Privacy vs Intrusion Detection Analysis

Emilie Lundin
emilie@ce.chalmers.se

Erland Jonsson
erland.jonsson@ce.chalmers.se

Department of Computer Engineering
Chalmers University of Technology
412 96 Göteborg, Sweden
Tel: +46 31 772 10 00
Fax: +46 31 772 36 63

Abstract

When auditing events in a computer system, a great deal of potentially sensitive information is recorded. Personal integrity is threatened, especially when the log files are used for intrusion detection. A solution for this may be to pseudonymize the sensitive information in the log files, i.e. exchange user names etc. for pseudonyms. This paper shows how this can be done. We also carried out a number of experiments using anomaly data collected from a proxy firewall. It was found that there were problems both with unwanted reidentification of the users and with ineffectiveness in the intrusion detection as such. It is discussed how and to what extent these problems can be remedied or circumvented.

1 Introduction

When auditing events in a computer system, much personal and sensitive information is recorded. Many users are unaware of what is in the log files and how this information can be used. Especially when the log files are used for intrusion detection (ID), personal integrity is threatened.

There are two main methods for intrusion detection, anomaly and misuse detection. An early anomaly detection model is described by [Den87]. Anomaly detection means establishing a "normal" behavior pattern for the users of the system and then looking for deviations from this behavior. Research efforts on anomaly detection have resulted in systems such as NIDES [JV94] and W&S [VL89].

Misuse detection means looking for known malicious or unwanted behavior. Examples of misuse detection systems are Haystack [Sma88] and Bro [Pax98].

To be able to use log files for intrusion detection purposes, there must be a way to connect events to a specific user. If we want to protect the privacy

of users, something must be done about the cleartext log files. A solution may be to pseudonymize the sensitive information in the log files, i.e. exchange user names etc. for pseudonyms. This paper shows how this can be done.

Further, we used an anomaly detection tool to perform experiments on pseudonymized data collected from a proxy firewall. We found that the pseudonymization caused problems both with unwanted reidentification of the users and ineffectiveness in the intrusion detection as such. It is discussed how and to what extent this problem can be remedied or circumvented.

Sobirey et al. [SFHR97] identified the need for privacy enhanced intrusion detection and presented a solution including real-time pseudonymization and intrusion detection. They discussed definitions of privacy and different directives and criteria for security evaluation on the subject. They do not seem to have had problems with the intrusion detection in combination with pseudonymized log files.

In the following, section 2 discusses sensitive information and how to protect privacy with pseudonymizers. In section 3, the implementation of our pseudonymizer is described, and section 4 describes the intrusion detection tool and the experiments. Possible remedies for the pseudonymization problems are discussed in section 5. Section 6 presents a final discussion and section 7 some concluding remarks.

2 Sensitive information and pseudonymizers

This section discusses what information can be considered sensitive. It is described how to use a pseudonymizer to enhance privacy, and the problems it causes are discussed.

2.1 Sensitive information

Any information that can be connected to a specific user can be sensitive. The sensitivity depends on what the information will be used for. Information that reveals the overall resource usage or user behavior in the system can also be considered sensitive. While the most obvious thing to do to reduce sensitivity is to remove the connection between a user and a specific event. This can be done by anonymizing or pseudonymizing certain data items. Evidently, the user name is one such item, but host names or use of certain files with user specific names or paths can also be disclosing.

Intrusion detection tools report not only what is certain to be an actual intrusion but also attempted intrusions and suspicious events. Especially anomaly detection tools are very inexact in what they report; misuse detection tools also often report things that are not certain to indicate intrusive behavior. Many alarms are false, and innocent people can be accused of something they have not done or at least not done on purpose. Furthermore, many intrusion detection tools can visualize the user's behavior and reveal things he did not wish to be public. There is thus a need for protecting the privacy of users; the use of intrusion detection tools and pseudonymization of log files accomplish this. However, there is a conflict between pseudonymizing enough information and not causing problems for the intrusion detection activity.

2.2 Pseudonymization and intrusion detection

To be able to use log files for intrusion detection purposes, there must be a way to connect events to a specific user. Otherwise, there would be no possibility of knowing who caused the suspected intrusion, stopping the intruder or examining the damage he has done. It would also be difficult to perform any advanced intrusion detection because this often requires sequences of events for the same user in order for something to be classified as an intrusion.

