ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 2, Issue 12, December 2013

HONEYPOT TECHNIQUE USED FOR INTRUSION DETECTION SYSTEM

Padmini Rathore [#1], Nitin Jain [##2]

[#1] *M.Tech Scholar, Dept. Of Electronics & Telecom.Engg.*
Chouksey Engineering College, Bilaspur
Chhattisgarh, - India

[2] *Assistant Professor. Dept. Of Electronics & Telecom.Engg.*
Chouksey Engineering College, Bilaspur
Chhattisgarh, - India

Abstract— **Deception systems using Honeypot presents a system that pretends to have alone or more network vulnerabilities that a blackhat or a hacker is looking for. But actually it does not have those vulnerabilities. It does so just to deceive the intruder. And it does so by stealthily monitoring the network, as an example assume an application which acts to serve on some well known port and logs all the attackers proceedings, this way Honeypot deceive the intruders at the same time manages to record some very valuable information. At present when there are many bad guys out there, these techniques not only protects the valuable network resources but also helps us to develop new tools to fight them based on their novice attack pattern which we record on Honeypot. Honeypots are digital network bait and use deception to attract intruders, thereby distracting them from real production systems.**

Index Terms— **Honeypot,VirtualMachines(VM),Intrusion Detection System(IDS),Hyper Text Trasport Protocol(HTTP).**

I. INTRODUCTION

Before discussing deception techniques in details lets first have a look at what Honeypot actually is:

1.1 Honeypot

The definition of a Honeypot as, defined by the Honeypot maillist, a public forum of over 5,000 security professionals, is [7]: "A Honeypot is an information system resource whose value lies in unauthorized or illicit use of that resource". A Honeypot is any system designed for the sole purpose of being exploited. This is a broad definition that can be implemented in many ways. Some Honeypot systems use software, some use actual production machines, and some even use virtual machines such as with VMware. Whichever Honeypot design method is chosen, the underlying goal is to create a system that appears to be vulnerable. What makes a Honeypot different from other vulnerable computer systems is its extensive logging capability. The systems most often include at least four layers of logging to capture attacker activity. Every file accessed, every connection made, every keystroke an attacker makes on a Honeypot is logged to a secure location. While this concept sounds very simple (and it is), it is this very simplicity that give Honeypots their tremendous advantages (and disadvantages).

1.1.1 Advantages and Disadvantages

Honeypots are a simple concept, with the following advantages [8].

I. Small data sets of high value: Honeypots collect small amounts of information. Instead of logging a one GB of data a day, they can log only one MB of data a day. Instead of generating 10,000 alerts a day, they can generate only 10 alerts a day. Remember, Honeypots only capture bad activity; any interaction with a Honeypot is most likely unauthorized or malicious activity. As such, Honeypots reduce 'noise' by collecting only small data sets, but information of high value, as it is only the bad guys. This means it's much easier (and cheaper) to analyze the data a Honeypot collects and derives value from it.

II. New tools and tactics: Honeypots are designed to capture anything thrown at them, including tools or tactics never seen before.

III. Minimal resources*:* Honeypots require minimal resources, they only capture bad activity. This means an old Pentium computer with 128MB of RAM can easily handle an entire class B network sitting off an OC-12 network.

IV. Encryption or IPv6: Unlike most security technologies (such as IDS systems) Honeypots work fine in encrypted or IPv6 environments. It does not matter what the bad guys throw at a Honeypot, the Honeypot will detect and capture it.

V. Information: Honeypots can collect in-depth information that few, if any other technologies can match.

VI. Simplicity: Finally, Honeypots are conceptually very simple. There are no fancy algorithms to develop, state tables to maintain, or signatures to update. The simpler a technology, the less likely there will be mistakes or misconfigurations.

Like any technology, Honeypots also have their weaknesses. It is because of this they do not replace any current technology, but work with existing technologies and have following disadvantages:

I. Limited view: Honeypots can only track and capture activity that directly interacts with them. Honeypots will not capture attacks against other systems, unless the attacker or threat interacts with the Honeypots also.

