TNC Transactions on Networks and Communications
SOCIETY FOR SCIENCE AND EDUCATION UNITED KINGDOM
VOLUME 2, ISSUE 5
ISSN: 2054 - 7420

An Organizational Role-based Extrusion Detection Model with Profile Migration

Tirthankar Ghosh, Kasun Abeykoon and Thusith Abeykoon
Department of Computer Science and Information Technology
College of Science and Engineering
St. Cloud State University
tghosh@stcloudstate.edu

ABSTRACT

Intrusion detection and prevention systems play a crucial role in the overall information security implementation of today's organizations. Traditionally, signature-based and anomaly-based detections have been the two main methods of detection and prevention techniques. Signature-based intrusion detection systems are excellent in detection and performance, but they are vulnerable to unknown threats like zero-day attacks. Extensive research have been conducted on anomaly detection and prevention based on users' behavior profiling. However, as insider attacks increase, it has become equally important to monitor and analyze extrusion attempts. Behavior-based profile creation has a promising future in extrusion monitoring. However, profiling individual behavior has its limitations in that it tends to incorporate unintended behavior into the normal profile. In this study, user's organizational role has been integrated into profile creation further reducing number of false positives. A prototype of the model is tested with three users belonging to three different roles. A profile migration scheme is proposed to import user profiles at various login locations.

Keywords – Extrusion detection, Role-based profile modeling, Profile migration

1 Introduction

For many years, security experts have given much priority to increase network perimeter security because most attacks were targeted on breaching perimeter security. But with the development of technologies like mobile computing, cloud computing, distributed processing and increasing reliance on web-based applications, cyber-attacks are becoming more and more complex and advanced with time. Even though it is extremely important to protect the networks from external attacks, internal attacks can do far more damage as these attacks come from trusted insiders.

Attacks do not always occur in the same pattern. Attackers will try different kinds of methods to achieve their goal. Intrusion Detection and Prevention Systems (IDPSs) traditionally use signature-based approach, which is a very effective method and achieves better performance in detecting known threats; however they are ineffective in detecting new and unknown attacks. To overcome this, researchers have developed anomaly based IDPSs, which use behavior analysis to detect intrusion. Using anomaly based detection, IDPSs compare current system, user, or network status with a previously

DOI: 10.14738/tnc.25.473
Publication Date: 23th September 2014
URL: http://dx.doi.org/10.14738/tnc.25.473

created profile to detect anomalous behaviors. The main drawbacks in anomaly-based detection methods are managing large number of profiles, increasing false negatives and false positives.

Anomaly based IDPSs create user profiles to identify each user in the system. These systems consist of a learning phase and a detection phase. In the learning phase they gather data from each user to create profiles, and in the detection phase they compare users' data in real time with previously gathered information stored in the profiles. The main drawback in this process is that there is no method to validate actions done by the users in the learning process. Therefore, unauthorized actions might be added to the users' profiles. To overcome this issue, the users' organizational role based on their job function gets introduced to the anomaly profile creation process. A group profile is created for each role by analyzing users' data within a specific group. Since there are standard authorized activities a user can do in a specific job role, the individual unauthorized activities will be ignored during profile creation. A good framework to build anomaly profiles for large organizations was introduced by role based profile analysis [1] which use both role based profiles and individual profiles to detect attacks. This research has extended their model in creating combined role-based and individual-based profiles by designing and implementing an algorithm to detect deviations.

The authors of this paper have also presented a prototype to migrate user profiles at multiple login locations. Most of the previous work related to IDPS user profiles have been done to create those profiles, and not much research is present in the literature involving their transferability. The need of designing a scheme to migrate user profiles within an organization has also been addressed in this research.

The rest of the paper is organized as follows. Section 2 discusses some relevant literature in the context of anomaly detection and profile migration, section 3 describes our design and prototype implementation, section 4 discusses the results, and section 5 describes our profile migration scheme. Finally, section 6 concludes the paper.

2 Literature Review

Research in anomaly based profile creation is a rapidly developing area in the information security field. Researchers use a wide range of methodologies to develop profiles to obtain high accuracy detection. One of the main issues in maintaining user profiles is how to handle users' behavior change over time. This is also known as Concept Drift [2], and anomaly profiles should be up-to-date to detect the new user behavior. A dynamic normal profiling system [3] was introduced to solve this problem and the special algorithms SVM (Support Vector Machine) and FLORA 2 (Floating Rough Approximation) were used to update user profile during extended period of time. SVM algorithm creates normal profiles by filtering irrelevant data, and FLORA2 uses a dynamic sliding window to update patterns with time. Multiple classifiers were used to improve performance in anomaly-based detection by using the Kyoto 2006+ dataset obtained from honeypots at the Kyoto University [4]. A Multi-Level Intrusion Detection System (ML-IDS) was proposed in [8] to use three parameters to detect intrusions: traffic flow, packet header, and payload. Instead of obtaining results from individual parameters, ML-IDS used a decision fusion technique to get better results. Another study was conducted in [5] by observing number of running applications, number of open windows, application performance, websites viewed, and keystroke analysis.

Further research have shown that intrusion detection on encrypted sessions is difficult because the system has to decrypt it before analyzing the data. Protomon [6] is one of the unique anomaly based IDS which was developed to detect malicious behaviors in cryptographic and application level protocols in encrypted sessions. Casual Relation Minor framework (CR-Miner) was developed to identify anomalous events on a host [9], by discovering the relations between user activities and networks traffic.