This means that the log files can not be anonymized if they are going to be used for intrusion detection. Anonymization in this case means that every user name, host name etc. is replaced with an arbitrary string and there is no way to recover the real names. Instead, pseudonymization can be used. Pseudonymization means that the information you wish to protect is exchanged for pseudonyms. A name should be exchanged for the same pseudonym every time it is encountered.

In [SFHR97], it is suggested that a pseudonym database or some kind of encryption can be used to pseudonymize data and to be able to recover true identities. There is a great deal of information in normal log files that needs to be protected. It is stated that user IDs, location IDs and conditional subdirectories and objects must be pseudonymized to attain effective protection and still be able to perform intrusion detection.

If log files are pseudonymized in an adequate way, the risk of personal information leaking out by mistake is reduced. The system administrator will not inadvertently discover that you are taking particularly long lunches or are working late every night on your secret project.

There must of course be a way to reidentify a user when suspicious behavior is encountered. The procedure of reidentification should be well specified in a security policy. This is preferably done by the site security officer (SSO) and only when there is a strong suspicion against a user. It may be done in cooperation with the system administrator, who probably has more thorough knowledge of the system and can explain many of the suspicious events without having to investigate them more closely.

It can be very important that the reidentification is done quickly to prevent more damage than necessary when an intrusion is detected. One such occasion is if the intrusion detection system (IDS) is of a type that has mechanisms for automatic response, e.g. interruption of suspicious network sessions. Then it is probably necessary to allow the IDS to reidentify users on its own initiative in real-time. Otherwise, there may be a problem if the person analyzing the log files is not authorized to reidentify suspected users in an emergency situation. There must be a well thought out balance between efficiency and personal integrity.

There are many ways to find out which pseudonyms correspond to which users. For example, a person who knows the system and its users as well as the time and date of an event can connect a user to the specific event. Thus, unwanted reidentification is a serious problem with pseudonymous log files. As stated in [SFHR97], the following data can be used:

- action in combination with date/time and the final action status;
- action under consideration of the access rights, in combination with date/time and the final action status;

- host identifier or name and host type.

There are probably more ways to do this if there is thorough knowledge of the system and its users.

Despite the problems, it probably will be necessary in many cases to work with pseudonymized log files. More and more people are beginning to understand the importance of protecting privacy. Companies also wish to protect information about the behavior in their computer system and the usage of their resources to protect themselves. If they can be convinced that it is safe to hand out anonymous log files, there will be a great deal more material available for computer security research.

3 A pseudonymizer tool

In a recent industrial project, we developed a pseudonymizer tool. This was done to obtain log files to use for intrusion detection experiments where sensitive information was removed. The company intended to distribute the tool to their customers to be able to obtain pseudonymized log files to use for debugging purposes and for further intrusion detection experiments.

3.1 Pseudonymizer input

The log files we worked with came from a special proxy firewall, that works as a "personal" firewall either between a company network and the Internet or between security partitions inside a network. "Personal" in this case means that the users are authenticated individually and that the services available are configured per user.

The proxy firewall registers information about the connections made through it. If it is placed between the company network and the Internet, it typically registers an employee trying to reach resources in the company network from home. If it is used to create secure partitions, it registers users who want to access secure servers protected by the firewall. The firewall authenticates the user and acts as a proxy, delivering the services the user asks for.

The log files register who attempted to connect and when and where the connection was attempted. They report whether the authentication succeeded or failed, what service the user asked for and whether the request was granted. After a service is started, nothing is reported for that session until it is closed and the user logs out. It also reports information about its internal state and changes to the service database, which determines what access rights a user has.

The firewall should be connected so that all traffic into the protected network is routed to the firewall. Outgoing traffic can use another way.

The contents of a log entry vary depending on what kind of event is reported. A "login" entry is depicted in figure 1.

time	event	status	server	user name	remote host	authentication method

Figure 1: "Login" log entry

- time: date and time of day

- event: login or logout
- status: failure or success
- server: name of server handling the request
- user name: user name
- remote host: name of host from which the user is connecting
- authentication method: ordinary password, one-time password, smart card etc.

In this case, the server, user name and the remote host fields must be pseudonymized to achieve basic privacy protection. Some entries also have a free text field, where user names, disclosing file names and host names, can appear. The text can be useful so it is convenient to look for known names or substrings having the form of e.g. a host name and just substitute these.

