Intrusion Detection Using Log Server and Support Vector Machines

Donghai Guan[O], Donggyu Yeo, Juwan Lee, Dukwhan Oh
Dept. of Computer Engineering, Kumoh National Institute of Technology
{eastsea[O], sylot, wanirang, dhoh}@se.kumoh.ac.kr

Abstract

With the explosive rapid expansion of computer using during the past few years, security has become a crucial issue for modern computer systems. Today, there are many intrusion detection systems (IDS) on the Internet. A variety of intrusion detection techniques and tools exist in the computer security community such as enterprise security management system (ESM) and system integrity checking tools. However, there is a potential problem involved with intrusion detection systems that are installed locally on the machines to be monitored. If the system being monitored is compromised, it is quite likely that the intruder will alter the system logs and the intrusion logs while the intrusion remains undetected. In this project KIT-I, we adopt remote logging server (RLS) mechanism, which is used to backup the log files to the server. Taking into account security, we make use of the function of SSL of Java and certificate authority (CA) based key management. Furthermore, Support Vector Machine (SVM) is applied in our project to detect the intrusion activities.

1.Introduction

As computer systems play increasingly vital roles in modern society, they have become the target of intrusions by our enemies and criminals. In addition to intrusion prevention techniques, such as user authentication (e.g. using passwords or biometrics), avoiding programming errors, and information protection (e.g., encryption), intrusion detection is often used as another wall to protect computer systems.

Intrusion detection (ID) is defined as "the problem of identifying individuals who are using a computer system without authorization (i.e., 'crackers') and those who have legitimate access to the system but are abusing their privileges (i.e., the 'insider threat')"[1].
Intrusion detection can be categorized into two main models [2]:

Misuse detection model: Detection is performed by looking for the exploitation of known weak points in the system, which can be described by a specific pattern or sequence of events or data.

Anomaly detection model: Detection is performed by detecting changes in the patterns of utilization or of behavior of the system. It is performed by building a model that contains metrics derived from system operation and flags as intrusive any observations that have a significant deviation from the model.

IDSs are usually classified as host-based or network-based. Host-based systems base their decisions on information obtained from a single host (usually audit trails), while network-based systems obtain data by monitoring the traffic of information in the network to which the hosts are connected [3].

Today, we can find many IDSs on the Internet. Most of the IDSs adopt different mechanisms and functions, however, especially in different organizations. It is difficult to communicate between these different systems. To cope with this problem, the enterprise security management (ESM), which main function is to provide convenient framework for easy interoperation/application of the security system, serving various purposes within the enterprise, has been developed and widely used. Considering the necessary of different ESMs to communicate, the IDWG of IETF has developed intrusion detection message exchange format (IDMEF) and intrusion detection exchange protocol (IDXP) for it.

Though these IDSs follow different approaches for intrusion detection, log files analysis has been used as an indispensable component since it can record most of the user activities. So we should try our best to prevent it from intruder's attacking. In this project KIT-I, we suggest the RLS mechanism to securely backup the log files to the server.

2. Related Work

With the development of computer security science, we can find many intrusion detection techniques and tools such as the intrusion detection message exchange format (IDMEF) and intrusion detection exchange protocol (IDXP). As for IDMEF and IDXP, their main function is to specify the format of message and exchange between different identities. Though many techniques follow different approaches for intrusion detection, log files analysis has been used as an indispensable component [4]. Because almost all the activities of the users are recorded in the log files, we can get enough information from it for intrusion detection system. Analysis of the log file is important even after an attack has occurred, for determining the extent of damage incurred. Also, this analysis helps in tracking attackers and in recording attack patterns for future prevention.

There are many methods have been used for the intrusion detection systems such as data mining [5][6], expert system [7] and neural network [8][9][10], which have been successfully used in the research area and practice.

However, there is a big challenge for a system, which only installed IDS on the local machine. If an intruder intrudes a system and changes the log file successfully, at that time, the data are not reliable and all the processing is useless. For solving this problem, we present a way that using the remote logging server (RLS) mechanism. As for RLS, every client automatically sends its log file to the server at intervals. Even if the local machine is compromised, we can get the backup information from the logging server to detect intrusions. So this method is a rational complement for the IDS which only be installed on the local machine. In addition, the RLS mechanism can also be integrate with the enterprise security management (ESM), which provide a more efficient way to manage audit data.

