Sweeter Honeynets

Kenneth Patton

December 2005

Abstract

The honeynet is a new technology used in the field of computer security for researching the actions of hackers. While honeynets have the potential to give us great insight into the hacker world, recent studies have shown that the rate of data collection by honeynets is far from optimal. This paper first discusses the motivation for using honeynets to track hacker's activities as well as a brief background on honeynets and the various types of hackers in the world. Then, solutions to the problem of increasing honeynet traffic are presented. The primary solution that the author develops is using a webserver as bait, and the steps needed to implement this approach are discussed in detail. Two additional methods are also presented as alternatives - advertising through hacker chatrooms and online contests. The author concludes that all three of these methods are feasible and intends to implement them for experimental confirmation.

1 Introduction

With computer systems playing a larger role in society today, computer security is now more important than ever to protect the confidentiality, integrity, and availability of digital information. Computer systems are also becoming increasingly complex, making it difficult to insure system security when hundreds of applications are run on a regular basis with just as many running in the background continuously. In addition, application developers typically possess a release and patch mentality in order to minimize the time to market, increasing the number of software bugs that external computer crackers can use to compromise a host system.

Due to the prevalence of hackers on the internet today, we recognize the need to research the methods that hackers use to compromise target systems. There are many different techniques employed by hackers to compromise external computers; these range from code analysis and the manual design of tools to the less sophisticated downloading of automated scripts from the internet. Ideally we would like to obtain information about all the different types of attacks that hackers employ, although the less sophisticated attacks are generally more prevalent.

In order to track the actions of hackers, we need a strategy for allowing hackers to do their work while they are unknowingly observed. One tool that is used to facilitate this is known as a honeynet. In a honeynet, a single secure computer monitors a group of insecure "bait" computers that are waiting to be compromised by external hackers. While still a relatively young technology, honeynets help facilitate research into computer crimes because they present hackers with an otherwise undisturbed environment, which makes it simple to identify what traffic and actions on a computer in the honeynet are due to hackers.

There are typically two types of honeynets in use today: production honeynets and research honeynets. Production honeynets are simple honeynets used to capture limited amounts of information, often employed by companies to help protect more valuable systems on a net-

Proceedings of the Class of 2006 Senior Conference, pages 45–52,
Computer Science Department, Swarthmore College ©2005

work. Research honeynets are more complex entities designed to capture as much information as possible about the behavior of intruders. Research honeynets are typically not designed to protect other systems on the network in the short-run, but ideally benefit systems in the future through analyzing the techniques that hackers use to compromise typical machines. All traffic on research honeynets is known to be intrusive in nature because they have no other intended purpose, which makes it easier to analyze a hacker's behavior.

However, as research honeynets are typically unadvertised, they attract relatively low amounts of traffic. In a recent study [1], the average amount of time it took an unpatched Linux system connected to the internet to become compromised was approximately 3 months. While data collected from individual break-ins is certainly valuable, with such sparse occurrences it is questionable whether this is the best method for collecting data. Instead, by making the honeypots more visible through actively advertising them, we can draw more hacker activity at the cost of additional legitimate traffic. As an example, by placing a webserver on a honeypot and designing a simple but enticing website that draws a small amount of web traffic, we present a bigger target for typical hackers than an anonymous machine on the network. Unfortunately this has the drawback that not all of the traffic on the machine will be illicit, but since we know exactly what traffic to expect it should not be difficult to filter out attacks on the machine. Standard web browsers that request valid pages of the website will not be considered attack traffic, but web requests for invalid pages and non-http traffic will be considered attack traffic.

2 Background

2.1 Hackers

There are a number of different types of hackers, each with different motivations and methods of attacking a remote computer. We classify as hackers individuals who, through direct or indirect action, causes a machine to behave in a manner other than intended by the owner. Often this results in the hacker gaining control over the system, but we still classify individuals who make the machine behave abnormally but do not gain control of the system as hackers (for example, due to DDoS attacks). Here we try to classify the different types of hackers that might typically be encountered by a system on the internet and their motives. A more in-depth classification of hackers is presented by Marc Rogers in [5].

