A Log Analysis based Intrusion Detection System for the creation of a Specification Based Intrusion Prevention System

Andre' Muscat

Department of Computer Science and AI,
University Of Malta

Abstract. We propose a novel Intrusion Prevention System (IPS) which would base its knowledge and operation on a higher level of abstraction than the processing of the contents of the network packets audit data themselves which is the source of data on which most current and proposed Intrusion Detection Systems (IDS) base themselves on. We focus on what is actually being asked of the system, and use that understanding together with research on prediction based systems to build a specification based Intrusion Prevention System based on the patterns extracted from higher level application or operating system logs.

1 Introduction

While the world of Intrusion Detection Systems (IDS) technologies and methodologies for detecting attacks seem to move ahead at the speed of light, this is not quite the case. By observing a report commissioned in 2000 by the Software Engineering Institute (SEI) [12] coupled with my experience in the security field for over four years, one would be able to notice that the systems used for IDS have not developed so much. Granted, features and functionality are being integrated into products rapidly, however the methodologies used in detecting attacks did not adhere to such a tight schedule. All detection mechanisms used to expose illicit activity on a network can be broken down into two main categories which do not overlap. These are Pattern Matching and Anomaly Detection.

Pattern Matching based IDS work by looking for patterns/signatures in the information data they are processing in order to detect attacks. When a pattern in the data fits a signature the system would issue an alert to the security administrator. As a direct result of this operational method, pattern matching based solutions are generally effective at detecting known current intrusion methods, but quite ineffective against novel or new attacks until a signature for that new attack method is released.

Anomaly Detection based IDS work by establishing a "normal" operating method of a system. They create patterns of usage of that machine. When some action/activity happens outside of the normal operational usage patterns, it would trigger an intrusion alert. Anomaly Based IDS, unlike Pattern Matching IDS are generally quite good at detecting novel attacks but can have a tendency to generate many false positives as a result of "off the normal" legitimate actions which a user of a system might perform. From personal experience we came to think that unlike several marketing campaigns claim, all IDSs on the market use some combination of the above methodologies for detecting attacks using different types of data sources.

In this paper we will be analyzing the reasons why a security administrator of an organization/networked system would require an Intrusion Detection System as part of their standard

collection of tools which are required for the proper daily operation of their administration and resource security enforcement job. However we will not stop there. After making a case on the importance of having an IDS and how they contribute to close "holes" inside a network setup, we will also proceed to analyze how totally unrelated (non IDS) tools, which a typical high end administrator uses in his daily task of administration and integrity keeping of his network can be used in order to create a system which based on IDS technology can actually be used to create an Intrusion Prevention System – an IDS with the capability to react based on intelligent information processing. This will allow the tool not only to detect an intruder but will also be able to stop the intruder point blank during the course of an attack.

We will be covering both common methods through which an attacker compromises an operating system for his own current and future needs, as well as analyze what is being done by security engineers and experts in order to learn how attacks are being made directly from the source, i.e. the attackers themselves without paying them or promising rewards. This system/method used is called a HoneyPot/HoneyNet [15] and is proving to be very useful to a security administrator who wants to be instantly aware of new attack techniques/variations of already existing attacks that are being used.

We will also be covering details on the pitfalls which we are expected to find on the road and how we will be addressing these issues such as noise in the information domain, the size of the information domain and how this data can be formatted to enable information coming from different sources and software products to contribute to the creation of the IPS intelligence.

At the end of the paper we will also be mentioning a novel intrusion prevention method which is aimed at blocking a specific type of attack – buffer overflow attacks which is still in the research phase.

2 The Need For Securing Resources

Any computer user has heard the buzzword "security" and the need to protect the information contained on a private computer from prying eyes. The main problem to be replied in most if not all cases is "How can this be achieved ?". From experience, a security aware administrator knows that the best solution is to deploy defenses on a network in a layered manner – not one defense mechanism, but many different types and at different levels addressing different monitoring needs. These defenses include the deployment of secure operating systems, the creation and deployment of security policies, vulnerability scanning, patching, access control and authentication, encryption, program wrappers, firewalls, intrusion detection, and intrusion response, and also disaster recovery software/procedures as well as proper configuration of the operating system features and services to support the required security policies. Although this list is quite long a security aware administrator would also be required to train the people making use of the system in order to ensure that the secure system remains secure. Irrespective how many systems/policies one would deploy all of them will be nullified by a user of the system who does not understand and follow the security practices required. For e.g. a user with an easily guessable password like "love", name of wife, name of car etc can nullify all of the security measures implemented since an intruder would use those credentials to make use of the system resources. More information on the dangers of weak passwords and how they can be prevented/enforced by an operating system based security policy can be found at [13].