3.Our Project

In this project KIT-I, we apply the RLS mechanism, which is applied to transfer the client's log file to the logging server at intervals. The main objective of RLS is to keep a duplicate copy of these crucial log files on the remote server so that in the event that some of the clients have been compromised, the system administrator can cross-reference the compromised machine's logs with the log kept in the logging server. We can regard the way that between a client and

server as a channel. It is a critical problem to assure whether the channel is reliable. If the intruder intrudes the channel and changes the data, the backup file is useless and we may get wrong information after we analyze it. As we know, there is a potential hazard that someone will intercept the data being read, masquerade as one of your applications, and fill your system with bogus data. To cope with this problem, we use the SSL capability of Java to create an encrypted channel between the server and the client and CA-based key management for verification of the client's identity. KIT-I is composed of two modules-- transferring module and SVM module. The function of the transferring module is to backup the log file through an encrypted channel. The function of the SVM module is to find abnormal activities and give a notice to the administrator. For our project, it can be extended to a part of enterprise security management (ESM).

3.1 System Architecture

As Figure 1 shows, KIT-I is composed of two function modules. The first module is transferring module, whose main function is to transfer the log files on the clients to the logging server at intervals. The second module is SVM module. As for SVM module, its function is analyzing the audit data of the log file that transferred from the clients. Considering the security, this system should be configured as a bastion host. For our project, it can be extended to a part of enterprise security management (ESM) as Figure 2 shows.

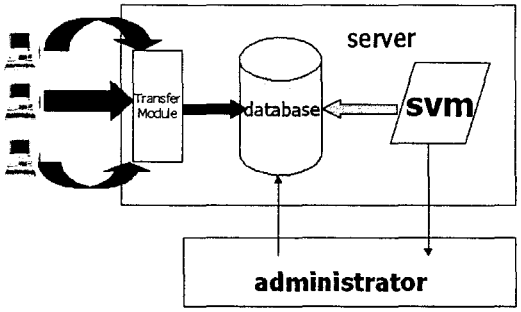

Figure 1. System Architecture

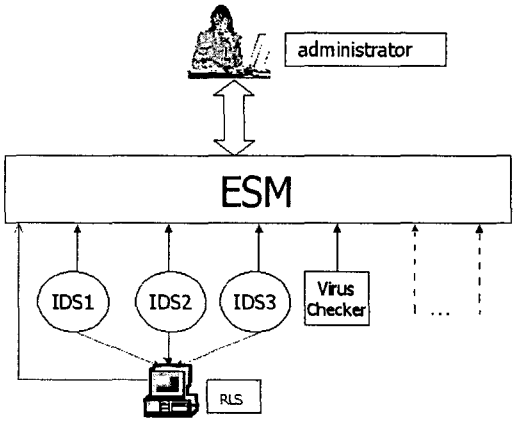

Figure 2. System Extension Architecture

The future system architecture is shown as Figure 2. In this architecture, the RLS mechanism is integrate into the enterprise security management (ESM). For the ESM, its main function is providing a convenient framework for management of security system. Considering the importance of log files, using the RLS mechanism, the ESM can provide a more efficient way for management audit data.

3.2 Transferring Module

In transferring module, it is a crucial problem to assure the transfer's reliability. Sun Microsystems has an implementation of SSL in its Java Secure Socket Extension (JSSE). The Java Secure Socket Extension (JSSE) is a set of Java packages that enable secure Internet communications. It implements a Java version of SSL (Secure Sockets Layer) and TLS (Transport Layer Security) protocols and includes functionality for data encryption, server authentication, message integrity, and optional client authentication. So in this module, we use the SSL capability of Java to create an encrypted channel between the server and the clients.