There were no demands for absolute anonymity of the users and the systems from which the logs came, so the above mentioned pseudonymizations were sufficient. It was also considered the maximum that could be done without disturbing the intrusion detection experiments.

3.2 Pseudonymizer function

The pseudonymizer program reads the binary log files, exchanges the sensitive information and generates "clean" log files on the same format as the input files. It also generates a database with the connections between pseudonyms and real names.

The pseudonymizer program looks for user names, host names and IP addresses in the server, user name, remote host and free text fields. If the name is found in the mapping database, the corresponding pseudonym is used. The user names are mapped to the word "user", followed by a sequential number, i.e. user0, user1, user2 etc. The host names consist of a computer name and a domain. The computer name is mapped to host0, host1, host2 etc., and each part of the domain is mapped separately to dom0, dom1, dom2, etc. For example, the host name *computer1.chalmers.se* may be mapped to *host3.dom4.dom1*. The IP addresses are mapped to an address in the same class, meaning it can be seen whether it is a class A, class B or class C address.

The program treats one log entry at a time and can be used for pseudonymizing data in real-time without writing the log files to disk.

3.3 Pseudonymizer weaknesses

The pseudonymizer program seems to work well in general but still has some weaknesses. The simple pseudonyms for users and hosts make it very easy to remember them after a reidentification.

The program may miss a disclosing file name. It pseudonymizes only known user and host names that appear as parts of a file name, and there may be other things in the file name that are of a disclosing nature as well.

It may also miss a user or host name in the free text fields. If the name has not appeared in the specified user or host fields before it appears in the free

text field, the program has no way of knowing that it is a name that should be pseudonymized. This never happened when we were testing the program, so it should be a very rare incident.

The program has no mechanism for reidentifying pseudonyms automatically but this is a function that can easily be implemented. This may be needed if it is used together with a real-time intrusion detection system.

4 Experiment with Anomaly Detection

We implemented an intrusion detection tool using statistical methods to detect anomalies. The objective of the experiments was to evaluate our tool and find out whether statistical anomaly detection was a practicable method to use for analyzing log files generated by the system described in section 3.1. The data flow in the experimental system is shown in figure 2.

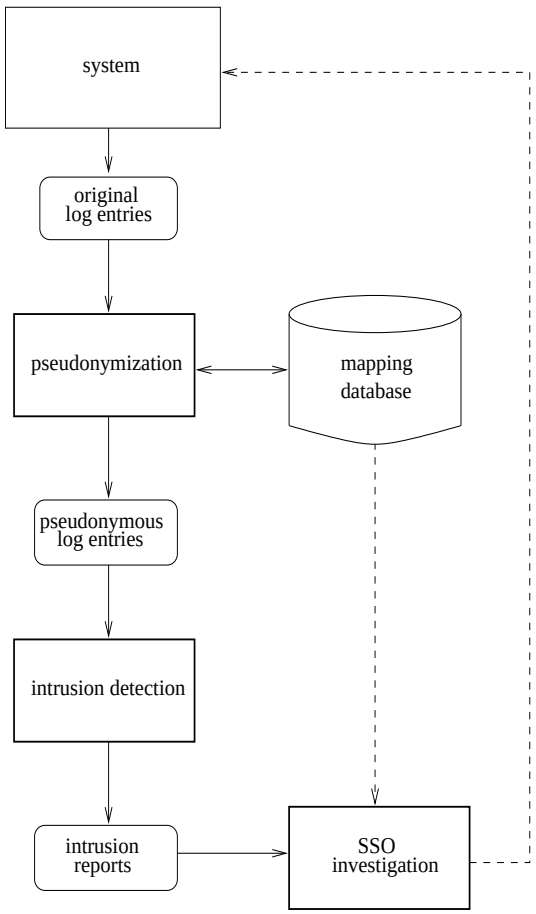

Figure 2: Dataflow in experiment system

4.1 Detection Tool

The primary intention was that the program should be able to detect masqueraders and, to some extent, insiders. A masquerader can be defined as someone using an account on the system that he is not authorized to use. It may be an external or an internal intruder. An insider is a legitimate user who misuses the system. If the insider misuses the system using someone elses account, he also falls under the category of masquerader. To make it easier to distinguish between the categories, the term insider is used in this paper only to describe someone who misuses the system with his own account.

A model for normal behavior in the system is necessary to be able to detect anomalies. We call this normal behavior model "profile". A profile should reflect a user's, program's or system's normal behavior during a period of time.