Since Role Based Access Control (RBAC) [7] was introduced, IDPSs have integrated it to get better performance and accurate detection. The advantages of role based access control along with scalable administration, separation of duties and least privilege were all taken into account when developing the Composite Role Based Monitoring (CRBM) system [10]. The main advantage of CRBM is it maps vertically and horizontally between different levels and domains. CRBM also used user's organizational role, applications, and operating systems in its behavior monitoring system. One of the other main technologies integrating RBAC is database management systems. Database traces stored in log files were used, to model user's normal behavior and identify intrusion [11]. Furthermore, they have integrated RBAC, which will reduce the number of user profiles to compare, and alert when a user tries to access system resources beyond the user's role permission.

Anomaly based detection systems use statistical analysis to compare current user data with predefined user profiles. Standard deviation is a commonly used measure to detect deviation. For more advanced comparisons, the Kullback-Leibler (K-L) divergence [12] has been used. K-L divergence was used to detect anomalous behaviors in wireless communication signals [13] from interferences and to analyze DNS traffic for Domain-Flux attacks from Botnets [14].

There has not been much discussion in the literature on designing an efficient communication framework for importing user profiles in an intrusion detection environment. Earlier studies suggested a communication mechanism for cooperation between multiple intrusion detection agents using a new protocol named Information Exchange Protocol to avoid limitations of intrusion detection system with a controlling center [15]. Information exchange protocol use UDP datagrams for its communications. It also uses four kinds of event formats in exchanging messages as event receiving, event sending, announcement receiving and announcement sending. The Intrusion Detection Working Group (IDWG) suggested a protocol called Intrusion Detection Exchange Protocol (IDXP) to enable communication between intrusion detection systems [16] over a connection oriented protocol. Also, IDXP is an application-level protocol which supports confidentiality, integrity and mutual authentication between intrusion detection systems. IDXP supports exchange of data in a structured or unstructured format. Structured data includes messages in a format defined in Intrusion Detection Message Exchange Format (IDMEF), while unstructured data includes unstructured text and binary data.

All of these protocols work to exchange data between intrusion detection systems, not between other systems. Therefore, the importance of a data exchange mechanism between intrusion detection systems and other devices such as an authentication server is growing.

3 Design and Implementation

The basic premise of the model is to integrate the role-based profile with the individual-based profile. Four parameters are chosen to represent the profiles, two for each profile: CPU and memory utilization for the individual profile, and the number of processes and network connections for the role-based profile. The deviation is measured by the Kullback-Leibler (K-L) divergence as given below in Equation 1.

The Kullback-Leibler divergence is a statistical function used to measure the proximity of two discrete probability distributions. If P and Q are two probability distributions, K-L divergence is give by Equation 1 below:

$$D_{KL}(P||Q) = \sum_i \ln(\frac{P(i)}{Q(i)})P(i) \quad (1)$$

K-L divergence satisfies the following properties:

- *Is always a positive number*
- *Is equal to zero if P(i) = Q(i) (If two distributions are identical)*
- *Is not defined when P(i) ≠ 0 and Q(i) = 0*

Instead of using Standard Deviation (SD), which measures how much the new data set is deviated from the mean, K-L divergence is used in this study to get a more precise comparison with user profiles and current data. A value of 0.5 is added to the values before calculating the K-L value to eliminate P(i) ≠ 0 and Q(i) = 0 [13].

Our model is divided into three sections, the learning phase, the profile creation phase, and the detection phase. The learning phase describes how the data is gathered and saved to create user profiles. The profile creation phase describes how individual and role-based baseline profiles are created. Lastly, the detection phase detects intrusions in real time by measuring deviation from baseline profiles. Each phase is described in details below.

3.1 Learning phase

In the learning phase the system collects user's CPU usage and memory usage (for individual profiles), and number of running processes and number of established network connections (for role-based profiles) every five seconds. These data are grouped into one-minute sections to find the average values for every minute and every third minute. The flowchart is shown in Figure 1.

Each value will be recorded and stored in a file for use during the profile creation phase.

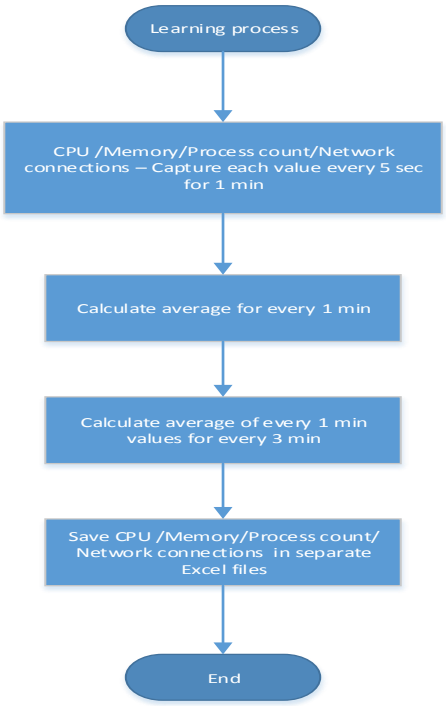

Figure 1 Learning phase flow chart

3.2 Profile creation phase

In the profile creation phase, the system reads each user's CPU and memory data and creates an individual profile for each user. Simultaneously, the system collects all users' process and network usage data based on their organizational role. Separate role-based profiles are created for each role. The flowchart is shown in Figure 2.

