New Mexico State University

Honeypots and Honeynets Technologies

Hussein Al-Azzawi

Final Paper

CS 579 Special Topics / Computer Security

Nov. 27, 2011

Supervised by

Mr. Ivan Strnad

Table of contents:

1. Abstract

Honeypots/nets can be considered as a part of the intrusion detection/prevision system. By establishing fake area inside the real environment, so when any suspicious activity identified, it will be redirected to this honey area, so we can know who is trying to attack us, why and how. One of the most important things about honeypots/nets is how to isolate them from the production environment so they cause no harm. There are a lot of methods and techniques to build, hide and secure honeypots/nets. [1]

2. Introduction

Honeypot is a part or complete system that is designed to be easily compromised by attackers, so we keep them away from our real environment since they think they already in, also we can track them and study their tactics, methods, behaviors and tools as if they inside our production environment. "A honeypot is a closely monitored computing resource that we want to be probed, attacked, or compromised. More precisely, a honeypot is "an information system resource whose value lies in unauthorized or illicit use of that resource" [2].

The concept of honeypots is somehow new but there is a lot of research in the field. Honeynet is a group of honeypots that simulate real network to false the attackers and collect as much as possible information about them, even if one ore more of the honeypots are compromised by the attacker, we still can track them and may gain more info about them since they become more aggrieve with the pots they got. Actually this is the idea of the honeypots/nets; to provide an environment to the attackers to play with them and to discover their ambiguous world, tactics, tools and the way they think and

start their attacks. "The value of a honeypot is weighed by the information that can be obtained from it. Monitoring the data that enters and leaves a honeypot lets us gather information that is not available to NIDS." [2]

The strength of honeypot/net comes by collecting huge amount of data from all the attacks that may happen, analyzing them and comparing them to what is going on our production environment, using the results to enhance our internal security system like IDS and firewalls. Most of the collected data are valuable since any traffic come to the honeypot/net is suspicious. The honeypots varies in the interaction they provide to the attackers, from the low interaction to medium and high, each type has its advantages and disadvantages. [1]

3. Types of honeypots

"Honeypots can run any operating system and any number of services. The configured services determine the vectors available to an adversary for compromising or probing the system. A high-interaction honeypot provides a real system the attacker can interact with. In contrast, a low-interaction honeypots simulates only some parts — for example, the network stack"[2]. So we can see that there are two major types of honeypots; high-interaction and low-interaction honeypots, many levels varies in between low and high, the following will be a discussion about the honeypots types:

3.1 Low-interaction honeypots

This kind of honeypots provide limited services that the attacker may use to go through, in this type we only simulate specific services that we want to study and

analyze to find out how these services might be attacked and compromised. Like simulating web server that only responds to what it's configured for. The advantage of this is the ease of installation and configuration with low risk. But it's easy to detect by expert attackers which may lead to more aggressive attacks against our production network, another thing is that the results will be limited to the current configuration of the system.

3.2 High-interaction honeypot

This type provides full environment to the attackers, most of the time we provide standard system with most of the known vulnerabilities in that system, like an operating system of a device like router or switch to study what can attackers do with it. This type is more difficult to deploy and configure, also the risk is higher here since attackers can take full control over the machine and may use it to start other attacks that may go through our production network or to take it as a zombie to generate attacks on the internet. But by taking the risk of giving full access to that system the results will be more valuable and interesting more than the low interaction honeypots. The combination of a group of honeypots will result in a honeynet which is a complete network environment that only used for tracking the attackers by simulating a real network that is isolated from our real network.

3.3 Physical Honeypots

Depending on this we have to have a dedicated physical machine to be deployed as a honeypot. "Physical often implies high-interaction, thus allowing the system to

be compromised completely" [2]. The problem with this approach is the cost and management especially if we want to study a large environment that contains a lot of servers and different types of clients.