With the developing complexities and improvements of operating system, security software, security standards and the need of people to know what is going on their network, the need to provide more human readable and processed data was required. This data is the first point of reference a security administrator would run to in order to check out what is going on the network, on the first

suspicion of an intrusion/abnormal/not-desired activity. This data takes the form of logs, which are outputted from all of the above mentioned types of software which are used to enforce security and understand what happened on that system in order to correct the security measures on that PC and prevent such incidents in the future.

Given a multilayered type of network security deployment even in a moderately sized network of 20 computers would generate more information and logs which an administrator would find heavy difficulties in be able to keep up with in order to monitor the daily activity on the network. From our own personal experience in the field, these logs are typically only accessed depending on the problems/reports encountered as the days go by. However this means that if a suspicion is in place the attack may have already been done and executed, which in itself may be too late to protect against for this attack, however on a brighter side the information collected from this attack may be used to prevent future similar attacks.

Current IDSs might inform a user that an intrusion has occurred, however to find out how this was done, an administrator would still be required to delve into these operational/trace logs to look for some indication on what mis-configuration/vulnerability was used by the intruder. Even more worse, without any form of monitoring a user might have left some files/tools around the operating system purposely. These files may be used to hide the actions performed or even worse replace normal operating system files containing custom backdoors in order to enable easier access on future attacks without being detected. Unfortunately, without total monitoring/detailed protections, once a system is compromised (after a successful intrusion), without a proper prevention system, the entire system would have to be re-installed from scratch in order to ensure the integrity of a computer system. This is the main concept used by RootKits to take control of a system. These kits are developed specifically to operate at the kernel level rendering them fully capable access everything and everywhere while being able to hide the operations of an attacker from the unwitting security administrator. RootKits replace central operating system files giving the impression that nothing changed, to the naked eye. RootKits can either take the form of Trojans attaching themselves to files or by totally replacing the original core operating system files with custom made system files with custom code specifically designed to hide illicit activity on that system by the attacker. More information on RootKits can be found at [11] and [12]. Special file system level integrity monitoring and Trojan detection tools can be used to be informed when such sensitive files are changed. Tools such as Tripwire [8] are used for such file level intrusion change detections.

Even well administered networks are vulnerable to attack. Recent work in network security has focused on the fact that combinations of exploits are typical means by which an attacker breaks into a network. [5] Without multiple layers of security software deployed on a system, that very same system would be just as strong as its weakest point of entry. This creates problems since there are automated tools through which attackers would simply supply a machine IP, let it operate and let it report the weaknesses of that system. Based on those weakness reports the attacker would be able to devise his next steps of attack. A growing tool in the open source community is working on a project whose output is a tool named Nessus [9] which although designed for administrators to find the "weakest link" in the security infrastructure of a system, can be itself used by an attacker to devise his attack pattern in order to go by undetected.

3 NIDS vs. HIDS

In the field of detection of unwanted intruders on a computer network system, there are two main types of Intrusion Detection Systems (IDS) which act on different sets of data. Network Intrusion Detection Systems (NIDS) operate on the network packets audit data being received at the network point of a machine, while Host Based Intrusion Detection Systems (HIDS) work by monitoring the actual applications behavior on the host and its interactions with the underlying operating system.

It is a fact that in the past data contained within a packet received at the network level was easier to access, decode and process. This resulted in a concentration of research and development of several IDS systems based on the processing of information contained in network data packets. The literature field also seemed to concentrate most on this type of data. However with the movement of high speed Ethernet cards from 100Mbs to 1GBps, coupled with the encryption of the data contained in the packets via the use of technologies such as IPSEC/SSH, it has become increasingly difficult for NIDS to keep up both the precision at which they can detect intrusions as well as guarantee the detection of attacks without a high rate of false positives.

In this research task, we are giving Host Based IDS the attention it needs in order to use current technologies to process as much of the information we may already have already at hand in order to build patterns and systems, which can actually not only detect, but also prevent attack methods.

In HIDS, the most common approach taken is to monitor relevant interactions between an application and an operating system by monitoring the system calls on that operating system. This resulted in the creation of systems which are based once again on monitoring a pattern of calls rather than the real logical operations that an attacker would be thinking in order to gain access to a system. Such a level of analysis also makes it easier for an attacker to hide the tracks of the attack by mimicking pause periods and fake system calls to hinder system call based HIDS by applications.

