Analyzing the Process of Installing Rogue Software

Robin Berthier*, Jorge Arjona‡*, and Michel Cukier*

*Center for Risk and Reliability
Department of Mechanical Engineering
University of Maryland, College Park, MD, USA
{robinb, jorarar, mcukier}@umd.edu

‡Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politecnica of Valencia, Spain
jorarar@teleco.upv.es

Abstract

This practical experience report presents the results of an experiment aimed at understanding the sequence of malicious actions following a remote compromise. The type of rogue software installed during attacks was used to classify and understand sequences of malicious actions. For this experiment, we used four Linux target computers running SSH with simple passwords. During the eight-month data collection period, we recorded a total of 1,171 attack sessions. In these sessions, attackers typed a total of 20,335 commands that we categorized into 24 specific actions. These actions were analyzed based on the type of rogue software installed by attackers.

1. Introduction

Most security analysis experiments focus on methods for keeping attackers out of target systems, but do little to address their behavior after a remote compromise. In this experiment, we focused exclusively on post-compromise attacker behavior. More specifically, we focused on the attackers' actions that led to the installation of rogue software.

We used four Linux target computers running SSH with simple passwords. During the eight-month data collection period, we recorded 1,171 attack sessions. In these sessions, attackers typed 20,335 commands that we categorized in 24 specific actions. These actions were analyzed to determine attackers' patterns and understand their motivation. Finally, we analyzed and classified the rogue software downloaded, installed and run on the target computers.

The paper is structured as follows. Section 2 describes the experimental setup used to collect the attack data. In Section 3, we discuss the analyses on the attack data, we present the attackers' action, and the rogue software that was downloaded, installed and run on the target computers. Section 4 discusses the related work. Finally, we present conclusions in Section 5.

2. Experimental Setup

To collect our data, we used a set of four high-interaction Linux honeypot computers on a sealed-off network that allowed all incoming connections, but severely limited outgoing connections to minimize damage by the attackers. The IP addresses of these honeypots were never advertised. The four honeypots all ran on an identical Linux disk image: a slimmed-down installation of Fedora Core 3. To monitor attacker activity, we used the following tools: a modified OpenSSH server to collect password attempts, syslog-ng [2] to remotely log important system events, including logins and password changes, strace [3] to record all system calls made by incoming SSH connections, and the Honeynet Project's Sebek tool [4] to secretly collect all keystrokes on incoming SSH connections. Each honeypot had one privileged root account plus five non-privileged user accounts. For additional details on the experimental setup, refer to [1].

3. Analyzing the Attacks

The first step of our analysis was to divide the information we collected into different attacks and sessions. We assumed that each IP address over a time window of 24 hours corresponded to an attacker. We defined an attack to be all interactions between an attacker and a honeypot. We then defined a session as a single SSH interaction between an attacker and a honeypot. We extracted a total of 1,171 different sessions. Differentiating attacks from sessions is important because we discovered that some attackers

978-1-4244-4421-2/09/$25.00 ©2009 IEEE

used several SSH sessions in parallel to perform their attack. Our results showed that 47% of attackers used a single session, and the remaining 53% of attackers used at least two parallel sessions. Table 1 provides the number of sessions associated to the four honeypots. After several months of collecting data, we changed the set of honeypot IP addresses (March 28 - August 18, 2007 we used a one set of IP addresses and August 19 - December 4, 2007 we used a second set). The change was a request from the network administrator on which our testbed was located.

Table 1: Number of sessions collected by each honeypot

Hosts	No. of Sessions (First set of IPs)	No. of Sessions (Second set of IPs)
Honeypot A	91	231
Honeypot B	155	262
Honeypot C	122	127
Honeypot D	101	81

3.1. Analyzing Attackers' Actions

In this section, we refined the analysis on the different sessions by introducing the concept of actions. An action is defined as a set of commands run by an attacker to reach a goal, for example, gathering information or installing software. Table 2 provides the detail of each action with examples of commands and overall statistics. We can see from Table 2 that typical attack sessions consist of three steps:

- First, attackers check the system configuration, by reviewing network settings, user accounts, processes running and software installed;
- Second, attackers change the system configuration by adding user accounts, modifying passwords and altering software settings; and
- Third, attackers download, unpack, install and run rogue software.