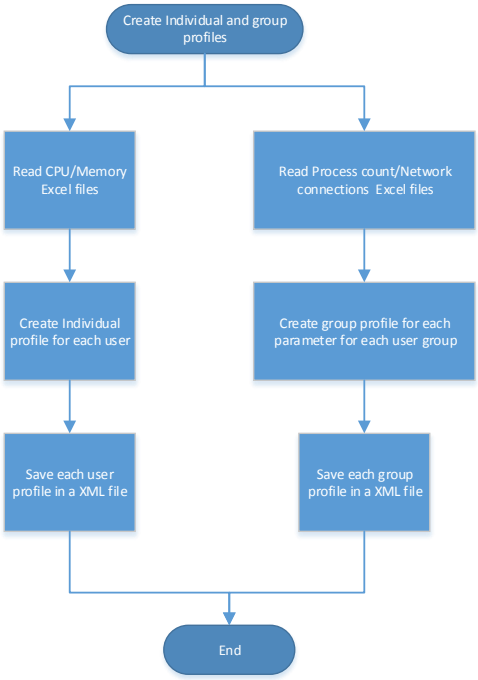

Figure 2 Profile creation phase flow chart

3.3 Detection phase

In the detection phase, before running the actual detection process, a trial detection process will run, which includes several tests to find the limits and thresholds of K-L values. The detection phase is divided into three sub phases as described below.

In the first phase, the system reads the user's individual and role-based profile data and stores them in memory. Then the system starts and captures the user's real-time CPU usage, memory usage, number of running processes, and the number of established network connections for every five seconds. While the system is computing K-L values by analyzing this data, several test tools are used to generate high CPU activity along with memory, process and network utilizations. This method will help to define what the lowest and highest K-L values are when the system is in normal use. It also shows how K-L values increase with high CPU, memory, process and network utilizations, and finally, to define the threshold K-L values for each parameter. The overall algorithm is shown in Figure 3.

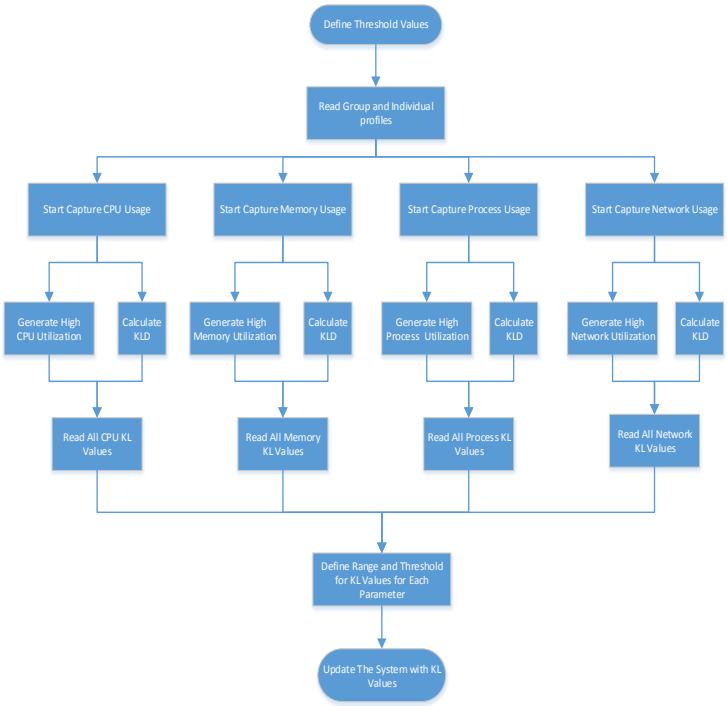

Figure 3 First detection phase flow chart

Before starting the first phase, the minimum utilization, maximum utilization, and intervals between them are defined for each parameter. When the process starts to capture usage of each parameter, the test tools which are used to simulate high utilization also starts from minimum utilization to a maximum utilization. If the normal system utilization is higher than the predefined minimum utilization, the test tool will start to simulate from normal system utilization. All these values are recorded in a file. At the end of this process, for each parameter, the K-L values when the system runs in normal behavior and in intervals between normal and maximum system utilizations are identified and inserted into the system to run in the next phase. The K-L value when the system runs in normal utilization will be the baseline K-

L value for each parameter, and the K-L values for intervals above the baseline will be used as the threshold values.

In the second phase, the system loads the user profiles first and starts to capture user data. At the end of each minute, the system compares data with the profile and then computes the K-L value and compares it with the current threshold K-L value. All observations during normal system behavior with K-L values greater that this threshold are flagged as false positives; all observations under stressed behavior with K-L values lower than the threshold are flagged as false negatives. False positives and negatives are plotted with increasing threshold values to find the crossover point. The algorithm is shown as a flowchart in Figure 4.

