NETWORK SECURITY: COMPROMISES AND COUNTERMEASURES

Thomas C Bryan
Department of Computing Sciences
800 E Lancaster Ave, Villanova, PA 19085
thomas.bryan@villanova.edu

KEY WORDS

network security mimicry ddos traceback deception mining honey pot

ABSTRACT

Network security is a long-standing and well-recognized problem in the computing world. Protecting information is a main priority of a network administrator, with security techniques becoming increasingly robust. As more and more information has become digital, security of this digital information has tightened and mitigation techniques to protect the information have become more complex. While approaches to security have improved, intrusion techniques have also become increasingly complex and difficult to detect. This paper discusses two common network security attacks and two approaches to defending against such attacks, as well as an examination of honey pots (a supplemental defense), and uses these to motivate a discussion of the bigger issues involved and what the future of network security might look like.

1 INTRODUCTION

Network security has been around since the idea of a network was conceived. Security has many reasons of importance, first and foremost being the protection of confidential or sensitive information from prying eyes. This confidential or sensitive information are things such as stored bank account numbers, personal identification numbers, credit card numbers, social security numbers, and the like.

There are many methods of security being used today, such as deception and traceback methods as well as various data mining methods. Many of these defense methods utilize a concept known as a honey pot, or a virtual network of machines that can be used to keep an attacker online while their location is traced. Also in use today are various methods of circumventing those security measures such as using mimicry or distributing a denial-of-service attack amongst a network of compromised systems.

Our proposal involves a thorough investigation of existing honey pots and creating a benchmark of their strengths and vulnerabilities to real attacks. Details of my proposed work are presented in Section 4. In the following we survey known types of network attacks and defenses.

Section 2 discusses two types of network attacks, namely mimicry and distributed denial-of-service attacks. Defenses to such attacks, such as Data Mining methods, Deception/Traceback methods, and honey pots will be discussed in Section 3.

2 NETWORK ATTACKS

In this section, two types of network attacks, mimicry and distributed denial-of-service, are discussed. In a mimicry attack, the attacker attempts to silently take control of the system and deceive the IDS, while in a distributed denial-of-service attack the attacker uses a network of compromised systems to launch a brutal denial-of-service attack on a network. By understanding the details of these two forms of attacks, a network administrator or other security expert can better design approaches to protect against such attacks.

2.1. MIMICRY ATTACKS

Mimicry attacks are one of the hardest attacks to detect in that they are done through stealth and deception. There are several methods of executing a mimicry attack on a system. Each method has a few assumptions, such as the idea that the attacker knows the inner-workings of the IDS in place, they know how to gain control of the system silently and quickly, or that no-ops are being utilized by the attacker. By putting the problem of detection into terms of the formal language theory, the system can successfully detect system calls containing malicious code sequences. [1]

Implementation of this system is fairly straightforward. The IDS must be trained to recognize malicious code sequences, and a database of normal sequences of system calls must be built. Many optimization paths are available by using ideas from the model-checking literature. By performing depth-first searches on the fly, memory space is greatly reduced, so the IDS is efficient in both space and time. By modifying an off-the-shelf exploit script (autowux.c), it was possible to successfully deceive and

compromise the well-known pH IDS (a derivative of Forrest, et al.'s early system, with the twist that pH responds to attacks by slowing down the application in addition to raising alarms for the system administrator).[1]

To protect against this type of attack IDSs must be able to analyze commands extremely rapidly, as well as recognize various techniques employed by mimics such as no-op flooding. Mimicry attacks are one of the most difficult attack methods to guard against today.[1]

2.2 DDoS ATTACKS

Distributed Denial-of-Service (DDoS) attacks are among the most difficult to defend against due to the sheer mass of the attacking swarm of computers. By examining the source code of four popular DDoS bots, or the robot responsible for automating the attack, various methods of mitigating systems against their respective botnet attacks can be explored. A botnet is a network of compromised systems that can be controlled by the bot to launch a cooperative attack on a target. [2]