The Accidental Hacker

> An accidental hacker is a user who, without previous intent, unknowingly compromises or disrupts the normal behavior of a supposedly secure computer. The user may realize the result of their actions after the fact but generally will not try to exploit the vulnerability that they found. Obviously such a user has no prior motives, which makes it difficult to attract these accidental hackers. Generally if a system vulnerability can be taken advantage of accidentally, it is a serious threat to the security of the computer and be prone to intentional exploitation by less scrupulous hackers. Since break-ins due to accidental hackers are often due to glaring security holes and occur sporadically, they have little research value when focusing on typical hacker threats.

Worm and Virus Creators

> In general virus and worm designers do not directly attempt to compromise particular machines on the internet, but through their actions they indirectly account for a portion of system break-ins. Their motives are often simply entertainment, but occasionally they write viruses for a purpose - usually with motives similar to those of blackhats. However, the automated nature of viruses and

worms cause them to attempt to compromise machines in the same way every time, making their actions very predictable.

"Script Kiddie"

"Script Kiddies" are relatively unskilled hackers that use automated tools downloaded from the internet in order to attempt to break into machines. These are the most prevalent type of intentional hackers, but generally their actions are easy to reproduce and identify. "Script Kiddie" motives typically range from simply the excitement of doing something illegal to collecting botnets for DDoS attacks and harvesting credit card information.

Blackhat / Cracker

Blackhats, often called crackers, are experienced hackers that are typically motivated to break into a protected computer system for personal benefit such as money or access to sensitive data. As the most advanced type of hacker, they generally have detailed knowledge about system exploits and are able to carefully take advantage of those exploits in order to gain full control of a system. Blackhats motives can range from the thrill obtained due to the challenge of hacking a machine to disgruntled employees trying to get back at an employer.

Whitehat

Whitehats are similar to blackhats, but differ in their motives for breaking into a machine; while blackhats compromise machines for personal benefit, whitehats claim to be "ethical" and break into machines in order to help make computer systems more secure. The difference between whitehats and blackhats can be narrow at times, with whitehats on occasion breaking into machines in order to investigate blackhats, but generally whitehats will leave less of a trace on a compromised machine than a blackhat would. Whitehats are often employed by security companies and rarely act on their own to compromise random hosts on the internet.

From a computer security standpoint, the most valuable information we could obtain would be the strategies used by blackhats / whitehats to break into systems, followed by information about typical "Script Kiddie" exploits and toolkits, and lastly how viruses and worms penetrate machines. Unfortunately, the traffic on an average machine connected to the internet will generally find more break in attempts in the reverse order; worms are likely to generate the most, albeit relatively simple, "attack" traffic, while "script kiddies" generally make up a much larger percentage of attempted break ins than blackhats. Regardless, we need a tool to investigate the methods that these different sources employ to compromise machines: a honeynet.

2.2 Honeynets

A honeynet is a collection of computers whose purpose is to track everything that occurs on designated "bait" computers, the honeypots. Honeypots are not used for any particular function on the network, but rather exist solely to be broken into by external hackers. The goal of a honeynet is to research the actions of hackers, which is best accomplished on honeypots since they contain essentially only attack traffic with little background noise.

In a basic honeynet setup as seen in figure 2.1, all traffic passing between local computers and external computers on the internet must pass through a honeywall. The job of the honeywall is similar to a firewall, with advanced filtering and logging capabilities. The honeywall may be set up in one of two configurations: as a standard network bridge or using Network Address Translation.

