Global Information Assurance Certification Paper

Honeypots - Weighing up the Costs and Benefits

GSEC Practical Assignment Version 1.4

Andrew Evans

28 October 2002

Honeypots are an evolving tool in the fight against hackers, allowing insights into their techniques, their motivations, and their targets. Lance Spitzner of the Honeynet Project defines a honeypot as "a security resource whose value lies in being probed, attacked, or compromised" (Spitzner, 2002). Honeypots are used to collect information on techniques used by blackhat hackers and malicious software, and to provide warning of attack. They can be considered a highly focused version of the traditional network intrusion detection sensor. Network intrusion detection sensors monitor all traffic on a network segment and log traffic that corresponds to signatures of malicious traffic. Honeypots log all activity directed at the honeypot system and since the honeypot has no other purpose, virtually all the activity being logged can be considered worthy of attention.

Honeypot technology has been developing since the early 1990's and is now being considered a valid tool in the security expert's arsenal. This paper discusses the history of honeypot usage and defines the different types of honeypots being used, then discusses the issues affecting an organization considering deploying a honeypot. Honeypot technology is developing fast, and will soon become a widely used and highly useful addition to network intrusion detection systems. Presently most complex honeypot systems are being deployed for research purposes, rather than being used in production systems. This situation will change as honeypot solutions mature and the level of risk and cost involved in honeypot deployment decreases.

The earliest documented instances of honeypot usage occurred during the late 1980s and early 1990s however the term 'honeypot' was not widely used until the late 1990s. The origins of the term date back to Cliff Stoll's paper "Stalking the Wily Hacker" where he mentions 'catching flies with honey' (Stoll 1988, p 494). In August 1986 Cliff Stoll, a systems administrator at the Lawrence Berkeley Laboratory (LBL) in California, discovered the activities of a hacker on the LBL network. After detecting the hacker Cliff and his team set up monitoring systems to record the activity of the hacker and discovered that he was searching for files containing keywords such as nuclear, NORAD and SDI. Eventually they set up a trap in the form of text files containing fake information on the Strategic Defence Initiative (SDI, also known as the 'Star Wars' project) to keep the hacker online long enough to trace him back to his location. After an investigation by American and German law enforcement agencies the hacker was found to be a twenty-five year old German named Markus Hess who was selling US government secrets to the KGB in exchange for money and cocaine (Hafner and Markoff, 1995). Cliff wrote a paper and later a book about his experience called 'The Cuckoo's Egg'. It is worth noting that Cliff Stoll did not create a honeypot in the sense that we know today. He monitored systems that had been compromised and added

information to keep the hacker interested. Similar techniques are often used today when setting up honeypots. Systems that are attracting large numbers of probes, or that have been compromised in the past can be replaced with a honeypot to determine what is being attempted and also to determine how much the attacker knows about the system and how it was compromised. This provides valuable intelligence to help secure vulnerable systems.

In January 1991 Bill Cheswick of AT&T Bell Laboratories added fake services to their Internet gateway in order to monitor hacker attempts to access it. He wrote the paper 'An Evening with Berferd' about his experiences monitoring a hacker he called Berferd (this was the username the hacker attempted to gain access with). He monitored Berferd's attempts to hack into the gateway, responded to the commands he attempted to perform and then constructed an actual honeypot system to further study his activities (Cheswick, 1991).

It is interesting to contrast the techniques used in the earliest honeypots with current honeypot techniques. In the early examples systems and networks were slow and the hackers operated manually (no rootkits, automated scanners or network worms). This allowed system administrators to monitor and restrict the activities of the hackers in real time before they could cause significant damage. In 'An Evening with Berferd', when the system was attacked, Cheswick emailed outputs of commands executed by the hacker to him manually a couple of times a day, obviously this kind of technique would be impossible in today's systems, where real-time processing is expected and systems can be probed and compromised within seconds.

The largest, most significant honeypot operation to date has been the Honeynet project. The Honeynet project is a non-profit research group founded in 1999 to 'learn the tools, tactics and motives of the blackhat community and share these lessons learned' (The Honeynet Project, 2002). The project aimed to remedy the lack of information on the motives and techniques of blackhat hackers. Since its inception the scope of the honeynet project has expanded and they have developed complex honeynet systems that simulate entire production networks and are working on easily deployable, standardized honeynets for general use. When the first phase of the honeynet project was over in 2001 they published a book, 'Know your Enemy' and the 'Know your Enemy' series of papers about their discoveries. Lance Spitzner, founder of the honeynet project, recently published a book on honeypot and honeynet technologies.