3.4 Virtual honeypots

Better way to implement honeypots/nets is by having dedicated hardware hosting many virtual honeypots, the idea of this is to implement honeypots on the same physical machine. Each administrator can use his/her favorite virtual environment like VMware or Xen to build honeypots/nets. This approach is cost effective and gives us more options and the ability to build many different types of pots so it will be easy to manage and react. Add to that the loss of compromising virtual machine is less than in case of full physical systems, and it's faster to rebuild new virtual machines if we have backup images.

3.5 Wireless honeynets

Till now the interest in the wireless honeypots is not that popular as in the wired networks, for many reasons; lack of knowledge in wireless vulnerabilities, considering it less dangerous since hackers have to be there to access the wireless network directly by searching for weaknesses in the network to gain access, but this thought is wrong based on the fact that most of the attacks against organizations nowadays are internal attacks so the attacker is already physically in and may have a good understanding of the network infrastructure. Wireless honeypots can be started by providing fake access points, to improve the design we can add some machines

connected to that access point to generate fake traffic to make that WLAN looks real for the attacker, that traffic can be generated using predefined scripts simulating some services, to track the attackers to get involved in, for the network analyst, it's known that if the traffic on the access point is not generated by our scripts then its suspicious so we start tracking the source, to fingerprint the attack type and to find out if this attack generated by an internal employee or an outsider. [5] [9]

There are some tools that provide such honeynet services like "Fake AP" by Black Alchemy, this software can up to 53000 fake access points, according to their website [13]. It's an interesting tool since it provide the ability to create such a huge number of access points, actually I couldn't find another tool that behave like this one, it gives an interesting wide space to play with the hackers, for me the only limitation in that tool is the supported wireless standard, it only supports the IEEE 802.11b standard.

4. Popular Tools

4.1 Honeystick: By UK honeynet project, it's a Gen III Honeywell with honeypots all on a usb stick, there is no specific implementation, but they provide the mechanism of how to build it since some products need to be licensed. To build it we need an operating system running Debian. It's useful when we need a honeypot to be up in small time without the need for any additional resources just a prepared stick carrying the honeystick on it. Still it does not provide full features as a real honeypot. [14].

4.2 Honeyd: It's is a virtual honeypot, "Honeyd is a small daemon that creates virtual hosts on a network." [15] these virtual hosts can be configured in deferent ways to let them look like a full system that is running, so even if an attacker try to identify the OS he/she will not be able to get real information, and while attackers wasting their time we are studying their behavior and tools.

4.3 eTrap: This tool works based on the open relay concept, by providing "spam honeypots", that assigns SMTP service to the honeypot and anyone access this service is considered as a spammer, while the attackers working on this trap we can build our own blacklists of IP addresses and emails that we don't want to receive traffic from by applying these blacklists to our gateway to block them. [7].

4.4 Honeywall CDROM: This technology which created by honeynet.org project is an example of the third Gen honeynets. The honeywall CDROM "is a bootable CDROM that installs all of the tools and functionality necessary to quickly create, easily maintain, and effectively analyze a third generation honeynet." [6]. It combines many tools so it's an easy and simple way to build a honeypot with small period of time since it comes ready, also it can be modified after the installation based on the needs of each scenario. This tool created by the honeynet.org to increase the numbers of the honeypots/nets users around the world so they get more feedback about the attacks that are captured and analyzed on those honeypots which will increase the research cycle in this field of security.

5. Collecting Malware with Honeypots

Malware is a name used to refer to software that used in malicious ways. Serious security problems happens cause of such software especially when they are automatically spreading in the networks. Such software used in DDoS attacks and if they are well managed they can take any site down by isolating it from the entire Internet. As the these malwares increasingly spread all over the Internet we need a way to analyze and identify them as soon as possible before they destroy a lot of resources.

