4 The Value of Honeypots

Now that we have defined honeypots and how they work, we can attempt to establish their value. As mentioned earlier, unlike mechanisms such as firewalls and intrusion detection systems, a honeypot does not address a specific problem. Instead, it is a tool that contributes to your overall security architecture. The value of honeypots, and the problems they help solve depend on how you build, deploy, and use them.

Honeypots have certain advantages and disadvantages that affect their value. In this chapter we will examine those advantages and disadvantages more closely. We will also look at the differences between production and research honeypots and their respective roles.

Advantages of Honeypots

Honeypots have several advantages unique to the technology. We will review four of them here.

Data Value

One of the challenges the security community faces is gaining value from data. Organizations collect vast amounts of data every day, including firewall logs, system logs, and Intrusion Detection alerts. The sheer amount of information can be overwhelming, making it extremely difficult to derive any value from the data. Honeypots, on the other hand, collect very little data, but what they do collect is normally of high value. The honeypot concept of no expected production activity dramatically reduces the noise level. Instead of logging gigabytes of data every day, most honeypots collect several megabytes of data per day, if even that much. Any data that is logged is most likely a scan, probe, or attack—information of high value.

Honeypots can give you the precise information you need in a quick and easy-to-understand format. This makes analysis much easier and reaction time much quicker. For example, the Honeynet Project, a group researching honeypots, collects on average less then 1MB of data per day. Even though this is a very small amount of data, it contains primarily malicious activity. This data can then be used for statistical modeling, trend analysis, detecting attacks, or even researching attackers. This is similar to a microscope effect. Whatever data you capture is placed under a microscope for detailed scrutiny.

For example, in Figure 4-1 we see a scan attempt made against a network of honeypots. Since honeypots have no production value, any connection made to a honeypot is most likely a probe or attack. Also, since such little information is collected, it is very easy to collate and identify trends that most organizations would miss. In this figure we see a variety of UDP connections made from several systems in Germany. At first glance, these connections do not look related, since different source IP addresses, source ports, and destination ports are used. However, a closer look reveals that each honeypot was targeted only once by these different systems. Analysis reveals that an attacker is doing a covert network sweep.

```
Date      Time      Src-IP          Dst-IP      Proto  Src-P  Dst-P
02/02/15  23:50:15  213.68.213.135  10.1.1.101  udp    5298   18030
02/02/15  23:50:15  213.68.213.134  10.1.1.109  udp    18986  10903
02/02/15  23:50:15  213.68.213.133  10.1.1.108  udp    16932  16219
02/02/15  23:50:16  213.68.213.140  10.1.1.107  udp    1348   5274
02/02/15  23:50:16  213.68.213.130  10.1.1.105  udp    19841  15316
02/02/15  23:50:17  213.68.213.144  10.1.1.104  udp    17773  3327
```

Figure 4-1 Covert network sweep by an attacker picked up by a network of honeypots

He is attempting to determine what systems are reachable on the Internet by sending UDP packets to high ports, similar to how traceroute works on Unix. Most systems have no port listening on these high UDP ports, so when a packet is sent, the target systems send an ICMP port unreachable error message. These error messages tell the attacker that the system is up and reachable.

The attacker makes this network sweep difficult to detect because he randomizes the source port and uses multiple source IP addresses. In reality, he is most likely using a single computer for the scan but has aliased multiple IP addresses on the system or is sniffing the network for return packets to the different systems. Organizations that collect large amounts of data would most likely miss this sweep, since multiple-source IP addresses and source ports make it hard to detect. However, because honeypots collect small amounts of, but high-value data, attacks like these are extremely easy to identify. This demonstrates one of the most critical advantages of honeypots.

Resources

Another challenge most security mechanisms face is resource limitations, or even resource exhaustion. Resource exhaustion is when a security resource can no longer continue to function because its resources are overwhelmed. For example, a firewall may fail because its connections table is full, it has run out of resources, or it can no longer monitor connections. This forces the firewall to block all connections instead of just blocking unauthorized activity. An Intrusion Detection System may have too much network activity to monitor, perhaps hundreds of megabytes of data per second. When this happens, the IDS sensor's buffers become full, and it begins dropping packets. Its resources have been exhausted, and it can no longer effectively monitor network activity, potentially missing attacks. Another example is centralized log servers. They may not be able to collect all the events from remote systems, potentially dropping and failing to log critical events.

Because they capture and monitor little activity, honeypots typically do not have problems of resource exhaustion. As a point of contrast, most IDS sensors have difficulty monitoring networks that have gigabits speed. The speed and volume of the traffic are simply too great for the sensor to analyze every packet. As a result, traffic is dropped and potential attacks are missed. A honeypot deployed on the same network does not share this problem. The honeypot only captures activities directed at itself, so the system is not overwhelmed by the traffic. Where the IDS sensor may fail because of resource exhaustion, the honeypot is not likely to have a problem. A side benefit of the limited resource requirements of a honepyot is that you do not have to invest a great deal of money in hardware for a honeypot. Honeypots, in contrast to many security mechanisms such as firewalls or IDS sensors, do not require the latest cutting-edge technology, vast amounts of RAM or chip speed, or large disk drives. You can use leftover computers found in your organization or that old laptop your boss no longer wants. This means that not only can a honeypot be deployed on your gigabit network but it can be a relatively cheap computer.