Honeypots are commonly split into 3 different complexities. These complexities are defined as low, mid and high involvement, referring to the level of interaction that an attacker is capable of having with the honeypot system (Baumann & Plattner, 2002). The low involvement honeypot is extremely simple; in its most common form it simulates network services by listening on specific ports and capturing any traffic directed at those ports. The low involvement honeypot does not allow for much two-way communication with the attacker so the level of risk is

minimal. The mid-involvement honeypot allows slightly more interaction, providing fake services that realistically simulate the behaviour of the real counterparts. The high involvement honeypot provides a real operating system for an attacker to interact with, allowing an attacker to compromise the system fully. High involvement honeypots have carefully hidden logging systems and firewalls to allow collection and control of data.

As the level of involvement increases the amount of information gathered increases, as does the risk and cost of honeypot deployment. Because low involvement honeypots do not allow any real interaction with an attacker, the risk of honeypot compromise is low, although it must be remembered that the underlying operating system may be vulnerable, systems should be thoroughly secured before running any simulated services. Low involvement honeypots are an excellent introduction to honeypot technology because of their low risk and ease of deployment (Spitzner, 2002).

Mid-involvement honeypots are also fairly low risk, however there is always a risk that there will be a vulnerability in the honeypots fake implementation of a network service that may allow compromise of the entire system. Mid involvement honeypots are very difficult to implement as the fake services must convincingly imitate the functionally and vulnerability of real services that may usually be run on a completely different operating system to that which the honeypot is running on (Baumann & Plattner, 2002). Mid-involvement honeypots are the area of greatest development and growth possibilities in honeypot technology, when correctly implemented, they represent the best trade-offs between risk and usefulness.

High involvement honeypots are extremely risky, the honeypot system itself is designed to be compromised. They must be surrounded by data control and collection mechanisms to ensure reliable information gathering, and to prevent the honeypot system being used as a launch platform for attacks on other external systems or on the internal network. These data control systems are usually implemented with a firewall that either restricts the number of outgoing connections or examines the signatures of outgoing connections and blocks those corresponding to attacks (The Honeynet Project, 2002). High involvement honeypots require significant resources to administer, they must be monitored at all times and the logging and data collection systems must be kept secret from an attacker who has compromised the honeypot system. High involvement honeypots are mainly used in research, to determine attacker motives and methods.

So, why are honeypots used? Roesch and Spitzner (2002) define two main reasons to deploy a honeypot within an organization. These reasons are research and intrusion detection (intrusion detection honeypots are usually referred to as production honeypots). Research honeypots such as those used by the honeynet project are set up to collect information and are usually isolated

from any production networks. Production honeypots are set up to alert systems administrators to compromises and provide information on the mechanisms hackers use to gain access the organization's systems.

Research honeypots are mostly used by educational institutions, computer security agencies, and government and military organizations. They are deployed to gain information on the latest vulnerabilities, worms and viruses. They are also used to gain insight into the activities of blackhat hackers, worms and network scanners (The Honeynet Project, 2002). Research honeypots require considerable resources to set up and monitor. They are normally set up to capture complex attacks and to fool experienced hackers into believing they are breaking into a production system. The best method of doing this is to use real systems running services and operating systems with real vulnerabilities. The system is then surrounded with multiple mechanisms for logging and traffic monitoring, as well as altering the logging systems to send all data to a remote log server while making it appear that the logs are stored locally (Spitzner, 2002). Research honeypots are designed to be fully compromised, that is, to allow a hacker to take complete control of the honeypot system. For this reason, research honeypots must be guarded with data control devices such as firewalls, to reduce the risk of the honeypot being used to launch attacks on other systems.

The extension to the research honeypot is the honeynet, a network of honeypots that simulate an entire organization including database, mail, web and DNS servers. Virtual honeynets have been developed that implement an entire honeynet on a single system, this vastly reduces the complexities of the data collection and control systems and makes honeynet deployment much easier than when using multiple systems (The Honeynet Project, 2002). The latest iteration of the research honeynet is the wireless honeypot set up by researchers at the Science Applications International Corporation in June 2002. The purpose of this project is to monitor what purposes hackers use wireless networks for and to determine the proportion of users who just want to use the internet and those who use the honeypot for unlawful means (Poulsen, 2002).