II. Risk: All security technologies have risk. Firewalls have risk of being penetrated, encryption has the risk of being broken, IDS sensors have the risk of failing to detect attacks. Honeypots are no different, they have risk also. Specifically, Honeypots have the risk of being taken over by the bad guy and being used to harm other systems. Depending on the type of Honeypot, it can have no more risk than an IDS sensor, while some Honeypots have a great deal of risk.

1.2 Types of Honeypot

Having defined a basic definition of Honeypot, it is time to see the various ways Honeypots are used commercially and academically. Although there are various configurations of Honeypots they can be broadly classified into two main types:

1. Production
2. Research

1.2.1 Production Honeypots

Production Honeypots are low–interaction Honeypots which have little or no interaction with the attacker or intruder in context. Also, they have less value to security of production resources. They try to create as less a realistic environment as possible i.e. when they are deployed they not necessarily emulate the whole system as a whole but try to emulate as much as possible within certain time and value. Once deployed they serve very little purpose, they capture data, In essence they just act as a basic event log, with a potential difference that they are not meant to be interacted with. For example, if you want to monitor web-based attacks, you just emulate a basic web server like Apache and listen to port 80(usually HTTP) connections. Once this is done, all the connections that scan the Honeypots for HTTP vulnerabilities will be logged.

Production Honeypots are made for mainly this reason; they capture data and send it to administrators. How they utilise this data and what precautions they take is left on to them. This has so many advantages as compared to competing technologies like Intrusion detection systems and firewalls. A Honeypot has no production value i.e. they do not act as servers and so are not meant to be interacted with. If any probe or access comes on it, it is most likely a malicious activity, unless there has been a misconfiguration by the administrator or someone has mistakenly accessed the wrong system. Nevertheless, *noise reduction* for malicious activity is the best advantage of these types of Honeypots. They indicate the administrator succinctly of what the attack or the probe is and how the intruder got access in. Thus, instead of browsing through 10,000 alerts from firewalls and IDS it makes the work of an administrator much easier. He can pinpoint the exact log of the attack without getting into the hassle of going through each and every record of the activity. Thus, the idea of less false positives giving efficient use of manpower gets practical here. Also, there occur no advanced algorithms or databases of attack signatures to be kept for validating packets entering your network. This is not just for detection. In fact, production Honeypots help widely in prevention. They can prevent worms from entering into the system, which work by scanning a complete network range

and targeting specific vulnerabilities. An example of this technique will be presented later in one of the deception techniques in later chapters. Also, they might slow down the inbound connections directed towards the network via them. An example of this sort of Honeypot is the LaBrea tarpit, which detects connection to non-existent IP addresses or overwhelming ARP requests to particular IP address (thus predicting it to be a DDoS attack). It then acknowledges the connection but keeps it incomplete and thus the intruder is 'stuck' by the Honeypot. However, bandwidth requirements can decide the time for which the connection can be left hanging. Also, production Honeypots often are used to deceive people as legitimate servers. An intruder might think he/she is interacting with the real system while they are just attacking a Honeypot. A recent example of this was cited in one of the large security firms – Internet Security Systems (ISS).

According to the firm, one of the web-server that suffered a breach and got defaced was just a Honeypot and was meant to get hacked. However, the article said the "X-force Internet Watch" was then properly monitored and the malware was removed.

But this all does not mean that production Honeypot are flawless. Firstly, they can only monitor anything that is coming their way. They cannot prevent anyone from opening confidential files from ports they are not listening, nor can they stop Trojan installs from ports they are not monitoring. Secondly, as stated earlier, it requires a high skill set for maintaining a Honeypot and it is a risky business if it is used as a "launch pad" for attacking other systems connected within the same network as the Honeypot.