For the purposes of strictly research in a non-production environment, the honeywall is best set up as a network bridge because it allows for multiple honeypots and the honeypots to appear

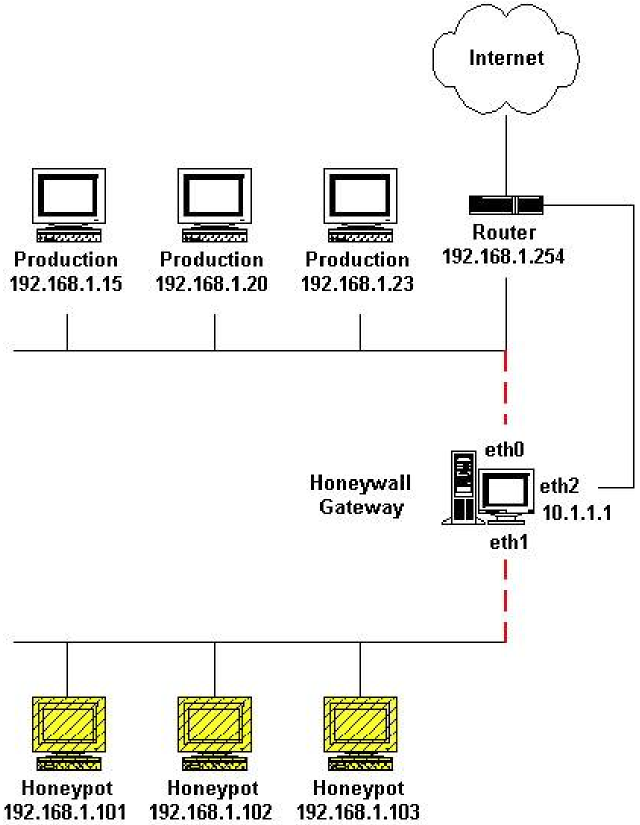

Figure 1: Standard Honeynet Setup

as standard machines on the network to hackers. In an environment where the honeynet is set up as a decoy to distract hackers from other critical network resources, NAT can be useful because it allows the honeypot to camouflage the production machines. For example, a company could put a webserver behind the honeywall and have all communication on port 80 forwarded to it while all other is forwarded to a honeypot. Overall it looks like the honeypot and the webserver are the same machine, and a hacker will be distracted trying to break into the honeypot while the webserver is fully secure. This allows the company to analyze the hacker's actions in addition to quickly highlighting attempts to break into the webserver.

In general the honeywall will log all packets that pass through it in order to correlate specific streams of packets as an attack. In addition, the honeywall will often receive data detailing what users do on the local honeypots. Most honeywalls will also have the ability to limit outbound connections from the honeypots in order to prevent a hacker from exploiting a honeypot for denial of service attacks. Combining these features, an administrator is often able to determine the precise method that the hacker employed in order to compromise a honeypot as well as identify the intentions of a hacker based on his or her actions on the honeypot.

Although research honeynets provide an excellent means to track hackers, they suffer from a major drawback: they are essentially passive devices, waiting for hackers to stumble across the honeypots. In a study study done by The Honeynet Project [1], 19 unpatched Linux systems had an average life expectancy of 3 months before getting compromised. While this may be a good phenomena for the average computer user, it becomes difficult to research the latest hacker techniques if attacks occur infrequently. In the same study [1] of Windows honeypots they found that unpatched Windows machines generally have life expectancies measured in hours. However, the short life expectancy of Windows machines was due primarily to worms rather than active hackers, which makes the attacks less valuable as research material.

The results of these previous studies are for research honeypots and it is expected that production honeypots recieve significantly more traffic. While we could consider this as one strategy to attract more activity on honeypots, we recognize that usually a company running production honeypots in addition to other production machines will focus more heavily on security than research capabilities. For example, in the event that a company needs to decide whether to sacrifice forensics in order to get a production machine back up and running, we assume that most likely the company will choose to sacrifice the data in order to revive a machine that may be crucial to their business. For this reason we focus mainly on standalone research honeynets and what can be done to make them more attractive.

Overall our goals for research honeynets are

1. Increase number of attacks per unit time
2. Increase overall "quality" of attacks
3. Increase variety of attacks
4. Minimize non-attack background traffic
5. Minimize overall cost of deployment

In the following sections we present strategies that attempt to accomplish these goals.

2.3 Related Research

Lance Spitzner developed the idea of honeytokens [2] that are similar to the concept of honeypots but on a smaller scale. Honeytokens are files on a system that are not meant to be accessed by anyone, but are developed to stand out if a user is browsing the system looking for interesting files. Accessing a honeytoken triggers an alarm in the system that notifies the supervisor about the illicit behavior, and is meant to be used as a first line of defense against improper insider activity. In this manner honeytokens are meant to be components of an otherwise functional production system, and differ from honeypots that are meant as stand alone systems.

