Detecting Early Signs of Insider Attack Using Role-Based Analysis and Classification

University of Colorado at Colorado Springs

CS 526

Jeff Hinson, Fadi Mohsen, Joe Taylor

May 4, 2009

Abstract

The main purpose of this paper is to detect insider attacks before they happen, or at least to mitigate attack consequences and loss, by monitoring users access and actions. First, we collect users normal actions, then using the learning classifier and genetic algorithm application, we can solidify a set of rules that represent a normal working environment. After the rules are set in place, the system will continue to learn and analyze the new user's actions and compare them to the stored rules. Any user action that contradicts the rules will be considered abnormal, and thereby initiating notifications to be sent to predefined administrators for further investigation.

Introduction

Information security has been studied and analyzed for years. This can be seen in the many security innovations available today such as intrusion detection systems (IDS) and centrally managed security systems similar to that used by Microsoft's Active Directory. Most of these security innovations are geared toward preventing malicious individuals outside the organization from gaining access to private information and networks within the organization.

The research tends to overlook and underplay the threat from within. What happens when a seemingly good employee decides to turn against his organization? This can be a very difficult problem to solve and the potential for organizational harm is very high. A well maintained and proactive security approach can mitigate this risk to a large degree but what happens if this employee is an administrator over the network? He

would have legitimate need to have almost complete control over all security within the network in order to complete his job duties.

Insider Attacks

Insider attacks involve legitimate network users that have decided to act against the organization that employs them or that had formerly employed them. Research has shown that insiders usually take noticeable action within the network prior to the attack [1]. Without active monitoring for this activity, it is likely to go unnoticed. The battle against insider attacks is waged with the following goals: preventing attacks, detecting attacks and mitigating the damage. To effectively prevent insider attacks, an organization must have a clear understanding and prioritization of its critical assets. They need to know what critical information and resources might be a target of an insider attack. If they are aware of what needs to be protected, they can then use the techniques, tools, and procedures more efficiently. Fighting the insider attack battle on multiple front lines is not a wise choice regardless of how secure the organization thinks it is. The second goal is detecting insider attacks and the activities that generally occur prior to the attack. This can be done by implementing the tools and techniques that can discover the insider activity before a big loss occurs. Honeypots and honeytokens are techniques used to detect the attacks. They set a trap for the insider who is looking for sensitive information by baiting them with information that seems sensitive. Although both honeypots and honeytokens are traps, the former works on the system level whereas the latter works on the file level. The third and last goal is to mitigate the insider attack. Once the attack occurs, the organization needs to take quick action to reduce damages. Mitigation of an insider attack depends on having a solid emergency plan in place and ready.

According to a study done at Carnegie Mellon University [2], three categories were found to summarize styles of insider attacks: theft of information, fraud, and IT sabotage. Theft of confidential or proprietary information covers the cases in which current or former employee or contractor intentionally misused an authorized level of access to networks, systems or data with the intention of stealing confidential or proprietary information from the organization. Broadcasting this sensitive information would likely be a big concern for a government organization, especially if the information could reach enemy hands. It also harms the business process for an organization if the competitors receive such sensitive information. It is important to say here that most of these attacks were detected using non-technical means such as notification by a customer or informant. The second type of insider attack is fraud in which the attacker intentionally misused an authorized level of access to data with the intention of committing fraud using this data to obtain property or services. Here the identification of the attacks were based on system logs including database logs, system file change logs and file access logs. Lastly, IT sabotage is when an attacker intentionally misuses authorized levels of access to networks and systems with the intention of harming the organization. The attacker here used his legitimate account, and in some cases used other accounts.

What prompts insiders to attack their own organization? According to a study that was conducted by the US Secrete Service and the Carnegie Mellon University Software Engineering Institute CERT program in which 150 insider cases from across US critical infrastructure sectors were analyzed [1], they found that most insiders had a personal predisposition to this type of activity which can be recognized by monitoring certain characteristics like mental problems, previous crimes, and interpersonal skills. Many of the insiders were found to commit an attack after becoming angry with their employer [3]. One of the key findings of the study was that insiders exhibited such behavior ahead

of their attack [3]. These findings require us to change our plan instead of just using techniques and tools we must also observe employee behavior. This new angle on insider attacks motivated the researchers to focus their research on building a model for capturing employee behavior [4]. Their model focuses on observable influences that affect employees.