1.2.2 Research Honeypots

Research Honeypots are more complex than production Honeypots and are kept in more secure environment since they do not comparatively have much valuable assets to protect in the backbone. However, they simulate the whole operating system and thus present the intruder with a known set of vulnerabilities within the system. For example, for web attacks a default installation of Linux 3.1 with Apache 1.1 can be installed and the results observed. Since, research Honeypots are a step ahead than production ones, they naturally get the backward compatibility and advantages. Thus, all the advantages of production Honeypots are present in research ones. Also, they are more stringent in their deployment and can serve response tasks like trace-back. Security firms might be interested in finding new attack tools and trends and thus keep their eye on research Honeypots. However, law enforcement agencies and government look more for early warnings and prediction from the analysis of research Honeypots. These are just a few examples to cite for the significance of research Honeypots to security circles. For example, a recent result from the Honeynet Project revealed a vast increase in organised credit card fraud. According to it a vast majority of stolen credit cards are used across relay channels thus increasing the illicit use of credit cards, performing identity thefts, compromising merchant sites and exchanging of these numbers. Also, commercially research Honeypots firms are now including the Microsoft vision – providing Honeypots as service rather than just providing support and installation at commercial sites. In a common configuration of Honeypots called bait-and-switch all traffic to the main server is routed to various Honeypots worldwide and they depict the main server completely. Once Honeypot-providing firms install the 'switches' or 're-routers' at the commercial site, they just can log all the activity passing through their Honeypots, depicting the main server. The customers wouldn't know the difference as they think they are interacting with the main server. So commercially as well Honeypot service could mount to a great profit margin in the near future. However, research Honeypots are still thought to be a subject of the nerds. But even the results displayed by these Honeypots have a substantial value and can be used for tightening the security of your network.

II. HONEYPOT CONCEPTS AND INPLEMENTATION

2.1 Honeypot Concept

We now take a look at the main concepts of Honeypots and a few different ways in which they can be implemented.

Honeypots are digital network bait and use deception to attract intruders, thereby distracting them from real production systems. A Honeypot with several layers can slow down an attack, increasing the possibility of the attack being detected, and the possibility of countering the intrusion before it succeeds. Intrusion detection and logging applications can be deployed within the Honeypot to listen for and log unauthorized activity. Since no interaction with a Honeypot is authorized, there is no need to filter through the information collected by a Honeypot for suspicious traffic. This information can then be used to learn how the intruders operate, and to come up with suitable countermeasures. In summary, the main concept of a Honeypot is to learn from the intruder's actions.

Additionally, Honeypots are not designed to be the sole source of security for any network; they should be used in conjunction with other security measures.

2.2 Approaches to Honeypot Implementation

To implement a Honeypot, some factors you need to consider include [5]:

I. Firstly, Kind of data that should be made available through the Honeypot, for the Honeypot to masquerade as an authentic system, realistic data needs to be used. However, there are also the consequences to consider when the Honeypot is compromised and the intruder uses the data against the organization. Measures need to be in place to handle such an occasion when it arises.

II. Secondly, how to prevent uplink liability**,** If a Honeypot is compromised, it could be used by the intruder to attack other systems (this is known as uplink liability). There are liability issues to consider if this happens, and preventative measures to take. Legal issues concerning Honeypots will be covered in more detail in the next section.

III. Building criteria**,** The Honeypot owner also has to decide between building a Honeypot and purchasing a commercially available one. Financial resources need to be considered. In addition, maintenance of the Honeypot requires knowledgeable personnel, as well as a considerable amount of time to examine the data collected by the Honeypot.

IV. Lastly, Location of Honeypots, Experts suggest isolating the Honeypot from your production system to prevent uplink liability. A lot more information on the considerations involved in Honeypot implementation can be found in.

III. COMMON DEPLOYMENT STRATEGIES

Many Honeypots do their job well, if they are deployed as single systems inside a network. Such deployments need a minimal administration effort. To build up an effective Honeypot environment, it is necessary to have a computer security policy. This policy defines what the threats are, where the important systems are located, what the main goals for attackers might be, how the systems will be protected and what to do when an incident occurs. This section contains some deployment strategies that can be used together with other security devices like firewalls and IDS to protect the network.