From Table 2, typical actions appear to include getting information related to users (present in 66% of the sessions), getting information related to the system (39%), and getting information related other parts (38%). Passwords were changed in 37% of the sessions and system files were modified in 34%. We observed that files were downloaded and unpacked in more than 41% of the sessions. A total of 2,437 commands (12% of the 20,335 commands) could not be identified. Most of these unidentified commands were discovered to be typos that attackers incorrectly typed.

Besides these typical sessions, some attackers took other interesting actions such as trying to hide their intrusion. In 26% of the sessions, attackers deleted log entries and the command history. A few attackers restored deleted files to make sure their intrusion would remain undetected. Another practice to hide intrusions was to obscure the name of the folder in which the rogue software was installed. The most popular folder names were: " ", "...", ".. ", ". " and "..". Finding one of these folders is an unmistakable sign that an attack occurred. We also found that some attackers were compulsively using the Unix command "w" or "who" to make sure no other legitimate user could connect while they were attacking the system.

Most of the attack sessions were short and lasted less than one minute. Some lasted a few hours because attackers would sometimes launch a rogue software such as a network sniffer and would return to check the output. The distribution of session durations is provided in Figure 1.

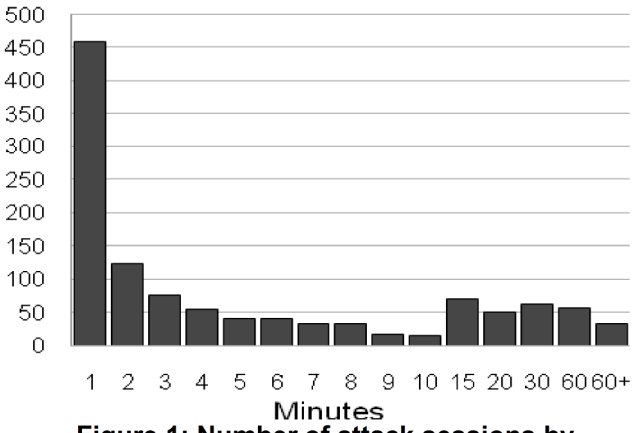


Figure 1: Number of attack sessions by duration

3.2. Analyzing Rogue Software

The honeypots were instrumented to record all network traffic. We analyzed the incoming traffic using tcpdump [5] and chaosreader [6] to extract files downloaded on the honeypots. We extracted 250 files from 379 attack sessions where attackers downloaded files. We could not extract files in the other 129 sessions because the attacker file download had failed mainly due to incorrect URLs or missing files. This result let us believe that attackers were neither skilled nor organized. Here is an example of an attack session where the attackers did not succeed downloading a remote file. We found that the attacker tried to download several files, was unsuccessful and then left.

```
9:53:31 w
9:53:33 uname -a
9:53:34 uptime
9:53:45 cat /cpuproc/cpuinfo
9:53:48 cd /tmp
9:53:49 ls -al
9:53:53 mkcd #
9:53:54 ls -al
```

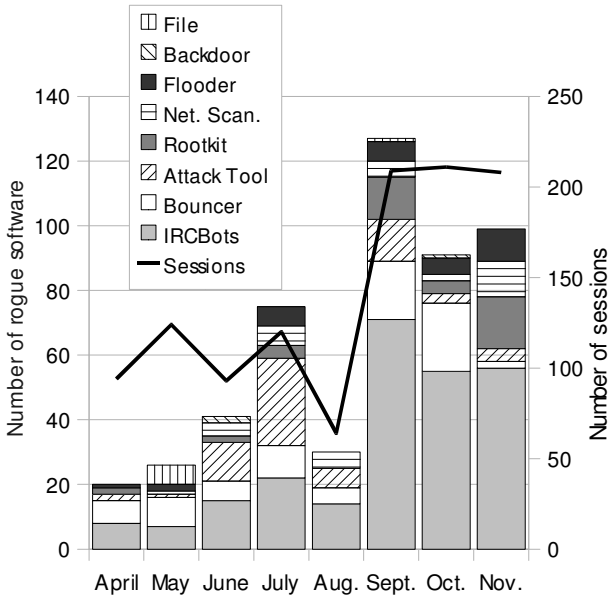
```
9:53:58 mkdir    " "
9:54:00 cd   " "
9:54:03 ls wget
9:54:07 /sbin/ifconfig
9:54:31 wget
9:55:09 ftp
9:55:47 o
9:55:54 ftplynx
        www.almerimur.com/capella/linux.tar.gz
9:56:08 wget
  www.geocities.com/capella99_2000/linux.tar.gz
9:56:18 ftp 207.150.179.22
9:57:18 wget
         www.almerimur.com/capella/linux.tar.gz
9:57:26 history -c -d offset
```