In the final phase, the threshold values are added systematically from the second phase. Then the system again starts to capture the user's real time data to compute K-L values. The inputs are passed through the individual profile-matching engine to analyze their behavior and measure deviation. If there are any deviations from the predefined threshold, output from the individual profile-matching engine are passed onto the role-based profile-matching engine. If deviation is detected in the second phase, the incident is marked as anomalous. The algorithm is shown in Figure 5. The rationale for using individual profile-matching engine before role-based profile-matching engine is as follows: if the role-based profile is checked first with the number of connections and number of processes running, a deviation may be passed onto the second stage with a higher number of process running, which may not necessarily mean that CPU and memory utilization will be higher. That will not be flagged as a deviation at the second stage. On the other hand, if the individual profile is checked first with the CPU and memory utilization, any deviation with high utilization will have a higher chance of getting flagged as a deviation at the second stage, as higher utilization may have been caused by more processes or more connections, thus reducing overall false negatives. In this phase, the system knows the K-L values for normal and suspicious usage, and from the K-L output of user, the system can define whether the current behavior is suspicious or not.

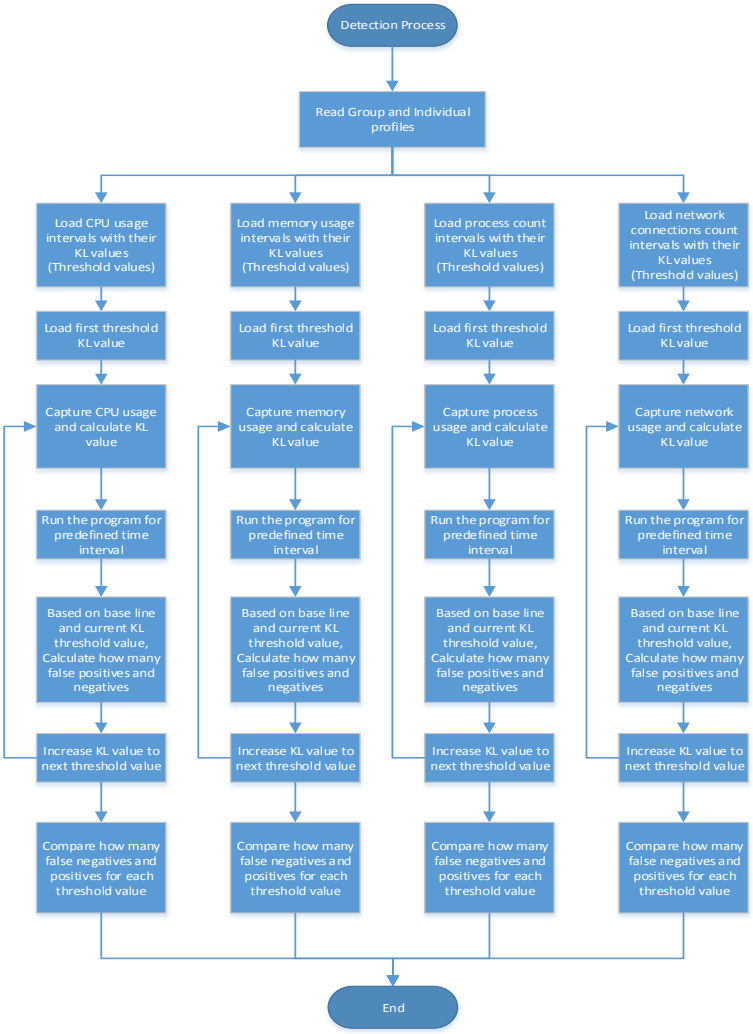

Figure 4. Second detection phase flow chart

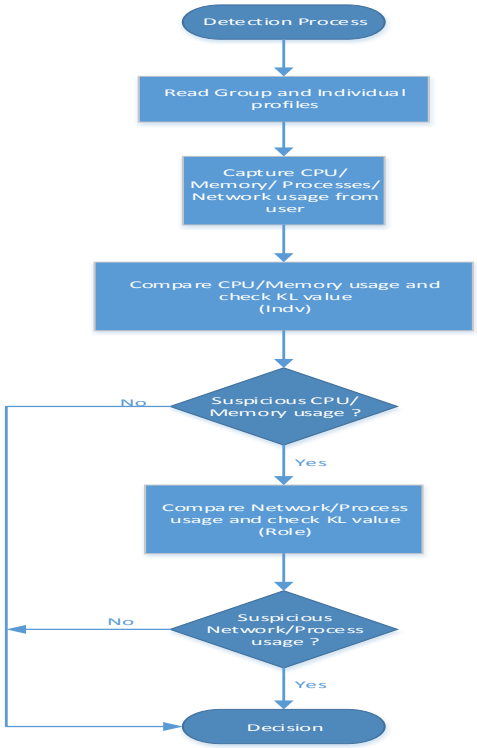

Figure 5. Final detection phase flow chart

4 Discussion of Results

Python 2.7 was used as the primary programming language to collect data for creating individual and role-based profiles. These profiles were saved as XML documents. Three users were selected belonging to different roles, and asked to run the programs in order to capture their CPU activity, memory usage, as well as process and network usages. Data was collected for three weeks in a supervised environment, and each user's individual and role-based profiles were created. One challenge was to determine the appropriate threshold value for using in the detection phase. We stressed the systems by injecting high CPU, memory, network connections, and processes, and recorded the parameters. Based on whether each value is between the baseline and threshold value or above the threshold value, the outcomes are defined as either a false negative or a false positive respectively. Plotting false positives and negatives with increasing threshold, we were able to find the optimum point, and use that as our predefined threshold. The tests ran for several days to capture user data and were grouped by each day.

