25848
25847

Honeypots: Why We Need A Dynamics Honeypots?

Rahmat Budiarto, Azman Samsudin, Chuah Wee Heong, Salah Noori
School of Computer Sciences Universiti Sains Malaysia
11800 Penang, Malaysia

{rahmat,azman}@cs.usm.my, chuah78@yahoo.com, salah@nrg.cs.usm.my

Abstract

Honeypots has emerged to become a great tool for administrator to track down the Intruder, prevent attack by intruder and log all the activity done by the intruder. In this paper, we will look into different type of Honeypots such as KFSensor, Specter, Honeyd and Honeynets. Compare the strength and the weakness of each Honeypots. We will discuss on how to implement a simple Honeypots. This paper will also look into the Dynamic Honeypots and the possible solution for Dynamic Honeypots that need minimum configuration from network administrator.

1. Introduction

Due to emerge of the internet and everyone are link together by network. The security and privacy of each local network or each user become more and more important issue. Administrator may use different kind of security tools and hardware to protect his own network such as Firewalls, Sniffer and many others that available in the market.

One of the solutions for the network security could be Honeypots (see Figure 1). It is the concept that introduced by several people in computer security, especially Bill Cheswick's [1] paper "An Evening with Berferd" and Cliff Stoll [2] in the book "The Cuckoo's Egg". After that, Honeypots has continued to evolved and develop to a powerful network security tools as it is today.

2. Value of Honeypots

We have two general categories of honeypots; honeypots can either be used for production purposes or research.

When honeypots is used for production purposes, honeypots are protecting an organization. This would include preventing, detecting, or helping organizations respond to an attack. When honeypots is used for research purposes, honeypots are being used to collect information of the attack. This information has different value and it is vary for each organizations. Some organization might want to be studying trends or behavior in attacker activity, while the others are interested in early detection, warning and prediction, or law enforcement. Generally, high-interaction honeypots are often used for research purposes, while the low-interaction honeypots are used for production purposes. Anyhow, either type of honeypot can be deploys for either purpose.

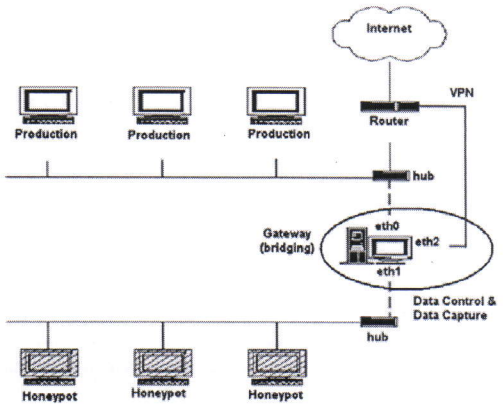

Figure 1: Example of Honeynet (2nd Generation) Architecture

When the Honeypots is used for production purposes, it can protect the organizations in one of three ways: Prevention, Detection, and Response to attack. Now we will look in more detail on how a honeypots can work in all three ways. Honeypots is capable to prevent attacks in many ways. The first way is against automated attacks, such as auto-rooters or worms. All these attacks are based on tools that is randomly scan entire networks and looking for vulnerable systems. If the vulnerable systems are found, these automated tools will then launch the attack and try to take over the system while the worms will self-replicating, copying themselves to the victim. The honeypots can help to defend against such attacks by slowing down their scanning down, potentially even stopping all these threat. It is called sticky honeypots, these solutions will monitor unused IP space. When it found a probed by such scanning activity, these honeypots will interact with attack and slow the attacker down. They do this using a variety of TCP tricks, such as a Windows size of zero, putting the attacker into a holding pattern. [3] This is excellent solution for slowing down or preventing the spread of a worm that has penetrated into the internal organization. One such example of a sticky honeypot is LaBrea Tarpit

[6]. Sticky honeypots are most often low-interaction solutions. It is a 'no-interaction solutions', as they slow the attacker down to a crawl. Honeypots can also protect your organization from human attackers by using the concept of deception or deterrence. The main idea is to confuse the attacker, and make him waste his time and resources interacting with the honeypots. Meanwhile, we can detect the attacker's activity and has the time to respond or stop the attacker's attack. It can be even more interesting when an attacker knows your organization is using honeypots, but does not know which systems are honeypots and which systems are legitimate computers. They may decide not to attack your system because afraid about being caught by honeypots. Thus the honeypot deters the attacker. The example of a honeypots that designed to do this is Deception Toolkit, a low-interaction honeypot.

