Scan 25
Analyze a Worm

*Ralf Spenneberg**

November 27, 2002

Contents

1 Overview

This document analyzes a worm captured by a Honeynet. Members form the Honeynet.BR team have captured a new worm from the wild. The file (.unlock) was used by teh worm to infect the honeypot.

Further information may be found at the webpage of the `honeynet project`: http://http://www.honeynet.org/scans/scan25/

* ralf@spenneberg.net

2 Analysis

Before the analysis was begun the file was checked using the command `md5sum`:

```
$ md5sum .unlock
a03b5be9264651ab30f2223592befb42  .unlock
```

The correct value garuanteed the successful download of the binary. The analysis could begin.

2.1 Which is the type of the `.unlock` file? When was it generated?

The type of the file is most easily determined using the UNIX command `file`. This command analyzes the file header and compares it to known file types.

```
$ file .unlock
.unlock: gzip compressed data, deflated, last modified:
Fri Sep 20 12:59:04 2002, os: Unix
```

The command `file` states, that the file `.unlock` has been compressed using gzip and was last modified on Friday September 20 at 12:59 CET using a UNIX operating system.

2.2 Based on the source code, who is the author of this worm? When it was created? Is it compatible with the date from question 1?

Based on the source code the worm was written by contem@efnet with some modification by aion (aion@ukr.net). Further analysis of the comments in the source code give the impression that contem wrote the peer-to-peer UDP DoS part (PUD) of the worm and aion added the worm features to it. Aion is responsible for the code in the file `.update.c` too.

The files were created at

- Sep 19 23:57 .update.c CET
- Sep 20 15:28 .unlock.c CET

This is not fully compatible to the timestamp of the gzip file. The gzip file was created on September 20 at 12:59 while its contents was last modified at 15:28. Both the `ls` and the `file` command take the timezone into account. Therefore it cannot be a mangled timezone. Maybe the attacker modified the systemtime or transferred the files between systems which were off by at least 150 minutes.

2.3 Which process name is used by the worm when it is running?

The worm uses the process name `httpd`. This name is probably used to hide the worm within the list of processes. The process name is independent of the commandline which was used to start the process. The process name is explicitly assigned at line 1805 of the source code:

```
Line   78: #define PSNAME          "httpd "
Line 1805: strcpy(argv[0],PSNAME);
```

2.4 In which format the worm copies itself to the new infected machine? Which files are created in the whole process? After the worm executes itself, wich files remain on the infected machine?

The worm is copied in one file. This file is copied to the machine uuencoded.

```
Line 1416: cat > /tmp/.unlock.uu << __ eof__; \n
```

The uuencoding ensures a successful copy of the worm across the remote shell which is used during the break-in. The shell can only transfer printable characters!

Then the file is uudecoded to `/tmp/.unlock`.

```
Line 1421: uudecode -o /tmp/.unlock /tmp/.unlock.uu;
```

The gezipped tar archive is then extracted using the following command

```
Line 1422: tar xzf /tmp/.unlock -C /tmp/;
```

During the extraction the files `/tmp/.unlock.c` and `tmp/.update.c` are created. The file `/tmp/.unlock.c` is then compiled to create the file `/tmp/httpd`. The crypto library is linked to it. This is needed for the OpenSSL attack (see below).

```
Line 1423: gcc -o /tmp/httpd  /tmp/.unlock.c -lcrypto;
```

The file `/tmp/.update.c` is then compiled to create the file `/tmp/update`. All in all the following files are chronologically created:

```
/tmp/.unlock.uu
/tmp/.unlock
/tmp/.unlock.c
/tmp/.update.c
/tmp/httpd
/tmp/update
```

After the worm is executed,

```
Line 1425: /tmp/httpd %s; /tmp/update; \n
```

the following files are removed:

```
/tmp/.unlock.uu
/tmp/.unlock.c
/tmp/.update.c
/tmp/httpd
/tmp/update
```

This is accomplished using the following lines of source code:

```
Line 1427: writem(sockfd,"rm -rf /tmp/.unlock.uu /tmp/.unlock.c \
                                                  /tmp/.update.c "
Line 1428:                "       /tmp/httpd /tmp/update; exit; \n");
```

The only file left over is `/tmp/.unlock`.

2.5 Which port is scanned by the worm?

The worm scans for opened ports 80. Usually this port is used by web servers. The worm sends the string `GET / HTTP/1.1\r\n\r\n` to the web server and analyzes the response.

```
Line 1154: write(sock,"GET / HTTP/1.1\r\n\r\n", \
            strlen("GET / HTTP/1.1\r\n\r        \n"));
```

The response is tested for the line defining the server.

```
Line 1163: for (d=0;d<n;d++) if (!strncmp(buf+d,"Server: ",\
           strlen("Server: "))) {
```

2.6 Which vulnerability the worm tries to exploit? In which architectures?

The worm tries to exploit the OpenSSL Vulnerability which is described in http://www.kb.cert.org/vuls/id/102795. If the worm finds a web server it tests the server line (see above) for the word **Apache**.

```
Line 1708: if (strncmp(a,"Apache",6)) exit(0);^M
```

