Data Collection and Analysis for Masquerade Attack Detection: Challenges and Lessons Learned

Malek Ben Salem and Salvatore J. Stolfo

Intrusion Detection Systems Laboratory
Computer Science Department
Columbia University
1214 Amsterdam Ave.
New York, New York 10027, USA
{malek,sal}@cs.columbia.edu

Abstract. Real-world large-scale data collection poses an important challenge in the security field. Insider and masquerader attack data collection poses even a greater challenge. Very few organizations acknowledge such breaches because of liability concerns and potential implications on their market value. This caused the scarcity of real-world data sets that could be used to study insider and masquerader attacks. In this paper, we present the design, technical, and procedural challenges encountered during our own masquerade data gathering project. We also share some lessons learned from this several-year project related to the Institutional Review Board process and to user study design.

1 Introduction

Lack of large-scale, real-world data has hindered the development of effective intrusion detection systems for the detection of insider attacks. Most organizations that undergo such types of attacks prefer not to announce them publicly out of liability and confidentiality concerns. According to the 2010 cyber crime watch survey, which was conducted by the Computer Emergency Response Team (CERT), and which surveyed 523 security executives and law enforcement officials, 72% of the insider incidents that occurred at the surveyed isntituions were handled internally without legal action or the involvement of law enforcement [3]. Another 13% of the insider incidents are handled internally with some legal action. Announcing such attacks may also have market share implications. For the same reasons, they are even less likely to share real-world data that could be used to study such attacks with the research community.

The study of masquerade attacks, a class of insider attacks in which a user of a system illegitimately poses as, or assumes the identity of another legitimate user, suffers similarly from the scarcity of real-world data, despite their significance. According to the 2010 cyber crime watch survey [3], 35% of the surveyed executives and law enforcement officials experienced unauthorized access and use of their information, systems, and networks. This type of intrusion, known as *a masquerade attack*, was second in the top five list of electronic crimes perpetrated by outsiders after virus, worms and other malicious code attacks.

In the absence of a real-world data set for the study of masquerade attacks, we had to launch our own data collection project. In this paper we present the methodology followed to gather our own data set for the evaluation of masquerade attack detection techniques. We discuss the challenges encountered during the collection and analysis of the data set. The rest of this paper is organized as follows. In Section 2, we describe the objectives of the masquerade attack data collection project. Section 3 covers the challenges encountered during the different phases of the data collection project, including design, procedural, and technical challenges. In Section 4, we present the lessons learned throughout the masquerade data gathering project. Finally, Section 5 concludes the paper by summarizing the main points of the paper.

2 Project Objectives

In the case of masquerade attack detection, most detection approaches used machine learning techniques to profile normal user behavior, and detect abnormal behavior that could be indicative of a masquerade attack. The vast majority of these techniques were evaluated using the the Schonlau dataset [6], gathered by Mathias Schonlau [2]. This dataset suffers from several shortcomings. The first shortcoming is the absence of any command arguments. Only simple commands have been collected. Another weakness is the lack of timestamps that indicate when the user commands were issued. No indication is available as to what time period is covered by the 15,000 commands collected from each user. It could take one user a few days to issue this number of commands, when it takes another a few months to do the same. This indicates another shortcoming of this dataset, namely the heterogeneity of the users. Not all users have the same expertise with Unix commands, nor do they have the same job functions. The wide variety of backgrounds amongst the users causes differences in their behaviors, such as the variety of commands that they use.

While all of the above shortcomings of the Schonlau dataset are important, perhaps the most significant weakness of this dataset is the **lack of** real **masquerader data**. All of the user command sequences gathered in this dataset were issued by normal users performing their regular day-to-day jobs. No command sequences were issued by attackers. Masquerade attacks were simulated by randomly inserting excerpts of command sequence from one user into the command sequences issued by another user. This practically turned the masquerade attack detection exercise into an author identification exercise. The fact that the command sequences belong to users with widely varying Unix proficiency and different job description further weakens the accuracy results achieved by the proposed "masquerade attack" detection classifiers, which have been only evaluated in an "author identification" exercise.