To analyze the nature of the files we extracted, we submitted them to the VirusTotal web service [7]. 50% of the files could be identified using VirusTotal. The remaining 50% were identified manually using their source code. Figure 2 provides the categories and the volume of files downloaded by attackers during the data collection period. Figure 2 indicates that the popularity of our honeypots grew over time. The gap in August 2007 is due to the change in IP addresses. Figure 2 also shows that the main interest of attackers was to install "**IRCBots**" (most were Mech-based IRCBots [8]). The motivation to deploy IRCBots is to make the compromised computer part of a botnet. An army of several thousand bots can be turned into profit by attackers who sell computer resources on the black market [9]. Here is an example of IRCbot based attack session:

```
 13:46:06 cd /var/tmp
 13:46:11 wget
http://www.shaq.profesor.info/like/error.tar.gz
 13:46:16 tar xzvf error.tar.gz
 13:46:18 rm -rf error.tar.gz
 13:46:19 cd error
 13:46:21 chmod +x*
 13:46:22 ./x
 13:46:28 exit
```

The second most popular software installed was "**Bouncers**", which are programs used to relay network connections, much like a proxy. Attackers often use this type of software to hide their source IP address and hostname. Most of the bouncers we collected were based on Psybnc [10]. Under "**Attack Tools**", we grouped various programs used by attackers to compromise the computer. These tools included non-malicious software, such as rogue SSH servers, and malicious programs, such as john-the-ripper [11], log cleaners, process hiders and network sniffers. In this category, we included rogue web servers installed by attackers to setup phishing websites [12]. The "**Rootkits**" type included system exploits and rogue binaries used to gain root privileges on the compromised computer. "**Network Scanner**" contained software to automatically probe for listening SSH servers or to perform port scans. The "**Flooder**" type consisted of network applications built to launch denial-of-service attacks [13] against a given target. "**Backdoor**" included programs to stealthily and remotely control the compromise computer. "**Files**" contained non-malicious files, such as movie trailers, computer drivers or even Windows update patches. Attackers who downloaded these files were simply using the compromised computer for storage. We even found an attacker who attempted to turn the compromised computer into a CounterStrike game server.

Figure 2: Number of files downloaded by attackers over time, sorted by type



The final step in our analysis was to label each session with the type of file downloaded. We discovered that in 58 out of the 379 sessions, attackers downloaded more than one type of file. This number increases to 158 if we aggregate files per attack instead of session. This is because attackers often used auxiliary sessions to download other files or performed other tasks in parallel. Main and auxiliary sessions were linked using the source IP and a time window of 24 hours.

Table 3 indicates that for all categories of rogue software, approximately half of the attackers used auxiliary sessions. These auxiliary sessions had, on average, fewer command lines compared to main sessions and tended to be shorter, except for the "**Flooder**" and "**File**" types. We investigated the delay between the first command typed by attackers and the time when they deployed their rogue software, in order to get insight on the detection time required by a security tool before damage occurred. On average, for main sessions, it took between 2 minutes 14 seconds and 30 minutes for attackers to deploy their rogue software.