3.1 Minefield

If the Honeypots are installed among production servers, it is called a minefield deployment. The Honeypots are distributed across the whole production network.

To lure attackers, they often mirror some of the real server data.

Most attacks do not just target a single system. Especially worms and netbots scan whole networks for vulnerable machines. If the Honeypots are distributed among production servers, there is a high chance to detect all attacks. They provide information about the attacked service and the changes made to the compromised system.

This deployment strategy takes a big security effort. Compromised systems are often used for further attacks. It requires a great effort to secure every single Honeypot from attacking other targets.

3.2 Shield

Important servers need special security measures. In a shield deployment, each Honeypot is paired with a server it is protecting. While normal traffic to the server is not affected, a firewall/router redirects all suspicious traffic to the Honeypot.

Every connection to a mail server that does not target the specific SMTP port is suspicious and can be redirected reliably to the Honeypot. However the Honeypot is not able to protect this server from attacks to the SMTP port, because this kind of traffic will be classified as normal.

3.3 Honeynet

A Honeynet is a type of Honeypot as well as a deployment strategy. In a Honeynet, the Honeypots build an own network segment with fake services and traffic to lure attackers and to learn from them. The goal of this sort of deployment has rather the function of detection than prevention or protection

3.3.1 Honeynet Architecture

As we stated earlier, Honeynets are nothing more than an architecture. To successfully deploy a Honeynet, you must correctly deploy the Honeynet architecture. The key to the Honeynet architecture is what we call a honeywall. This is a gateway device that separates your Honeypots from the rest of the world. Any traffic going to or from the Honeypots must go through the honeywall. This gateway is traditionally a layer 2 bridging device, meaning the device should be invisible to anyone interacting with the Honeypots. Below we see a diagram of this architecture. Our honeywall has 3 interfaces. The first 2 interfaces (eth0 and eth1) are what separate our Honeypots from everything else; these are bridged interfaces that have no IP stack. The 3rd interface (eth2, which is optional) has an IP stack allowing for remote administration.There are several key requirements that a honeywall must implement; Data Control, Data Capture, Data Analysis, Data Collection. Data Control defines how activity is contained with the Honeynet without an attacker knowing it. Its purpose is to minimize risk. Data Capture is capturing all of the attacker's activity without the attacker knowing it. Data Analysis is the ability to analyze this data. Data Collection is the ability to collect data from multiple Honeynets to a single source. Of all these requirements, Data Control is the more important. Data Control always takes priority as its role is to mitigate risk. Honeynets are a form of a high-interaction Honeypot. Their primary advantage is their ability to gather extensive information. A Honeynet is an architecture, similar to a fishbowl. Within this architecture you can deploy any type of system or application you desire. The critical requirements for this architecture are Data Control, Data Capture, Data Analysis and Data Collection, with Data Control taking the priority. While very powerful, Honeynets present unique risks.

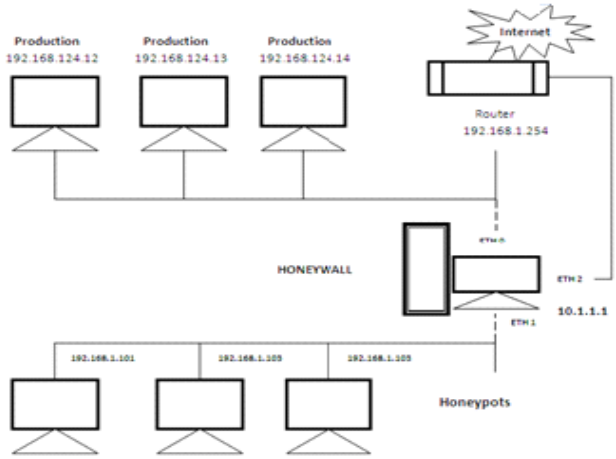

Figure 1 Honeynet Architecture

Mechanisms can be put in place to mitigate these risks, but there is no way to eliminate all risk [10].

