Honeypots

Definitions and Value of Honeypots

Lance Spitzner
With extensive help from Marty Roesch and David Dittrich
http://www.enteract.com/~lspitz
Last Modified: 17 May, 2002

Note: Coming in September, the only book dedicated to honeypots, *Honeypots: Tracking Hackers*!

Over the past several years there has been a growing interest in honeypots and honeypot related technologies. Honeypots are not a new technology, they were first explained by a couple of very good papers by several icons in computer security, Cliff Stoll's book the Cuckoo's Egg", and Bill Cheswick's paper "An Evening with Berferd." This paper attempts to take their work further and discuss what honeypots are, how they can add value to an organization, and several honeypot solutions. There are a variety of misconceptions on what a honeypot is, how it works, and how it adds value. It is hoped this paper helps clear up those issues. Also, few people realize the risk and issues involved with honeypots. Though honeypots can add value, the time and resources involved may best focused on greater priorities. If after reading this paper you are interested in learning more about honeypot technologies, I've created a website dedicated just to honeypots, at http://www.tracking-hackers.com.

Definitions

Before we jump into the paper, we should first agree on several definitions. Far too often I've seen people arguing on maillists about honeypots. What is so amusing is you can tell they are talking about two entirely different concepts. If they had taken a moment to agree on what they were arguing about first, life would have been much simpler for everyone (including my mailbox) To make sure we are all on the same sheet of music, I would like to first agree on some definitions. For this paper, I will first standardize on the definition of a honeypot, then the two different types of honeypots, and finaly the different categories of security and how they apply to honeypots.

I define a honeypot as "*a security resource who's value lies in being probed, attacked or compromised*". This means that whatever we designate as a honeypot, it is our expectation and goal to have the system probed, attacked, and potentially exploited. Keep in mind, honeypots are not a solution. They do not 'fix' anything. Instead, honeypots are a tool. How you use that tool is up to you and depends on what you are attempting to achieve. A honeypot may be a system that merely emulates other systems or applications, creates a jailed environment, or may be a standard built system. Regardless of how you build and use the honeypot, it's value lies in the fact that it is attacked.

We will break honeypots into two broad categories, as defined by Snort creator Marty Roesch. Marty pointed out to me that the two types of honeypots are "*production*" and "*research*", a breakdown I found to be very useful. The purpose of a production honeypot is to help mitigate risk in an organization. The honeypot adds value to the security measures of an organization. Think of them as 'law enforcement', their job is to detect and deal with bad guys. Traditionally, commercial organizations use production honeypots to help protect their networks. The second category, research, are honeypots designed to gain information on the blackhat community. These honeypots do not add direct value to a specific organization. Instead they are used to research the threats organizations face, and how to better protect against those threats. Think of them as 'counter-intelligence', their job is to gain information on the bad guys. This information is then used to protect against those threats. Traditionaly, commercial organizations do NOT use research honeypots. Instead, organizations such as Universities, government, military, or security research organizations use them.

Before discussing how honeypots add value to security, lets first define what security is. Security is the reduction of risk. One can never eliminate risk, but security helps reduce risk to an organization and its information related resources. When discussing security, I like to break it down into three areas, as defined by the infamous Bruce Schneier in *Secrets and Lies*. Bruce breaks security down into the three categories as follows.

- **Prevention:** We want to stop the badguys. If you were to secure your house, prevention would be similar to placing dead bolt locks on your doors, locking your window, and perhaps installing a chain link fence around your yard. You are doing everything possible to keep the threat out.
- **Detection:** We want to detect the badguys when they get through. Sooner or later, prevention will fail. You want to be sure you detect when such failures happen. Once again using the house analogy, this would be similar to putting a burglar alarm and motion sensors in the house. These alarms go off when someone breaks in. If prevention fails, you want to be alerted to that as soon as possible.
- **Reaction:** We want to react to the badguys once we detect them. Detecting the failure has little value if you do not have the ability to respond. What good does it to be alerted to a burglar if nothing is done? If someone breaks into your house and triggers the burglar alarm, one hopes that the local police force can quickly respond. The same holds true for information security. Once you have detected a failure, you must execute an effective response to the incident.

Now that we have a better idea of what security is, lets see how honeypots add value to each one of these three categories.