There are alternatives methods to research honeypots that are being used to gain information on the techniques used by hackers. A number of websites have sprung up over the past few years offering hacking competitions where points are awarded for skill and documentation and the level of access gained to the target system. These competitions have the advantage that usually the hackers are required to identify themselves and fully document the techniques used to compromise the system, therefore the data collection and control mechanisms are not required to be as sophisticated as those needed when utilising a research honeypot. However hacking competitions have the distinct disadvantage in that they are artificial situations and the only useful information collected is technical in nature. Research honeypot administrators can often gain valuable information about hacker motives and information sharing mechanisms. This type of

information is much more valuable in the long term than information on technical techniques that will become obsolete.

Production honeypots are set up in or near production networks to allow systems administrators to detect and react to malicious activity on the network. Up until quite recently, there have been few production honeypot systems, as most organizations were unwilling to accept the risk of purposefully having a vulnerable system on the network, nor willing to expend the considerable time and effort associated with setting up and monitoring a honeypot. Over the past few years this situation has begun to change as more commercial honeypot products come on to the market and the existing honeypot solutions become more mature. Production honeypots are useful tools but they must not be used in isolation due to their limited scope of operation. Production honeypots are only useful as a complement to existing intrusion detection systems, and some intrusion detection vendors are now offering solutions that include honeypot sensors that integrate into the system with the traditional 'sniffer' sensors. Also included in the production honeypot category are the 'sticky honeypots' or tarpits. These are products specifically designed to thwart worms and network scanners by holding on indefinitely to any connection attempts they receive (Liston, 2002).

In order to be the most useful, production honeypots should simulate the services running on production systems the organization is using. The reasons for this are twofold, the first is that using or simulating the operating systems that systems administrators and security analysts are familiar with makes analysis of any data collected much easier than with unfamiliar systems. The second advantage lies in the usefulness of the information gathered, for example, a Linux honeypot gives little useful information on the vulnerabilities that are being exploited on a network if Microsoft Windows 2000 is the standard system in use.

For the most part, production honeypots are not required to be as sophisticated as research honeypots as they do not need to accurately simulate real systems, merely to act as 'burglar alarms' and respond when they are probed. The main exception to this rule is when a honeypot is being used to replace a real production system that has been compromised in the past or is being probed. In this case the honeypot should simulate the production system as closely as possible in order to gain valuable information on what is being targeted or how a system was compromised (Roesch & Sptizner, 2002).

Honeypots in general have a number of advantages over standard intrusion detection sensors. The most important advantage is the fact that virtually all the data collected by the honeypot is of use to the administrator as any connection attempts to the honeypot system can be considered suspicious (Spitzner, 2002). This is a stark contrast to most network intrusion detection systems, which monitor all traffic on the network and are subject to high levels of false positives and false negatives in their detection systems and can be bypassed using sophisticated but well documented insertion and evasion techniques (Newsham

& Ptacek, 1998). Network intrusion detection systems are also vulnerable to denial of service if the traffic volume exceeds the ability of the intrusion detection system to collect information. The denial of service condition on intrusion detection systems differ from that of firewalls in that intrusion detection systems inherently 'fail open', that is, failure of the intrusion detection system does not stop traffic flow on the network. Another advantage is the low initial cost of the honeypot, as honeypots do not need to process high levels of network traffic they can also be quite low specification machines, and there are a number of free, open-source honeypot packages available.

Honeypots also have an advantage over traditional intrusion detection systems in that they do not require the same level of intelligence or continual signature updates. The most complex part honeypot deployment is configuring the data collection and control systems surrounding the honeypot. This contrasts to intrusion detection systems, which need to be kept up to date with the signatures of the latest worm or exploit circulating in the wild (much like traditional virus scanning systems), while honeypots merely log all data directed at them.