4.1 Individual Profiles

Figures. 6, 7, and 8 show each user's CPU usage. For the first user, the normal CPU usage is 10% or below, and the maximum usage is generally 50%. The user's threshold K-L value is between 10-20%. The optimal K-L value is 0.6, and the relative CPU usage is 15.69%.

For the second user, the normal CPU usage is 20% or below, and the maximum usage is generally 50%. The user's threshold K-L value is between 30-40%. The optimal K-L value is 0.92, and the relative CPU usage is 32.09%.

For the third user, the normal CPU usage is 30% or below, and the maximum usage is generally 90%. The user's threshold K-L value is between 40-50%. The optimal K-L value is 0.45, and the relative CPU usage is 44.47%.

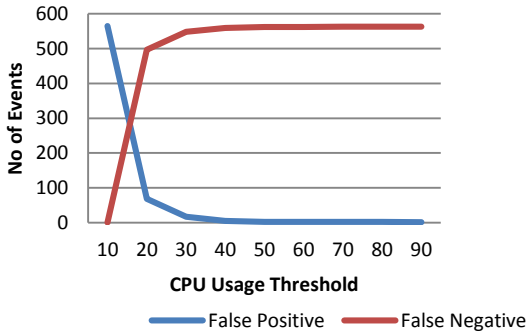

Figure 6. Optimum threshold for the first user's CPU usage

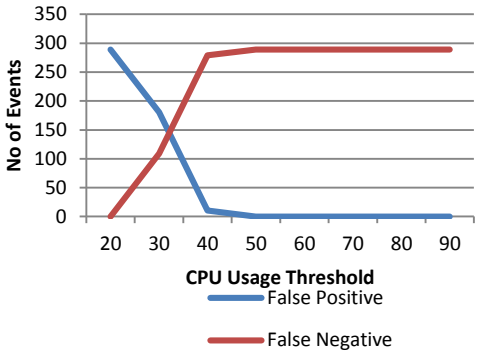

Figure 7. Optimum threshold for the second user's CPU usage

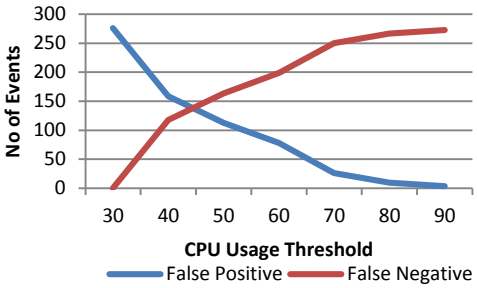

Figure 8. Optimum threshold for the third user's CPU usage

Figures 9, 10, and 11 show each user's memory usage. For the first user, the normal memory usage is 60% or below, and the maximum usage is generally 90%. The user's threshold K-L value is between 60-70%. The optimal K-L value is 1.56, and the relative memory usage is 65.69%.

For the second user, the normal memory usage is 70% or below, and the maximum usage is generally 80%. The user's threshold K-L value is between 70-80%. The optimal K-L value is 1.275, and the relative memory usage is 75%.

For the third user, the normal memory usage is 50% or below, and the maximum usage is generally 90%. The user's threshold K-L value is between 60-70%. The optimal K-L value is 2.0, and the relative memory usage is 66.9%.

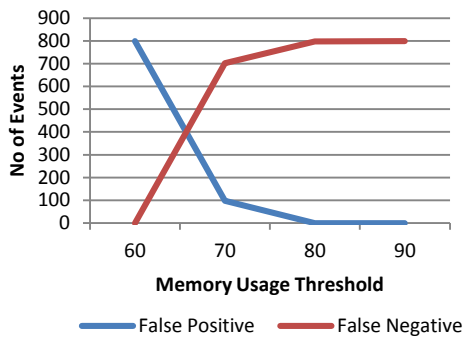

Figure 9. Optimum threshold for the first user's memory usage

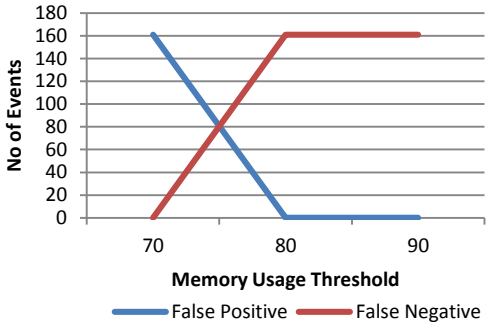

Figure 10. Optimum threshold for the second user's memory usage

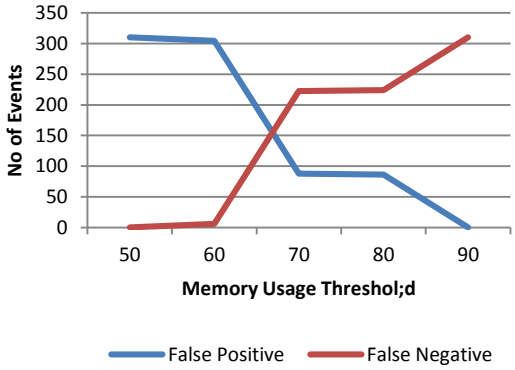

Figure 11. Optimum threshold for the third user's memory usage

4.2 Role-based Profiles

Role-based profiles show each user's process (application) and network connections usage, where each user is selected to represent a different organizational role.