Value of Honeypots

Honeypots have certain advantages (and disadvantages) as security tools. It is the advantages that help define the value of a honeypot. The beauty of a honeypot's lies in its simplicity. It is a device intended to be compromised, not to provide production services. This means there is little or no production traffic going to or from the device. Any time a connection is sent to the honeypot, this is most likely a probe, scan, or even attack. Any time a connection is initiated from the honeypot, this most likely means the honeypot was compromised. As there is little production traffic going to or from the honeypot, all honeypot traffic is suspect by nature. Now, this is not always the case. Mistakes do happen, such as an incorrect DNS entry or someone from accounting inputing the wrong IP address. But in general, most honeypot traffic represents unauthorized activity.

Because of this simplistic model, honeypots have certain inherent advantages and disadvantages. We will cover several of them.

1. Advantage - Data Collection
   Honeypots collect very little data, and what they do collect is normally of high value. This cuts the noise level down, make it much easier to collect and archive data. One of the greatest problems in security is wading through gigabytes of data to find the data you need. Honeypots can give you the exactly the information you need in a quick and easy to understand format. For example, the Honeynet Project, a group researching honeypots, collects on average only 1-5MB of data per day. This information is normally of high value also, as not only can you show network activity, but what the attacker does once he or she gets on the system. We will go into greater depth in these advantage when we discuss how honeypots add value to detection.
2. Advantage - Resources
   Many security tools can be overwhelmed by bandwidth or activity. Network Intrusion Detection Devices may not be able to keep up with network activity, dropping packets, and potentially attacks. Centralized log servers may not be able to collect all the system events, potentially dropping some events. Honeypots do not have this problem, they only capture that which comes to them.

1. Disadvantage - Single Data Point
   Honeypots all share one huge drawback; they are worthless if no one attacks them. Yes, they can accomplish wonderful things, but if the attacker does not send any packets to the honeypot, the honeypot will be blissfully unware of any unauthorized activity.
2. Disadvantages - Risk
   Honeypots can introduce risk to your environment. 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. Risk is variable, depending on how one builds and deploys the honeypot.

It is because of these disadvantages that honeypots do not replace any security mechanisms. They can only add value by working with existing security mechanisms. Now that we have reviewed the overall value of honeypots, lets apply them to security.

As we discussed earlier, there are two types of honeypots, production and research. We will first discuss what a production honeypot is and its value. Then we will discuss research honeypots and their value.

A production honeypot is one used within an organization's environment to help mitigate risk. It adds value to the security of production resources. Lets cover how production honeypots apply to the three areas of security, Prevention, Detection, and Reaction.

Prevention
I personally feel honeypots add little value to prevention, honeypots will not help keep the bad guys out. What will keep the bad guys out is best practices, such as disabling unneeded or insecure services, patching what you do need, and using strong authentication mechanisms. It is the best practices and procedures such as these that will keep the bad guys out. A honeypot, a system to be compromised, will not help keep the bad guys out. In fact, if incorrectly implemented, a honeypot may make it easier for an attacker to get in.

Some individuals have discussed the value of deception as a method to deter attackers. The concept is to have attackers spend time and resource attacking honeypots, as opposed to attacking production systems. The attacker is *deceived* into attacking the honeypot, protecting production resources from attack. While this may prevent attacks on production systems, I feel most organizations are much better off spending their limited time and resources on securing their systems, as opposed to deception. Deception may contribute to prevention, but you will most likely get greater prevention putting the same time and effort into security best practices.

Also, deception fails against two of the most common attacks today; automated toolkits and worms. Today, more and more attacks are automated. These automated tools will probe, attack, and exploit anything they can find vulnerable. Yes, these tools will attack a honeypot, but they will also just as quickly attack every other system in your organization. If you have a coffee pot with an IP stack, it will be attacked. Deception will not prevent these attacks, as there is no consciously acting individual to deceive. As such, I feel that honeypots add little value to prevention. Organizations are better off focusing their resources on security best practices.

Detection
While honeypots add little value to prevention, I feel they add extensive value to detection. For many organizations, it is extremely difficult to detect attacks. Often organizations are so overwhelmed with production activity, such as gigabytes of system logging, that it can be extremely difficult to detect when a system is attacked, or even when successfully compromised. Intrusion Detection Systems (IDS) are one solution designed for detecting attacks. However, IDS administrators can be overwhelmed with false positives. False positives are alerts that were generated when the sensor recognized the configured signature of an "attack", but in reality was just valid traffic. The problem here is that system administrators may receive so many alerts on a daily basis 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, similar to the story of "the boy who cried wolf". The very IDS sensors that they were depending on to alert them to attacks can become ineffective unless these false positives are reduced. This does not mean that honeypots will never have false positives, only that they will be dramatically fewer than with most IDS implementations.

