

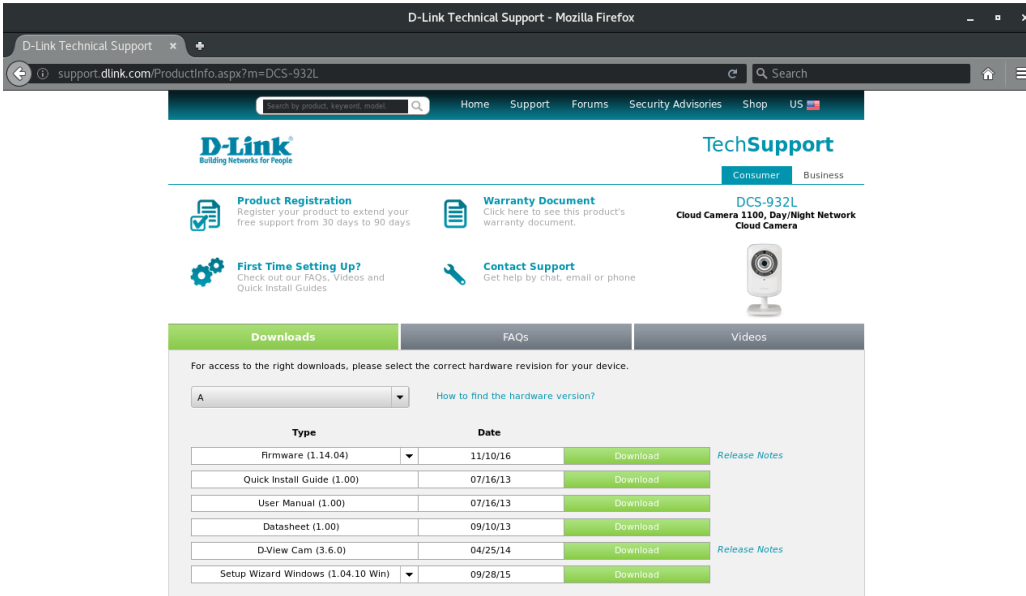
Reversing Firmware- How does that work?

March 8, 2017

Last week I wrote about a backdoor vulnerability in a device used by spammers. The team at Spider Labs (<https://www.trustwave.com/Resources/SpiderLabs-Blog/Undocumented-Backdoor-Account-in-DBLtek-GoIP/>) discovered it by reverse engineering a piece of firmware. If you've never seen anything like that before, here's a quick walk-through that'll take a piece of firmware from a binary file to an extracted file system you can explore on your own. Let's get started!

1.) Download the firmware

Download the firmware from D-Link (<http://support.dlink.com/ProductInfo.aspx?m=DCS-932L>). This walkthrough used hardware version A, firmware version 1.14.04 (ftp://ftp2.dlink.com/PRODUCTS/DCS-932L/REVA/DCS-932L_REVA_FIRMWARE_1.14.04.ZIP).



The screenshot shows the D-Link Technical Support website for the DCS-932L Cloud Camera. The page is viewed in Mozilla Firefox. The Downloads section is active, showing a table of available files for hardware version A. The table lists the following files:

Type	Date	Download	Release Notes
Firmware (1.14.04)	11/10/16	Download	Release Notes
Quick Install Guide (1.00)	07/16/13	Download	
User Manual (1.00)	07/16/13	Download	
Datasheet (1.00)	09/10/13	Download	
D-View Cam (3.6.0)	04/25/14	Download	Release Notes
Setup Wizard Windows (1.04.10 Win)	09/28/15	Download	

2.) Unzip the archive

Each “hit” binwalk gets is recorded on a single line, and comes in three parts:

- A file location in decimal format
- A file location in hexadecimal format
- A description of what was found at that location