To do this we need special environments to allow these malwares to come in to study and analyze them using tools built for this purpose, which is known as honeypots/nets. Every time we capture a new type we block it or close the vulnerability that allowed it to work. This technique is being used by the largest antivirus and spam organizations, and this is how they generate their periodic updates for their software and products. The following are some tools used to collect malware, these tools are of the type of low-interaction honeypots that they only simulate vulnerabilities, so the system will not be harmed by another types of attacks, at the same time it will provide specific information about the type of malware we want to study:

5.1 Nepenthes:

This software is designed to emulate services that will track malwares to them, once they get the malware or part of it will capture it and store it on the hard disk and it mostly will not be able to execute since this software is designed to capture malwares that attacks Windows while it runs on Linux environment and as we know windows binaries cant execute on Linux operating systems. Nepenthes can create

several thousands of honeypots concurrently on the same machine. It consists of five different types of modules, the first type is vulnerability modules: emulates the vulnerabilities of network services, this provides efficacy since it will emulate only the necessarily part of that service to trick the attacker. The second is Shellcode parsing modules: their job is to analyze the payloads received by the vulnerability modules the analysis done using assembly programs to extract data from the malware. Third type is Fetch modules: use the information gathered by the Shellcode modules to download the malware from the remote site. Fourth type is Submission modules: deals with the downloaded malware binary and may save into database and can send it to antivirus venders to update their databases. And the last type is Logging modules: log all of what happen while the emulation process is going on. [2].

5.2 Honeytrap

Honeytrap is similar to Nepenthes in the idea but the difference is that Nepenthes can't detect unknown attacks since it depends on the signature of the attack, so it can't capture the zero day attacks which are new attacks that still not well known. What Honeytrap do is to open ports on demand so whenever it receives a request on a TCP port it will respond dynamically, following this approach it can serve most of network based attacks. There is no need to let many ports to be open since Honeytrap depends on connection monitors to open ports on demand. There is two types of connection monitors: the first one is a network sniffer based on libpcap: by searching for packets with RST flag which means that these are reject connection request so after receiving such a packet it will open the requested port since the attack will keep

sending for that port for a while, so when it send it again the port will be open and will allow the malware to establish the connection and download it to the honeypot, this is the default type of connection monitors. The second is only for Linux depending on iptables: where the kernel firewall module is running on Linux machines, depending on iptables we can make a rule to redirect the traffic with SYN flags to Honeytrap, so by this way we can capture the malware from the first hit not like the Nepenthes, but the draw back is its no t stealth.[2]

6. Malware Analysis

There are many approaches to analyze malwares, here are the most important types to do the analysis in an automated way, we need automation since the rate of malware generation is increasing so we cant analyze all of this malware with the traditional ways which needs human interaction, the new generation of malware analysis tools analyze malwares and generate solutions for new types immediately by going through the code or studying their behavior, then they build mechanisms to stop and block these types of malware by generating machine understandable documents that can be transformed to commands that will update or modify the security systems like and IDPS, two major types of malware analysis are there:

6.1 Malware Analysis - Code Analysis:

This type analyze the code of the malware, since it looks inside the code it will find all the functions that the captured malware can do even if some of those functions only run in specific circumstances, the code analysis will not be based on

the source code even if it's the easiest way to analyze the codes, but since its hard to get the source code of the malware and then if we can get it, we will be not sure that the code is not changed, so we perform another type of code analysis which is static analysis, "the code analysis is normally performed at the machine code level. The sample is disassembled and then inspected manually (static analysis). "Sometimes even a decompilation is possible, resulting in a higher-level program that could be understood easier and analyzed faster" [2]. Depending on the resulted code we can know what the malware is designed to do or where it will send the information after compromising the machine since such information will be in the code like which server to contact and where to set it self to hide inside the system or if it will write on the registry and so on.

But there is a big problem with code analysis which is the encrypted binaries of the malware, the hackers recognized the code analysis danger so most of them are encrypting their binaries and using tools known as binary packers and crypters. Making the process of reverse engineering very complicated especially if it's done in an advanced way that may trick the dissassemblers so they will generate more complicated and wrong codes.