Figs. 12, 13, and 14 show each group's (role) process usage. The normal process usage for the first group is 120 processes or below, and the maximum usage is generally 130 processes. The user's threshold K-L value is between 120 and 130 processes. The optimal K-L value is 1.362, and the relative application usage is 125 applications.

The normal process usage for the second group is 80 processes or below, and the maximum usage is generally between 100 to 150 processes. The user's threshold K-L value is between 90 and 100 processes. The optimal K-L value is 1.4, and the relative application usage is 97 applications.

The normal process usage for the third group is 80 processes or below, and the maximum usage is generally 100 processes. The user's threshold K-L value is between 80 and 90 processes. The optimal K-L value is 1.362, and the relative application usage is 85 applications.

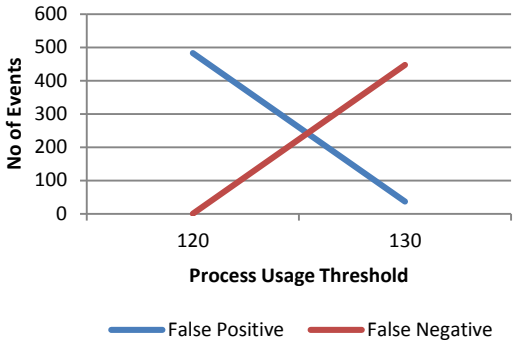

Figure 12. Optimum threshold for the first group's process usage

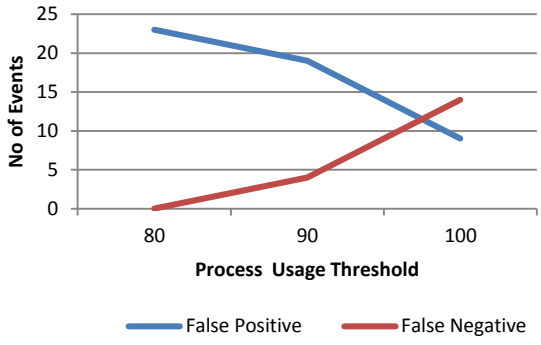

Figure 13. Optimum threshold for the second group's process usage

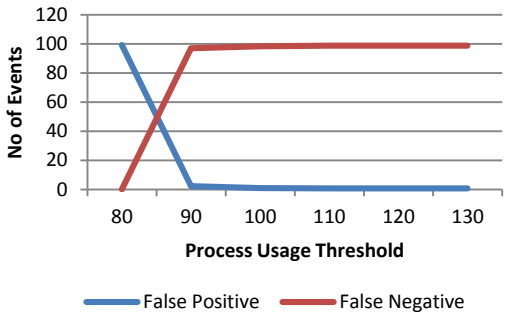

Figure 14. Optimum threshold for the third group's process usage

Figures 15, 16, and 17 show each group's (role) network usage. The normal network usage for the first group is 30 connections or below, and the maximum usage is generally 120 connections. The user's threshold K-L value is between 40 and 50 connections. The optimal K-L value is 2.4, and the relative network usage is 41 network connections.

The normal network usage for the second group is 20 connections or below, and the maximum usage is generally 140 connections. The user's threshold K-L value is between 20 and 40 connections. The optimal K-L value is 2.49, and the relative network usage is 38 network connections.

The normal network usage for the third group is 10 connections or below, and the maximum usage is generally 90 connections. The user's threshold K-L value is between 30 and 40 connections. The optimal K-L value is 1.7, and the relative network usage is 33 network connections.

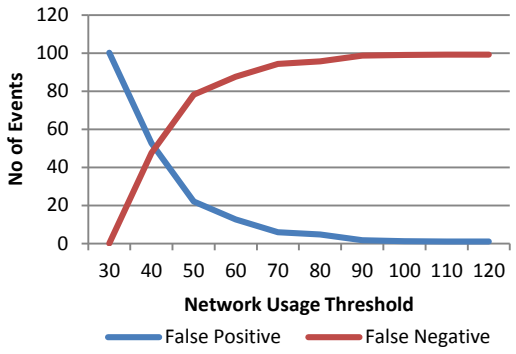

Figure 15. Optimum threshold for the first group's network usage

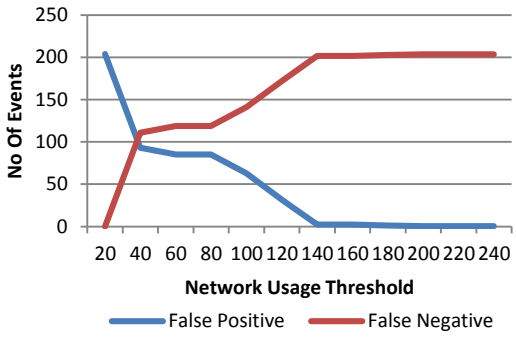

Figure 16. Optimum threshold for the second group's network usage

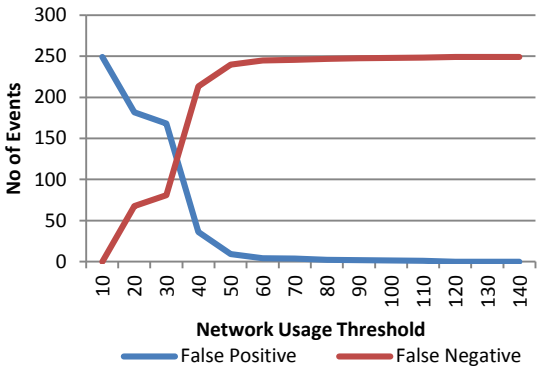

Figure 17. Optimum threshold for the third group's network usage

4.3 Final detection and false positive analysis

Our algorithm uses the individual profile-matching engine before the role-based profile-matching engine (rationale already given in section 3.3). In this section we analyze false positives, and show that our approach succeeds in reducing false positives. The analysis is done on data collected during the detection phase with real-time data under supervised condition with no malicious activity.