4 Data Mining Applications

Typically most IDS technologies require the use of pattern constructs which are used to either define attacks/abnormalities. Whether we are working on an IDS or an Intrusion Prevention System (IPS), we still require the creation of patters via manual Knowledge Engineering from a network/security expert, or the use and application of modern methodologies to process all of the large amounts of information, in order to extract patterns which can be monitored and adapted by the knowledge engineers. Given the complexities of today's network environments, needs and the sophistication of the increasingly hostile attacks the knowledge constructed based outputted from the knowledge engineer only is often limited and unreliable [1]. Data Mining applications is a field which is directly addressed towards large data set information grouping and searching. Data mining is an area which can offer large opportunities to the refining of the pattern matching rules used by IDS.

"Data mining tasks and algorithms refer to the essential procedure where intelligent methods are applied to extract useful information patterns from huge data sets. There are many data mining tasks such as clustering, classification, regression, content retrieval and visualization etc. Each task can be thought as a particular kind of problem to be solved by a data mining algorithm. Generally there are many different algorithms which could serve the purpose of the same task. Meanwhile some algorithms can be applied to different tasks"[16].

5 Pre-Processing

As mentioned earlier in the paper, we are proposing the creation of a specification based IPS based on the patterns which are extracted from the various forms of logs which are outputted from all of the various types of software mentioned earlier on. However this is not as simple as it might sound. Most of the logs which are output from the various applications are proprietary and not conforming to a specific standard of any form. This means that the information contained is essentially different from one log to another. What makes this task even more complicated is that different logs will

offer different views to the system. In essence this task involved the analysis of various types of log files in order to help find commonalities and also group different type of logs in order to help create a framework which would allow the attainment of our aims.

Another aspect which is involved in log analysis is the inherent noise, missing values and inconsistent data in the actual information contained and written in the logs. It is a fact that all attackers know that all applications are vulnerable in some form of another. During research we also came across situations whereby attackers found and used methods which by supplying say a firewall with a particular type of data (in UNICODE format) they could literally crash the firewall and hence have absolute access to the system, as well as hide details of their activity since the firewall was not logging the information coming from the source of the attack. A log analyzer suffers from that very same danger. The importance to transform and process the data supplied to the log analyzer in a standard format is a key process to the standardization of the logs to be processed as well as offer an extra layer of protection before actual processing of the data, for e.g. Microsoft use their Event Viewer for log output, third party tools their own etc. A standard by a security company named NETIQ proposes a format named the WebTrends Enhanced Log Format (WELF) format which proposes a standard which is most ideal for large scale text based logs. With this format we can transform practically every log entry/system call equivalent to a standard format on which operations will take place. In addition real world data sets tend to be too large and some multi dimensional. Therefore we need data cleaning to remove noise, data reduction to reduce the dimensionality and complexity of the data and data transformation to convert the data into suitable form for mining etc.

6 Common Pitfalls

The use of anomaly detection in practice is hampered by a high rate of false alarms. Specification based techniques have been shown to produce a low rate of false alarms, but are not as effective as anomaly detection in detecting novel attacks. While detection of new novel attacks can be an issue, we are working on a system which will prevent known future attacks which are based on common methods. As an example in windows operating system environments, it is common for attacks to originate from the usage of particular sensitive operating system files – `cmd.exe`, `ftp.exe` etc. Indeed these files can be protected by basic file system level protection, however in large systems, file system based protection is not always possible and may require another type of level of protection. The rationale is that although an attacker may find a common operating system weakness, an extra level of protection based on specifications coming from previous analysis steps can be used to stop the attacker even if the underlying operating system defenses have been beaten.

7 Modern Techniques Used For Knowledge Discovery

To collect information on the way attackers are using system vulnerabilities, organizations and research groups are making use of what is called honey pots and empty systems whose function is to react like a normal operating system, however while giving the impression to the attacker that they got access to a legitimate system, the attacker is actually revealing his methods, tools and tricks to a highly monitored and specialized environment based exactly to root out the modern attacks. These systems are what are known as Honey Pots. The information collected from the honey pots and nets can be then combined, pre-processed, transformed and processed in order to devise specifications which can be then fed into our IPS in order to identify activities on a high

level logistical later and prevent by reacting to/redirecting attackers as we may need in order to subvert attacks. The simplest form of reaction is to close a connection, however unless the IPS is capable of detecting complex logistical issues and be able to think on a high level like a user, it will always be open to further attacks and weaknesses.