Looking at the first line, we see that binwalk found a U-Boot string at 106352. U-Boot is a popular bootloader. When a device is powered on, it’s the bootloader’s job to load up the operating system. And sure enough, at 327680, we can see a ulmage header telling us that we’ll find the OS kernel image in a LZMA archive that starts at 327744. If you’re having a hard time following it, I cleaned up the formatting in step 6.

```

File Edit View Search Terminal Help
root@kali:~/project# binwalk dcs932l_v1.14.04.bin
-----
DECIMAL      HEXADECIMAL  DESCRIPTION
-----
106352      0x19F70      U-Boot version string, "U-Boot 1.1.3"
106816      0x1A140      CRC32 polynomial table, little endian
124544      0x1E680      HTML document header
124890      0x1E7DA      HTML document footer
124900      0x1E7E4      HTML document header
125092      0x1E8A4      HTML document footer
125260      0x1E94C      HTML document header
125953      0x1EC01      HTML document footer
327680      0x50000      ulmage header, header size: 64 bytes, header CRC: 0x88345E96, created: 2016-09-09 13:52:27, image size: 380495
327744      0x50040      8 bytes, Data Address: 0x80000000, Entry Point: 0x803B8000, data CRC: 0x531E940E, OS: Linux, CPU: MIPS, image type: OS Kernel Image, compression type: lzma, image name: "Linux Kernel Image"
                    LZMA compressed data, properties: 0x5D, dictionary size: 33554432 bytes, uncompressed size: 6558763 bytes
root@kali:~/project#

```

6.) Carve out the LZMA archive

Before we can unpack that LZMA archive and dig through it, we need to carve it out of the larger binary. We’ll do that by running: `dd if=dcs932l_v1.14.04.bin skip=327744 bs=1 of=kernel.lzma`

(Optional) You can check to ensure the LZMA archive came through OK by running `file kernel.lzma`.

```

File Edit View Search Terminal Help
-----
DECIMAL      HEXADECIMAL    DESCRIPTION
-----
106352      0x19F70        U-Boot version string, "U-Boot 1.1.3"
106816      0x1A140        CRC32 polynomial table, little endian
124544      0x1E680        HTML document header
124896      0x1E7DA        HTML document footer
124900      0x1E7E4        HTML document header
125092      0x1E8A4        HTML document footer
125260      0x1E94C        HTML document header
125953      0x1EC01        HTML document footer
327680      0x50000        uImage header, header size: 64 bytes,
                    header CRC: 0x88345E96,
                    created: 2016-09-09 13:52:27,
                    image size: 3804958 bytes,
                    Data Address: 0x80000000,
                    Entry Point: 0x803B8000,
                    data CRC: 0x531E94DE,
                    OS: Linux, CPU: MIPS,
                    image type: OS Kernel Image,
                    compression type: lzma,
                    image name: "Linux Kernel Image"
327744      0x50040        LZMA compressed data, properties: 0x5D,
                    dictionary size: 33554432 bytes,
                    uncompressed size: 6558763 bytes
root@kali:~/project# dd if=dcs932l_v1.14.04.bin skip=327744 bs=1 of=LinuxKernelImage.lzma
3866560+0 records in
3866560+0 records out
3866560 bytes (3.9 MB, 3.7 MiB) copied, 5.07149 s, 762 kB/s
root@kali:~/project# file LinuxKernelImage.lzma
LinuxKernelImage.lzma: LZMA compressed data, non-streamed, size 6558763
root@kali:~/project#

```

7.) Another data file...

Now you can unpack that LZMA archive by running `unlzma kernel.lzma`. To learn what we've unpacked let's use the `file` command again by running `file kernel` ...looks like we've got another data file.

```

root@home:~/project/dlink# unlzma kernel.lzma
root@home:~/project/dlink# file kernel
kernel: data
root@home:~/project/dlink#

```

8.) Time to rinse...

Just like before, we're going to run `binwalk` against the data file with `binwalk kernel`.