3.3.2 Risk and Issues

I. Harm is when a Honeynet is used to attack or or harm other, non-Honeynet systems. For example, an attacker may break into a Honeynet, then launch an outbound attack never seen before, successfully harming or compromising its intended victim. Data Control is the primary means of mitigating this risk. Multiple layers of Data Control are put in place to make it more difficult for the attacker to cause damage. However, there is no guaranteed method to ensure that a Honeynet can not used to attack or harm someone else. No matter what mechanisms are put in place, an attacker can eventually bypass them. Your organization will have to decide how much risk it is willing to assume. For low risk organizations, you may want to minimize the activity allowed outbound (to zero, perhaps.) For organizations with greater risk thresholds, you may decide to allow greater outbound activity.

II. There is the risk of detection. Once the true identity of a Honeynet has been identified, its value is dramatically reduced. Attackers can ignore or bypass the Honeynet, eliminating its capability for capturing information. Perhaps even more dangerous is the threat that once identified, an attacker can introduce false or bogus information into a

Honeynet, misleading your data analysis. For example, with local access to the Honeynet, an advanced attacker, or an attacker armed with proper tools, can potentially identify that a Honeynet is in place and may even identify the Honeynet Data Control and/or Data Capture mechanisms themselves.

III. There is the risk of disabling Honeynet functionality. This could be an attack against either Data Control or Data Capture routines. Attackers may want to not only detect a Honeynet's identity, but disable its Data Control or Data Capture capabilities, potentially without the Honeynet administrator knowing that functionality has been disabled. For example, an attacker may gain access to a Honeypot within the Honeynet, then disable Data Capture functionality on the Honeypot. The attacker could then feed the Honeypot with bogus activity, making administrators think Data Capture is still functioning and recording activity, when it's not. Having multiple layers of Data Control and Data Capture helps mitigate this risk as there is no single point of failure.

IV. Violation is the catch all of remaining risk. Attackers may attempt criminal activity from your compromised Honeynet without actually attacking anyone outside your Honeynet. One example is an attacker using a Honeypot to upload, then distribute contraband or illegal material, such as illegal copies of movies, music, stolen credit cards, or child pornography. Remember, this individual broke into your system on their own initiative. You are not dealing with the most law abiding of cyber citizens. If detected, this illegal activity would be attributed (at least initially) to you by way of it being on your system. You may then have to prove to that it was in fact not you who was responsible for this activity [10]

IV.DECEPTION TECHNIQUES

4.1 Simple Port Listener

This technique is basically used for low interaction Honeypots. By listening to some port and emulating some application on that port can be used to fool attackers. Deception Services are specially designed to listen on an IP service port and respond to network requests. They can be used to emulate services like for example Sendmail. When an attacker connects to the Honeypot, he or she may receive a banner that identifies the service as being some version of Sendmail. If the attacker is fooled by the deception, he or she may try to gain access to the system. Such actions will provide the administrator the logs of attackers' movement.

In this low interaction configuration, the Honeypot is just being set up to listen to various port activities and raise appropriate alerts once certain threshold is exceeded. For example, if from an open port an attacker is able to gain access to the root shell a suitable alarm/email is delivered to the system administrator. This seems similar a lot to various logging capabilities built-in in many operating systems, but this basic functionality adds to the fact that Honeypots are passive devices and are not supposed to be interacted with. If however, there is access to it, as their definition suggests they have to blow their bells and whistles. Also, another thing that has to be kept in mind is the logs regarding these port activities have to be preserved in systems other than Honeypot because most of the attacks are directed to gaining root access and once that is done an apt attacker can easily erase his records. Thus, the overall idea can be described just in three words listen, log and alert.