8 Being Realistic

For the development of an IPS we have to be able to assume that an attacker can silently take control of an application/operating system itself without being detected, via some form of social engineering paradigm or system vulnerability/exploit. Our system must also be able to cater for the unknown. There are actual penetration methods which can leave no trace in the system call trace especially if no logging is enabled. For instance exploiting a buffer overflow attack involves only a change in the control flow of the program, but does not itself cause any system calls to be invoked and thus no system call based IDS can detect the buffer overrun itself. Since no system call is made even the operating system would not report anything. Indeed an IDS which monitors the integrity of the code segment of an application would be able to report such an attack, but would be difficult to prevent it. Work is being done to see whether a way can be found to block this type of attack as soon as the MemCpy Command is used.

Contrary to popular belief there is little to be done against such attacks. Fortunately there is little harm that an attacker can do to the rest of the system without executing any system call. However this proves the point that different IDSs are required at different levels. An IDS which checks for the integrity of the code segment of a program can be sufficient to detect such an attack but not block it/prevent it.

In order to address such attacks and also prevent them were possible we are thinking of applying the same concept used in the processing of the logs to treat the parameters passed to a system call as a field entry in the logs. Like that we may be able to perform checks on the parameters of the call and hence also start preventing buffer overflow attacks even the actual call is executed by monitoring the request. This part is still in research and treating it as a section for Future Research.

9 System Implementation

As a result we need to find patters in order to model and understand what a user is effectively using and doing rather monitor only the patter of operating system calls. Based on the patterns which are generated either by Knowledge Engineering or patterns resulting from Data Mining Techniques we can build a Finite State Machine Based System through which we can determine/predict the next state based on a number of previous states and if our prediction comes to reality also decide to block the operation from taking place.

$$\underset{(E0,\ E1,\ E2,\ E3) \rightarrow (E1,\ E2,\ E3,\ E4)}{\text{Action}}$$

By monitoring a sequence of operations one can already predict the next operation based on past experience. The real trick in our proposal is to use the past experience patterns not to predict but to know what is going on on a system secured by our final system. This would allow instant intrusion prevention if our IDS is intelligent enough to understand and react at levels which are higher than operating system calls.

10 Conclusion

In these couple of pages we tried to present an outline of our work, however due to space limitations we had to skip some details which might have been more illuminating to the shady areas of the most discussed but yet unknown and mysterious field that is the security section of the computing world. The research covers various technologies used for security, intrusion detection, prevention, as well as systems and algorithms used by the above mentioned technologies including automation data mining systems and algorithms as well as the development of a framework which is operating system independent to be used for intrusion prevention as a protection against the attackers of tomorrow.

References

1. Wenke Lee, 2003. Applying Data Mining to Intrusion Detection: The quest for Automation, Efficiency and credibility.
2. B.Schneier, 2000. Secrets & Lies: Digital Security in a Networked World, John Wiley & Sons Inc.
3. Peng Ning, Yun Cui, Douglas S.Reeves, 2003. Constructing Attack Scenarios through correlation of intrusion alerts.
4. Cliff Changchun Zou, Weibo Gong, Don Towsley, 2003. Code Red Worm Propagation Modeling and Analysis.
5. Paul Ammann, Deminda Wijesekera, Saket Kaushik, 2003. Scalable, Graph Based Network Vulnerability Analysis.
6. David Wagner, Paolo Soto, 2003. Mimicry Attacks on Host Based Intrusion Detection Systems.
7. WebTrends Enhanced Log Format (WELF) `http://www.netiq.com/partners/technology/welfcert.asp`
8. Tripwire. `http://www.tripwire.com/literature/`
9. Nessus Project. `http://www.nessus.org`
10. Eric Cole, 2002. Hackers Beware – Defending your network from the wily hacker, New Riders Publishing.
11. Stuart McClure, Joel Scambray, George Kurtz. McGraw Hill, 2003. Hacking Exposed – Network Security Secrets & Solutions, Fourth Edition.
12. Julia Allen, Alan Christie, William Fithen, John McHugh, Jed Pickel, Ed Stoner, 2003. State of the Practice of Intrusion Detection Technologies `http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html`
13. Benny Pinkas, Tomas Sander, 2003. Securing Passwords Against Dictionary Attacks.
14. CERT Coordination Center. `http://www.cert.org`.
15. The Honeynet Project, 2001. Addison-Wesley. Know Your Enemy,
16. Tao Li, Qi Li, Shenghuo Zhu, Mitsunori Ogihara, 2003. A Survey on Wavelet Applications in Data Mining.