In order to overcome these weaknesses, we launched an initiative to collect our own dataset for masquerade attack detection and to make it available for the broader research community. We call this dataset the RUU (Are You You?) dataset [1]. The tasks consisted of gathering computer usage data belonging to a large homogeneous set of *normal* users, and simulating masquerader attacks. Each task posed its own challenges. However, both tasks were subject to the Institutional Review Board (IRB) review first, a process that we describe in the following section.

3 Challenges

3.1 Procedural Challenges

The major procedural challenge encountered during the concept phase of our data collection project was the IRB process. Obtaining IRB approval to conduct user studies is a costly process, bot in time and effort. During the IRB process, the research plan and objectives including the detailed description of the planned experiments are reviewed in advance in order to protect the privacy rights of the human subjects involved in the research. This process is required in all institutions that receive research funding from the US federal government. It is a lengthy process and may be iterative in some cases, as some clarifications may be requested by the IRB. For example, we had to submit the exact text of the *call for participation* to students in our user studies. We had to specify in advance what pieces of data we anticipated to collect, for how many users, and for how long. We did not necessarily know the answers to all of these questions when we initiated the IRB review process. In order to continue working on the project, we had to extend or re-initiate the review processes on several occasions.

3.2 Design Challenges

Once the user studies were approved, we could run our experiments. We simulated masquerade attacks by developing a very specific attack scenario, which user study participants had to follow. The masquerade attack scenario had to be: 1) representative of masquerade attacks, *i.e.* generalizable, and 2) easily executable in a user study. The latter condition meant that the execution of the masquerader scenario had to be time-limited. Not specifying a time limit for the attack adds a lot of uncontrolled variability to the experiments. Furthermore, it makes the experiments costly both in participants and researcher time.

We developed an attack scenario, where the attacker gets an opportunity to access a coworker's computer during a 15-minute lunch break, while the coworker leaves the office and stays logged in to their computer. In this scenario, we described the financial difficulties that the masquerader was going through, and the personal conflict that they had with the coworker. The attacker's objective was to find any information that could be used for financial gain. We strove to ensure that the task of the user study participants was goal-oriented, thus revealing the malicious intent of the attacker.

We conjectured that the malicious intent of a masquerader to steal information will be manifested in the attacker's search behavior when they access the victim's computer. Our goal was to confirm this conjecture and to show that the attacker's search behavior is different from a normal user's search behavior, and that monitoring search behavior could be used for the detection of a masquerader's attack. When we conducted the first user study, we only used the 'Malicious attacker"' scenario described above. However, we soon realized that we had to control the 'user's intent' in order to ensure the validity of our experiment and of the resulting collected data. To that extent, we conducted another a second experiment, where the intent to leak information was the independent variable that we controlled. Besides the 'malicious attacker' scenario described above, we developed two other scenarios: a 'benign masquerader' scenario, and a 'neutral' scenario. In the benign scenario, the participants experienced a hard drive failure and could access a coworker's computer while their coworker left out for 15 minutes, in order to finish working on an urgent project. In the neutral scenario, the participants in this scenario had no compelling reason to access the co-worker's computer. They were left to freely choose whether they wanted to access their coworker's desktop when the coworker left during a lunch break.

In order to increase the sensitivity of our experiment, we had to reduce uncontrolled variability. This in turn requires controlling user bias which makes up the largest source of error variance in user study experiments [4]. There are three techniques used in behavioral sciences to reduce subject variability, or user bias. The first and preferred technique is the use of the same subject in all 'treatment conditions' of the experiment, that is in all three scenarios. This procedure could not be used in our experiment as it undermined the assumption that masqueraders were not familiar with the file system under attack. Using the same subjects in different treatment conditions of the experiment means that they will be exposed to the file system more than once. This implies that, in the second and third treatment condition or scenario, the subjects have prior knowledge about the file system, which introduces a new type of error in our experiment.