Simplicity

I consider simplicity the biggest single advantage of honeypots. There are no fancy algorithms to develop, no signature databases to maintain, no rulebases to misconfigure. You just take the honeypot, drop it somewhere in your organization, and sit back and wait. While some honeypots, especially research honeypots, can be more complex, they all operate on the same simple premise: If somebody or someone connects to the honeypot, check it out. As experienced security professionals will tell you, the simpler the concept, the more reliable it is. With complexity come misconfigurations, breakdowns, and failures.

Return on Investment

When firewalls successfully keep attackers out, they become victims of their own success. Management may begin to question the return on their investment, as they perceive there is no longer a threat: “We invested in and deployed a firewall three years ago, and we were never attacked. So why do we need a firewall if we have never been hacked?” The reason they were never hacked is the firewall helped reduce the risk. Investments in other security technologies, such as strong authentication, encryption, and host-based armoring, face the same problem. These are expensive investments, costing organizations time, money, and resources, but they can become victims of their own success.

In contrast, honeypots quickly and repeatedly demonstrate their value. Whenever they are attacked, people know the bad guys are out there. By capturing unauthorized activity, honeypots can be used to justify not only their own value but investments in other security resources as well. When management perceives there are no threats, honeypots can effectively prove that a great deal of risk does exist.

For example, once I was in Southeast Asia conducting a security assessment for a large financial organization. I was asked to do a presentation to the Board of Directors on the state of their security. As always, I had a honeypot running on my laptop. About 30 minutes before the presentation, I connected to their network to make some last-minute changes. Sure enough, while I was connected to their network, my system was probed and attacked. Fortunately, the honeypot captured the entire attempt. In this case the attack was a Back Orifice scan. When the attacker found my system, they thought it was infected and executed a variety of attacks, including attempting to steal my password and reboot the system. I then went with this captured attack and used it to open my presentation to the Board. This attack demonstrated to the Board members that not only did active threats exist but they tried, and succeeded, in penetrating their network. It is one thing to talk about such threats, but demonstrating them, keystroke by keystroke, is far more effective. This proved extremely valuable in getting the Board's attention.

Disadvantages of Honeypots

With all of these wonderful advantages, you would think honeypots would be the ultimate security solution. Unfortunately, that is not the case. They have several disadvantages. It is because of these disadvantages that honeypots do not replace any security mechanisms; they only work with and enhance your overall security architecture.

Narrow Field of View

The greatest disadvantage of honeypots is they have a narrow field of view: They only see what activity is directed against them. If an attacker breaks into your network and attacks a variety of systems, your honeypot will be blissfully unaware of the activity unless it is attacked directly. If the attacker has identified your honeypot for what it is, she can now avoid that system and infiltrate your organization, with the honeypot never knowing she got in. As noted earlier, honeypots have a microscope effect on the value of the data you collect, enabling you to focus closely on data of known value. However, like a microscope, the honeypot's very limited field of view can exclude events happening all around it.

Fingerprinting

Another disadvantage of honeypots, especially many commercial versions, is fingerprinting. Fingerprinting is when an attacker can identify the true identity of a honeypot because it has certain expected characteristics or behaviors. For example, a honeypot may emulate a Web server. Whenever an attacker connects to this specific type of honeypot, the Web server responds by sending a common error message using standard HTML. This is the exact response we would expect for any Web server. However, the honeypot has a mistake in it and misspells one of the HTML commands, such as spelling the word length as *legnht*. This misspelling now becomes a fingerprint for the honeypot, since any attacker can quickly identify it because of this error in the Web server emulation. An incorrectly implemented honeypot can also identify itself. For example, a honeypot may be designed to emulate an NT IIS Web server, but the honeypot also has certain characteristics that identify it as a Unix Solaris server. These contradictary identities can act as a signature for a honeypot. There are a variety of other methods to fingerprint a honeypot that we discuss later in the book.

If a blackhat identifies an organization using a honeypot on its internal networks, he could spoof the identity of other production systems and attack the honeypot. The honeypot would detect these spoofed attacks, and falsely alert administrators that a production system was attacking it, sending the organization on a wild goose chase. Meanwhile, in the midst of all the confusion, an attacker could focus on real attacks.

Fingerprinting is an even greater risk for research honeypots. A system designed to gain intelligence can be devastated if detected. An attacker can feed bad information to a research honeypot as opposed to avoiding detection. This bad information would then lead the security community to make incorrect conclusions about the blackhat community.

This is not to say all honeypots must avoid detection. Some organizations might want to scare away or confuse attackers. Once a honeypot is attacked, it can identify itself and then warn off the attacker in hopes of scaring him off. However, in most situations organizations do not want honeypots to be detected.

Risk

The third disadvantage of honeypots is risk. They can introduce risk to your environment. By risk, we mean that a honeypot, once attacked, can be used to attack, infiltrate, or harm other systems or organizations. As we discuss later, different honeypots have different levels of risk. Some introduce very little risk, while others give the attacker entire platforms from which to launch new attacks. The simpler the honeypot, the less the risk. For example, a honeypot that merely emulates a few services is difficult to compromise and use to attack other systems. In contrast, a honeypot that creates a jail gives an attacker an actual operating system with which to interact. An attacker might be able to break out of such a cage and then use the honeypot to launch passive or active attacks against other systems or organizations. Risk is variable, depending on how one builds and deploys the honeypot.