Detection is the second way honeypots can help to protect an organization. Detection is critical; the honeypots purpose is to identify a failure or breakdown in prevention system. No matter of how secure an organization is, there will always be failures. There are always humans that involved in the process. By using an efficient detection tools to detect an attacker, we can quickly response to their attack, stopping or mitigating the damage they did to the organization. From the history shows, it is extremely difficult to do a good detection. Technologies such as Intrusion Detection System sensors and systems logs haven proved ineffective for some reasons. Intrusion Detection system often generates far too much data, a very large percentage of false alarms that will confuse the administrator, inability to detect new attacks, and it is unable to work in encrypted or IPv6 environments. Honeypots perform very well in detection purpose; it addresses many of these problems of traditional detections. Honeypots reduce false alarm by capturing small data sets with high value, it also able to capture unknown attacks such as new exploits tactics or polymorphic shellcode, and honeypot also can work in encrypted and IPv6 environments. Generally, low-interaction honeypots such as KFSensor make good solutions for detection. They are easier to deploy and maintain compare to high-interaction honeypots that need complex configuration and risk.

The final way a honeypot can help in protecting an organization is through response to attack. When an organization has detected a failure or a security breach, how do they respond? It is often be one of the greatest challenges faced by the organization. There is always lack of useful information such as who are the attacker, how did they get in, and how much damage they have done to the organization. In the situations like this, detailed information on the attacker's activity is very critical. There are two problems compounding to incidence response. First, often the affected systems compromised cannot be taken offline to analyze the root cause and the damage done by the attacker. Production systems, such as an organization's mail server, that it is so critical that even though it's been hacked, and security administrator may not be able to take the system down and do a proper forensic analysis. Instead, they are limited to analyze the live system while still providing production services. This will affect the ability to analyze what is happening, how much damage the attacker has done, and even if the attacker has broken into other related systems. The other problem is even if the system is pulled off-line, there is too much data. The security professional will have difficulty to determine what the bad guy did. If the data is polluted with others information such as user's logging in, mail accounts read, files written to databases and etc. It will be very difficult to determine what is normal day-to-day activity, and what is activity done by the attacker.

Honeypots could be very helpful in addressing both problems. Honeypots can be an excellent incident response tool, as they can quickly and easily taken offline for a full forensic analysis and further analysis, without impacting day-to-day business operations of the organization. Also, the honeypot only captures unauthorized or malicious activity done to it. These will makes hacked honeypots much more easy to analyze then hacked production systems, as any data we retrieve from a honeypot is most likely to be related to the attacker. The value honeypots provide here is quickly giving organizations the in-depth information of the attacker that they need to rapidly and effectively respond to the incident. Generally, high-interaction honeypots is the best solution for response to the attack. In order for us to respond to an intruder, we need in-depth knowledge on what the attacker did, how they broke in, and what are the tools they used. For this kind of information we most likely need the capabilities of a high-interaction honeypot such as HoneyNets.

3. Dynamic Honeynet

3.1. Comparison of Static Honeypot and Dynamic Honeypot

The main issue with all the network security technologies including Honeypot is to configure them. Anything in the network security technologies from a simple router to firewall rules require a human to analyze the environment and problem, and then come up with a solution to configure and implement the best solution. But, the work is not done there. Once we have implemented the technology, we have deployed, it need our daily attention and resources.

The current static Honeypots face the same issue, regardless of what type of honeypot you are building. From something as complex as Honeynets to something as simple as KFSensor, configuration is still required. With current honeypots, one of the configuration issues is where will the honeypot sit at? Which Operating

System will host the honeypot and what are the environments that we want the honeypot to emulate, will it be Solaris system, Linux, Windows, or maybe Cisco IOS? We might want to ensure we are emulating the same operating systems that are deployed within your organization, so the honeypots can easily blend into our environment. After we have determined the Operating System, there come some questions like what are the services do we want to run: is it web, email, or perhaps file sharing? Failing to run the correct services means missed probed or attacks. However, monitoring the wrong services can be just as harmful, as the wrong service can be a dead give away for a honeypot. If we are having a Linux honeypot emulate Sub7 or NetBus services, then it would look quite odd. Or we are having a Windows honeypot that emulate the sadmind, dtscpd, or rpc.statd service would also have a wired appearance about it. As such, we have to make sure the correct honeypots are running relevant services. Also, the issue becomes of where do we deploy our honeypot, and how many do we need?

As if configuring and deploying was not enough, someone has to maintain the honeypots once they go live and it can be more challenging then it sounds. It is not only is honeypot technologies that rapidly developing and changing that require updates for our honeypots but also the organization networks. New systems such as the latest Linux server are constantly being added into the network while old systems such as that Novell NetWare server still running IPX were removed or upgraded, The old system such Gopher or Telnet may be phased out while new applications like Instant Messaging, P2P, or live video was introduced. The organization and networks are constantly changing. To stay current, the honeypots have to adapt to the changes. Traditionally, this means someone (Honeypots administrator) need to manually updating or modifying the honeypots to better mirror the real production environment. That means it cost time, money, and sometime mistakes do happen.