Agobot, one of the most popular bots (with over 600 variants are listed by the Anti-Virus vendor Sophos), is the first bot studied. Written in a modular structure in C++, the bot provides cross-platform capabilities, as well as the ability for new attacks to be easily added as the necessity arises. This bot has the most comprehensive list of attack tools. SDBot, another popular bot (with over 1800 variants listed by Sophos, a popular anti-virus company with its main focus on business application), is widely available. This bot is written in C++ and targets Windows systems. RBot has over 1600 variants listed by Sophos, and is also written in C++ to target Windows systems. Spybot has over 200 variants listed by Sophos, and is written in C, also affecting Windows systems. [2]

IP address spoofing, or faking the IP address of a system, is in play for all four bots examined, as well as source and destination port randomization. This makes mitigation against such attacks difficult, as identification is more difficult. However, as the randomization gets greater, it will increase the number of anomaly packets, which aids in the identification of an attack. Randomization throughout an attack causes a multitude of problems for a system administrator. [2]

Many of the mitigation techniques in use today can be easily bypassed. Source spoofing remains an issue, not in locating the attack controller, but in neutralizing the botnets used to launch the DDoS. Network protection providers must incorporate learning techniques and adaptive mechanisms to respond and repel an attack in a timely manner. [2]

3 NETWORK DEFENSES

In this section, three types of network defenses (data mining, deception/traceback, and honey pots) are discussed. In the data mining technique, users of a system are profiled based on their command sequences, spelling mistakes, and actions as a whole. If a user cannot be identified using these characteristics, or a known user begins executing an attack sequence, the system will alert the appropriate administrators of the attack. In the deception/traceback method, multiple modules are brought together to make up a defense system that both deceives the attacker by using a honey pot and tracks the attacker's location by using packet traceback. The honey pot technique is basically using a virtual machine to trick an attacker into trying to gain access to it while the system decides how to respond to the occurring attack. By understanding the details of these two forms of defense, a network administrator or other security expert can be ready to utilize one or more defense schemes to protect their resources

3.1 DATA MINING DEFENSES

Most systems today use user IDs and passwords to identify users. A system called the Intrusion Detection and Identification System (IDIS) mines log data to identify command sequences used by an operator, and then identifies the operator instead of relying on IDs and passwords. The system consists of a command monitor, a detection server, a mining server, and a honey pot. The monitor collects input commands from the user and transfers the collection to a detection server, which compares the commands with attack patterns in real-time to identify the sources of the attack. By examining the interactions of the user with the system, user profiles can be built, and referenced in order to identify users at later times. The mining server extracts the individual commands used by each user from their log files, counts the number of times each command sequence appears in the log file, and stores the result in the user's habit file. This habit file is then compared with all other users' habit files in the underlying system to identify user-specific command sequences and common command sequences to all users. The detection server is used to detect malicious command sequences on the fly, and triggers the honey pot, which will collect attack patterns in its own log file. After examining 655 students' log files (424 being computer science students and 231 being Management or Social Science students), it was determined that 95.03% of the computer science students were identified correctly, and 90.82% of the non-computer science students were correctly identified. [4]

3.2 DECEPTION/TRACEBACK DEFENSES

The deception and traceback defense method uses multiple modules to combine the intrusion deception technique called a Honey pot with a basic traceback technique. The new scheme is called PDDT (**P**roactive **D**efense Model based on Intrusion **D**eception and **T**raceback). [3]

Consisting of three separate modules, the system effectively traps an intruder in a trap environment, and then executes a traceback program to locate the origin of attack. The intrusion deception module implements deception to the invader and then guides it into the trap environment, all the while collecting attack data while interacting with the attacker. The traceback module receives and disposes of the data stream captured, then traces back and locates the source of attack by way of advanced probabilistic packet marking. The management and control module is made up of the local control center and a teleconsole. The local control center is in charge of setting up the trap environment and configuring the firewall and IDS. The teleconsole is in charge of logging and communicating with each of the other modules.[3]

