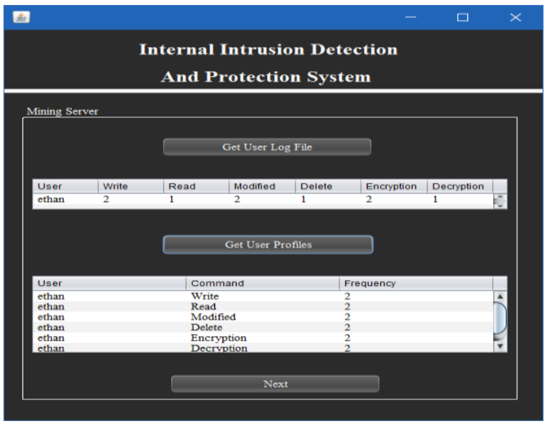
Fig. 5.12 View of predicate rating of each operation made by user

In the mining server as shown in Fig. 5.14, the admin compares the number of operations assigned by the admin to the user to be performed and the operation performed by the user.



Command	Frequency	Predictability	Predictiveness
Write	2	0.16666666666666...	0.2
Read	2	0.16666666666666...	0.2
Modified	2	0.16666666666666...	0.2
Delete	2	0.16666666666666...	1.0
Encryption	2	0.16666666666666...	1.0
Decryption	2	0.16666666666666...	0.2

Fig. 5.13 View of predicate reading

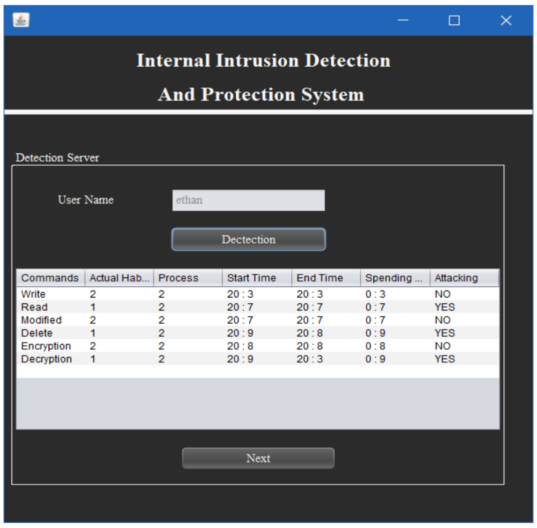


User	Write	Read	Modified	Delete	Encryption	Decryption
ethan	2	1	2	1	2	1

User	Command	Frequency
ethan	Write	2
ethan	Read	2
ethan	Modified	2
ethan	Delete	2
ethan	Encryption	2
ethan	Decryption	2

Fig. 5.14 View of mining server i.e. getting user log file

The experimental results demonstrate that the IIDPS's user identification accuracy is 94.29%, whereas the response time is less than 0.45 s, implying that it can prevent a protected system from insider attacks effectively and efficiently. This is shown in Fig. 5.15 and 5.16.



User Name: ethan

Commands	Actual Hab...	Process	Start Time	End Time	Spending ...	Attacking
Write	2	2	20 : 3	20 : 3	0 : 3	NO
Read	1	2	20 : 7	20 : 7	0 : 7	YES
Modified	2	2	20 : 7	20 : 7	0 : 7	NO
Delete	1	2	20 : 9	20 : 8	0 : 9	YES
Encryption	2	2	20 : 8	20 : 8	0 : 8	NO
Decryption	1	2	20 : 9	20 : 3	0 : 9	YES