4.2 Honeypot Farms

There is one possible solution to simplifying large Honeypot deployments, Honeypot farming. The concept of farming is simple. Instead of deploying large numbers of Honeypots, or Honeypots on every network, you simply deploy your Honeypots in a single, consolidated location. This single network of Honeypots becomes your Honeypot farm, a dedicated security resource. Attackers are then redirected to the farm, regardless of what network they are on or probing. Remember, Honeypots don't passively capture traffic from your network, so they don't have to be physically connected to the network. Honeypots only have to be virtually on your network. Honeypots farms do this by deploying redirectors. A redirector acts as a proxy or 'worm hole', it transports an attacker's probes to a Honeypot within the Honeypot farm, without the attacker ever knowing it. The attacker thinks they are interacting with a victim on a local network, when in reality they have been transported to your Honeypot farm.

Figure 2 below demonstrates the concept of redirecting attackers to Honeypot Farms. The potential advantages are enormous. Deploying Honeypots becomes an extremely simple affair. Instead of having individuals all over the world build, customize, deploy, and maintain a separate Honeypot for your every network (keeping in mind many organizations have thousands of networks), you build and deploy a single, centralized Honeypot farm. This Honeypot farm could become part of your SOC (Security Operations Center) where you have the manpower and resources already dedicated to build such a solution. You then have the trained professionals on site that can deploy and maintain the farm. Instead of having Honeypots distributed all over the world, you have them all in one place.

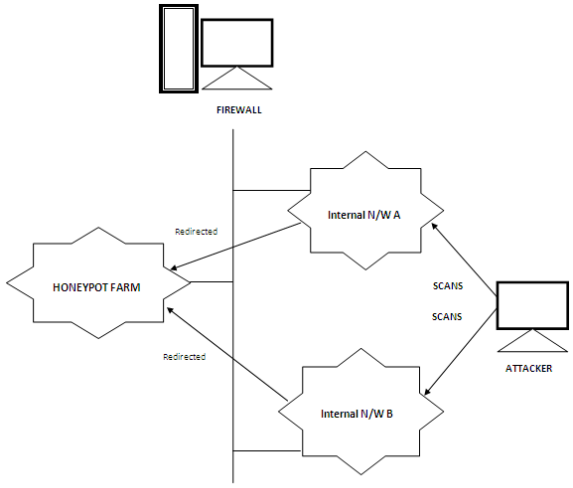

Figure 2 Honeypot Farm

The farm can be as simple as several low-interaction Honeypots, to something as involved as a large Honeynet made up of hundreds of real systems and applications waiting to be attacked. This centralization then makes maintaining, standardizing, and analyzing the Honeypots far simpler. To 'virtually' deploy your Honeypots on other networks, you simply mail a redirector to each network administrator you want to have your Honeypots on. These redirectors, once physically placed on a network, will then redirect all attackers or unauthorized activity to the centralized Honeypot farm, where SOC personnel will monitor and analyze all of the captured information in real time. To deploy the redirectors, you simply ship one to each network admin for each network you want monitored. These redirectors can be nothing more than a black box, one which the network administrators simply plug into their local networks. No configuration is required, as all configurations were done by the SOC personnel. That black box can then monitor predetermined IP's (or dynamically monitor unused IP's). When attackers interact with those IP's, or interact with systems in a malicious or unauthorized manner, the black-box magically transports the attacker to the Honeypot farm. This simplifies deploying distributed Honeypots to nothing more than FedEx'ing a box to your network administrators [11].

Updating and administering your Honeypots, especially high-interaction Honeypots, also far becomes easier. Instead of reaching out to each remote system, security administrators merely update the Honeypots in a single location. Instead of having to maintain multiple Honeynets distributed around the world, they have only one physical Honeynet to maintain, saving them a great deal of time. This time can then be used to develop more advanced Honeypots that mirror your environment. For example, instead of having a Honeypot emulate a Solaris box with a SQL emulator, you now have the time and resources to deploy a real Solaris Honeypot running Oracle. You can then even populate the database with bogus information, to see how attackers interact with the database, and what data they attempt to recover. Honeypots farms can exponentially increase the effectiveness of Honeypots, especially high-interaction solutions such as Honeynets.