Because experience has shown that insider attacks are becoming an increasingly big concern, it is recommended to use every possible means to prevent, detect, and mitigate them. Windows Server 2003 and Windows Server 2008 support the ability to prevent and detect insider attacks to some degree. However, knowledge of correct security procedures are not always known within an organization. Educating the department responsible for monitoring employees is a key to successfully decreasing the attacks. They must be aware of the tools' capabilities and limitations. As mentioned above, Windows Server 2008 has a number of capabilities such as auditing, but integration with other tools is also likely to be beneficial.

In this paper, our concern is with designing a tool that can help the security department detect attempts to sabotage the IT resources of an organization. The tool will depend on the input received from Windows Server 2008 auditing and logging.

Role-Based Analysis

Role-based access control has been heavily used since the 1990s in enterprise level systems. Roles provide administrators of large organizations an efficient and organized way of assigning permissions to individual users. These roles can be assigned to an individual user and that user inherits the security permissions assigned to that role [5].

Similar benefits can be gained by using roles for activity analysis. Users within a role have the same permission within the network and also usually have similar usage patterns. By analyzing these activities in groups by role, rules can be created for these roles that accurately classify this activity as normal or abnormal.

Even though role-based analysis appears to be the best option for analyzing a large group of users for unusual activity, research also finds that organizations often fail to maintain their role-based access control systems. Exception management, documentation maintenance and policy changes are sited as frequent issues within this type of access control [6]. Improper maintenance of the roles will have an impact on the effectiveness of the rules generated for those roles.

Normal vs. Abnormal Activity

Normal activities are those activities that a person would be expected to perform on a regular basis according to the function that they perform for the organization. Abnormal activities are activities that are unusual or unnecessary for a person to be involved with based on their function with the organization. Installing network monitoring software might be considered normal for an administrator while it would be highly unusual for a member of the marketing team to do the same. Deleting account history for a customer might be a normal clean up activity for someone in the finance group but it would be high unusual for a system administrator to do the same thing.

Insider attack perpetrators regularly display noticeable abnormal activity prior to an attack. These abnormal activities were displayed by 87% of the MERIT research insiders and included such activities as setting up and using extra accounts, installing

and using hacking software, failing to document systems, failing to backup systems and accessing inappropriate web sites from work [1].

Research by the University of Texas at Austin has found that the number of systems calls made by a user can help predict whether the user's activity is normal or abnormal [7]. System call monitoring has been proven effective in detecting external attack scenarios but research has also shown that the characteristics of system calls vary greatly between internal versus external threats [7]. This appears to be a promising research direction for identifying normal versus abnormal user behavior.

Information Gathering

Information gathering is a central issue in any system seeking to prevent insider attacks. Important considerations have to be made as to what information should be collected and how it is to be collected. Data can be collected by either the server or by a local client. A server-side solution might seem appealing, but poses multiple risks. For example, if information is collected solely by the server, a user could create a local account on a computer that would go undetected by the server. A client-side solution, however, would require a lot of programming, as well as strictly-defined rules for detecting when a client is manually turned off. For the purposes of this paper, we will consider Windows Server 2008 for a server-side solution of information gathering, and a custom client as a client-side solution.

Windows Server 2008 Event Logging

Windows Server 2008 has several new features in the auditing configuration which allow system administrators to target more distinct types of activities [8]. These options

allow administrators to easily turn on or off auditing in many specific ways, which significantly reduces the effort needed for collecting data. It also contains options for exporting the gathered information into an XML document, which can be easily parsed or decoded.

Although Windows Server 2008 may seem to be a simple, all-in-one solution for information gathering, it poses some potential security risks. For example, it is incapable of monitoring local accounts. This introduces the risk of an employee generating a local account that would bypass the server's auditing system. Furthermore, given how easily an auditing feature could be turned on or off, a system administrator could easily turn off auditing for his own account and plant malicious code before anyone notices.

Custom Client

Another option for consideration is the creation of custom, local, network-specific monitoring client. Such a client could be designed to collect a specific set of information. As opposed to using Windows Server 2008, where all of the data is collected by the server and the data is limited to the auditing features available, a custom client could be installed on the individual computers to manually and locally collect data. A custom client would have more freedom in what data is collected (e.g. local accounts), but would require more programming effort. Some examples of the data that could be collected by such a client are listed below.

Registry Analysis

The user's registry - in part or in whole - could be scanned or copied periodically. This would allow the server to see how the user's computer is changing over time. This

could be especially important in determining how a system is affected during normal operations. Because the registry is so large and complex, it is highly unlikely that someone could make a serious system change without it being detectable by the registry. On the same note, however, an exhaustive analysis of the entire registry would be complicated and difficult.