The profiles used in our tool consist of a number of parameters for which the mean value and standard deviation are determined. The values for the parameters are updated automatically once daily. In this early version of the tool, only the information about login events is used.

The parameters used in the profiles are:

- number of successful logins
- number of failed logins
- number of successful logins per IP address
- number of failed logins per IP address
- time since last login
- session time

The first four parameters are interval based, meaning that the day is divided into a number of intervals and the statistics for each interval are calculated. An interval may be one hour, which means that you can find out e.g. how many logins the user carries out between 8:00 and 9:00 every day.

The tool creates one profile for each user and one for the system. When the program is run, one log entry at a time is read and analyzed and the recent behavior of the user is updated. The recent behavior is compared with the long-term profiles for both the individual user and the system. If the recent behavior shows too great a deviation from the profile values, an anomaly is reported. In this tool, the long-term profiles should reflect the users' behavior during the previous month. Recent behavior is defined as behavior during the present day or hour.

Several different output files are generated by the tool for the reason that the administrator should be able to find out what has happened when he receives an alarm. The anomalies or "alarms" reported are in the form of entries in a text file. This file contains information about each reported event with user name, date and time, from which parameter the behavior deviated and by how much. To help with the investigation, different types of graphs are produced. They show e.g. profiles of how many logins and failed logins the user makes during different intervals in the day and from which hosts these logins have been made. There are graphs over how many reported anomalies the user has caused for each of the profile parameters every day. Another type of graph presents the number of logins and failed logins the user makes every day.

4.2 Experiments

The experiments were carried out on authentic log files gathered over a ten-month period. Before the experiments, the log files were pseudonymized with the tool described in section 3. The log files came from two different installations of the proxy firewall and were not known to contain any intrusions.

The pseudonymized log files were analyzed in chronological order, and the prototype was configured to generate all possible kinds of output files. The detection output was then evaluated manually.

Some interesting problems appeared during the evaluation. While it was not believed that the log files contained any intrusions, some users showed very strange behavior. For example, one user suddenly started to make 150 failed logins per day. We found no evident explanation for this behavior until we reidentified the user and found out that it was not a regular user at all but an erroneous process. Other anomaly alarms also turned out to be set of by "special" users rather than regular ones. Our analysis would have been much simpler had we known this in advance.

4.3 Experienced problems

Pseudonymization solves some privacy problems, but we found during the experiments that it leads to other problems instead.

4.3.1 Reidentification problems

Our experiments confirmed the reidentification problems described in section 2.2, and we found that there are probably even more ways to reidentify a user if there exists thorough knowledge of the system and its users. With knowledge of the users in the system, it is fairly easy to reidentify a user only by looking at his normal behavioral profile. There is a conflict between pseudonymizing enough information and not causing problems for the intrusion detection activity. Also, after reidentification of a user because of the suspicion of intrusion, it is difficult not to remember the connection between user and pseudonym.

4.3.2 System privacy

Another problem is that information that reveals the overall resource usage or user behavior in the system can be considered sensitive under certain circumstances. Thus, the pseudonymization does not protect the privacy of the system as a whole. This became evident when we studied various graphs of data. For example, we could easily infer the working hours of the employees and the workload on the system.

4.3.3 Pseudonym efficiency problem

It takes a considerable amount of time to investigate each user. Many graphs must be studied and the user must often be reidentified and questioned about his behavior. A problem is that the pseudonyms make the intrusion detection process even less effective. If the person who investigates the reported events is familiar with the system and has the user names in cleartext, he could probably

sort out a large number of false alarms without having to investigate them in detail. This would save a great deal of time.

5 Possible Remedies

5.1 Reidentification and system privacy

A possible solution to the problem of unwanted reidentification is to add "noise" to the log files. You can add fake users showing behavior that "fills out" characteristics in the user patterns of the whole system. This can be done to protect information leakage about the usage of the system. You can also add fake entries for existing users to make it more difficult to reidentify them.

The person receiving the intrusion alarms would not know whether it was a real or a fake alarm if he had no access to the information about which were the fake entries and users. A person with access to that information would in theory be able to quickly identify the false alarms caused by the fake entries and users.

The fake entries may pose a problem for the IDS. Anomaly detection would probably work badly if the system did not automatically ignore or treat the fake entries in a special way.