Because of their disadvantages, honeypots cannot replace other security mechanisms such as firewalls and intrusion detection systems. Rather, they add value by working with existing security mechanisms. They play a part in your overall defenses.

The Role of Honeypots in Overall Security

Now that we have reviewed the advantages and disadvantages of honeypots, let's apply them to security. Specifically, how do honeypots add value to security and reduce your organization's overall risk? As we discussed earlier, there are two categories of honeypots: production and research. We will review how honeypots add value in relation to these two categories.

Production Honeypots

Production honeypots are systems that help mitigate risk in your organization or environment. They provide specific value to securing your systems and networks. Earlier we compared these honeypots to law enforcement: Their job is to take care of the bad guys. How do they accomplish this? To answer that question, we are going to break down security into three categories and then review how honeypots can or cannot add value to each one of them. The three categories we will use are those defined by Bruce Schneier in *Secrets and Lies* [1]. Specifically, Schneier breaks security into prevention, detection, and response. Although more complex and extensive models for security exist, I find them confusing and difficult to apply. As such, we will stick with Schneier's simple and useful prevention, detection and response model.

Prevention

In terms of security, *prevention* means keeping the bad guys out. If you were to secure your house, prevention would be similar to placing deadbolt locks on your doors, locking your windows, and perhaps installing a chainlink fence around your yard. You are doing everything possible to keep out the threat. The security community uses a variety of tools to prevent unauthorized activity. Examples include firewalls that control what traffic can enter or leave a network or authentication, such as strong passwords, digital certificates, or two-factor authentication that requires individuals or resources to properly identify themselves. Based on this authentication, you can determine who is authorized to access resources. Mechanisms such as encryption prevent attackers from reading or accessing critical information, such as passwords or confidential documents.

What role do honeypots play here? How do honeypots keep out the bad guys? I feel honeypots add little value to prevention, since they do not deter the enemy. In fact, if incorrectly implemented, a honeypot may introduce risk, providing an attacker a window into an organization. What will keep the bad guys out is best practices, such as disabling unneeded or insecure services, patching vulnerable services or operating systems, and using strong authentication mechanisms.

Some individuals have discussed the value of deception or deterrence as a method to prevent attackers. The deception concept is to have attackers waste time and resources attacking honeypots, as opposed to attacking production systems. The deterrence concept is that if attackers know there are honeypots in an organization, they may be scared off. Perhaps they do not want to be detected or they do not want to waste their time or resources attacking the honeypots. Both concepts are psychological weapons, used to mess with and confuse a human attacker.

While deception and deterrence may prevent attacks on production systems, I feel most organizations are much better off spending their limited time and resources on securing their systems. What good is deploying a honeypot to deceive an attacker if your production systems are still running vulnerable services, applications need to be patched, and personnel are using passwords that are easy to guess? Deception and deterrence may contribute to prevention, but you will most likely get greater value putting the same time and effort into security best practices. It's not nearly as exciting or glamorous, but it works.

Deception and deterrence also fail to prevent the most common of attacks: targets of opportunity. As we discussed in Chapter 2, most attackers are focused on attacking as many systems as possible—the easy kill. They do this by using scripted or automated tools that hack into systems for them. These attackers do not spend time analyzing the systems they target. They merely take a shotgun approach, hitting as many computers as possible and seeing what they get into. For deception or deterrence to work, the attacker must take the time to input the bad information that honeypots are feeding them. Most attackers today do not bother to analyze their targets. They merely strike at a system and then move onto the next. Deception and deterrence are designed as psychological weapons to confuse people. However, these concepts fail if those people are not paying attention. Even worse, most attacks are not even done by people. They are usually performed by automated tools, such as auto-rooters or worms. Deception or deterrence will not prevent these attacks because there is no conscious individual to deter or deceive.

Deception and deterrence can work for organizations that want to protect high-value resources, In those cases there is a human attacker analyzing the information given out by the honeypot. The intent would be to confuse skilled attackers focusing on targets of choice. Unlike attackers who focus on the easy kill, these attackers carefully select their targets and analyze the information they receive from them. In such a case, honeypots could be used to deceive or deter the attacker.

One example of deception would be for deployment in a large government organization that conducts highly sensitive research. This research could have extreme value to other nations, that would target specific systems to obtain the classified material. A honeypot could be used to deceive and confuse the attacker, preventing further attacks. A honeypot fileserver could be created, acting as a central repository for classified documentation. However, instead of recording valid documentation, bogus material could be created and planted on the honeypot. Then the attackers would be given access to the honeypot fileserver, where they could obtain the documentation. For exmple, an attacker would believe she captured the plans for an advanced jet fighter when in reality shee has bogus plans for some nonexistent plane that will never fly.

This model only works for attackers who focus on targets of choice. In our honeypot fileserver example, for the deception to work, the attacker must obtain the documents, read the documentation, and understand its content. Attackers focusing on targets of opportunity would bypass this deception. They are not interested in documents; they are interested in compromising large number of systems. In many cases the attackers may not even be able to understand the documents, especially it it is not in their native language.