All the information exchanged between the server and the client is encrypted after the encrypted channel is established. So keys are needed to decrypt the data. As for the key, the first thing that needed to be created is a key-manager factory, which provides objects (KeyManagers) that manage particular types of key stores. Just as a key ring keeps your house keys organized, a KeyManager keeps the public and private keys needed for SSL communications organized. A KeyStore contains the keys while the KeyManager knows how to provide access to the particular keys in the key store.

An important ingredient of SSL is the ability to use public-key cryptography. Each party to the communication has two keys – one public key that can be freely sent to others and one private key that is kept by individuals. The beauty of these systems is that anything encrypted with the public key can be decrypted with the private key, and vice versa. However, data encrypted with the public key cannot be decrypted with the public key, which is why the key can be sent out freely.

The important part of the project KIT-I in the scope of the class is the CA-based key management between the client and the server. The server will not accept SSL connections from clients that use encryption keys that have not been signed by the Certificate Authority (CA), which is the logging server in this case. The public key for the client can be produced either from the client machine or the server. But this key needs to be signed by the server so that the clients can be authorized for connection to the logging server.

Until now, there have been so many ways to be applied to store keys for access by clients and servers. The easiest is to put each server's public key into the key store of each client, and each client's public key into each server's key store. This means everyone has each other's key and all can communicate with each other. The reason this works is that each person's public key signed by its private key. When the programs validate the keys they were sent, they find that the key was signed by a trusted key in their key store. However, this gets to be a real problem if the applications involved can be both client and server. Every time you bring up a new client, you potentially have to go visiting a bunch of different nodes updating keys.

Instead of putting every key in every store, in this project we create a master key that we can put it in all the keys stores when we install the application. Then we can use the master key to sign all the client and server keys we create. As in figure 3 shows, when the applications send their keys, the keys will be signed by a trusted key in the key store. Because nothing in the Java Development Kit or JSSE lets us set up a CA and sign keys. So we have to go elsewhere for tools to do this. In this project, we chose to go with the OpenSSL toolkit running on Linux.

The basic method for the key management is using the CA private key (made by CA) to sign the client' s public key and then distributing the certificates (include the CA public key) to the clients. In my case,

considering the security, the certificates distributing process is off-line using floppy disk. But in the situation that some clients are far away from the server (etc. in different countries), it is not convenient to delivery the certificate. At that time, we can adopt more than one CA and distribute the certificates between different CA.

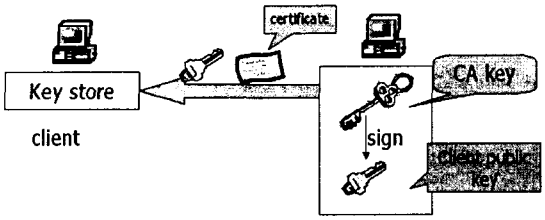

Figure 3. Key management

After the key management finished, the protocol in KIT-I is straightforward: the server and the client will exchange the certificate and validate it. After validating it, the server will accept the connecting request and then establish an encrypted channel between the client and server.

3.3 SVM module

After the server receives the log files, the SVM module begins to operate on the audit data of the log file. Since most of the intrusions can be located by examining patterns of user activities, many IDSs have been built by utilizing the recognized attack and misuse patterns. But a main issue concerning misuse detection is how to develop signatures that include all possible attacks to avoid false negatives, and how to develop signatures that do not match non-intrusive activities to avoid false positives.

There are so many algorithms have been used to cope with the problem, such as neural networks and support vector machines (SVM). The main ideas are to discover useful patterns or features that describe user behavior on a system, and use the set of relevant features to build classifiers that can recognize anomalies and known intrusions. For neural networks or support vector machines (SVM), they can be trained to learn the normal behavior and attack patterns, then significant deviations from normal behavior are flagged as attacks.

Nowadays both neural networks and SVM have been used within the security community. However, in our project, we apply SVM for our intrusion detection. There are mainly two reasons that we experiment with SVM for intrusion detection. The first is speed: as real-time performance is of primary importance to intrusion detection systems, any classifier that can potentially outrun neural networks is worth considering. The second reason is that SVM is based on linear classification, in which a small set of training data can achieve a good performance.