That's why the idea of the Dynamic Honeypots comes out. It is just a plug-n-play solution. We can simply plug it into the network and the honeypot does all the work for us. Dynamics Honeypots will automatically determines how many honeypots need to deploy, how to deploy them, and what they should look like so that they can blend in with the organization environment. Even better, the deployed honeypots can change and adapt to the organization environment. When we add Linux to the network, we suddenly have Linux honeypots. If we remove a Novell server from organization network, the Novell honeypots automatically disappear. When we replace our Juniper routers with a Cisco IOS, and our honeypot routers will change. The ultimate goal is an appliance, a solution where we just simply plug into our network, it will learn the environment, deploys the proper number and configuration of honeypots, and most important is adapts to any changes in our networks. Sound like a great magic but the technology is there. What we need is just have to put all of them together.

3.2. Ways to make Dynamics Honeypot

The most critical part of a dynamic honeypot is how the Dynamic Honeypots learns about our network, what systems our organization using and how these systems are being used. With this knowledge, the dynamic honeypot can intelligently map and respond to our environment. One possible approach is to actively probe the organization network, determine what systems are live, types of systems they are, and what kind of services they are using. Nmap is one such scanning tool with that capable to do all mentioned activities. However, there are some drawbacks to such an active method. Firstly, it will introduce more activities to our networks. It not only affecting the network bandwidth or network activity, but with the scanning process it can cause services, even entire systems to shutdown. Other than that, it is possible to miss a system, as the system may be firewalled. The probes are sent to the system, but nothing is returned. The third drawback will be an active scanning takes only a snapshot of a point of time; it cannot give is the real-time changes update. We would constantly need to scan our environment to get the latest update of the system. That's why it's not a very elegant approach.

The idea is relatively simple, to map and identify systems on your network and it do not actively probe the systems, instead it passively capture network activity, analyze that activity, and then determine the system's identity. This technology uses the same approach as active scanning. Scanning Tools such as Nmap will build a database of known operating systems and services. These tools will then actively send packets to the target, packets which illicit a response. These responses which are unique to most operating systems and services are then compared with the database of known signatures to identify the operating system and services of the remote system.

Passive fingerprinting also takes the same approach; it has a database of known signatures for specific systems. However, the data is taken passively. Instead of actively probing the remote systems, the passive fingerprinting sniff traffic from the network and analyzes the packets from that network. Then, it will compare the packets against a database of signatures to identify the remote system and also the services. Passive fingerprinting is not just limited to TCP, it can also be used in other protocols. Passive technologies have several advantages. First it's not intrusive. It is passively gathering data rather than actively interacting with systems. This will reduce the network bandwidth and network traffic or damaging or taking down a system or service in the network. Second, although the systems are using host-

based firewalls, passive fingerprinting will still be able to identify the system, if nothing else then it will map a MAC address to an IP. Lastly, this method is continuous -- as organization networks changes, these changes can be captured in real time and this becomes critical for maintaining realistic honeypots over the long term. But we do have some disadvantage of passive mapping, it may not work well across routed networks; it's more effective on organization local LAN. In some cases, more then just one dynamic honeypot would have to be physically deployed in the organization, depending on the organization size, number of networks, and configuration.

The dynamic honeypot could leverage this concept of passive fingerprinting to learn our networks. The honeypot could be deployed as an appliance or single box. This device is then physically connected to your network. Once connected, it spends the some time watching and learning the organization network. By passively analyzing all of the traffic it sees, it will then determine how many systems are on your networks, what are the operating system types, the kind of the services they offer, and potentially even which systems are communicating with whom and how often is it. All these information is then used to learn and map the organization network. Once the honeypot learns the environment, it can begin deploying more honeypots. The strong point of the Dynamics Honeypots here is that the honeypots are crafted to mirror your environment. And by appear and behaving the same way as the organization production environment, the honeypots seamlessly blend in, making them much more difficult for attackers to identify as honeypots. Moreover, this passive learning does not stop there. It continuously monitors the organization network and whenever it found a change, this change is identified and the deployed honeypots will adapt to the changes. If the organization is a typical Windows environment, we may begin deploying some Linux servers. Then the dynamic honeypot, using passive fingerprinting, can determine that Linux systems have been deployed and the honeypot would then deploy Linux honeypots, or update existing honeypots, based on the same Linux makeup and using similar services. The dynamic honeypot vastly reduces not only the work involved in configuring your honeypots, but also maintains them in a constantly changing environment. All this will save up a lot of cost.