Table 2: Grouping of attacker's actions and statistics on the number of commands captured

Group	Action	Commands	Commands		Sessions	
			Number	Percentage	Number	Percentage
Get information	get information related to users	`w, whoami, who, last, id, finger, lastlog`	1,464	7.20%	768	65.64%
	get information related to the system	`uptime, uname, ifconfig, netstat, locate, php -v, hostname, whereis, nmap, cat /etc/<system file>`	894	4.40%	454	38.80%
	get other type of information	`ps -a, ps -ax, history, cat <file>`	876	4.31%	445	38.03%
Change configuration	add one or multiple users and set passwords	`adduser, useradd, passwd`	116	0.57%	48	4.10%
	gain root privileges	`su, sudo`	85	0.42%	40	3.42%
	change the password	`passwd`	569	2.80%	432	36.92%
	modify system files	`cp, mkdir, mv, rm`	1,030	5.07%	400	34.19%
	change system configuration	`path, userdel, iptables, ln, export, chown, chmod, rhsnd`	610	3.00%	236	20.00%
Edit files	read system files	`nano, pico, vi <lastlog, wtmp, bash_history, /etc/passwd>`	139	0.68%	13	1.11%
	read other files	`nano, pico, vi <file>`	39	0.19%	86	7.35%
	edit system files	`nano, pico, vi <file>`	430	2.11%	63	5.38%
Hide intrusion	tamper with system files or user variables to hide traces of the intrusion	`unset <ENV_VAR>, rm [-rf] <file>, export <ENV_VAR>=/dev/null, cat /dev/null > <file>`	859	4.22%	313	26.75%
Restore deleted files	restore deleted files to hide traces	`touch`	95	0.47%	27	2.31%
Kill process	terminate the execution of processes	`kill, cat <filename>.pid, ps`	362	1.78%	109	9.32%
Fetch rogue software	download a file from a remote host and unpack it	`lwp-download, scp, curl, ftp, wget, unzip, tar`	2,339	11.50%	484	41.37%
Deploy rogue software	run a rogue software	`perl, ./<command>`	1,065	5.24%	459	39.23%
	install a rogue software	`./configure, make, make install, ./setup, gcc`	25	0.12%	11	0.94%
Tool command	Unix tool commands	`cd, ls, pwd, clear`	5,700	28.03%	892	76.24%
Other actions	use SSH	`ssh, "yes"`	47	0.23%	23	1.97%
	launch a new console	`sh, bash`	79	0.39%	69	5.90%
	launch a new console	`screen`	68	0.33%	30	2.56%
	failed attempt of getting files using SCP	`winscp unsuccessful attempt`	611	3.00%	8	0.68%
	chat with other users	`wall`	56	0.28%	6	0.51%
Exit	exit the session	`exit`	340	1.67%	291	24.85%
Total		**Identified commands**	**17,898**	**88.02%**	**1171**	
		Unidentified commands	**2,437**	**11.98%**		
		Total commands	**20,335**	**100.00%**		

Table 3: Comparison between attack sequences based on the type of rogue software downloaded