The performance evaluation of this system was carried out on a SUN Solaris ftp server and the Fang Zheng's Yuanming web server located in the researchers' lab. Several deception hosts running *honeyd*, a free honey pot application, were deployed on the network, each composing a unique trap environment. Snort was selected as the IDS of the deception module. Boson NetSim software was run on one host to imitate a bank of Cisco routers, and also to assist in the traceback by way of packet marking. Another host was running the CGI procedure established by Java to play the role of the teleconsole and logging server. MD5 was used as the encryption standard between hosts and modules. Linux Red Hat 9.0 was deployed as the universal operating system.[3]

20 DoS attacks were launched against the defense system including TCP SYN Flood and IP Spoofing attacks routed through various numbers of intermediate nodes. The experiment determined that the PDDT system has effective real-time capability and flexibility, as well as good proactive defense capabilities. There is still work to be done; large-scale DoS defense, as well as locating the attack's source precisely in the event of IP forging or using compromised hosts as new attack sources. [3]

3.3 HONEY POTS

Honey pots, while not generally viewed as an active defense mechanism but more of a supplement to an already-in-place IDS, utilize simple deception. Zhang et al. [6] investigated honey pots fairly extensively and divided them into four categories: prevention, detection, reaction, and research. Prevention honey pots are primarily used to delay an attacker from getting to valuable information by simply wasting their time with false information. Detection honey pots generate alerts to security officers and administrators. This is easily done because no legitimate traffic should be using the honey pot's resources; all traffic to the honey pot is considered hostile. Reaction honey pots are used to be a companion system to an IDS, and is used as a trial environment to test for system vulnerabilities. A research honey pot is used in order to gain information about an intruder by keeping logs and studying and reporting on the intruder. Honey pots can fulfill any of these roles or any combination of them at once.[5]

Honey pots are beneficial for a number of reasons such as keeping intruders occupied, studying the actions and attack patterns of attackers, being a safety net for IDSs by trapping undetected attackers, as well as reducing the processor overhead in a system.[5]

If an administrator or security officer chooses to use a variety of resources for the honey pot to emulate as bate, they can then determine the interests of the attacker. The honey pot will allow the attacker to go about his business, and will be logging each and every move the attacker makes. This log can then be studied by the security officer or administrator in real time so an effective response to the attack. This method of buying time for the response also allows the security officer to learn the path that the attacker took to get to the system. Tracking this path generally requires a good deal of time and experience to determine the full path of attack. An added benefit of this method of response is that it simply wastes the attacker's time, delaying him from compromising the valuable resources being protected by the honey pot. The more convincing the honey pot is, the more time will be wasted.[5]

The honey pot often acts as a safety net for the IDS that is in place. The IDS will never be perfect, but with the honey pot in place, the reliability of the defense system as a whole is greatly heightened. An added benefit of the honey pot is that while IDS detection often depends on the type of attack being experienced, honey pot detection is completely independent of the type of attack; a mimic will be treated exactly the same way as a DDoS attack.[5]

Processor overhead is often a concern when IDSs are in play. Each packet transaction in the system must be examined, so the IDS generates a high demand within the system. In addition to this, data transfer also adds a great deal of processor overhead into the system. The honey pot greatly reduces this overhead due to the fact that the resources are only needed when the honey pot system is being utilized, which is only during an attack.[5]

Given that honey pots are not the only system visible to the attacker (valuable resources are also visible), the honey pot is not always the system that gets attacked first. In order to increase the chance that the honey pot will be the first thing to be intruded, there are usually a fairly large number of honey pots on any given network. These are generally close to the valuable resources to act as both guards and decoys. Such a group of honey pots on a network is called a honey pot farm.[5]

The concept of honey pots is not privileged information, so attackers know of their existence. There are tools out there that can be used to detect honey pots, and once identified the honey pots can be disengaged. Once the honey pot is disengaged, the resource being protected by it is completely exposed to the attacker. In order to defend against such tools, Zheng et al. [6] proposed a design for a dynamic honey pot, which is capable of changing its configuration to match the dynamic environment of its network. [5]