If the worm finds an apache server it tests for 21 different Linux architectures. These are:

- Gentoo
- Debian Apache 1.3.26
- Red-Hat Apache 1.3.6, 1.3.9, 2x1.3.12, 1.3.19, 1.3.20, 1.3.22, 1.3.23, 1.3.26
- SuSE Apache 1.3.12, 1.3.17, 1.3.19, 1.3.20, 2x1.3.23
- Mandrake Apache 1.3.14, 1.3.19, 1.3.20, 1.3.23
- Slackware Apache 2x1.3.26

Some architectures are mentioned twice, because the worm test different addresses for the attack. If no architecture is matched the worm will use Red-Hat Apache 1.3.23 by default:

```
Line 1715: if (arch == -1) arch=9;
```

2.7 What kind of information is sent by the worm by email? To which account?

The worm sends some information to aion@ukr.net at the mailserver freemail.ukr.net. This is probably an email account used by the attacker to collect the IP addresses of the infected hosts.

The Registrant of the mailserver is UkrNet. This is a russian ISP offering free email accounts.

The following information is send by email:

- hostid: IP-Address
- hostname: Hostname
- att_from: IP-Address of the parent it connects to in the UDP P2P network

2.8 Which port (and protocol) is used by the worm to communicate to other infected machines?

The worm uses the UDP port 4156 to communicate with other infected machines. Using this port the different nodes in the P2P network communicate. Several different messages may be send:

- 0x20 Info - Send some information about the machine: worm version, uptime etc.

- 0x21 Open a bounce - Opens a proxy running on port 1080
- 0x22 Close a bounce - Closes the proxy
- 0x23 Send a message to a bounce
- 0x24 Run a command in a local shell
- 0x26 Route - Get the routing table
- 0x28 List - Lists the connected P2P servers
- 0x29 UDP Flood
- 0x2a TCP Flood
- 0x2b IPv6 TCP Flood
- 0x2c DNS Flood
- 0x2d Email Scan - All files are scanned for email addresses
- 0x70 Incomming client - New worm on P2P network
- 0x71 Receive List - Get list of P2P servers
- 0x72 Send the list - Send the list of P2P servers
- 0x73 Get my IP
- 0x74 Transmit their IP
- 0x41-0x47 Relay to client

Since this worm is a variant of the original Slapper worm (see below) it is important to note, that the commands 0x25 (Worm ping), 0x27 (Update worm code), 0x30 (Exploit and propagate), 0x2e (Retrieve web page), 0x2F (Send spam) have been removed from the code.

2.9 Name 3 functionalities built in the worm to attack other networks.

The worm offers the following functionalities:

- UDP Flood using random data of maximum 9216 bytes for a specified time
- TCP Flood for a specified time
- TCP IPv6 Flood for a specified time
- DNS Flood
- Open a proxy

The TCP Flood capability seems broken, since the worm opens a TCP connection and immediately closes it again without sending any data.

2.10 What is the purpose of the .update.c program? Which port does it use?

The .update.c program uses the port 1052 and the password "aion1981" It opens a shell when it is contacted on this port and the password is issued.

```
Line 63: if( !strncmp(temp\_buff,PASS,strlen(PASS)) )
Line 64:          execl("/bin/sh","sh -i",(char *)0);
```

2.11 Bonus Question: What is the purpose of the SLEEPTIME and UPTIME values in the .update.c program?

The .update.c program opens the port and accepts connections for exactly UPTIME seconds (10),

```
Line 52: for(stimer=time(NULL);(stimer+UPTIME)>time(NULL);)
```

then it goes to sleep again for SLEEPTIME seconds (300).

```
Line 73: sleep(SLEEPTIME);
```

When a connection is made during UPTIME seconds, a shell is started and bound to the port. The attacker can then enter any command on the compromised host.

3 Conclusion

This worm is a variant of the Slapper worm. The slapper worm is a relative of the Scalper worm. All in all 7 different variants are known to this date. These variants are called:

- Scalper - A worm attacking the chunked encoding security hole of apache.
- Slapper.A - The original worm. The infective file is called `.bugtraq`
- SlapperII.A - A variant which connects to an IRC server.
- SlapperII.A2 - A variant which connects to the same IRC server but a different web server to download its code.
- Slapper.B - The worm discussed in this analysis. The infective file is called `.unlock`
- Slapper.C - Another variant. The infective file is called `.cinik`
- Slapper.C2 - Again a variant. The infective file is called `.cinik2`

A nice genealogy of the worms can be found at `http://isc.incidents.org/analysis.html?id=177`.

The global slapper worm center at `http://www.f-secure.com/slapper/` shows nicely that the analyzed worm was especially active around September 28th, about one week after Slapper.A and about a week before Slapper C.

4 Recommendations for Protection

To protect yourself do the following:

- Update your apache web servers and openssl libraries.
- Analyze your web servers for the files `.bugtraq`, `.unlock`, `.cinik` or `.cinik2` in the `tmp` directories
- Filter all email leaving your site for the address aion.ukr.net
- Filter all UDP traffic on port 4156
- Filter all TCP connects to port 1052