Another risk is false negatives, when IDS systems fail to detect a valid attack. Many IDS systems, wheter they are signature based, protocol verification, etc, can potentially miss new or unknown attacks. It is likely that a new attack will go undectected by currently IDS methodologies. Also, new IDS evasion methods are constantly being developed and distributed. It is possible to launch a known attack that may not be detected, such as with K2's ADM Mutate. Honeypots address false negatives as they are not easily evaded or defeated by new exploits. In fact, one of their primary benefits is that they can most likely detect when a compromise occurs via a new or unknown attack by virtue of system activity, not signatures. Administrators also do not have to worry about updating a signature database or patching anamoly detection engines. Honeypots happily capture any attacks thrown their way. As discussed earlier though, this only works if the honeypot itself is attacked.

Honeypots can simplify the detection process. Since honeypots have no production activity, all connections to and from the honeypot are suspect by nature. By definition, anytime a connection is made to your honeypot, this is most likely an unauthorized probe, scan, or attack. Anytime the honeypot initiates a connection, this most likely means the system was successfully compromised. This helps reduce both false positives and false negatives greatly simplifying the detection process. By no means should honeypots replace your IDS systems or be your sole method of detection. However, they can be a powerful tool to complement your detection capabilities.

Reaction
Though not commonly considered, honeypots also add value to reaction. Often when a system within an organization is compromised, so much production activity has occurred after the fact that the data has become polluted. Incident response team cannot determine what happened when users and system activity have polluted the collected data. For example, I have often come onto sites to assist in incident response, only to discover that hundreds of users had continued to use the compromised system. Evidence is far more difficult to gather in such an environment.

The second challenge many organizations face after an incident is that compromised systems frequently cannot be taken off-line. The production services they offer cannot be eliminated. As such, incident response teams cannot conduct a proper or full forensic analysis.

Honeypots can add value by reducing or eliminating both problems. They offer a system with reduced data pollution, and an expendable system that can be taken off-line. For example, lets say an organization had three web servers, all of which were compromised by an attacker. However, management has only allowed us to go in and clean up specific holes. As such, we can never learn in detail what failed, what damage was done, is there attacker still had internal access, and if we were truly successful in cleanup.

However, if one of those three systems was a honeypot, we would now have a system we could take off-line and conduct a full forensic analysis. Based on that analysis, we could learn not only how the bad guy got in, but what he did once he was in there. These lessons could then be applied to the remaining webservers, allowing us to better identify and recover from the attack.

Research
As discussed at the beginning, there are two categories for honeypots; production and research. We have already discussed how production honeypots can add value to an organization. We will now discuss how research honeypots add value.

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 possibly when will they attack? It is questions like these the security community often cannot answer.

For centuries military organizations have focused on information gathering to understand and protect against an enemy. To defend against a threat, you have to first know about it. However, in the information security world we have little such information.

Honeypots can add value in research by giving us a platform to study the threat. What better way to learn about the bad guys then to watch them in action, to record step-by-step as they attack and compromise a system. Of even more value is watching what they do after they compromise a system, such as communicating with other blackhats or uploading a new tool kit. It is this potential of research that is one of the most unique characteristics of honeypots. Also, research honeypots are excellent tools for capturing automated attacks, such as auto-rooters or Worms. Since these attacks target entire network blocks, research honeypots can quickly capture these attacks for analysis.

In general, research honeypots do not reduce the risk of an organization. The lessons learned from a research honeypot 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 their production environment, they may want to consider production honeypots, as they are easy to implement and maintain. If organizations, such as universities, governments, or extremely large corporations are interested in learning more about threats, then this is where research honeypots would apply. The Honeynet Project is one such example of an organization using research honeypots to capture information on the blackhat community.

Honeypot Solutions

Now that we have been discussing the different types of honeypots and and their value, lets discuss some examples. The more and more I work with honeypots, the more I realize that no two honeypots are alike. Because of this, I have identified what I call *level of interaction*. Simply put, the more an attacker can interact with a honeypot, the more information we can potentially gain from it, however the more risk it most likely has.