Honeypots are also touted as having the advantage of using deception as a defence mechanism, that is, hackers can never be sure if they are compromising a real system or wasting their time on a honeypot, and also that the knowledge that they are being monitored may dissuade them from further activities for fear of prosecution. The Deception ToolKit (DTK) honeypot system was designed with this in mind. A DTK host will respond to a probe at port 365 to identify itself as the Deception toolkit, and it is a trivial matter to configure non-honeypot systems to perform the same activity (Piscitello, 2001). This is a form of 'security through obscurity' that was an excellent concept at the time when DTK was first released, as most hacking attempts were still performed manually and took significant effort on the hacker's part. However, the vast majority of present-day hacking attempts either come from network worms or tools such as network scanners and autorooters. These systems can scan and attack thousands of systems in a short space of time and there are many of them operating on the Internet almost constantly (The Honeynet Project, 2001). They are more often than not operated by 'script kiddies', hackers with little real understanding of the techniques they are using. That said, it is not difficult to prevent 'script kiddie' attacks as the represent the lowest skill level amongst hackers, searching for obvious and well-documented security holes. The highly skilled 'überhackers', who write the 'script kiddie' tools, are those who are difficult to protect against. Ironically, these are the people against whom deception is the greatest use, but they are also those who are most likely to be capable of recognizing and avoiding honeypot systems. These types of hackers are few and far between and typically target government, military, or large e-commerce sites. Some analysts argue that for the above reasons deception is not a useful defence mechanism in the modern environment, this is true in most cases, however in combination with other techniques it can be used to great effect to reduce the usefulness of scanners and the speed with which network worms can propagate.

One of the biggest disadvantages of honeypot usage is that they can only detect traffic directed at them. If a hacker becomes aware of the existence of a honeypot and avoids it or happens to attack everything except the honeypot, then it cannot detect the attack. This disadvantage shows the importance of using the honeypot as part of a 'defence in depth' strategy to complement existing intrusion detection sensors that examine all network traffic (Spitzner, 2002). Honeypot technologies also suffer from risk to performance trade-offs. Low-risk, easy to deploy honeypots are easily identified as honeypots and are mainly useful for capturing 'one shot' attacks such as those instigated by worms. Realistic honeypots are usually real systems that introduce high levels of risk to an organization and carry significant management overhead. The middle ground between these extremes will one day result in low-risk, realistic honeypot systems, however honeypot technology is not yet mature enough to be ready for widespread use. Another very important danger of honeypot deployment is the significant risk of attack for other systems belonging to the organization deploying the honeypot. An attacker could bring all their attentions to bear on the IP block containing the honeypot, looking for other systems of interest. If a hacker spends a period of time compromising a honeypot and then discovers that they have wasted their time, they are likely to be annoyed enough to attack other systems belonging to the organization or perhaps even initiate a denial of service attack against the organization. This was shown to be a real threat after the honeynet project published their research, systems belonging to the project came under severe and concerted attack from the blackhat community.

When considering whether to deploy a honeypot, the total cost of ownership must be considered carefully. There are a number of honeypot solutions freely available that can be run on almost any spare computer. Monitoring and administering a honeypot is usually a non-trivial activity, especially given that it adds another set of logs to check and another system to maintain. If an organization is running a high-involvement honeypot, then the administrator responsible for the honeypot needs to be available and able to control the machine twenty-four hours a day, seven days a week to ensure the safety of other systems on the network (Baumann and Plattner, 2002).

Here I will cover some of the more popular ready-to-deploy honeypots that are available. These descriptions are brief as comparisons and reviews of these products can be found in a many other documents written by those who have in-depth experience of their use. Spitzner (2002) provides an in depth analysis of a number of these products that is summarized below.

- BackOfficer Friendly – This is a simple low-involvement honeypot that is easy to deploy and runs on windows systems. It provides emulation of basic services.

- Specter – Another windows based low-involvement honeypot similar to BackOfficer Friendly, it is also capable of performing whois lookups and reverse port scans of attackers.

- Honeyd – An open-source Unix based honeypot released in 2002 that provides greater realism than the windows based honeypots above. Emulates operating systems at the application and IP stack. Uses ARP spoofing to assume the identify of unused IP addresses in the network.

- Mantrap – A high involvement commercial honeypot solution for Solaris. Mantrap creates 'jails' in which applications and services can be installed. The jails are complete, separate operating systems that can be compromised by intruders without compromise of the underlying operating system.