Another similar idea is the Catering Framework [4] designed by Xuxian Jiang and Dongyan Xu that "caters" to the desires of hackers by analyzing network traffic. This framework is designed to dynamically modify honeypots to keep services open that hackers are more likely to use; the Catering Framework makes this distinction by profiling the random network traffic received by outside sources. Any random traffic that is received is assumed to be illicit, and by keeping track of the most prevalent types of network traffic the framework determines what services are best to run on honeypots. While the Catering Framework presents a good strategy for holding onto hackers that find the honeypot, it suffers from the fact that it fails to draw in additional hackers that did not randomly encounter the honeypot in the first place.

Maximillian Dornseif and Sascha May examined models of the cost versus benefit of running a honeynet [3] and found that the cost of running a honeynet can be modeled as $C(t) = S + Mt$ while the utility gained from the honeynet can be expressed as $U(t) = PtM/I$, where S is the initial startup cost, M is the maintenance cost per unit time, P is the amount of utility gained per attack, and I is a factor by which higher investments in the maintenance cost influence the chance of being attacked. Under their model we would like to minimize the overall cost while maximizing the utility gained from the honeynet. We see that in order to do this we would like to minimize S in relation to M. However, their model does not account for methods that artificially influence the chance of being attacked, which would allow us to increase U(t) while not significantly affecting C(t).

3 Improving Honeynets

3.1 Running a Webserver

The easiest method to increase traffic and the visibility of a machine is to setup a webserver that the outside world can visit. Then by registering a domain name and listing the machine with search engines we can increase the overall traffic on the machine. At first this may result in just an increase in benign traffic, but in the long run it provides hackers with another method through which they can encounter the honeypot.

Here we present a brief description of the steps necessary to setup a more enticing honeypot with a webserver

1. Obtain webserver for operating system of choice

   The Apache HTTP Server is one of the most common webservers on the internet due to its powerful feature set, simple installation, and ease to maintain. In addition the Apache HTTP Server is freely available for almost all commonly used operating systems - setup is simple on Windows and most UNIX variants, including Linux, Mac OS and the BSDs. Nearly all package based distributions provide precompiled versions of the HTTP server, but complete sources are available at http://httpd.apache.org/download.cgi and can be configured and installed fairly simply on any machines with an ANSI-C compiler.

   Several versions of Windows and Mac OS X also come with built in webservers for those adverse to the thought of installing the Apache HTTP Server. Mac OS X's built in webserver is apache with a more user friendly interface. Window's built in webserver, Internet Information Services (IIS), is fairly simple to set up and consists of adding virtual directories to the default website through Administrative Tools / Internet Information Services.

2. Building the Website

   The next crucial step in attracting hackers is to design a site that has a tendency to draw illicit behavior. While designing an interesting site that brings in a lot of traffic from average internet users is appealing, we do not gain anything from users who visit the site for legitimate purposes. For this reason it is important to design a site that standard internet users have no interest in, but that a hacker would come across when looking for targets.

   Money is a typical motivating factor for blackhats, and so we recognize that hackers will be more likely to attempt to break into a honeypot if they believe it will result in monetary gain. An easy method to present this illusion is by designing a site that mimics a financial institution, but with relatively relaxed security measures on the site. If the site is visually well designed and attractive but lacks even basic security measures such as SSL security that many users may not notice (although ideally a blackhat would), it gives the appearance of a relatively inept IT department - an ideal target for a blackhat looking to make money through illicit means.

3. Registering a Domain

   Obtaining a domain name is the next step in making a site appear legitimate. There are many different sites on the internet that allow you to register a domain name. After registering the domain name you also need to find a service willing to host your DNS records for you, although, many sites provide a primary DNS server in a package with registering for the domain name. One site that we recommend which provides these services is Yahoo! Domains at http://domains.yahoo.com - they provide a number of tools and DNS servers and only cost 2.99 per year for the domain name.