There's a ton of output there, including another LZMA archive at `4038656`. If you scroll up to the top of the `binwalk` output, you'll also see the Linux kernel version.

```

File Edit View Search Terminal Help
size < 1530failed
3463610 0x34D9BA Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert pRxBl
k->pRxPacketfailed
3463714 0x34DA22 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert pHead
er802_3failed
3464130 0x34DBC2 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert pTask
failed
3464462 0x34DD0E Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert pNetD
evfailed
3464766 0x34DE3E Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert (pref
ixLen < IFNAMSIZ)failed
3464878 0x34DEAE Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_linux.c:%d assert ((slo
tNameLen + prefixLen) < IFNAMSIZ)failed
3480306 0x351AF2 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_ate.c:%d assert (TxPowe
r >= -7)failed
3484022 0x352976 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_ate.c:%d assert (BbpVal
ue == 0x00)failed
3484126 0x3529DE Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_ate.c:%d assert (BbpVal
ue == 0x04)failed
3485810 0x353072 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_ate.c:%d assert bbb_dat
a == valuefailed
3486762 0x35342A Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_ate.c:%d assert pRacfg
!= NULLfailed
3487126 0x353596 Unix path: /net/wireless/rt2860v2_sta../rt2860v2/os/linux/rt_pci_rbus.c:%d assert pA
dfailed
3491536 0x3546D0 Unix path: /etc/Wireless/RT2860STA/RT2860STA.dat
3573187 0x3685C3 Neighborly text, "neighbor %.2x%.2x%.2x%.2x%.2x%.2x%.2x%.2x lost on port %d(%s)
(%s)"
3807776 0x3A1A20 CRC32 polynomial table, little endian
4038656 0x3DA000 LZMA compressed data, properties: 0x5D, dictionary size: 1048576 bytes, uncompressed
size: 8072704 bytes
root@kali:~/project#

```

9.) ...And repeat.

Now let's extract that LZMA we saw in there. We'll use `dd if=kernel skip=4038656 bs=1 of=mystery.lzma`, and unpack the results with `unlzma mystery.lzma`

```

File Edit View Search Terminal Help
root@kali:~/project# dd if=kernel skip=4038656 bs=1 of=mystery.lzma
2520107+0 records in
2520107+0 records out
2520107 bytes (2.5 MB, 2.4 MiB) copied, 3.18272 s, 792 kB/s
root@kali:~/project# file mystery.lzma
mystery.lzma: LZMA compressed data, non-streamed, size 8072704
root@kali:~/project#

```

10.) The CPIO archive

Run `file mystery`. It's a CPIO archive, which is yet another archive format...and it's the kind of place you're likely to find the file system.

```

File Edit View Search Terminal Help
root@kali:~/project# file mystery
mystery: ASCII cpio archive (SVR4 with no CRC)
root@kali:~/project# mkdir cpio; cd cpio
root@kali:~/project/cpio# cpio -idm --no-absolute-filenames < ../mystery
cpio: Removing leading '/' from member names
15767 blocks
root@kali:~/project/cpio#

```

Create a directory to unpack the CPIO archive and get in there with `mkdir cpio; cd cpio`. Now unpack the CPIO with `cpio -idm --no-absolute-filenames < ../mystery`.

11.) Explore the file system

If everything went well, congrats! The file system is unpacked, and you're able to explore it on your own.

```
File Edit View Search Terminal Help
root@kali:~/project/cpio# ls -l
total 60
drwxrwxr-x 2 501 501 4096 Apr 23 19:40 bin
drwxrwxr-x 3 501 501 4096 Apr 23 19:40 dev
drwxrwxr-x 2 501 501 4096 Apr 23 19:40 etc
drwxrwxr-x 9 501 501 4096 Apr 23 19:40 etc_ro
drwxrwxr-x 2 501 501 4096 Sep 9 2016 home
lrwxrwxrwx 1 501 501 11 Apr 23 19:40 init -> bin/busybox
drwxr-xr-x 4 501 501 4096 Apr 23 19:40 lib
drwxrwxr-x 2 501 501 4096 Sep 9 2016 media
drwxrwxr-x 2 501 501 4096 Sep 9 2016 mnt
drwxrwxr-x 2 501 501 4096 Apr 23 19:40 mydlink
drwxrwxr-x 2 501 501 4096 Sep 9 2016 proc
drwxrwxr-x 2 501 501 4096 Apr 23 19:40 sbin
drwxrwxr-x 2 501 501 4096 Sep 9 2016 sys
drwxrwxr-x 2 501 501 4096 Sep 9 2016 tmp
drwxrwxr-x 5 501 501 4096 Apr 23 19:40 usr
drwxrwxr-x 2 501 501 4096 Sep 9 2016 var
root@kali:~/project/cpio#
```