Another proposed system is that of physically and logically roaming honey pots to defend against service-level DoS attacks. This system would allow the honey pot to randomly move its position within a server pool, which would add the beneficial features of filtering and connection dropping. Filtering would result in an attacker's address being blacklisted on the network. Connection dropping would involve the server dropping all connections when the server changes from idle, acting as a honey pot, to active, acting as a legitimate server. The dropped connections would all be hostile connections, and once dropped would allow space for legitimate connection requests. In a logically roaming scheme, the system would coordinate legitimate clients and the server to randomly change destination addresses within their packets. By doing this, hostile traffic would be dropped due to their packets containing incorrect destination addresses. [5]

It is well-known that honey pots are not a defense in their own right, but are an invaluable method of deception that often buy security officers and system administrators valuable time to devise an effective response. There is much room for improvement, and as the attackers gain more tools to render honey pots inoperable, the defenders need to improve them constantly or else they will actually become a weak spot in the defense.[5]

4 PROPOSED WORK

4.1 INTRODUCTION

We propose to examine the currently available IDS and honey pot systems to determine their unique behaviors and limitations. The goal of this research is to improve existing IDS and honey pots to better detect and deceive attackers in real time. The objective of this is to examine the three most widely used IDS systems in combination with the top three honey pot systems, a total of nine defense systems, and report on their robustness after executing various standard attacks on them. The research is divided into two phases, discussed in sections 4.2 and 4.3 respectively. Project design and timeline are discussed in section 4.4. Anticipated results and dissemination methods are discussed in section 4.5. A personal research statement is presented in section 4.6 and the proposed project budget is discussed in section 4.7.

4.2 PHASE ONE: ATTACK

Phase one of the research would consist of executing various attacks, such as mimicry and DDoS attacks as well as others, on the systems. By doing this we can determine which are the most robust combinations of systems. The robustness of the defense system would be determined by comparing each combination of IDS and honey pot in terms of their ability to identify the path of attack, their degree of interaction with the attacker, and their ability to successfully defend the system against an attack.

4.3 PHASE TWO: ANALYZE

The data stream of each attack would be captured for analysis in phase two of the proposed research. In this phase we would analyze all network packets generated during each of the launched attacks on each of the IDS/honey pot combinations. This would be valuable in order to determine the attack patterns that affected the system the most, as well as determine patterns that may have gotten past the IDS and honey pot system. By determining which attack patterns were unsuccessfully detected, current IDSs could be improved to better detect them and variations upon them.

4.4 PROJECT DESIGN/TIMELINE

In order to do this research, the IDS and honey pot software must be procured. This will be done legally and the numbers involved in this are included in the budget proposed below in section 4.7. A "clean" host system and virtual network must also be obtained, which can easily be done from the Computer Sciences Department at Villanova University. Knowledge of how to successfully execute each attack outlined in the proposal must be gained, so there will be quite a bit of research and practice necessary.

We will begin the proposed work by doing research into the execution of attacks on a system, as well as how to correctly configure each IDS and honey pot configuration to ensure the greatest protection possible for each combination. Following this research period (most likely two weeks), the system will be set up, consisting of one

physical computer hosting a network of virtual "physical" systems as well as virtual "virtual" systems. In order to attack this virtual network, we feel that a separate system should be used to launch the various attacks, or else the analysis could get too confusing with all the data coming and going to the same host computer.

We feel an adequate amount of time to set up each combination of IDS and honey pot and then execute the various attacks would be three days per IDS/honey pot combination. After the attacks are completed, the robustness must be examined for each combination. We feel one week is sufficient to accurately determine the order of robustness for the set of combinations. In this manner, it will take about five weeks to do the first phase of the research in totality, plus the two-week front-end for research and setup.

Phase two will take a good amount of time due to the sheer volume of data that would need to be examined. We feel that one week per IDS/honey pot combination would be enough time to analyze the data streams. On this time schedule, phase two would be completed in nine weeks.