Table 1 summarizes results for the first user. It can be seen that about 14% of events were flagged as intrusion in the first stage by the individual-based detection engine, while only 4% of events were flagged as intrusion by the system. This significantly reduces false positives. Similarly, tables 2 and 3 summarize results for the second and third users respectively. Flagged events were reduced from 18% to null for the second user, and from 53% to 3.5% for the third user. These tests show that our system is capable of significantly reducing false positives.

Table 1. False Positive Analysis for the First User

	No of Events	Percentage
Total	2453	
Events passed Individual profile test	2104	85.70%
Events failed Individual profile test but passed group profile test	250	10.19%
Events failed both tests	100	4.07%

Table 2. False Positive Analysis for the Second User

	No of Events	Percentage
Total	1078	
Events passed Individual profile test	878	81.44%
Events failed Individual profile test but passed group profile test	200	18.55%
Events failed both tests	0	0%

Table 3. False Positive Analysis for the Third User

	No of Events	Percentage
Total	958	
Events passed Individual profile test	451	47.07%
Events failed Individual profile test but passed group profile test	473	49.37%
Events failed both tests	34	3.54%

5 Profile Migration Scheme

There is a need to migrate user and group profiles, as users are becoming mobile and login to the domain from various locations. We have proposed such a scheme to migrate these profiles. Also to move the user profiles from one location to another, they have to be saved in one or more locations where they can be accessed from any host within the organizational network. These locations should act like data-stores, which could house all user profiles in specific format and profiles should be identified according to the user credentials.

Our design has three main components: client, authentication server, and data store. Client is the user entry point to the organizational network. Mainly this would be a computer where users could log into

the organization network and their credentials will be authenticated by an authentication server. After the profiles are moved to the client machine, they are being used by the IDPS system residing in the client machine.

The authentication server is the main device that glues the system togeather. It communicates with both client and the data store, and transfers profiles to the client. It also authenticates each user logging into the host machine. The main tasks of this server are to authenticate the user, apply any policies related to the user, and transfer the user profiles (both individual and role-based) from the data store to the client.

The data store is the repository of IDPS user profiles. These user profiles are indexed according to any user credentials such as user name or user id. Each user profile will be saved as an XML file to be supported by the data store. It would be an added advantage if the data store supports any regular expression based searches within the data store. The data store only communicates with the authentication server to transfer profiles.

The process flow is shown in Figure 18 below. Users are assumed to connect to the organization's network domain for authentication by the domain controller. The process starts from the user. Initially, the user provides login credentials to the client machine. The client machine initiates the login process, and provides the supplied credentials to the authentication server. On success, the authentication server initiates the profile transfer process to identify and transfer the IDPS individual and role-based profiles to the client machine. After the user related processes are done, the user login process will complete its task and the user can access the machine.

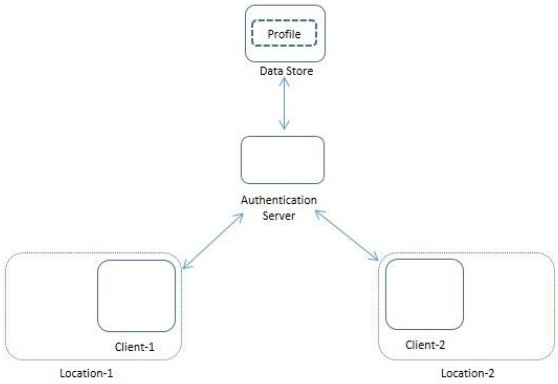

Figure 18. Process flow for user profile migration scheme

The design was implemented using Windows Server 2003 as the directory server (domain controller), and Microsoft SQL Server 2008 as the data store, all running on virtual hypervisors. VMWare Workstation was used as the virtual hypervisor. Selecting a platform for a data store was based on several considerations. First was a directory server, second was a database server, and third was storing the profiles in a file system. Storing the profiles in a directory server was not considered because of two reasons; one, it would add an extra load to the current directory servers; two, adding the profile data in a seperate system will make the data store an isolated system which can be managed and controlled seperately without entering to the directory server. Selecting a database as a data store rather than a

flat file system or directory service provides several advantages such as easier indexing, iplementing security, and open to sql queries for easier data searching and manipulation.