4. Listing with Search Engines

   Unforunately, many search engines such as google no longer allow you to list your site manually anymore. Instead crawlers automatically prowl the internet for new sites that are linked in from existing ones. Danny Sullivan discusses tips for making websites more visible to search engines in [6]. Some of the biggest tips are to make sure your website is listed in the major website directories and to carefully craft the title / content of the page with regard to certain search terms. The primary directory service that many major search engines use is the open directory project at dmoz.org, and submitting websites is simple using the "suggest URL" feature.

3.2 Hacker chatrooms

Another strategy to increase hacker traffic on honeypots is to go straight to the source - finding the hackers themselves and convincing them to attack your machine. On any decently sized IRC server, for example Undernet, the list of most popular channels includes a number of hacker chatrooms such as #cc-web where the operator advertises rooted machines, credit card numbers, senders, mailers, and hacked ebay accounts. However, as one would not like to reveal to potential hackers the true nature of the honeypots, it is difficult to find ways to coax hackers into attacking your machine without suspicion.

The simplest strategy available is simply to tell the chatrooms that you had a personal machine that you wanted to check the security of - and that you would be glad to have anybody attempt to break into it. Often hackers are motivated by a need for personal acknowledgment, and by presenting a challenge to the hacker you will be acknowledging the hacker's skills if he or she is successful at breaking in to your machine.

Other strategies range from advertising the machine as a valuable box that likely has credit card numbers on it (possibly in conjunction with a webserver set up on the machine) to inciting hackers through insults in order to try and get them to attack the machine in retaliation.

3.3 Obvious Advertising - contests

A third method for developing the traffic on a honeypot is through active advertising such as that done by http://www.rootthisbox.org/. Using an ingenious method for attracting traffic, http://www.rootthisbox.org/ relies on attracting hackers to the site through a challenge - to see who is the best hacker. Machines are submitted to http://www.rootthisbox.org and the goal of a number of different teams is to gain root control of as many machines as possible and hold onto that control for as long as they can. Throughout this process, each team is competing against everyone else in what resembles a virtual game. Setting a machine up to act as a honeypot and submitting it to the contest would certainly generate a large amount of research data and benefit the security community greatly. One of the drawbacks of this approach, however, is it relies on the ego and competitive nature of hackers who are trying to show off their skills. I believe that this strategy will attract more "script kiddies" than any other type of hacker because while appealing, blackhats have better things to do with their time than participate in this kind of game. Still, this type of experiment would nonetheless produce interesting and valuable results.

4 Conclusions

Having highlighted a major drawback of honeypots I believe this is an area of important research if we want to fully track the evolving attacks that hackers employ. I have illustrated a number of different methods that one could use to cope with this problem, but there are certainly more methods out there. In future experiments

I would ideally like to test some of these methods, in particular that of setting up an active webserver on a honeypot. Honeynets are still a young technology and as such there are many different experiments that can be done with them. By expanding the rate at which data acquisition is performed on honeynets we essentially expedite all future experiments, which is why I believe this is an important first step in the field of hacker analysis.

References

[1] The Honeynet Project. *Know Your Enemy - Trend Analysis* http://project.honeynet.org/papers/trends/life-linux.pdf.

[2] Lance Spitner. *Honeypots: Catching the Insider Threat* Annual Computer Science Security Applications Conference, December 2003.

[3] Maximillian Dornseif, Sascha May. *Modeling the costs and benefits of Honeynets* The Third Annual Workshop on Economics and Information Security, May 2004.

[4] Xuxian Jiang, Dongyan Xu. *BAIT-TRAP: a Catering Honeypot Framework* http://www.cs.purdue.edu/homes/jiangx/collapsar/publications/BaitTrap.pdf.

[5] Marc Rogers. *A New Hacker Taxonomy* http://homes.cerias.purdue.edu/ mkr/hacker.doc, 2000.

[6] Danny Sullivan. *Search Engine Placement Tips* http://searchenginewatch.com/webmasters/article.php/2168021, 2000.