The more a honeypot can do and the more an attacker can do to a honeypot, the more information can be derived from it. However, by the same token, the more an attacker can do to the honeypot, the more potential damage an attacker can do. For example, a low interaction honeypot would be one that is easy to install and simply emulates a few services. Attackers can merely scan, and potentially connect to several ports. Here the information is limited (mainly who connected to what ports when) however there is little that the attacker can exploit. On the other extreme would be high interaction honeypots. These would be actual systems. We can learn far much more, as there is an actual operating system for the attacker to compromise and interact with, however there is also a far greater level of risk, as the attacker has an actual operating system to work with. Neither solution is a better honeypot. It all depends on what you are attempting to achieve. Remember, honeypots are not a solution. Instead, they are a tool. Their value depends on what your goal is, from early warning and detection to research. Based on 'level of interaction', lets compare some possible honeypot solutions.

For this paper, we will discuss six honeypots. There are a variety of other possible honeypots, however this selection covers a range of options. We will cover BackOfficer Friendly, Specter, Honeyd, homemade honeypots, Mantrap, and Honeynets. This paper is not meant to be a comprehensive review of these products. I only highlight some of their features. Instead, I hope to cover the different types of honeypots, how they work, and demonstrate the value they add and the risks involved. If you wish to learn more about the capabilities of these solutions, I highly recommend you try them out on your own in a controlled, lab environment.

BackOfficer Friendly

BOF (as it is commonly called) is a very simple but highly useful honeypot developed by Marcus Ranum and crew at NFR. It is an excellent example of a low interaction honeypot.

The reason I am such a big fan of this is due to BOF's simplicity. It is a great way to introduce a begginer to the concepts and value of honeypots. BOF is a program that runs on most Window based operating system. All it can do is emulate some basic services, such as http, ftp, telnet, mail, or

BackOrrifice. Whenever some attempts to connect to one of the ports BOF is listening to, it will then log the attempt. BOF also has the option of "faking replies", which gives the attacker something to connect to. This way you can log http attacks, telnet brute force logins, or a variety of other activity (Screenshot). I like to run BOF on my laptop, as it gives me a feel for what type of activity may be occuring. The value in BOF is in detection, similar to a burglar alarm. It can monitor only a limited number of ports, but these ports often represent the most commonly scanned and targeted services.

Specter
Specter is a commercial product and what I would call another 'low interaction' production honeypot. It is similar to BOF in that it emulates services, but it can emulate a far greater range of services and functionality. In addition, not only can it emulate services, but emulate a variety of operating systems. Similar to BOF, it is easy to implement and low risk. Specter works by installing on a Windows system. The risk is reduced as there is no real operating system for the attacker to interact with. For example, Specter can emulate a webserver or telent server of the operating system of your choice. When an attacker connects, it is then prompted with a http header or login banner. The attacker can then attempt to gather web pages or login to the system. This activity is captured and recorded by Specter, however there is little else the attacker can do. There is no real application for the attacker to interact with, instead just some limited, emulated functionality. Specters value lies in detection. It can quickly and easily determine who is looking for what. As a honeypot, it reduces both false positives and false negatives, simplifying the detection process. Specter also support a variety of alerting and logging mechanisms. You can see an example of this functionality in a screen shot of Specter.

One of the unique features of Specter is that it also allows for information gathering, or the automated ability to gather more information about the attacker. Some of this information gathering is relatively passive, such as Whois or DNS lookups. However, some of this research is active, such as port scanning the attacker. While this intelligence functionality may be of value, many times you do not want the attacker to know he is being watched. Be careful when implementing any active, automated responses to the attacker.

Homemade Honeypots
Another common honeypot is homemade. These honeypots tend to be low interaction. Their purpose is usually to capture specific activity, such as Worms or scanning activity. These can be used as production or research honeypots, depending on their purpose. Once again, there is not much for the attacker to interact with, however the risk is reduced because there is less damage the attacker can do. One common example is creating a service that listens on port 80 (http) capturing all traffic to and from the port. This is commonly done to capture Worm attacks. One such implementation would be using netcat, as follows:

```
netcat -l -p 80 > c:\honeypot\worm
```

In the above command, a Worm could connect to netcat listening on port 80. The attacking Worm would make a successful TCP connection and potentially transfer its payload. This payload would then be saved locally on the honeypot, which can be further analyzed by the administrator, who can assess the threat of the Worm. Organizations such as SANS and SecurityFocus.com have had success using homemade honeypots to capture and analyze Worms and automated activity.