The second approach and probably the most obvious approach is to select a homogeneous group of subjects, *i.e.* subjects with similar characteristics that are relevant to the experiment, such as their familiarity with the use of computers, their ability to search for information in a file system, and their acuity or sense of cyber-security. Finally, the third approach for reducing subject variability is the use of several small subject sets with characteristics that are highly homogeneous within one set, but widely varying between sets. We have chosen the second approach, and selected subjects who were all students at the Computer Science department of Columbia University, so that they have comparable skills. This should minimize the variability between subject with respect to their familiarity with computer usage, and how to search a desktop in order to steal information, or how to perform a data theft attack without being detected. This should reduce confounds and bias in the results of this user study.

To further reduce subject variability, we anticipated all questions that could be raised by the participants and included answers to them in the user study scenarios. The goal was to minimize any verbal communication between the researcher and the participant in the experiment. That ensured that all participants received the same instructions, thus minimizing the participant bias. One may argue that simulating a masquerade attack is not appropriate, and that it is hard for an innocent student to act as a masquerader. We argue that, if the scenario of the experiment is well-written, and with very clear instructions, the participants in the experiment will follow the instructions. To this extent, we refer the reader to the very well-known Milgram experiment, which showed how subjects obey an authority figure and blindly follow instructions, even when they contradict their own values and ethics [5].

Besides reducing subject variability, we strove to reduce the experimental treatment variability by presenting each user study participant with the same experiment conditions. In particular, we used the same desktop and file system in all experiments. We also ensured that the desktop accessed by the subjects looked the same to each participant. In particular, we cleaned up the list of recently accessed documents, and opened MS Office documents before the start of each experiment, and automated the data collection and uploading to a file server so that the data collected does not reside on the desktop used in the experiment and does not bias the results of the experiment. Finally, we strove to limit the number for unanalyzed control factors. For example, we ensured that all the experiments were run by the same research assistant.

All of the aboves measures to reduce subject variability and control user bias and attacker intent were not necessarily planned at the beginning of the experiment. We leaned them the hard way. We had to redo the user studies and hire new participants, thus increasing cost and effort.

3.3 Technical Challenges

To collect normal user data, we developed a first sensor that gathers user command data, including command arguments and timestamps. Recall that our objective is to collect a dataset that collects Unix and Linux user command data from a homogeneous set of users and overcomes the weaknesses of the Schonlau dataset. We built a first sensor for the Linux operating system. The sensor uses a kernel hook to audit all events on the host. It collects all process IDs, process names, and process command arguments in real time. The hooking mechanism used is the *auditd* daemon included in most modern Linux distributions. When we deployed the sensor, we could not get enough adopters. Most students on campus did not run the Linux operating system on their personal computers. Therefore, we had to develop a second sensor that runs on Windows systems. This delayed the project by several months.

We developed a second sensor for the Windows XP platform. The Windows sensor monitors all registry-based activity, process creation and destruction, window GUI access, and DLL libraries activity. The data gathered consists of the process name and ID, the process path, the parent of the process, the type of process action (e.g., type of registry access, process creation, process destruction, etc.), the process command arguments, action flags (success/failure), and registry activity results. A timestamp is also recorded for each action. The Windows sensor uses a low-level system driver, DLL registration mechanisms, and a system table hook to monitor user activity. It relies on hooks placed in the Windows ServiceTable, which is a typical approach used by malicious rootkits.

In the first data collection round, we had about 15 volunteers who accepted to install the sensor on their computers and to share the data collected about their normal activities on the computer. All of these students were taking the Intrusion Detection Systems (IDS) class at Columbia University. This sample was not large enough to conduct experiments and achieve results with high statistical significance. Therefore, we had to collect more data when the IDS class was offered the following year. Meanwhile we had prepared a second sensor for Windows Vista. Unfortunately, we realized that most students have upgraded their operating systems from Windows XP to Windows 7 directly.