- Deception Toolkit (DTK) – This is a package of Unix perl scripts that emulate various operating system vulnerabilities. DTK also identifies itself when probed on the correct port. The rationale behind this is that it may scare hackers into going elsewhere.

- Labrea Tarpit – This is a 'sticky honeypot' system developed shortly after the first outbreak of 'Code Red'. It is designed to capture, log and hold scanning threads from worms in order to slow the worm's progress and give systems administrators warning of an outbreak on their system.

The future of honeypots is unclear, the technology is not yet mature or widely deployed. One of the most useful techniques that honeypot systems today are beginning to include is ARP spoofing capabilities, allowing them to assume the identity of any unused IP address. This technique works by sniffing for replies to ARP requests, if no reply is broadcast for a particular ARP request the honeypot assumes that the IP address is unused and broadcasts its own ARP reply (Liston, 2002). This technique, in conjunction with 'sticky honeypot' techniques, has the ability to vastly reduce the speed of vulnerability scanners and the impact of network worms. The Raptor firewall product has a similar capability to this in that it takes on the identity of unused IP addresses in the range it is protecting and logs any data sent to those addresses.

The main reason for the difficulty in predicting honeypot technologies of the future is that they are largely dependent on the IT threats we will face in the future. Attack mechanisms adapt to defensive techniques, just as defence mechanisms adapt to attacker techniques. An excellent example of an attack system adapting in response to a honeypot is the open-source Nessus vulnerability scanner's ability to detect IP addresses being spoofed by the Labrea Tarpit 'sticky honeypot' product. As honeypot technologies evolve and become more popular, techniques for fingerprinting and avoiding honeypots will be developed in response. Therefore the most important direction for the future of

honeypots is the realistic simulation of a wide range of operating systems and products at all levels of resolution.

The greatest potential of honeypots lies in their ability to capture 'zero-day' exploits and vulnerabilities, that is, exploits and vulnerabilities that the industry is not yet aware of. The honeynet project illustrated this advantage with their capture of the dtspcd exploit for Unix systems in January 2002 (Spitzner, 2002). The vulnerability in this service was well known, however no exploits for the vulnerability had yet been captured 'in the wild'. One of the greatest disadvantages that the IT security industry faces is its defensive posture and the time it can take for vulnerable organizations to discover the problems and patch their systems. If a malicious individual or organization discovered an unpublished vulnerability in a widely deployed piece of software, they could compromise thousands of critical systems before anyone was aware of the problem, widely deployed honeypots could capture exploit code early and allow quick analysis of the vulnerability. Honeypots could especially be useful in detecting attacks and exploits in situations where patching all vulnerable systems is impossible or impractical. A good example of this is the Abstract Syntax Notation 1 (ASN.1) implementation vulnerabilities discovered within the last year that affected a number of SNMP products. Many other systems built on ASN.1 could be vulnerable, but fixing the problem involves thoroughly testing and altering every system individually (Poulsen, 2002).

There has been significant debate over the legal and ethical issues of deploying honeypots or honeynets. These issues will not be explored in detail here as they are highly dependent on the laws and culture of the country where an organization operates and on the type of organization. Many people incorrectly assume that honeypots are of little use for collection of evidence for legal purposes as the use of a honeypot constitutes entrapment, however this is generally not the case. Entrapment laws usually only cover law enforcement organizations and only where a person is induced to commit a crime they would otherwise not have committed. The real issues affecting organizations deploying honeypots are privacy of information and liability for damage to other organizations should a hacker use their system as a launch pad for other attack. The issues surrounding honeypots have yet to have a major outing in legal circles, so the best option is to consult the organizations legal representative about any plans prior to deploying a honeypot (Spitzner and Roesch, 2002).

It became apparent during this research that a large number of honeypot type solutions are being applied to the fight against spam (unsolicited bulk email or 'junk' email). Many individuals and organizations run relay honeypots (Spitzner, 2002), which appear to be open relays, mail servers that will relay mail from external sources to external addresses, spammers connect to these honeypots to send their email, the honeypot logs the connection, deletes the emails and sends an automated complaint to the spammers internet service provider. Another clever and easy to implement honeypot style solution in use is the fake

web site incorporating thousands of incorrect email addresses (which will be harvested by automated web 'bots and added to bulk email lists, reducing their usefulness). Commercial spam blocking organizations also use honeypot mailboxes with purposely publicized addresses to collect email which they use to develop signatures and heuristics for use in blocking spam.

