

2018-01-18

file – 用来检测是否是有效的文件和文件类型

hexdump – 16进制导出工具

strings 跟hexdump类似但是可以以可读的形式展示

dd – 从二进制文件中挖掘数据

lzma – 解压LZMA文件

binwalk – 通过固件文件头来分析文件和文件系统

Fireware Mod Kit – 自动化分析固件文件的一系列脚本

squashfs-tools – 可以通过apt-get squashfs-tools 来安装。

1. 初步分析

通常来说逆向工程的第一步就是使用上面列出的通用linux工具从要分析的文件中挖掘出尽量多的信息。通过这些信息来决定下一步进行的分析。比如使用hexdump发现sqsh意味着文件中有一个squashfs文件系统或者识别出了固件包中常用的boot loader U-Boot。

A.File Tool

文件工具通常只是告诉我们这个文件是否是已知的文件类型，某些情况下可以识别出文件的种类，比如数据文件。

file分析的结果可以告诉我们该文件是否是已知的类型，是否需要进一步分析。这里的结果我们可以看到仅仅是显示数据文件。

B.hexdump

hexdump 工具可以让你分析文件中的每一个字节，这是非常有价值的。使用hexdump分析固件如下：

上面的命令会把hexdump的结果写入到文件hexTP.txt中做进一步分析。-C选项是设置hexdump输出为hex+ASCII的方式，更便于阅读。输出的文件十分巨大，这个例子中我们可以从文件头的地方看到这个固件是属于TP-Link的，但是这个信息我们已经知道了。所以下一步就是尝试strings命令，看看是否可以获得更多的信息。

```
root@SafeCode:~# hexdump -C router.bin > data.txt
root@SafeCode:~# head data.txt
00000000  01 00 00 00 54 50 2d 4c 49 4e 4b 20 54 65 63 68 |....TP-LINK Tech|
00000010  6e 6f 6c 6f 67 69 65 73 00 00 00 00 76 65 72 2e |nologies....ver.|
00000020  20 31 2e 30 00 00 00 00 00 00 00 00 00 00 00 00 | 1.0.....|
00000030  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
00000040  08 41 00 08 00 00 00 01 00 00 00 00 4d 72 92 38 |.A.....Mr.8|
00000050  30 60 49 64 58 57 22 00 d7 ec 0d 25 00 00 00 00 |0`IdXW"....%....|
00000060  b1 12 89 aa af 45 a9 2b f2 b5 78 de a0 cc 48 d5 |.....E.+..x...H.|
00000070  00 00 00 00 80 00 20 00 80 1a a2 40 00 3e 02 00 |.....@.>..|
00000080  00 00 02 00 00 0c 68 1c 00 10 00 00 00 2c 00 00 |.....h.....|
00000090  00 00 00 00 00 00 bf f4 00 03 00 0d 00 21 00 00 |.....!..|
root@SafeCode:~#
```

C.strings

作为初始的信息收集，strings可能是最常用和做好用的工具之一，因为它可以显示文件中所有可打印的数据。跟使用hexdump一样，最好把strings的结果写入文件分析，以免下次想要分析的时候还需要重复一遍strings命令。搜索一下文件系统常用的boot loader名字比如U-boot，就可以找到这些信息。现在我们已经知道这个嵌入式系统使用U-boot作为boot loader，而且知道了它的版本信息。

D. Binwalk

binwalk会分析二进制文件中可能的固件头或者文件系统，然后输出识别出的每个部分以及对应的偏移量

binwalk给出了大量有用的信息。从这些信息中可以得知这个固件是运行在MIPS架构上的一个linux系统。使用了squashfs文件系统。同时也再次确认boot loader是U-boot。

2.提取文件系统

终于到了最关键的一步，我们要从固件镜像中分离出文件系统的内容。因为是linux系统，可以预见一些标准的linux文件比如passwd和shadow可能会有一些敏感信息。很多人使用dd来分离文件系统的内容。使用binwalk和Firmware Modification Kit来解包最简单方便。

使用binwalk的-e参数可以自动把固件镜像中的所有文件都解出来。

binwalk -e <input file>

使用Firmware Modification Kit中的extract-firmware.sh脚本会更加高效。如果你想重打包修改之后的固件的话可以使用build-firmware.sh来打包固件。这样可以节省大量时间自己解包和管理所有的偏移。

```
root@SafeCode:~# binwalk -Me router.bin

Scan Time:      2018-01-18 21:36:15
Target File:    /root/router.bin
MD5 Checksum:   c672cf7d54a04c24abeaa3f9efa78771
Signatures:     344

DECIMAL          HEXADECIMAL      DESCRIPTION
-----
0                0x0              TP-Link firmware header, firmware version: 0.-16396.3, image version: "", product ID: 0x0, product version: 138477576, kernel load address: 0x0, kernel entry point: 0x80002000, kernel offset: 63744, kernel length: 512, rootfs offset: 813084, rootfs length: 1048576, bootloader offset: 2883584, bootloader length: 0
13392            0x3450           U-Boot version string, "U-Boot 1.1.4 (May  6 2013 - 13:20:35)"
13440            0x3480           CRC32 polynomial table, big endian
14728            0x3988           uImage header, header size: 64 bytes, header CRC: 0xBA7F2047, created: 2013-05:20:35, image size: 34860 bytes, Data Address: 0x80010000, Entry Point: 0x80010000, data CRC: 0x263C3839, Linux, CPU: MIPS, image type: Firmware Image, compression type: lzma, image name: "u-boot image"
14792            0x39C8           LZMA compressed data, properties: 0x50, dictionary size: 33554432 bytes, uncompressed size: 99104 bytes
131584           0x20200          TP-Link firmware header, firmware version: 0.0.3, image version: "", product ID: 0x0, product version: 138477576, kernel load address: 0x0, kernel entry point: 0x80002000, kernel offset: 391, kernel length: 512, rootfs offset: 813084, rootfs length: 1048576, bootloader offset: 2883584, bootloader length: 0
132096           0x20400          LZMA compressed data, properties: 0x50, dictionary size: 33554432 bytes, uncompressed size: 2317284 bytes
1180160          0x120200         Squashfs filesystem, little endian, version 4.0, compression: lzma, size: 2652 bytes, 537 inodes, blocksize: 131072 bytes, created: 2013-05-06 05:32:12
```

```
root@SafeCode:~#  
root@SafeCode:~# cd _router.bin.extracted/  
root@SafeCode:~/_router.bin.extracted# cd squashfs-root/  
root@SafeCode:~/_router.bin.extracted/squashfs-root# ls  
bin dev etc lib linuxrc mnt proc root sbin sys tmp usr var web  
root@SafeCode:~/_router.bin.extracted/squashfs-root#
```