One remedy for the problem of a system administrator remembering the connection between the real name and pseudonym after a reidentification is to create sufficiently complicated pseudonyms. This prevents the administrator from unintentionally remembering the connections. Another solution, also proposed in [SFHR97], is to change the connections at regular intervals. The problem with this is that the intrusion detection tool will lose the possibility to check users' past behavior.

5.2 Pseudonym efficiency

A partial solution to the efficiency problem may be to identify and label the different kinds of users. If there are special user accounts that are not used by physical persons, these will probably be used differently than other accounts. Different kinds of groups of users may also be labelled, so that some of the false reports can be identified on the basis of knowledge about the groups' normal behavior. This would probably have helped in our case.

6 Discussion

When an intrusion is reported, it is first necessary to find out what has happened and in many cases also investigate why the user did what he did. Was he looking for weaknesses in the system to use for his own malicious purposes? Was he doing it by mistake or did someone else use his account? To find out why something was reported as a suspected intrusion, you must know the users and the system to be able to do the job within a reasonably short time.

If different groups of users are labelled, the job may become easier, but this will be at the expense of privacy. Knowledge of the group to which users belong makes it easier to reidentify them. The more specific the groups are, the easier it will be to do intrusion detection, but the less privacy will be achieved and

vice versa. There is a contradiction between protecting the privacy of users in the log files and using them for efficient intrusion detection.

We have reason to believe that there will be fewer problems with a more exact intrusion detection tool. If you can trust the system enough to let it investigate all alarms itself and respond actively to them, the system can make the reidentifications and it will not be necessary to interfere. The pseudonymous log files would then not be a problem and perhaps would not be needed at all. However, it is not at all clear that it is possible to construct such a system and in any case, it will not happen in the near future.

7 Conclusion

We have found that there is trade-off to be made between user privacy and effectiveness of the intrusion detection analysis. There are no evident ways to get around this problem completely but it is indeed possible to make some improvements, e.g. using a pseudonymizer tool. However, even if such a tool is utilized, information can be inferred from known data, which is an example of a classical database problem. Also, the use of a pseudonymizer is at the expense of intrusion detection efficiency.

References

[Den87] Dorothy E. Denning. An intrusion-detection model. In *IEEE Transactions on Software Engineering*, volume SE-13(2), pages 222–223, Feb 1987.

[JV94] Harold S. Javitz and Alfonso Valdes. The nides statistical component: Description and justification. Technical report, SRI Computer Science Laboratory, March 1994. Available for download at: http://www.sdl.sri.com/nides/index5.html.

[Pax98] Vern Paxon. Bro: A system for detecting network intruders in real-time. In *Proceedings of the Seventh USENIX Security Symposium*, pages 31–51, San Antonio, Texas, January 1998. USENIX.

[SFHR97] Michael Sobirey, Simone Fischer-Hübner, and Kai Rannenberg. Pseudonymous audit for privacy enhanced intrusion detection. In *Proceedings of the IFIP TC11 13th International Conference on Information Security (SEC '97)*, pages 151–163, Copenhagen, Denmark, May 1997. IFIP, Chapman & Hall.

[Sma88] Steven E. Smaha. Haystack: An intrusion detection system. In *Fourth Aerospace Computer Security Applications Conference (IEEE Cat.No.CH2619-5)*, volume xii+440, pages 37–44, Washington, DC, USA, 1988. IEEE Comput. Soc. Press.

[VL89] H. S. Vaccaro and G. E. Liepins. Detection of anomalous computer session activity. In *1989 IEEE Symposium on Security and Privacy*, pages 280–289, Oakland, CA, USA, May 1989. IEEE Piscataway, NJ, USA.

Emilie Lundin is a PhD student at the Department of Computer Engineering at Chalmers University of Technology, Goteborg, Sweden. She is part of the Computer Security group and her main research interest is intrusion detection. Emilie Lundin received a MS in Computer Science and Engineering from Chalmers.

Erland Jonsson is an associate professor of Computer Security and head of the Department of Computer Engineering at Chalmers University of Technology, Goteborg, Sweden. Prior to taking up his present post, he worked in industry for almost 20 years with hardware and software design and quality assurance for telecommunications and space applications. His research interests include intrusion detection, security mechanisms and quantitative security modelling. Jonsson recieved a MS in Electrical Engineering and a PhD in Computer Engineering from Chalmers. He is a board member of the Special Interest Group for Security of the Swedish Information Processing Society. He is a member of the IEEE Computer Society and ACM.