In our project, the SVM is trained based on the user activity record. Here we use the method mentioned by Srinivas [11] to change the activities of users to numerical value, which can be treated as the input of SVM. The activities of users are extracted from the history log file and changed to the level of different commands. Let us assume a simple attack pattern that include four commands: **su**, **reboot**, **cd** and **man**. For their different danger level, we can set the values for these four commands 10, 10, 2 and 2 respectively.

We can get the following values:

Avg (average)=(10*4+10*2+2*5+2*10)/(4+2+5+10)=4.29

HV (highest value) =10

SV (sum of value) = 10*4+10*2+2*5+2*10=90

So the input of SVM is {4.29, 10, 90}, and output is –1 (if it is not an attack pattern, the output value is 1).

After training the SVM, we can apply it for out ID.

4. Conclusions and future works

In this paper, we make a project—KIT-I aimed to provide a secure way to the log files since they are very important for intrusion detection systems. KIT-I is mainly composed of two modules—transferring module and SVM module. For transferring module, Remote logging server (RLS) mechanism is used to transfer the log files from the clients to the server at intervals. Furthermore, the SSL mechanisms of Java and certificate authority (CA) are adopted to establish an encrypted channel between the client and the server. Even if some clients are comprised, using KIT-I, we can get the backup data from the server.

SVM module is used to process the audit data of the log files and support vector machines are applied. In fact, other methods like expert systems and neural network also can do it. But compared with them, SVM has obvious advantage on speed and it is less sensitive to the size of test data.

Although this project has been aimed primarily at providing secure protect measures for the log files, we believe it can also be extended to a part of enterprise secure management (ESM). A number of future work will be done.

- The computer used as the CA is the weak link. If unauthorized access is gained to this computer, the CA key could be stolen or, at the very least, used to sign bogus keys. So if possible, put this computer in a secure room, do not connect it to a network, and carry the keys in and out on floppy disks.
- Since in this project the clients will transfer the log data to the server at intervals, we should take into account the CPU resource of the server. If the ability of server's CPU is not enough, we can add more servers in this project.
- There is a delay in this project, so in the future we should try our best to decrease the latency of it.
- We will merge IDXP and IDMEF technique into this project to make it fit with the standard of the IDS.

REFERENCES

[1] Denning D, An Intrusion-Detection Model, *IEEE Transactions on Software Engineering, Vol. SE-13, No 2, 1987*

[2] Kumar S, Spafford EH, An Application of Pattern Matching in Intrusion Detection, *Technical Report CSD-TR-94-013, Purdue University, 1994*

[3] T.Lunt, A survey of intrusion detection techniques, *Computers and security, vol. 12,pp.405-418, 1993*

[4] Shambhu J. Upadhyaya, Kevin Kwiat, A Distributed Concurrent Intrusion Detection Scheme Based On Assertions, SCS International Symposium on Performance Evaluation of Computer and Telecommunication Systems, Chicago, IL, pp.369-376, 1999

[5] Wenke Lee, Salvatore J. Stolfo, Data Mining Approaches for Intrusion Detection, *Proceedings of the 7th USENIX Security Symposium, pp.120-132, 1998*

[6] Wenke Lee, Salvatore J. Stolfo, Kui W. Mok, A Data Mining Framework for Building Intrusion Detection Models, *IEEE Symposium on Security and Privacy, 44, EE, 120-132, 1999*

[7] Koral Ilgun, State Transition Analysis: A Rule-Based Intrusion Detection Approach, Vol. 21, No 3, pp. 181-199, 1995

[8] Debar H, Becke M, Siboni D, A Neural Network Component for an Intrusion Detection System. *Proc. of the IEEE Computer Society Symposium on Research in Security and Privacy, pp.240-250, 1992*

[9] Cannady J, Artificial Neural Networks for Misuse Detection. *National Information System Security Conference, pp443-456, 1998*

[10] Vladimir VN, The Nature of Statistical Learning Theory. Springer, *Berlin Heidelberg New York,* ISBN: 0387945598, *1995*

[11] Srinivas Mukkamala, Guadalupe I. Janoski, Andrew H.Sung, Intrusion Detection Using Neural Networks and Support Vector Machines. Proceedings of *IEEE IJCNN, pp. 1702-1707, 2002*