Where psychological weapons may fail, other honeypot can contribute to prevention. Later in the book we discuss LaBrea Tarpit, a unique honeypot that can slow down automated attacks, especially worms. While solutions such as these do not directly prevent attacks, they can be used to potentially mitigate the risk.

However, the time and resources involved in deploying honeypots for preventing attacks, especially prevention based on deception or deterrence, is time better spent on security best practices. As long as you have vulnerable systems, you will hacked. No honeypot can prevent that.

Detection

The second tier of security is *detection*, the act of detecting and alerting to unauthorized activity. If you were to secure your house, detection would be the installation of burglar alarms and motion detectors. These alarms go off when someone breaks in. In case the window was left open or the lock on the front door was picked, we want to detect the burglar if they get into our house. Within the world of information security, we have the same challenge. Sooner or later, prevention will fail, and the attacker will get in. There are a variety of reasons why this failure can happen: A firewall rulebase may be misconfigured, an employee uses an easy-to-guess password, a new vulnerability is discovered in an application. There are numerous methods for penetrating an organization. Prevention can only mitigate risk; it will never eliminate it.

Within the security community we already have several technologies designed for detection. One example is Network Intrusion Detection Systems, a solution designed to monitor networks and detect any malicious activity. There are also programs designed to monitor system logs that, once again, look for unauthorized activity. These solutions do not keep out the bad guys, but they alert us if someone is trying to get in and if they are successful.

How do honeypots help detect unauthorized or suspicious activity? While honeypots add limited value to prevention, they add extensive value to detection.

For many organizations, detection is extremely difficult. Three common challenges of detection are false positives, false negatives, and data aggregation. False positives are when systems falsely alert suspicious or malicious activity. What a system thought was an attack or exploit attempt was actually valid production traffic. False negatives are the exact opposite: They are when an organization fails to detect an attack. The third challenge is data aggregation, centrally collecting all the data used for detection and then corroborating that data into valuable information.

A single false positive is not a problem. Occasionally, there is bound to be a false alert. The problem occurs when these false alerts happen hundreds or even thousands of times a day. System administrators may receive so many alerts in one day that they cannot respond to all of them. Also, they often become conditioned to ignore these false positive alerts as they come in day after day—something like "the boy who cried wolf." If you received three hundred e-mails a day that were false alerts, you would most likely start to ignore your detection and alerting mechanisms. The very systems that organizations were depending on to notify them of attacks become ineffective as administrators stop paying attention to them.

Network Intrusion Detection Systems are an excellent example of this challenge. They are very familar with false positives. NIDS sensors are designed to monitor network traffic and detect suspicious activity. Most NIDS sensors work from a database of recognized signatures. When network activity matches the known signatures, the sensors believe they have detected unauthorized activity and alert the security administrators. However, valid production traffic can easily match the signature database, falsely triggering an alert. For example, I subscribe to Bugtraq[2], a public mailing list used to distribute vulnerability information. E-mails from Bugtraq often contain source code or output from exploits. These e-mails contain the very same signatures used by the NIDS sensors. When I receive these e-mails, the sensors see the source code, match that against their database, and then trigger an alert. This is a false positive. There are a variety of other types of traffic that can accidentally trigger NIDS sensors, including ICMP network traffic, file sharing of documents, or Web pages that have the same name as known Web server attacks.

The only solution to false positives is to modify the system to not alert about valid, production traffic. This is an extremely time-consuming process, requiring highly skilled individuals who understand network traffic, system logs, and application activity. People have to recognize valid traffic on the network and then compare that traffic to the NIDS signature database. Any signatures that are causing false positives must be either modified or removed entirely. It is hoped this will reduce the number of false positives, making the detection and alerting process far more effective. However, there is another challenge to reducing false positives: By modifying and eliminating a large number of signatures, an organization can have the problem of false negatives.

A false negative is when a system fails to detect a valid attack. Just as one may receive too many alerts, one can also receive too few. The risk is that a successful attack may occur, but the systems fail to detect and alert to the activity. NIDS not only face the challenge of false positives but also have problems with false negatives. Many NIDS systems, whether they are based on signatures, protocol verification, or some other methodology, can potentially miss new or unknown attacks.

For example, a new attack may be released within the blackhat community. Because this is a new attack, most NIDS sensors will not have the proper signatures to detect the attacks. Blackhats can use the new attacks with little fear of detection. Therefore, as new tools, attacks, and exploits are discovered, NIDS signature databases have to be updated. If a NIDS fails to update its signature database, it may once again miss an attack. In addition, new evasion methods are constantly being developed. These methods are designed to bypass detection. There are a variety of techniques for obscuring known attacks so NIDS and other detection mechanisms will fail to detect them. One example is ADMmutate [3], created by K2. This utility will take a known exploit and modify its signatures. Detection systems will fail to see attacks wrapped by ADMmutate, since the attack signatures have been modified.