A third advantage becomes one of mitigating risk. High-interaction Honeypots have a great deal of inherent risk in them. The risk being that once an attacker takes over one of the Honeypots, they can then use that Honeypot to attack other non-Honeypot systems. This risk is compounded when multiple high-interaction Honeypots are deployed. Honeypot farms can help address this. By consolidating all of your high-interaction Honeypots to a single farm, you mitigate risk by having all of your Honeypots in a single location. This means you require only a single place for data control, where you contain the attacker's inbound and outbound activity. Your security team can then monitor this single network, as

opposed to having to monitor and administer multiple Honeypots. This also allows them to ensure they have the most current technologies and solutions in place.

4.3 Random Servers

This deployment comes under the notion of security through obscurity. However, there is nothing confidential, but the network itself is just so probabilistic that you never know which node you are touching. The basic principle is to simulate the main servers within different Honeypots and make a demilitarised zone of them. Whenever the request for particular server(s) comes in, the request is assigned a server based on some predefined rule set or mathematical function. It is left only to probability that you get either the real server or a 'Honeypotted' server. Although, this might prove to be less legal prone deployment, subtle attacks may be missed and the server still might get compromised. However, the success rate is at least better than having just main servers placed on the network. Also, the centralised management is another ease in this deployment.

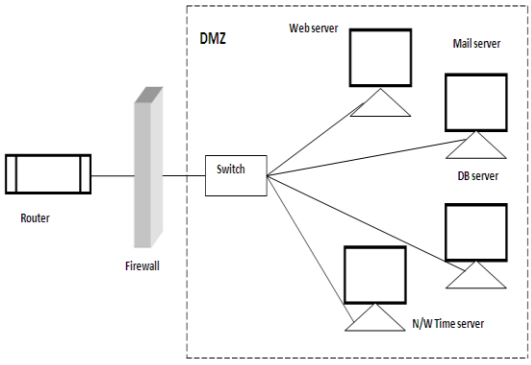

Figure 3 Random Servers as Honeypot

4.4 Honeytokens

Simply put, a Honeytoken is a fake digital entity that can have many different applications. Although the term "Honeytoken" was coined in 2003 by Augusto Paes de Barros , the concept of Honeytokens is not new. For years dictionaries, encyclopaedias, maps and directories have used fake entries or deliberately erroneous entries as copyright traps to facilitate detection of copyright infringement or plagiarism. In computer security, Spitzner defines a Honeytoken as a Honeypot that is not a computer, but a digital entity. A Honeytoken can exist in many forms such as a credit card number, an Excel spreadsheet, a PowerPoint presentation, a database entry, or even a fake login. Like other types of Honeypots, no Honeytoken has any authorized use. This gives Honeytokens the same power and advantages as traditional Honeypots, but extends their capabilities beyond physical computers.

The basic idea behind this is that whenever an attacker accesses data from a system or network, he would either just access (read/write/delete) it or steal it. To prevent this, bogus data is embedded within actual data and when this bogus data is accessed alerts are raised. Still further, if the data is stolen then the embedded 'bogus data' can call back the host system giving its location, however on suitable activation only. The just-access concept comes from a term coined, by Augusto Paes de Barros, 'Honeytokens' in one of the mailing lists. Honeytokens are just bogus data, accounts, database entries, SSN or NI numbers which have no value in real world. However, they are embedded within real data for deceiving users. If they are accessed, alarms are raised to the system administrators cautioning them of malicious activity. According to a recent article by Lance Spitzner he called it the other Honeypot. There is no infrastructure needed, no signatures to update, no constant monitoring required nothing at all. Besides cost, they gain all the advantages of Honeypots as they themselves are a part of Honeypots. Just like traditional Honeypots, Honeytokens do not solve a specific problem. They are not designed specifically to detect blackhats or prevent attacks. Instead, they are a highly flexible and simple tool with multiple applications to security, everything from detection to identifying who your threat is and their motives. Honeytoken's value lies in their simplicity. You deploy a digital file or record and if anyone accesses them, you potentially have a problem. You leverage the fact that the insider attacker knows your internal environment and has access to files, information and records (including our Honeytokens) that untrusted outsiders would not have access to. Their real value is when they are combined with other solutions. For example, in many cases Honeytokens may not prove unauthorized activity. Instead, they may merely indicate you have unauthorized behaviour.