Process Monitoring

Monitoring a process, or a class of processes, provides a real-time map of what the user is actively doing. Information about what resources are being collected, what files are being accessed, etc. could provide valuable insight for preventing an insider attack. Process monitoring allows the network to carefully monitor every step a user makes, as well as provides early detection for actions that could be malicious.

Peer to Peer Data Sharing

Peer to peer data sharing serves two purposes within the custom client: improving data delivery and being secure against a direct attack. By allowing clients to share data with each other, the likelihood of data loss at a single client is reduced. A globally unique identifier is assigned to each set of client data to assist in getting a single copy of the information to the server. Should the server be down for some reason, the clients will continue to gather data and share it amongst themselves until the server can be contacted for final delivery.

The clients will be able to communicate with each other as well as the server. This allows for a mesh of monitoring to prevent direct attacks on the clients and the server. If the client is disabled on a machine, the other clients and the server will notice and provide an alert. If the server is disabled, the clients will notice and be able to provide

an alert. The communication scheme will need to be planned out to prevent all nodes from monitoring all other nodes since this would cause excess network traffic.

Classifier Systems

Learning Classifier Systems (LCS) have been in use since the late 1970's to build rules based on environmental input. In general terms, a classifier system learns by analyzing input from the current environment, responds to this input with an action and receives a reward based on the action taken. This analyze-act-reward process encourages good rules by assigning points and discourages bad rules by withholding points. As this process continues to analyze input, the good rules are separated from the bad by points, and new, potentially good, rules are created based on the best current rules. These new rules replace old unsuccessful rules and the process continues analyzing input and improving its rules.

The original LCS systems suffered from a common problem called overgenerals. These overgenerals are rules that cover a large area of the problem space and often cover up small areas where their overly general rule is not appropriate. For example, an LCS might be attempting to learn which street corners require a vehicle to stop. If a particular neighborhood had many corners but only one stop sign, an overgeneral rule might be created that says that vehicles do not have to stop at any corner in that neighborhood.

Accuracy-Based Classifier Systems

In 1995, Wilson solved the overgeneral problem of LCS with the Accuracy-Based Classifier System (XCS). [9] An XCS resists overgenerals by altering the way it assigns

rewards. For each input received from the environment, a set of matching possible rules is created from the entire population of rules in the system. From this matching set of rules, only those with the correct associated action are selected and pulled into an action set. Rules in the action set receive rewards while all other rules do not.

Genetic Algorithm

Classifier systems typically have a genetic algorithm component that assists them in the creation of new potential rules. There are three primary genetic functions that act on the rules: reproduction, crossover and mutation. [9] Reproduction is usually implemented by selecting two parents from the existing rule set based proportionately on their fitness. These two parents may be used in crossover, mutation or both. Crossover takes both parents and selects certain attributes from each parent for the new child rule and the remaining attributes are assigned to the second child. This allows crossover to produce two new children that a similar to their parents but decidedly different as well. Mutation is typically implemented by introducing a fairly small chance to randomly alter attributes in the new child rules. By combining reproduction, crossover and mutation, new rules are introduced into the rule set to compete for rewards. Just as new rules are added, so low performing rules are removed.

XCS systems use a specific approach when it comes to these genetic algorithm functions. First, reproduction only occurs among the action set rules meaning that only rules that have matched an actual environmental event and provided the correct action are allowed to be considered for reproduction. This insures high quality parents for reproduction. Second, rules are also removed from the action set based not only on their fitness or rewards but also on how specific they are in comparison to other rules in

that action set. This allows for accurate removal of overly specific rules when a more general rule applies the correct action. Third, the rewards calculation is somewhat different from the standard implementation. Rather than simply assign points to all action set rules, the XCS proportions the points to action set member rules based on a probability of accuracy which considers how specific and how fit a rule is deemed to be. [9]

Lessons Learned

Administrative Access Complicates Monitoring

Since it is frequent for administrators to also be the inside attackers, their inherit privilege level within the network creates difficulties for monitoring activity and detecting insider attacks. Administrators, depending on the organization, usually have complete control over the network. This security level allows administrators to stop programs, disable logging and generally circumvent most security precautions.