The third challenge to detection is data aggregation. Modern technology is extremely effective at capturing extensive amounts of data. NIDS, system logs, application logs—all of these resources are very good at capturing and generating gigabytes of data. The challenge becomes how to aggregate all this data so it has value in detecting and confirming an attack. New technologies are constantly being created to pull all this data together to create value, to potentially detect attacks. However, at the same time, new technologies are being developed that generate more forms of new data. The problem is technology is advancing too rapidly, and the solutions for aggregating data cannot keep up with the solutions that produce the data.

Due to their simplicity, honeypots effectively address the three challenges of detection: false positives, false negatives, and data aggregation. Most honeypots have no production traffic, so there is little activity to generate false positives. The only time a false positive occurs is when a mistake happens, such as when a DNS server is misconfigured or when Martha in Accounting accidentally points her browser at the wrong IP address. In most other cases, honeypots generate valid alerts, greatly reducing false positives.

Honeypots address false negatives because they are not easily evaded or defeated by new exploits. In fact, one of their primary benefits is they can detect a new attack by virtue of system activity, not signatures. This can be demonstrated in the case of ADMmutate. ADMmutate defeats NIDS by altering the network signature of common attacks. However, a honeypot does not use a signature database. It works on the concept that anything sent its way is suspect. If an attacker wrapped an exploit with ADMmutate, NIDS sensors would most likely miss the attack because the signature was modified and did not match its database. A honeypot, on the other hand, would quickly detect the attack, ignoring any modifications made by ADMmutate, and alert the proper security personnel. Additionally, honeypots do not require updated signature databases to stay current with new threats or attacks. Honeypots happily capture any attacks thrown their way. This was demonstrated in January 2002 when the Honeynet Project caught the unknown dtspcd exploit in the wild with a honeypot.

The simplicity of honeypots also addresses the third issue: data aggregation. Honeypots address this issue by creating very little data. There is no valid production traffic to be logged, collected, or aggregated. Honeypots generate only several megabytes of data a day, most of which is of high value. This makes it extremely easy to diagnose useful information from honeypots. We demonstrated this earlier with the covert UDP network sweep when discussing the advantages of honeypots and how they collected data of high value.

One example of using a honeypot for detection would be deployment within a DMZ, often called the Demilitarized Zone. This is a network of untrusted systems normally used to provide services to the Internet, such as e-mail or Web server. These are systems at great risk, since anyone on the Internet can initiate a connection to them, so they are likely to be attacked and potentially compromised. Detection of such activity is critical. However, such attacks are difficult to detect because there is so much production activity. All of this traffic can generate a significant amount of false positives. Administrators may quickly ignore alerts generated by traffic within the DMZ. Also, because of the large amounts of traffic generated, data aggregation becomes a challenge. However, we also do not want to miss any attacks, specifically false negatives.

To help address these issues, a honeypot could be deployed within the DMZ (see Figure 4-2) to help detect attacks. The honeypot would have no production value. Its only purpose would be to detect attacks. It would not be in any DNS entries nor would it be registered or virtually linked to any systems. Since it has no production activity, false positives are drastically reduced. Any connection from the Internet to the honeypot indicates someone is probing the DMZ systems. If someone were to connect to port 25 on the honeypot, this indicates someone is most likely scanning for sendmail vulnerabilities. If someone were to connect to port 80 on the honeypot, this indicates a potential attacker scanning

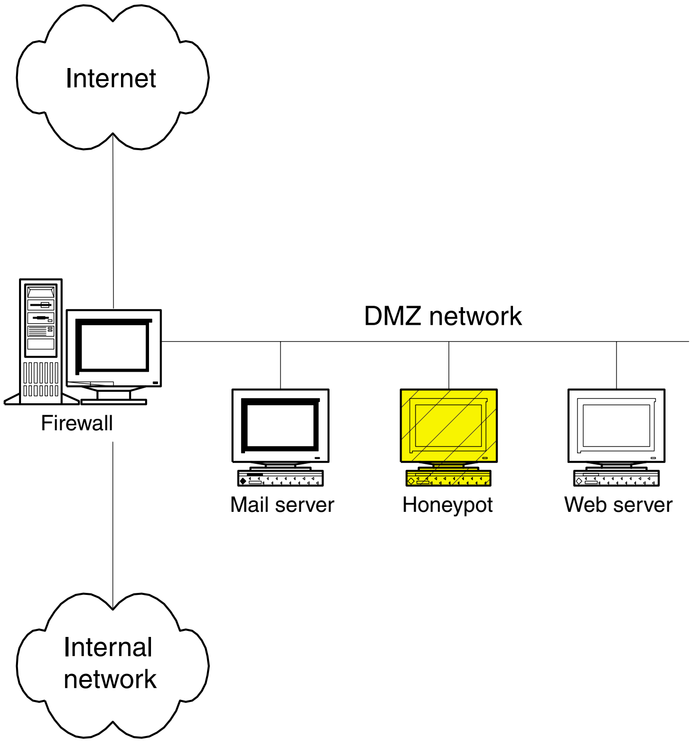

Figure 4-2 Network diagram of a honeypot deployed on a DMZ to detect attacks

for HTTP vulnerabilities. Even more telling would be if either the Web server or the mailserver initiated connections to the honeypot. If the honeypot detected any activity from these systems to itself, this would indicate that these systems had been compromised and were now being used to scan for other vulnerable systems.