At the end of the second phase of research, the data would need to be documented, edited, formatted, and approved by colleagues and superiors. This would add another week or two to the back-end of the research. To account for unforeseen circumstances, we will add two buffer weeks to the timeline. In total, this project should take about 20 weeks, or 5 months, to complete.

4.5 ANTICIPATED RESULTS AND DISSEMINATION METHODS

The anticipated results of this project would be in the form of a publishable manuscript of the research findings as well as a conference paper to be submitted to the National Information Systems Security Conference. These results will point out the behavior of the different combinations of IDS and honey pot systems, as well as their unique shortcomings and limitations. This research would hopefully spur improvements on the IDSs and honey pots examined.

4.6 PERSONAL STATEMENT

This topic is important to the forward movement of the network security field as a whole. In addition to that, it is personally interesting to me, as I plan on pursuing a career in the network security field once I graduate from Villanova University. By doing this research, it will be evident to future employers that I have intimate knowledge of the currently available IDS and honey pot systems, as well as knowledge of a number of attacks that I may be tasked with defending a system, or a number of systems, against in the future on the job.

4.7 PROPOSED PROJECT BUDGET

The proposed budget for this project is roughly $6,000. This can be broken down into $199 for the Specter Research package of the Specter IDS (which already has a honey pot system integrated into it. We may add one test to the research, but will disregard the built-in honey pot and use Specter in combination with the honey pots chosen). The other two popular IDSs are open source, and therefore free. The honey pots are all available for timed trials, so they are free for enough time to do the testing. The only other thing requiring financing would be the publishing of the resulting paper, which we allotted $100 for. Unforeseen expenses have been allowed for, with $300 going towards those. The remaining portion of the budget (roughly $5,400) will be used to pay the primary researcher and one assistant (assuming they each work 15 hours per week, at $10 and $8 per hour, respectively). By using borrowed equipment and free trials, this research is fairly cost-effective.

5 CONCLUSION

By examining two types of both network attacks and network defenses, one can begin to approach the topic of network security in a general sense. The deployment of honey pots is both a basic and effective method of deception and systems mitigation. In order to move network security forward as a field, honey pots must be improved upon, as well as used in various ways. In the event of an attack, a honey pot system should be dynamic enough to keep an attacker online and interested in the system itself, but at the same time should be safe enough and have the capabilities to ensure the attacker can be identified. The cat-and-mouse nature of network security ensures that it will continue to evolve as a field, and will become a much more specialized field as it does evolve.

REFERENCES

1. D. Wagner, P. Soto, "Mimicry attacks on host-based intrusion detection systems", Proceedings of the 9th ACM conference on Computer and communications security, Washington, DC, USA, November 18 - 22, 2002, pp. 255–264

2. V. L. Thing , M. Sloman , M. Dulay, "A Survey of Bots Used for Distributed Denial of Service Attacks", Proceedings of the IFIP TC-11 22nd International Information Security Conference (SEC 2007), 14–16 May 2007, Sandton, South Africa, Volume 232/2007, pp. 229-240

3. J. Tian, N. Li, “A Proactive Defense Scheme Based on the Cooperation of Intrusion Deception and Traceback”, 2007 International Conference on Computational Intelligence and Security Workshops, Hebei University, Baoding, China, Proceedings of the 2007 International Conference on Computational Intelligence and Security Workshops, Volume 00/2007, pp. 502-505

4. F.Leu , K. Hu , F. Jiang, “Intrusion Detection and Identification System Using Data Mining and Forensic Techniques”, Second International Workshop on Security, IWSEC 2007, Nara, Japan, October 29-31, 2007. Proceedings, Volume 4752/2007, pp. 137-152

5. P. Kabiri, A. Ghorbani, “Research on Intrusion Detection and Response: A Survey”, International Journal of Network Security, Fredericton, Canada, Sep 2005, Volume 1, Number 2, pp 84-102

6. F. Zhang, S. Zhou, Z. Qin, and J. Liu, “Honeypot: a supplemented active defense system for network security,” Proceedings of the Fourth International Conference on Parallel and Distributed Computing, Applications and Technologies, pp. 231–235.