Fig. 5.15 View of detection server to detect attack

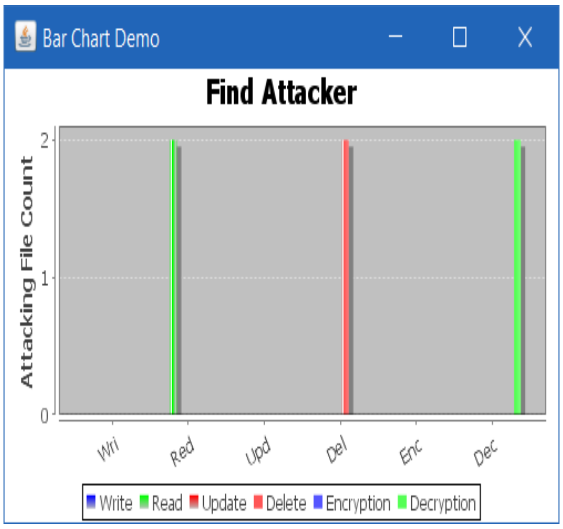


Fig. 5.16 View of find attacker graph

VI. RESULTS

The mining server analyzes the log data with data mining techniques to identify the user's computer usage habits as his/her behavior patterns, which are then recorded in the user's user profile. The detection server compares users' behavior patterns with those SC-patterns collected in the attacker profile, called attack patterns, and those in user profiles to respectively detect malicious behaviors and identify who the attacker is in real time. When an intrusion is discovered, the detection server notifies the SC monitor and filter to isolate the user from the protected system. The purpose is to prevent him/her from continuously attacking the system. Both the detection server and the mining server are run on the local computational grid to accelerate the IIDPS's online detection and mining speeds and enhance its detection and mining capability. If a user logs in to the system by using another person's login pattern, the IIDPS identifies who the underlying user is by computing the similarity scores between the user's current inputs, i.e., SCs, and the behavior patterns stored in different users' user profiles. This system can detect the insider attacker and thus help in intrusion detection.

VII. CONCLUSION AND FURURE SCOPE

In this paper, an IIDPS is developed to detect insider attacks at SC level by using data mining and forensic techniques. The experimental results show that the IIDPS can effectively resist several aforementioned attacks. This process confirms that data mining and forensic techniques used for intrusion detection provide effective attack resistance and also shows IIDPS may detect inaccurately when user's habit suddenly changes. Nevertheless, in most cases, the IIDPS can still identify the legality of a login user. When a user inputs a command, hundreds or thousands of SCs will be generated. Analyzing a huge number of SCs often takes a long time. The IIDPS spends0.45 s to identify a user. Although other systems consume longer time for data analysis than the IIDPS does, how to mine SCs in an efficient method should be addressed. Employing a local computational grid can accelerate the processing speed of the miming server and detection server. A mathematical analysis on the IIDPS's behaviors is helpful in deriving its formal performance and cost models, with which users can predict performance and cost of the IIDPS before using it. This can also detect malicious behaviors for systems employing GUI interfaces and then prevent the protected system from being attacked. The proposed model can be further used to increase detection accuracy and improve the decisive rate.

REFERENCES

[1] S. Gajek, A. Sadeghi, C. Stuble, and M. Winandy, "Compartmented se-curity for browsers— Or how to thwart a phisher with trusted computing," in Proc. IEEE Int. Conf. Avail., Rel. Security , Vienna, Austria, Apr. 2007, pp. 120–127.
[2] C. Yue and H. Wang, "BogusBiter: A transparent protection against phishing attacks," ACM Trans. Int. Technol., vol. 10, no. 2, pp. 1–31, May 2010.
[3] Q. Chen, S. Abdelwahed, and A. Erradi, "A model-based approach to self-protection in computing system," in Proc. ACM Cloud Autonomic Compute. Conf., Miami, FL, USA, 2013, pp. 1–10.
[4] F. Y. Leu, M. C. Li, J. C. Lin, and C. T. Yang, "Detection workload in a dynamic grid-based intrusion detection environment," J. Parallel Distrib. Comput., vol. 68, no. 4, pp. 427–442, Apr. 2008.
[5] H. Lu, B. Zhao, X. Wang, and J. Su, "DiffSig: Resource differentiation based malware behavioral concise signature generation," Inf. Commun. Technol., vol. 7804, pp. 271–284, 2013.
[6] Z. Shan, X. Wang, T. Chiueh, and X. Meng, "Safe side effects commit-ment for OS-level virtualization," inProc. ACM Int. Conf. Autonomic Comput., Karlsruhe, Germany, 2011, pp. 111–120.
[7] M. K. Rogers and K. Seigfried, "The future of computer forensics: A needs analysis survey," Comput. Security, vol. 23, no. 1, pp.12–16, Feb. 2004.
[8] J. Choi, C. Choi, B. Ko, D. Choi, and P. Kim, "Detecting web based DDoS attack using MapReduce operations in cloud computing environment," J. Internet Serv. Inf. Security, vol. 3, no. 3/4, pp. 28–37, Nov. 2013.
[9] Q. Wang, L. Vu, K. Nahrstedt, and H. Khurana, "MIS: Malicious nodes identification scheme in network-coding-based peer-to-peer streaming," in Proc. IEEE INFOCOM, San Diego, CA, USA, 2010, pp. 1–5.
[10] Jonathon T. Giffin, SomeshJha, and Barton P. Miller "Automated Discovery of Mimicry Attacks", 2006.