Since it only detects and logs unauthorized activity, the honeypot also helps reduce the amount of data collected, making data aggregation much easier. For example, when the honeypot is scanned by a source, the organization can flag the source IP address as potentially hostile and then use that to analyze the data it has already collected, such as in firewall logs or systems logs.

Finally, false negatives are also reduced, since the honeypot will detect any activity sent its way. In the case of ADMmutate, such attacks could potentially be successful against production systems and would never be detected. The attacker could get into a compromised system, bypassing any detection mechanisms. However, in the case of our DMZ honeypot, such an attack would easily be detected.

Keep in mind that honeypots are not the ultimate solution for detection. They are merely a technology to help detect unauthorized activity. At the beginning of the chapter, we discussed several disadvantages of honeypots. The largest issues with honeypots is they only detect activity directed *at* them. In our example with the DMZ, the honeypot would not detect any attacks sent to either the mailserver or the Web server. An attacker could have successfully attacked and exploited any system on the DMZ, and the honeypot would have never detected it. The only way the honeypot will detect activity is if the honeypot itself is also attacked. By no means should honeypots replace your NIDS systems or be your sole method of detection. However, they can be a powerful tool to complement your detection capabilities.

Response

Once we detect a successful attack, we need the ability to respond. When securing our house, we want to be sure someone can protect us in case of a break-in. Often house burglar alarms are wired to monitoring stations or the local police department. When an alarm goes off, the proper authorities are alerted and can quickly react, protecting your house. The same logic applies to securing your organization. Honeypots add value to the response aspect of security.

The challenge that organizations face when reacting to an incident is evidence collection—that is, figuring out what happened when. This is critical not only if an organization wants to prosecute an attacker but also when it comes to defending against an attack. Once compromised, organizations must determine if the attacker hacked into other systems, created any back doors or logic bombs, modified or captured any valuable information such as user accounts. Have other people infiltrated their networks?

When an attacker breaks into a system, their actions leave evidence, evidence that can be used to determine how the attacker got in, what she did once she gained control of the system, and who she is. It is this evidence that is critical to capture. Without it, organizations cannot effectively respond to the incident.

Even if the attackers take steps to hide their actions, such as modifying system log files, these actions can still be traced. Advanced forensic techniques make it possible to recover the attacker's actions. For example, it is possible to determine step by step what an attacker did by looking at the MAC (modify, access, change) times of file attributes. On most operating systems, each file maintains information on when that file was last modified, accessed, or changed. Determining what time certain files were accessed or modified can help determine the attackers actions. There are tools designed to look at systems files and determine the sequence of events based entirely on MAC times. The Coroner's Toolkit [4], designed by Dan Farmer and Wietse Venema, is a good one, and there are many others.

However, this evidence can quickly become polluted, making it worthless. A great deal of activity is almost always happening on production systems. Files are being written to the hard drive, processes are starting and stopping, users are logging in, memory is paged in and out—all this activity is constantly changing the state of a system. The more activity on a system, the more likely the attacker's actions will be overwritten or polluted. Even with the advanced tools and techniques, it can be very difficult to recover data that has been damaged. Think of a busy train station, one with people constantly coming and going. The activity of the people arriving at and departing from the train station represents the constant activity on a computer. When a crime is committed in the subway, certain evidence is left, such as fingerprints or hair samples. However, the greater the activity in the train station, the more likely this evidence will be contaminated. Perhaps someone else's fingerprints are on top of the attacker's, or hair samples are blown away by a passing train. The same forms of data pollution happen on computers. Recovering unpolluted evidence from a compromised system is one of the biggest challenges facing incident response teams.

A second challenge many organizations face after an incident is that compromised systems frequently cannot be taken offline. To properly obtain evidence from a compromised system, the attacked systems must be pulled offline and analyzed by other computers. This often means having to pull the actual hard drives from the compromised computer. Obviously, the attacked system can no longer do its job if it is taken offline. Many organizations cannot afford to lose the functionality of systems and will not allow an attacked system to be pulled offline for analysis. For example, an organization may have a critical Web server or database they cannot afford to have down. Instead, many organizations will attempt to minimize the attackers damage while leaving the resource online. Instead of taking down the system, they merely patch the system in an attempt to block anyone from coming in again. No attempt is made to learn how the attacker compromised the system, let alone recover any detailed evidence. The problem now is that organizations cannot react to a system compromise because they cannot properly analyze attacked systems.

Honeypots can help address these challenges to reaction capability. Remember, a honeypot has no production activity, so this helps the problem of data pollution. When a honeypot is compromised, the only real activity on the system is the activity of the attacker, helping to maintain its integrity. If we look at our train station analogy, imagine a crime at a train station where there are no people or trains coming or going. Evidence such as fingerprints or hair samples are far more likely to remain intact. The same case is true for honeypots. Honeypots can also easily be taken offline for further analysis. Since honeypots provide no production services, organizations can easily take them down for analysis without impacting business activity.