Main rogue software:	**IRCBots**		**Bouncer**		**Attack Tool**		**Rootkit**		**Net. Scanner**		**Flooder**		**Backdoor**		**File**		**None**	
Other rogue software often associated:	Bouncer (12) Rootkit (11) Flooder (11) Net. Scan. (9)		IRCBots (5) Att. Tool (6) Rootkit (4) Backdoor (2)		IRCBots (7) File (4) Bouncer (4) Net. Scan. (4) Rootkit (1)		IRCBots (4) File (4) Net. Scan. (3) Att. Tool (3) File (2)		Rootkit (2) Att. Tool (2) IRCBots (1) Flooder (1)		Rootkit (3) Att. Tool (2) File (1)		Rootkit (1) Att. Tool (1)		File (6)			
Number of attackers:	**83**		**29**		**24**		**22**		**17**		**18**		**4**		**4**		**182**	
Attackers using aux. sessions:	*53*		*15*		*11*		*9*		*9*		*8*		*0*		*1*		*89*	
Type of session:	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*	**main**	*aux.*
Number of sessions:	**144**	*136*	**40**	*48*	**29**	*45*	**24**	*24*	**22**	*17*	**21**	*17*	**7**		**6**	*3*	**262**	*326*
Avg. session duration (s):	**685**	*253*	**761**	*306*	**980**	*252*	**936**	*541*	**1,094**	*192*	**680**	*1,700*	**563**		**742**	*755*	**543**	*854*
Min. session duration (s):	**4**	*0*	**33**	*0*	**88**	*0*	**60**	*0*	**45**	*14*	**19**	*0*	**155**		**366**	*0*	**0**	*0*
Max session duration (s):	**11,470**	*3,339*	**3,645**	*2,611*	**2,750**	*2,664*	**4,183**	*4,582*	**4,512**	*998*	**2,819**	*19,672*	**1,227**		**1,976**	*2,264*	**7,647**	*76,629*
Avg. number of lines:	**28**	*17*	**31**	*15*	**212**	*30*	**41**	*18*	**36**	*10*	**21**	*16*	**28**		**18**	*4*	**15**	*13*
Min. number of lines:	**2**	*1*	**3**	*1*	**15**	*1*	**9**	*1*	**4**	*2*	**3**	*1*	**12**		**6**	*1*	**1**	*1*
Max. number of lines:	**676**	*702*	**93**	*88*	**914**	*859*	**116**	*95*	**261**	*26*	**98**	*59*	**47**		**39**	*9*	**217**	*237*
Avg. delay before exploit. (s):	**1,030**	*204*	**939**	*371*	**715**	*94*	**1,891**	*987*	**2,105**	*74*	**1,119**	*21*	**358**		**96**	*0*	**581**	*432*
Percentages of attackers for each group of actions:																		
Get info.	**87%**	*94%*	**90%**	*100%*	**96%**	*91%*	**91%**	*89%*	**94%**	*100%*	**89%**	*88%*	**100%**		**100%**	*100%*	**80%**	*87%*
Edit files	**17%**	*11%*	**31%**	*33%*	**33%**	*9%*	**18%**	*11%*	**29%**		**17%**	*25%*			**25%**		**18%**	*27%*
Change conf.	**80%**	*87%*	**83%**	*80%*	**88%**	*82%*	**96%**	*67%*	**59%**	*67%*	**72%**	*75%*	**100%**		**75%**	*100%*	**62%**	*81%*
Fetch rogue software	**100%**	*55%*	**100%**	*53%*	**100%**	*82%*	**100%**	*78%*	**100%**	*67%*	**100%**	*38%*	**100%**		**100%**	*100%*	**50%**	*60%*
Deploy rogue software	**76%**	*40%*	**69%**	*33%*	**83%**	*36%*	**82%**	*44%*	**77%**	*44%*	**67%**	*25%*	**25%**		**25%**		**41%**	*51%*
Kill process	**16%**	*21%*	**24%**	*7%*	**17%**	*9%*	**14%**	*11%*	**18%**		**11%**	*25%*					**7%**	*23%*
Hide intrusion	**30%**	*34%*	**45%**	*40%*	**38%**	*46%*	**36%**	*11%*	**29%**	*11%*	**28%**	*38%*	**25%**		**25%**		**26%**	*30%*
Restore deleted files	**1%**		**7%**	*7%*	**4%**		**9%**	*11%*				*13%*	**25%**				**5%**	*5%*
Other actions	**17%**	*13%*	**28%**	*27%*	**33%**	*46%*	**14%**	*33%*	**29%**	*11%*	**6%**		**25%**				**12%**	*20%*
Tool commands	**98%**	*72%*	**97%**	*87%*	**100%**	*82%*	**100%**	*67%*	**94%**	*67%*	**94%**	*75%*	**100%**		**100%**	*100%*	**72%**	*84%*

This delay is on average shorter for auxiliary sessions, and can even reach 0 seconds for the "**Backdoor**" and "**File**" types, because auxiliary sessions were sometimes used only to deploy the rogue software previously downloaded in the main session. This explanation is confirmed if we consider the breakdown of attacker actions per type of session and category of rogue software. We can see that all attackers used a main session to fetch rogue software, while using auxiliary sessions to obtain information about the system or to change program settings. We can also see that for all categories except for "**Rootkit**" and "**Network Scanner**", attackers primarily used auxiliary sessions to hide their intrusion and restore deleted files.

3.3. Discussion about Attackers

The data we analyzed provided evidence that attackers targeting weakly secured SSH servers tend to be low skilled humans. The large number of typos in recorded commands and the timing between commands indicate that attacks are rarely from automated scripts, but from human beings who used interactive terminals. For scalability reasons, automated attacks are predominantly on the Internet. However, our dataset shows that this rule does not apply for the specific service we opened on our honeypots (SSH on port tcp/22). We believe that human attackers use automated

scripts to scan and find SSH servers to compromise. Once successfully logged onto the machine they proceed to manually install rogue software.