You most likely will have to use other tools to confirm if someone is acting with malicious intent. An employee may access a Honeytoken that is a Microsoft Word file posing as your company's Research and Development plans. If an employee attempts to copy and transfer the file, you have identified a problem. Either you have a confused employee looking where they shouldn't, or you have someone who is knowingly attempting to access high-value documents they do not have authorization for. Regardless, once you have identified this activity, you can then use other measures to confirm the individual's intent.

V. CONCLUSIONS

This paper shows how Honeypots deception techniques can be used as a trustworthy means to learn about strategy and various tools used by attackers. A Honeypot is a valuable resource, especially to collect information about proceedings of attackers. As Honeypot has its advantages and disadvantages so knowing the limits of the Honeypot it should not be thought as a network security panacea. They are one facet of Network Security along with other Network Security tools such as firewall, IDS etc. Also Chapter 3 of this report talks about the various deployment strategy of Honeypots, we must be careful about choice we make for the deployment as it affects the effectiveness of Honeypot.

REFERENCES

[1] Dacier Marc, Pouget Fabien and Debar EurecomHervé, "Honeypots: Practical Means to Validate Malicious Fault Assumptions", In the proceedings of the 10th IEEE Pacific Rim International Symposium on Dependable Computing (PRDC'2004), February 2004

[2] Gupta Nirbhay, "Improving the Effectiveness of Deceptive Honeynets through an Empirical Learning Approach", In the proceeding of Australian Information Warfare and Security Conference, Perth Western Australia 2002

[3] Spitzner Lance,"Know Your Enemy: The Tools and Methodologies of the Script Kiddie", Honeynet Project, http://project.Honeynet.org , February 2000

[4] McGrew Robert and B. Vaughn Rayford, "Experiences With Honeypot Systems: Development, Deployment, and Analysis", In the proceedings of the 39th Hawaii International Conference on System Sciences, On page(s): 220a- 220a, January 2006

[5] Mokube Iyatiti and Adams Michele, "Honeypots: concepts, approaches, and challenges", In the proceedings of the 45th annual southeast regional conference (ACM-SE), New York, USA, On Pages(s): 321 – 326, 2007

[6] Sink Michael, "The Use of Honeypots and Packet Sniffers for Intrusion Detection", SANS Institute , As part of GIAC practical repository 2000 - 2002

[7] Spitzner Lance, "Honeypots: Catching the Insider Threat", In the proceeding of 19th Annual Computer Security Applications Conference (ACSAC), On page(s): 170- 179, December 2003

[8] Spitzner Lance, "Honeypots, definitions and value of Honeypots" http://www.spitzner.net/Honeypots.html, May 2003.

[10] Spitzner Lance, "Know Your Enemy: Honeynets What a Honeynet is, its value, overview of how it works, and risk/issues involved'', Honeynet Project http://www.Honeynet.org, May, 2006

[11] Spitzner Lance, "Honeypot Farms", http://www.securityfocus.com/, August, 2008

[12] http://www.Honeypots.net/Honeypots/links

[13] http://www.project.Honeynet.org/

[14] Kreibich Christian and Crowcroft Jon, "Honeycomb – Creating Intrusion Detection Signatures Using Honeypots", In the proceedings of the 2nd Workshop on Hot Topics in Networks (Hotnets II), Boston, November, 2003