6.2 Malware Analysis - Behavior Analysis

"In contrast to code analysis, behavior analysis handles the malware binary as a black box and only analyzes its behavior, as can be seen from the outside during its execution (dynamic analysis)." [2]. So the behavior analysis is better than the code analysis, since it give what the malware exactly will do and what it will change in the

system during and after compromising it. There is two methods of behavior analysis, the first one is done by comparing the system state before and after the malware run, so it will show all the changes in the system that happen as a result of the malware run. The second method is to monitor the system while the malware is running, so we can recognize how the malware executes step by step and the changes it did on the system. One of the most powerful tools is CWSandbox and here we go with more details about it:

6.2.1 CWSandbox:

This tool is designed for Windows32 environment, which is one of the most vulnerable environments, it's an automated tool that analyzes and fixes Windows operating systems. The concept of its work is to provide the malware with the environment that it's expecting to let it execute, but this is not a real environment, it's a simulated "sandboxed" which means that the execution of the malware will not cause any harm to the other running and production environment. This tool is based on the behavior analysis so it will only analyze one way that this malware will execute at the same environment so if inside the malware code there is other ways to execute, they will not be detected. "Effectiveness is achieved by using the technique of API hooking. API hooking means that calls to the Windows application programmers' interface (API) are rerouted to the monitoring software before the actual API code is called" [2]. Still the API hocking can be bypassed by talking directly to the kernel so there is no role for the APIs there, but the good news is that few malwares are designed to talk to the kernel directly, to do so the malware has to be designed for a specific operating system with specific options and will not be able

to run over another operating system nor on same the same operating system with different updates, while the target of the malware builders is to reach as much as possible machines with the same malware.

7. Honeypot/net Detection

When we say hacker or any other kind of attacker who have the ability to pass through firewalls and other security methods then he/she is not an ordinary person that will be easily fooled by a honeypot/net. Savvy attackers will search and study the place they reach to make sure that it's what they want. So if it's low or medium level interaction honeypot/net then they will suddenly recognize the case, so they will accept the challenge and go aggrieve towards the real network, and maybe they will use the provided honeypot/net to go inside the real network. [4]

Most likely, the attacker will study the processes going on the system, when the last files were created, accessed or modified. Also will check the log files of the system so if its only built on and the admin set away waiting for attacks then the logs will be empty and maybe if the admin forgot to cleanup his/her activities while configuring the honeypot/net it will be a clear message for the attacker that this is not a real system. [4]

Another way to identify honeypot/net is by sniffing the traffic over the network, to see if there is a direct communication as expected in an ordinary environment or its nothing more than few dummy data generated by the one who created that honeypot. Also they can search for virtualization specifications to see if this entire network provided just as punch of virtual machines providing simulated services. But if the attacker identifies the used tools within the honeypot/net then he will try to compromise

the system through weaknesses or vulnerabilities in those tools themselves. Such as sending unexpected TCP packet with SYN and RST flags are set, the ordinary system supposed not to answer for such traffic but honeyd mostly will do. Also using simulation is detectable, like when using VMware, since there is a magic value for each instance so by identifying it we can say that the system is running VMware, another interesting thing, is it possible to have a server running with small RAM like 128 or 256 MB and with small amount of physical memory, its defiantly a virtual machine sampling a target to be attacked! [4]

Even if the tool is hidden well as in sebek case, if the attacker is smart enough he can find it by going to the hardware level and looking into hidden parts of the system this is the hard way, still there is some easier way which can be done by trying things that are unexpected to the tool, so it's response will be in a strange way, like the sebek case where the kernel rootkit detection tool (KProcCheck) is used to find that sebek is installed, "This detection tool is able to detect loaded modules by direct traversal of the PsLoadedModuleList. It is also able to detect Native APIs that are hooked by various modules." [17].

Other measures could be like the limited number of connections, the honeypot/net will block the connections when they exceed predefined number when configured. Another clear thing is the modification of the packets, since honeypots modify packets that seem to be harmful so the attacker can identify their existence by sending traffic that its answer is known for the attacker.