Homemade honeypots can be modified to do (and emulate) much more, requiring a higher level of invovlement, and incurring a higher level of risk. For example, FreeBSD has a *jail* functionality, allowing an administrator to create a controlled environment within the operating system. The attacker can then interact with this controlled environment. The value here is the more the attacker can do, the more can be potentially learned. However, care must be taken, as the more functionality the attacker can interact with, the more can go wrong, with the honeypot potentially compromised.

Some additional examples of homemade honeypots:

- Port listener coded in PERL by Johannes B. Ullrich, used to capture the W32/Leaves Worm.
- Windows Inetd emulator for Windows NT and Win2000.
- Sendmail Honeypots, used to identify sendmail spammers.
- LaBrea Tarpit is a unique approach to honeypots, allowing you not only to capture worm activity, but potentially slow or disable worm attacks.

Honeyd

Created by Niels Provos, Honeyd is an extremely powerful, OpenSource honeypot. Designed to run on Unix systems, it can emulate over 400 different operating systems and thousands of different computers, all at the same time. Honeyd introduces some exicting new features. First, not only does it emulate operating systems at the application level, like Specter, but it also emulates operating systems at the IP stack level. This means when someone Nmaps your honeypot, both the service and IP stack behave as the emulated operating system. Currently no other honeypot has this capability (CyberCop Sting did have this capability, but is no longer available). Second, Honeyd can emulate hundreds if not thousands of different computers all at the same time. While most honeypots can only emulate one computer at any point in time, Honeyd can assume the identify of thousands of different IP addresses. Third, as an OpenSource solution, not only is it free to use, but it will expotentially grow as members of the security community develop and contribute code.

Honeyd is primarily used for detecting attacks. It works by monitoring IP addresses that are unused, that have no system assigned to them. Whenever an attacker attempts to probe or attack an non-existant system, Honeyd, through Arp spoofing, assumes the IP address of the victim and then interacts with the attacker through emulated services. These emulates services are nothing more then scripts that react to predetermined actions. For example, a script can be devloped to behave like a Telnet service for a Cisco router, with the Cisco IOS login interface. Honeyd's emuilated services are also OpenSource, so anyone can develop and use their own. The scripts can be written in almost any language, such as shell or Perl. Once connected, the attacker belives they are interacting with a real system. Not only can Honeyd dynamically interact with attackers, but it can detect activity on any port. Most low interaction honeypots are limited to detecting attacks only on the ports that have emulated services listening on. Honeyd is different, it detects and logs connections made to any port, regardless if there is a service listening. The combined capabilities of assuming the identify of non-existant systems, and the ability to detect activity on any port, gives Honeyd incredible value as a tool to detect unauthorzied activity. I highly encourage people to check it out, and if possible to contribute new emulated services.

Now we begin to move into more honeypots with greater levels of interaction. These solutions give us far greater information, but potentially have far greater risk. We will be discussing to such honeypots, Mantrap and Honeynets. We will begin with Mantrap.

Mantrap

Produced by Recourse Mantrap is a commercial honeypot. Instead of emulating services, Mantrap creates up to four sub-systems, often called 'jails'. These 'jails' are logically discrete operating systems separated from a master operating system (see Diagram.) Security administrators can modify these jails just as they normally would with any operating system, to include installing applications of their choice, such as an Oracle database or Apache webserver. This makes the honeypot far more flexible, as it can do much more. The attacker has a full operating system to interact with, and a variety of applications to attack. All of this activity is then captured and recorded. Not only can we detect port scans and telnet logins, but we can capture rootkits, application level attacks, IRC chat session, and a variety of other threats. However, just as far more can be learned, so can more go wrong. Once compromised, the attacker can used that fully functional operating system to attack others. Care must be taken to mitigate this risk. As such, I would categorize this as a mid-high level of interaction. Also, these honeypots can be used as either a production honeypot (used both in detection and reaction) or a research honeypot to learn more about threats. There are limitations to this solution. The biggest one is you are limited to what the vendor supplies you. Currently, Mantrap only exists on Solaris operating system.