As an example of how a honeypot can add value to incident response, consider a large organization with multiple Web servers. Instead of having everyone on the Internet connect to a single Web server, the organization distributes the load across multiple Web servers, helping to improve performance. In such an environment, a honeypot could be deployed for not only detection purposes, as discussed earlier, but for incident response purposes. Once again, let's look at a DMZ but this time with multiple Web servers, all listening on port 80, HTTP (Figure 4-3). In this deployment we have three Web servers and one honeypot. All four systems are listening on port 80, HTTP, which the firewall allows inbound. However, only the three Web servers have entries in DNS, so these are the only three systems that will get a valid request for Web pages. Since the honeypot is not listed in DNS, it will not get any requests for Web pages, and it will not have any production traffic. Our honeypot, however, is running the same applications as our Web servers.

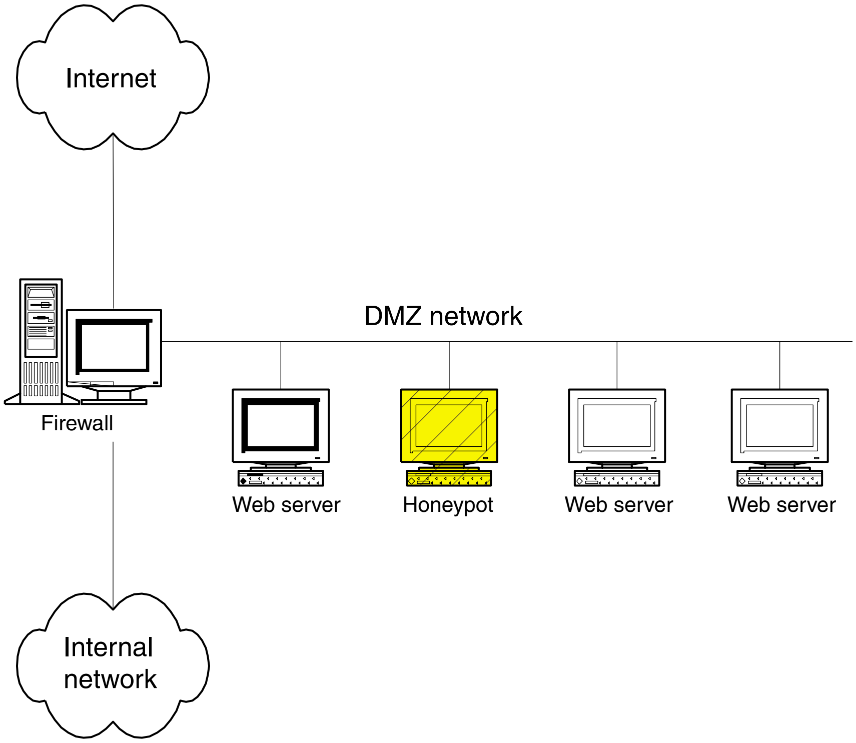

Figure 4-3 Honeypot deployed within a DMZ used for incident response

Notice how the honeypot in Figure 4-3 is in the middle of the network. Now if an attacker sequentially scans each system in our DMZ for any Web-based vulnerability, the scan will most likely also hit our honeypot. If one of the Web servers is successfully attacked, so too may be the honeypot. If multiple systems are compromised, the attacker most likely used the same tools and methods on all the systems, including the honeypot. Organizations can then focus on the honeypot for data collection and analysis and then apply the lessons learned to the other compromised systems.

Keep in mind that honeypots are not the single solution for incident response, they are only a tool to assist. The most critical step any organization can take is preparing *before* an incident. Examples include having a documented response plan, taking images of critical files for future analysis, and having the technical tools to quickly recover evidence. It is these best practices that will ensure effective incident responses. However, honeypots can be a powerful tool to complement your reaction capabilities by capturing details on how the attacker got in and what they did. If you are interested in learning more about incident response and forensics, I highly recommend the books *Incident Response* [5] or *Computer Forensics* [6].

Research Honeypots

One of the greatest challenges the security community faces is lack of information on the enemy. Questions like who is the threat, why do they attack, how do they attack, what are their tools, and when will they strike again often cannot be answered. The intelligence and counterintelligence community spend billions of dollars on information-gathering capabilities because knowledge is such a critical asset. To defend against a threat, you have to first be informed about it. However, in the information security world we have little such information.

The problem has been the source of data. Traditionally, security professionals have learned about blackhats by studying the tools the blackhats use. When a system was compromised, security administrators would often find the attacker's tools left on the attacked system. A variety of assumptions are then made about the attacker's based on these captured tools. This technique is similar to archaeology, where trained professionals attempt to understand centuries-old cultures by the tools they leave behind for us to find. While this technique is effective, so much more can be learned about attackers. Instead of learning only about the attacker's tools, it makes sense to identify these cyberadversaries, determine how well organized they are, and determine their methods. Honeypots can help us learn these things.

Research honeypots offer extensive value in information gathering by giving us a platform to study cyberthreats. What better way to learn about the bad guys than to watch them in action, to record step by step as they attack and compromise a system. Imagine watching an attack take place from beginning to end. Instead of just finding the attacker's tools, you can watch the attacker probe the system and launch his attacks. You can see exactly what he does, keystroke by keystroke, after he gains access.

The value of such information is tremendous, and it offers a variety of potential uses. For example, research honeypots can be used for the following.