Developing a sensor for Windows 7 required rewriting the core parts of the sensor, since Windows 7 no longer allowed placing hooks in the Windows ServiceTable to intercept system calls. Moreover, the sensor could not run on 64-bit versions of the Windows operating systems. Even certain updates to the operating system, such as the Windows XP Service Pack 3 (SP3), which includes security, performance, and stability updates to Windows XP, caused the sensor to crash in some instances. This caused our server to lose contact with some user sensors. The data collected for some users covered only intermittent periods of time. In some cases, users had to re-install the sensor. In other cases, users decided to run the sensor on virtual machines, where the guest operating system is Windows XP. This posed a data quality issue. User data collected from virtual machines is only a subset of the user's interaction with their personal computer. Therefore, it may not fully reflect the user's typical behavior. All these technical issues posed data sanitization challenges, which we present in the following section.

3.4 Data Sanitization Challenges

Each installed sensor was given a unique sensor ID. Data collected from one sensor was uploaded to a central server and stored under its own directory. Many users had to to re-install the sensor due to some incompatibilities with their operating system. Therefore, data belonging to one user was stored under different directories. Linking or combining this data was not straightford was in the absence of any user or system identification mechanism, other than the sensor ID. Extensive data analysis was required to find clues that could be used to link the data collected from one user.

Our sensors provided mechanisms for the users to protect their private data and sanitize it if they wished to. Unfortunately, many users did not take advantage of these mechanisms, either due to laziness, or due to lack of awareness of the consequences of revealing their identities to the research and broader communities by sharing their data. It is also possible that users did not care about the consequences if revealing their identity when their data was shared. Whatever the reason was, we had a moral obligation of sanitizing the data and ensuring that the identities of the students were anonymized if possible. This proved to be a major challenges because we did not know the names and user ids of our users. In the absence of the list of user names and IDs, we had to manually review all records of data collected and anonymize these user names and IDs wherever they showed up, such as in file or directory names. This was a very time-consuming process, and the results are probably less than fully-satisfactory.

4 Lessons Learned

After encountering all data collection challenges, we have learned some lessons we share below:

- Initiate the IRB review process as early as possible and anticipate future data and experiment needs of your research project.
- Identify the independent variable controlled in a user study and its dependent variables
- When designing a user study, identify ways for baselining the users and reducing variability, and especially user bias.
- List all the assumptions made about the participants in a user study. Ensure the assumptions are described clearly in the user study scenario. For example, does the participant know whether they are being monitored? Do they know whether the system is baited?
- Ensure that all participants in a user study, who are willing to share their data, take measures to sanitize their own data.
- Anticipate the technology market trends, and ensure that your data collection tools follow these trends.

5 Conclusion

In this paper, we presented several challenges encountered during a data collection project for the evaluation of masquerade attack techniques. Challenges were encountered in all phases of the data gathering project, including the concept, design, and implementation phases. The data sanitization step also presented its own challenges. We highlighted some lessons learned throughout the project. We believe that data collection projects that involve human subjects, *i.e.* user studies, require extensive planning. Many stakeholders are involved, including the IRB. Many sources of variability should be controlled or eliminated, and special efforts are required to reduce user bias and protect user privacy.

Acknowledgment

This material is based on work supported by the US Department of Commerce, National Institute of Standards and Technology under Grant Award Number 60NANB1D0127, the US Department of Homeland Security under grant award number 2006-CS-001-000001-02 and the Army Research Office under grant ARO DA W911NF-06-10151. The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the US Department of Commerce, National Institute of Standards and Technology, the U.S. Department of Homeland, or the Army Research Office.

References

1. Ben-Salem, M. RUU dataset: http://www1.cs.columbia.edu/ids/ruu/data/.
2. Ben-Salem, M., Hershkop, S., and Stolfo, S. J. A survey of insider attack detection research. In *Insider Attack and Cyber Security: Beyond the Hacker* (2008), Springer.
3. CERT. 2010 e-crimes watch survey, 2010.
4. Keppel, G. *Design and analysis : a researcher's handbook.* Pearson Prentice Hall, 2004.
5. Milgram, S. *Obedience to Authority: An Experimental View.* Harpercollins, January 1974.
6. Schonlau, M. Schonlau dataset: http://www.schonlau.net.