The next problem related to Dynamics Honeypots to be solved is how do the honeypots get deployed? As we discussed, passive fingerprinting offers a powerful tool, but how do we can actually get it to populate into the network with honeypots? Traditionally, this need physically deploying a new computer for each IP address we wanted to monitor. However, this has defeats the purpose of a dynamic honeypot if we still need a person to physically deploy multiple honeypots. What we need is a hands free, fire-and-forget solution. A far more simple and effective approach is not to deploy any physical honeypots. We try to deploy virtual honeypots in our network. All of these virtual honeypots are deployed and maintained by our appliance, our single physical device. The virtual honeypots monitor only unused IP space, so we can be highly confident that any activity to or from those IPs is most likely malicious or unauthorized behavior. Based on our previous discussion about the passive mapping of the network, we can determine how many honeypots we should deploy, what the types are, and where to deploy. Our honeypots now match the type of production systems in use and their services, but also the ratio of systems used. Other than that, the virtual honeypots can also monitor the same IP space as the systems themselves. Just for example, the honeypot learns that the Windows XP workstations are DHCP systems in the 192.168.1.100 - 192.168.1.250 range. Then the Windows XP honeypots would reside in the same IP space, while the other honeypots are monitored their respective IP space.

Once again, this ability to dynamically create and deploy virtual honeypots already exists. The Open-Source honeypot like Honeyd allows users to deploy virtual honeypots throughout an organization. In addition, this honeypot can also emulate over 500 operating systems, both at the IP stack and application level. As an Open-Source solution, it's highly customizable, allowing it to adapt to almost any environment. By combining the abilities of a solution like Honeyd, with the capabilities of a passive fingerprinting tool, we come very close to an ideal dynamic honeypot. We can have a hands free, fire-and-forget solution. We just deploy Honeyd honeypot by connecting it to your network. The passive fingerprinting tool will kicks in, passively monitoring and mapping the organization network. After sometime, it learns what systems our have, what services they are running, where do they located, and may be how they are being used. Based on this data, our honeypot will create virtual honeypots that mirror the makeup of the organization network, and subtly blend in with our production systems. Attackers can no longer tell which one is the honeypot and what is really part of your network. Once these honeypots are virtually deployed, it will continue to monitor your networks. The virtual honeypots adapt in real time to any additions, changes, or removal of existing systems. The security administrator just needs to sit back and catch the bad guys (attacker).

Dynamic honeypots can radically revolutionize the deployment and maintenance of honeypots. With the abilities of learning and monitoring the organization networks in real time, they become a fire-and-forget solution. Not only do they become cost-effective to deploy and maintain, but they have better integration into the organization network.

4. Conclusions and Future Works

Information becomes more and more important to every organization and the emerged to be a competitive advantage. And the information security also becomes the main priority for the security professional. There are many technologies that provide this kind of abilities such as Intrusion detection system, Firewall and other security measures. But all these tools often give us too many information that need us to dig the useful information from a few gigabytes data a day.

Honeypots come in to help us in three ways that is prevention, detection and how we react to an attack. There are two general types of honeypots which is Low interaction honeypots such as Honeyd, Specter and KFSensor. The highly interactive honeypots is like Honeynet. Honeypots basically is sit on an unused IP where any attempt connection to that IP will consider as an authorized and malicious attack. This will help to reduce the size of the information logged and the security professional can easy detect an intrusion and can response to it more effectively and fast.

Although the current technologies have the abilities to perform an ideal Dynamics Honeypots they really do not need any configuration from the system administrator. It will have the abilities of learning and monitoring the organization networks in real time and then deploy a virtual honeypots that will suite the organization environment.

The future task is to combine all the technologies to come out with an ideal Dynamics honeypot that will have its own intelligent and continuously change to mirror the actual production network. The future honeypots will become more cost effective to deploy and maintain and also better integration into the organization network.

5. References

[1] Bill Cheswick. An Evening with Berferd.In Which a Cracker is Lured, Endured, and Studied AT&T Laboratories.

[2] Cliff Stoll. The Cuckoo's Egg. Pocket Books. (1990)

[3] Lance Spitzner. Honeypots: Definitions and Values of Honeypots, http://www.tracking-hackers.com/papers/honeypots.html (2003)

[4] Lance Spitzner. Open Source Honeypots: Learning with Honeyd. http://www.securityfocus.com/infocus/1659 (2003)

[5]NetSec. Specter http://www.specter.com/default50.htm

[6] LaBrea. Tarpit http://labrea.sourceforge.net/README

[7] Honeynet Research Alliance http://www.honeynet.org/alliance/index.html (2003)

[8] M Lance Spitzner. Dynamic Honeypots. http://www.securityfocus.com/infocus/1731 (2003)

[9] Lance Spitzner. Knowing your enemy: Honeynets http://www.honeynet.org/know Know Your Enemy Honeynets.htm (2003).