One of the initial requirements of the design is to transfer the profiles which are saved in the data store to the host machine. First, the profile requesting component, which is the client or the server, has to identify the logon process and initiate a communication channel to the data store to get the profiles. Then the data store has to be queried to get the specific users' profiles. Since the profiles are indexed according to user credentials such as user name or user id, the initiating component has to have the requesting user details with the query. A successful query will identify the profiles (both individual as well as role-based) for the user which will be transferred to the client machine to be used by the IDPS. Since the authentication server runs as a directory server, it has the ability to identify the logon process for a user. Also since the data store works as a database, a database client tool will help to connect the datastore and run a query to get the profiles. The client tool should have the user's id to get the profiles. To complete this process a Visual Basic script was written to connect to the database and get the profiles. This script stays in the "sysvol" directory on the directory server and runs on the client machine when the user logs on. The script is executed as a part of the user logon process, and executes the database client tool to connect to the data store and query the user's profiles. If found, the script gets the profiles and transfers them on to the client machine.

6 Conclusion and Future Work

As insider attacks are growing, anomaly-based extrusion detection systems are becoming popular in many organizations. In this research we have designed such a system based on integration of individual profiles and organizational role-based profiles. We have implemented a prototype of the system, and demonstrated that our system is capable of reducing false positives significantly. We have also implemented a profile migration scheme, which helps to migrate users IDPS profiles to various locations within and outside the organizational boundary as long as the users log in to the domain.

A future extension of this research will be to make the user profiles (both individual and role-based) dynamic based on a feedback mechanism. Adding parameters such as user's geographical location and devices used will also help in building better profiles.

REFERENCES

[1]. Park, J. S., & Giordano, J., "Role-Based Profile Analysis for Scalable and Accurate Insider-Anomaly Detection", *25th IEEE International Performance, Computing, and Communications Conference*, pp. 463-470, 2006.

[2]. Wang, S., Schlobach, S. & Klein, M. C. A., "What Is Concept Drift and How to Measure It", In P. Cimiano & H. S. Pinto (eds.), *EKAW* , pp. 241-256, : Springer, 2010.

[3]. Kim, J. Y., Gantenbein, R. E., & Sung, C. O., "Dynamic Normal Profiling for Anomaly Detection Systems", in *Proc. 3rd Int. 3rd International Conference on Convergence Technology and Information Convergence*, pp. 27-32, 2008.

[4]. Kishimoto, K., Yamaki, H., & Takakura, H., "Improving Performance of Anomaly-based IDS by Combining Multiple Classifiers", in *Proc. IEEE/IPSJ International Symposium on Applications and the Internet*, pp. 366-371, IEEE Computer Society, 2011.

[5]. Pannell, G., & Ashman, H., "Anomaly Detection over User Profiles for Intrusion Detection", *Australian Information Security Management Conference,* Perth, Australia, 2010.

[6]. Joglekar, S. P., & Tate, S. R., "ProtoMon: Embedded Monitors for Cryptographic Protocol Intrusion Detection and Prevention", in *Proc. International Conference on Information Technology: Coding and Computing,* pp. 81- 88, 2004.

[7]. Ferraiolo, D. F., & Kuhn, D. R., "Role-Based Access Controls", in *Proc. 15th National Computer Security Conference*, pp. 554 - 563, 1992.

[8]. Al-Nashif, Y., Kumar, A. A., Hariri, S., Luo, Y., Szidarovsky, F., & Qu, G., "Multi-Level Intrusion Detection System (ML-IDS)", in *Proc. International Conference on Autonomic Computing*, pp. 131-140, 2008.

[9]. Zhang, H., Banick, W., Yao, D., & Ramakrishnan, N., "User Intention-Based Traffic Dependence Analysis for Anomaly Detection", *in Proc. IEEE Symposium on Security and Privacy Workshop*, pp. 104-112, 2012.

[10]. Park, J. S., & Ho, S. M., "Composite Role-Based Monitoring (CRBM) for Countering Insider Threats", in *Proc. Second Symposium on Intelligence and Security Informatics*, pp. 201-213, Heidelberg, Germany, 2004.

[11]. Kamra, A., Terzi, E., & Bertino, E., "Detecting Anomalous Access Patterns in Relational Databases", *The VLDB Journal — The International Journal on Very Large Data Bases*, vol 17(5), pp. 1063-1077, August 2008.

[12]. Kullback, S., Leibler, R. A., "On Information and Sufficiency", *The Annals of Mathematical Statistics*, vol 22(1), pp. 79-86, March 1951.

[13]. Afgani, M., Sinanovic, S., & Haas, H., "Anomaly Detection using the Kullback-Leibler Divergence Metric", in *Proc of the 1st International Symposium on Applied Sciences in Biomedical and Communication Technologies (ISABEL 08), 2008.*

[14]. Yadav, S., Reddy, A. K. K., Reddy, A. L. N. & Ranjan, S., "Detecting Algorithmically Generated Domain-Flux Attacks With DNS Traffic Analysis", *IEEE/ACM Trans. Netw.*, 20, pp. 1663-1677, 2012.

[15]. Zhang , W., Wang , L., Fu , X., & Teng, S., "Research on Communication Mechanism Among Cooperating Multi-Intrusion Detection Agents", *ICCI '06 Proceedings of the 2006 5th IEEE International Conference on Cognitive Informatics*, *pp.* 743-748, 2006.

[16]. Feinstein, B., & Matthews, G., "RFC4767 *The Intrusion Detection Exchange Protocol (IDXP)*", retrieved from http://www.ietf.org/rfc/rfc4767.txt, March 2007.