If honeypot is detected it will not only be useless but it also will be harmful to the network and it may be used to attack people outside which will put the organization in troubles with the low.

8. Anti honeypot technology

Not only the security guys are doing their job, the hackers also do, and as the game go on there is some kind of community that works in the background to find and develop ideas and tools to not only pass the honeypots/nets but also to destroy them, like Send-Safe Honeypot Hunter [16] tool which is used for such reasons and its becoming like a business since these tools are not for free and sometimes you have to by licenses to use them.

9. Legal Issues

Implementing honeypots may include some legal issues, this when an attacker comprises the honeypot and use it to attack people or organizations on the Internet, and may cause real damages or problems that will drive the victims to investigate to find the source of the attacks they suffered from, and they will find the that implemented honeypot/net is behind that harm, so victims will suppose that the owner of the honeypot/net is the attacker and its hard to consider that it's a testing environment used to track attackers to study their behaviors. And the problem become more complicated when the attacked organization is a competitor or in other country so the law is deferent which will lead for more troubles and misunderstanding. [3]

10. Conclusion

Security is like a race between hackers and security guys, so while using honeypots/nets and such tools we have to consider the efforts that hackers do to pass through by studying the current security tools and building more advanced ways to achieve their goals. So the security guys have to keep working to improve security tactics and build multi layer security policies and systems, so if the attacker pass one of the layers, we still can catch him and prevent him from accessing our resources.

As we see, there are many tools that are ready to use to let most of the computer users able to build their honeypots, for fun, research or security. These tools are different in the way they work and the advantages and disadvantages for each one of them. So we can't say that this tool is better than that, as every one of these tool is designed to do something different even if they are serving the same general idea. So before implementing your own honeypot/net you have to study your environment to know what exactly you have to do.

Lastly, it's a great idea to implement honeypot/net inside our environment to find our enemies and play with them so we can identify the weaknesses in our security system and study them to make the system more secure. But we have to secure ourselves from any potential harm may come from our honeypot/net, this can be done by implementing the honeypots/nets inside a DMZ that is disconnected from the production environment, and we should always consider the legal issues in case of compromising our honeypots/nets.

11. References

The honeynet project.

Books:

[1] L. Spitzner. *Honeypots: Tracking Hackers*.

[2] N. Provos, T. Holz. *Virtual Honeypots: From Botnet Tracking to Intrusion Detection.* Addison Wesley Professional.

[3] M. Warkentin, R. Vaughn. *Enterprise Information Systems Assurance and Systems Security: Managerial and Technical*.

Presentations:

[4] K. Piller, S. Wolfgarten. *Honeypot forensics*. (Berlin presentation, 28.12.2004)

Papers:

[5] L. Oudot. *Wireless Honeypot Countermeasures*.

[6] Know Your Enemy: Honeywall CDROM Roo

[7] Fighting Spammers With Honeypots: Part 1

[8] Felix C. Freiling and Thorsten Holz and Georg Wicherski. *Botnet Tracking.* RWTH Aachen University.

[9] R. Siles. *HoneySpot: The Wireless Honeypot.* The Spanish Honeynet Project. 2007.

[10] J. Oberheide and M. Karir. *Honeyd Detection via Packet Fragmentation.* Merit Network Inc.

[11] L. Spitzner. *The Value of Honeypots, Part Two: Honeypot Solutions and Legal Issues.*

[12] S. Krasser, J. Grizzard, H. Owen. *The Use of Honeynets to Increase Computer Network Security and User Awareness*. Georgia Institute of Technology.

Other websites:

[13] Fake AP, http://www.blackalchemy.to/project/fakeap/

[14] Honeystick, http://www.ukhoneynet.org/research/honeystick-howto/

[15] Honeyd, http://www.honeyd.org/

[16] Send-safe, http://www.send-safe.com/honeypot-hunter.html

[17] Sebek, http://www.security.org.sg/vuln/sebek215.html