Honeynets
Honeynets represent the extreme of research honeypots. They are high interaction honeypots, you can learn a great deal, however they also have the highest level of risk. Their primary value lies in research, gaining information on threats that exist in the Internet community today. A Honeynet is a network of production systems. Unlike many of the honeypots we have discussed so far, nothing is emulated. Little or no modifications are made to the honeypots. This gives the attackers a full range of systems, applications, and functionality to attack. From this we can learn a great deal, not only their tools and tactics, but their methods of communication, group organization, and motives. However, with this capability comes a great deal of risk. A variety of measures must be taken to ensure that once compromised, a Honeynet cannot be used to attack others. Honeynets are primarily research honeypots. They could be used as production honeypots, specifically for detection or reaction, however it is most likely not worth the time and effort. Most of the low interaction honeypots we have discussed so far give the same value for detection and reaction, but require less work and have less risk. If you are interested in learning more about Honeynets, you may want to review the book Know Your Enemy.

We have reviewed six different types of honeypots. No one honeypot is better then the other, each one has its advantages and disadvantages, it all depends on what you are trying to achieve. To more easily define the capabilities of honeypots, we have categorized them based on their level of interaction. The greater interaction an attacker has, the more we can learn, but the greater the risk. For example, BOF and Specter represent low interactions honeypots. They are easy to deploy and have minimal risk. However, they are limited to emulating specific services and operatings systems, used primarily for detection. Mantrap and Honeynets represent mid-to-high interaction honeypots. They can give far greater depth of information, however more work and greater risk is involved.

Legal Issues

No discussion about honeypots would be complete without covering the legal issues. Honeypots are just too cool not to have some legal issues. I am not a laywer. I have no real legal training or background. In fact, I was a History major at college, and not a very good one at that. So what I'm about to discuss are my own opinions, and not based on any legal precedent. When discussing honeypots, there are often two legal issues; entrapment and privacy. We will briefly review these issues. Lets start first with the issue of entrapment. The legal definition of entrapment is

A person is 'entrapped' when he is induced or persuaded by law enforcement officers or their agents to commit a crime that he had no previous intent to commit.

I personally feel that entrapment is not an issue. First, most individuals or organizations are not law enforcement, nor agents of law enforcement. We are not acting under the control of law enforcement, and we don't even have prosecution as an intent. Therefore, the legal definition of entrapment does not apply. Even for law enforcement, honeypots most likely do not represent entrapment, as they are not used to induce nor persuade attackers. Nothing is done to induce or persuade attackers to target Honeypots. Instead, attackers target and attack honeypots are there own initiative. As such, entrapment is most likely not an issue with honeypots technologies.

The next potential issue is privacy, either in the files placed on compromised systems by intruders and the interception of communication (usually IRC) relayed through Honeynets. While there is case law about the loss of the right of privacy in storing files on a stolen computer, or one that an intruder has compromised and is using without the owner's authorization, there is less case law surrounding interception of communication that is relayed through a compromised host. Privacy laws exist in the form of state statutes and federal statutes. State statutes may supersede, or may be superseded by, the federal ones.

At the federal level, the two main statutes concerning communications privacy are the Electronic Communication Privacy Act (18 USC 2701-11), and federal Wiretap Statute (Title III, 18 USC 2510-22).

And don't forget that other countries may have similar privacy laws that must be considered if you are implementing honeypots outside the U.S.

The Honeynet Project is attempting to determine what issues exists and how they apply to most organizations today. Until they can establish the legal issues involved, organizations are recommended to ***review all legal issues with their own legal counsel before proceeding***.

Conclusion

A honeypot are just a tool. How you use that tool is up to you. There are a variety of honeypot options, each having different value to organizations. We have categorized two types of honeypots, production and research. Production honeypots help reduce risk in an organization. While they do little for prevention, they can greatly contribute to detection or reaction. Research honeypots are different in that they are not used to protect a specific organization. Instead they are used as a research tool to study and identify the threats in the Internet community. Regardless of what type of honeypot you use, keep in mind the 'level of interaction'. This means that the more your honeypot can do and the more you can learn from it, the more risk that potentially exists. You will have to determine what is the best relationship of risk to capabilities that exist for you. Honeypots will not solve an organization's security problems. Only best practices can do that. However, honeypots may be a tool to help contribute to those best practices. For additional information on honeypot technologies, check out http://www.tracking-hackers.com.

Author's bio

Lance Spitzner is currently an active member of the Honeynet Project. He enjoys learning by blowing up systems in his home lab. Before this, he was an Tanker in the Rapid Deployment Force, where he blew up things of a different nature. You can reach him at lance@honeynet.org .

Whitepapers