- To capture automated threats, such as worms or auto-rooters. By quickly capturing these weapons and analyzing their malicious payload, organizations can better react to and neutralize the threat.
- As an early warning mechanism, predicting when future attacks will happen. This works by deploying multiple honeypots in different locations and organizations. The data collected from these research honeypots can then be used for statistical modeling, predicting future attacks. Attacks can then be identified and stopped before they happen.
- To capture unknown tools or techniques, as demonstrated with dtspcd attack, or covert NVP communications, discussed later in the book.
- To better understand attacker's motives and organization. By capturing their activity after they break into a system, such as communications among each other, we can better understand who our threat is and why they operate.
- To gain information on advanced blackhats.

This final point is one of the most exciting applications of research honeypots. As we discussed in Chapter 2, very little is known about how advanced blackhats operate, since they are extremely difficult to detect and capture. Research honeypots represent one method for gaining intelligence on this small but notably skilled group of individuals. Imagine building a honeypot that appeared to have high value, such as an emulated e-commerce site. An advanced attacker could identify and attack such a site, exposing his tools and tactics for the world to see. The CD-ROM contains the entire "Know Your Enemy" series of whitepapers published by the Honeynet Project. This series presents in-depth information on the gathering capabilities of research honeypots.

In general, research honeypots do not reduce the risk to an organization, but the information learned can be applied, such as how to improve prevention, detection, or reaction. However, research honeypots contribute little to the direct security of an organization. If an organization is looking to improve the security of its production environment, it may want to consider production honeypots because they are easy to implement and maintain. If organizations such as universities, governments, or very large companies are interested in learning more about threats, then this is where research honeypots would be valuable. The Honeynet Project is one such example of an organization using research honeypots to gain information on the blackhat community.

Honeypot Policies

For honeypots to be effective, any organizations using them must have a clearly defined security policy. A security policy defines how an organization approaches, implements, and enforces security measures to mitigate the risk to its environment. A honeypot is a technical tool used to enforce that policy. If the policy is not clearly defined, then a honeypot cannot contribute much. For example, if a honeypot is used for detection, its value is to detect unauthorized activity, such as scans, probes, or attacks. Such a honeypot may detect an employee sequentially scanning every system within an organization's network for open file shares on fellow employee workstations. The honeypot is successful in that it detects the probes and alerts the security administrator. However, was this unauthorized activity? That depends on the company's security policy. Organizational policy is critical for a second reason: the legality of honeypots. Chapter 14 covers the legal issues involved with honeypot technologies. However, organizations must also determine if it is legal to us a honeypot. Can a honeypot record the activities of an employee, even if that employee is conducting unauthorized activity? The answer may sound simple, but if the security policy does not clearly define authorized monitoring activities, the legality of honeypots may become an issue. It is critical that security policies indicate what monitoring functionality, not monitoring technology is permitted. For example, a security policy may state that honeypots are authorized, but what exactly does that mean? An organization may allow honeypots to detect scans or probes, but does it allow research honeypots that may capture keystrokes or the conversations of online chat sessions? These issues must be clearly defined before honeypots are deployed.

Summary

Honeypots are a highly flexible technology that can be applied to a variety of situations. As security tools, they have specific advantages. Specifically, honeypots collect small amounts of data, but most of this is information of high value. They have the ability to effectively work in resource intensive environments, and conceptually they are very simple devices. Also, they quickly demonstrate their value by detecting and capturing unauthorized activity.

However, honeypots share several major disadvantages. The most critical is they have a narrow field of view. If they are not attacked, they have no value. Second, certain honeypots can be fingerprinted, making detection possible. The third disadvantage is that honeypots can add additional risk: The honeypot may be used to attack or harm other systems or organizations. Any time you add additional services or applications to your environment, there are more things that can go wrong.

Within the three areas of security—prevention, detection and response—the primary value of production honeypots is detection. Because production honeypots greatly reduce the problem of both false negatives and false positives, they make an extremely efficient technology for detecting unauthorized activity. They also have some value with respect to reaction and, relatedly, helping organizations to develop their incident response skills. For prevention purposes, production honeypots are of minimal value. The concepts of deception and deterrence can be applied with honeypots to prevent attacks, but most organizations are better off spending their limited resources on security best practices, such as patching vulnerable services. Honeypots will not stop vulnerable systems from being hacked.

Research honeypots do not mitigate risk, but they primarily are used to gain information about threats. This information is then used to better understand and protect against these threats. When deploying honeypots, it is critical that organizations have a clearly defined security policy stating what activity is and is not authorized, including the use of honeypots to detect and monitor.

References

[1] Schneier, Bruce. 2000. *Secrets and Lies*. New York, New York: John Wiley and Sons, Inc.

http://www.counterpane.com/orderac2.html

[2] Bugtraq Mailing List
http://www.securityfocus.com

[3] ADMmutate
http://www.ktwo.ca/security.html

[4] The Coroner's Toolkit
http://www.porcupine.org/forensics

[5] Mandia, Kevin and Chris Prosise. 2001. *Incident Response*. Berkeley, California: Osborne/McGraw-Hill.

[6] http://www.incidentresponsebook.com
Heiser, Jay, and Warren Kruse. 2002. *Computer Forensics*. Boston, Massachusetts: Addison-Wesley.