Further evidence of the low skill level is found from the relatively low percentage of attackers who attempted to hide their intrusions as well as the large volume of attackers who were not able to complete their attacks. We found a number of attackers did not complete their attacks because some system tools were missing on the honeypot, or because the URL from which they tried to download rogue software was invalid. These findings confirm the conclusions of [14] and [1].

4. Related Work

In [15], the authors performed an in-depth forensic analysis of post-compromise attacker behavior. Their primary focus was on investigating the actions of more sophisticated attackers. The main difference between their project and our experiment was that we focused on a larger set of less sophisticated attackers and gathered aggregate statistics about their actions rather than investigating individual incidents in detail.

In [16], the author described the login attempts on a single honeypot over a 22-day period. A modified SSH server was used to collect password attempts, and most of the article was dedicated to the analysis of these attempts. During a period of seven days where Sebek was also installed on the honeypot, the author recorded one successful login attempt, providing some insight into attacker behavior.

The project which is the most similar to our study is [14], in which the authors collected SSH intrusions during six months from a total of 35 attackers. By comparing IP addresses of attackers with the ones collected using a large distributed low interaction Honeynet, the authors determined that intruders and scanners were two distinct sets of attackers. They also came to the same conclusion that attackers targeting weakly secured SSH servers were low skilled. Our work differs with [14] due to our larger data collection (we collected attacks from 305 distinct attackers), which allowed a more precise quantification of actions performed by attackers and rogue software downloaded.

5. Conclusions

We found that a typical attack session consisted of: 1) checking the system configuration, 2) changing the system configuration, and 3) downloading, installing and running rogue software. In about 25% of the cases, attackers will try to hide their intrusion.

We identified 250 rogue software files of various types. The most popular were IRC bots, bouncers, attack tools, root kits, network scanners, flooders and back door programs. We also found that attackers often launched more than one attack session at a time. We compared the main session with the auxiliary ones. We also found that in 27% of the sessions, attackers did download some software which was never used. This is an indication that we might have not given attackers enough time before redeploying the honeypot.

6. References

[1] Daniel Ramsbrock, Robin Berthier, Michel Cukier, "Profiling Attacker Behavior Following SSH Compromises," dsn, 37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07), pages 119—124, 2007

[2] http://www.balabit.com/network-security/syslog-ng/, 2008

[3] http://strace.sf.net, 2008

[4] http://www.honeynet.org/tools/sebek/, 2008

[5] http://www.tcpdump.org, 2008

[6] http://chaosreader.sourceforge.net, 2008

[7] http://www.virustotal.com, 2008

[8] Mech IRCBot: http://www.energymech.net, 2008

[9] Friess, N. and Aycock, J. “Black Market Botnets”, http://hdl.handle.net/1880/45380, 2007

[10] PsyBNC: http://www.psybnc.at, 2008

[11] John-the-ripper password cracker: http://www.openwall.com/john, 2008

[12] R. Dhamija, JD. Tygar and M. Hearst, “Why phishing works”, in Proceedings of the SIGCHI conference on Human Factors in computing systems, ACM New York, NY, USA, pages 581—590, 2006

[13] D. Moore, C. Shannon, D.J. Brown, G.M. Voelker and S. Savage, “Inferring Internet denial-of-service activity”, ACM Transactions on Computer Systems (TOCS), ACM New York, NY, USA, vol. 24, num. 2, pages 115—139, 2006

[14] E. Alata, V. Nicomette, M. Kaâniche, M. Dacier and M. Herrb, “Lessons learned from the deployment of a high-interaction honeypot”, 6th European Dependable Computing Conference (EDCC'06), pages 39—46, Coimbra, Portugal, October 18-20, 2006

[15] F. Raynal, Y. Berthier, P. Biondi, and D. Kaminsky, “Honeypot forensics”, In IEEE Information Assurance Workshop, 2004

[16] C. Seifert, “Malicious SSH Login Attempts”, http://www.securityfocus.com/infocus/1876, August 2006