The real question facing IT security professionals is 'Should we deploy a honeypot within our organization?' In general, I believe that at the current level of honeypot technology the answer should be 'no'. Honeypots are in danger of becoming a fashionable item for organizations to deploy, however the cost/benefit ratio of honeypot deployment is extremely high in most cases. With many IT security professionals finding themselves stretched to the limit as it is, the time required to maintain a honeypot or honeynet within an organization could result in other, more critical activities being neglected. A honeypot should only be deployed when an organization is sure that its intrusion detection systems; configuration management procedures and firewalls are all operating as they should. The organization needs to be sure that the time, effort and increased risk of operating a honeypot are warranted. When considering honeypot usage the first point of reference should be the IT security policies for your organization. Consider whether a honeypot would breach these policies and how to implement a honeypot within existing policy structure. Anyone seriously considering implementing a honeypot should start small, installing a low or mid-involvement honeypot and learning the issues that may affect them. Honeypots pose a significant risk and should be treated in the same manner as vulnerability scanners and password crackers are. In other words, it is advisable to get written permission to operate the honeypot and to carefully define the scope of the operation before going ahead with it. Ensure that the operation of the honeypot does not breach any organizational policies and preferably write a standalone operational policy for the honeypot.

References

Baumann, Reto and Plattner, Christian. "White Paper: Honeypots." 26 Feb. 2002. URL: http://security.rbaumann.net/download/whitepaper.pdf (12 Oct. 2002).

Cheswick, Bill. "An Evening with Berferd In Which a Cracker is Lured, Endured and Studied." 1991. URL: http://www.tracking-hackers.com/papers/berferd.pdf (12 Oct. 2002).

Hafner, Katie and Markoff, John. Cyberpunk. New York: Touchstone, 1995.

Liston, Tom. "LaBrea – The Tarpit." URL: http://www.hackbusters.net/LaBrea/ (10 Oct. 2002).

Newsham, Timothy N and Ptacek, Thomas H. "Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection." January 1998. URL: http://secinf.net/info/ids/idspaper/idspaper.html (15 Oct. 2002).

Piscitello, Dave. "Honeypots: Sweet Idea, Sticky Business" TISC Insight. Volume 3 No.2. 23 Jan. 2001. URL: http://www.tisc2001.com/newsletters/32.html (14 Oct. 2002).

Poulsen, Kevin. "Feds, Industry, Battle Biggest Bug." SecurityFocus Online News. 12 Jun. 2002. URL: http://online.securityfocus.com/news/474 (14 Sep. 2002).

Poulsen, Kevin. "Wi-Fi Honeypots a new hacker trap." SecurityFocus Online News. 29 Jul. 2002. URL: http://online.securityfocus.com/news/552 (10 Sep. 2002).

Roesch, Marty and Spitzner, Lance. "The Value of Honeypots." 10 Oct. 2001. URL: http://online.securityfocus.com/infocus/1492 (5 Oct. 2002).

Spitzner, Lance. "Honeypots – Definition and Value of Honeypots." 17 May 2002. URL: http://www.enteract.com/~lspitz/honeypots.html (2 Oct. 2002).

Spitzner, Lance. Honeypots: Tracking Hackers. Boston: Addison-Wesley Professional, September 2002.

Spitzner, Lance. "Honeypot Solutions." URL: http://www.tracking-hackers.com/solutions/ (21 Oct. 2002).

Stoll, Clifford. "Stalking the Wily Hacker." Communications of the ACM. Vol. 31 No. 5 (1988): 484-500.

The Honeynet Project. Know Your Enemy. Boston: Addison-Wesley, October 2001.

The Honeynet Project. "About the Project." URL: http://project.honeynet.org/misc/project.html (17 Oct. 2002).

The Honeynet Project. "Know Your Enemy: Defining Virtual Honeynets." 4 Sep. 2002. URL: http://project.honeynet.org/papers/virtual/ (10 Oct 2002).

The Honeynet Project. "Know Your Enemy: Honeynets." 8 Sep. 2002. URL: http://project.honeynet.org/papers/honeynet/ (22 Sep. 2002).