Larger organizations that need multiple administrators can work around this administrator permission issue. Administrators can be granted high privilege access to select areas and other administrators can be assigned to perform security audits over the other areas. Newer technologies such as smart-cards and RFID badges can also be used to restrict and track administrators within the network. [6]

Windows Server 2008 Auditing

To learn about the auditing abilities of Server 2008, we created a virtual lab consisting of a Server 2008 domain controller virtual machine and three domain client virtual

machines running Windows XP. We were able to configure the auditing, utilize the client VMs to generate events and observe the recorded results in the server event log.

We exported the test events into XML to allow us to examine how a processing application would be able to access this information. The table below shows examples of useful audit event types that can be captured using the Windows Server 2008 auditing features.

Audit Type	**Description**
Audit account logon events	Tracks user account logon and logoff information. Domain accounts are logged to the domain server and local accounts are logged to the local computer's security log.
Audit account management	Tracks the creation of new user accounts and changes to user accounts including password changes.
Audit object access	Tracks user access to objects including files, folders, registry keys and printers.
Audit policy change	Tracks changes to policies with the domain.
Audit process tracking	Tracks detailed audit information about applications starting and stopping.
Audit system events	Tracks events that affect the system security or security log and includes events such as restarting a computer or shutting down a computer.

Microsoft .NET Socket Communications

Code samples demonstrating socket communications with client and server applications were explored. Sharing information across sockets was proven out by connecting multiple application processes via sockets and exchanging XML information.

Future Direction

To move this project forward further, the rule layout will need to be defined and additional details will need to be decided for the client and server processes. Defining the rule layout will involve analyzing the events and information to be tracked and translating this into a chromosome layout usable by the classifier system and genetic algorithm. The client process definition will need to determine if and how peer-to-peer data sharing will work between the other clients and the server. The server process definition will need define how events will be managed as they are reported from multiple clients.

References

[1] Moore, Andrew P.; Cappelli, Dawn M.; & Trzeciak, Randall F. “The ‘Big Picture’ of Insider IT Sabotage Across U.S. Critical Infrastructures” Software Engineering Institute, Carnegie Mellon University, May 2008. http://www.cert.org/archive/pdf/08tr009.pdf

[2] D.M. Cappelli, A.P. Moore, and T.J. Shimeall, Common Sense Guide to Prevention/ Detection of Insider Threats, tech. report, Carnegie Mellon Univ., CyLab and the Internet Security Alliance, July 2006; www.cert.org/archive/pdf/ CommonSenseInsiderThreatsV2.1-1-070118.pdf.

[3] Cappelli, D. M.; Desai, A. G.; Moore, A. P.; Shimeall, T. J.; Weaver, E. A.; & Willke, B. J. “Management and Education of the Risk of Insider Threat (MERIT): Mitigating the Risk of Sabotage to Employers’ Information, Systems, or Networks.” Proceedings of the 24th International System dynamics Conference. Nijmegen, Netherlands, 2006. http://www.albany.edu/cpr/sds/conf2006/proceed/proceed.pdf

[4] Puleo, A.J., "Mitigating Insider Threat Using Human Behavior Influence Models", Master's thesis, Air Force Inst Of Tech Wright-Patterson AFB OH School Of Engineering And Management ,June 2006.

[5] J.S. Park, J. Giordano, "Role-based profile analysis for scalable and accurate insider-anomaly detection," pcc, pp.62, 2006 IEEE International Performance Computing and Communications Conference, 2006

[6] Bauer, L., Cranor, L. F., Reeder, R. W., Reiter, M. K., and Vaniea, K. 2009. Real life challenges in access-control management. In Proceedings of the 27th international Conference on Human Factors in Computing Systems (Boston, MA, USA, April 04 - 09, 2009). CHI '09. ACM, New York, NY, 899-908.

[7] Liu, A.; Martin, C.; Hetherington, T.; Matzner, S., "A comparison of system call feature representations for insider threat detection," Information Assurance Workshop,

2005. IAW '05. Proceedings from the Sixth Annual IEEE SMC , vol., no., pp. 340-347, 15-17 June 2005

[8] Microsoft TechNet, "What's New in Windows Security Auditing", http://technet.microsoft.com/en-us/library/dd560628.aspx, April 17, 2009.

[9] Butz, M. V., Kovacs, T., Lanzi, P. L., and Wilson, S. W. (2001). "How XCS Evolves Accurate Classifiers". In Spector, L., Goodman, E., Wu, A., Langdon, W., Voigt, H., Gen, M., Sen, S., Dorigo, M., Pezeshk, S., Garzon, M., and Burke, E., editors, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO'2001), pages 927-‒934. San Francisco, CA: Morgan Kaufmann.