提取：extrator.py

总述：extractor.py 将.bin文件输入，然后将kernel和filesytem文件提取出放在，/images下面分别为image和.tar.gz文件

options: -nf 禁用提取root -nk 禁用提取kernel -np禁用平行模型提取 -b品牌 -d打出debug的信息

commod：extractor.py -b **D-link** -sql **ip** -np -nk **file.bin**

bug修改：

```
./init.sh
python3 ./sources/extractor/extractor.py -b D-link -sql 127.0.0.1 -d
dcs930lb1_v2.16.01.bin  images

# 需要把他的两个提取 -nk -nf 去掉统一换成-np,两句话变成一句话
python3 ./sources/extractor/extractor.py -b D-link -sql
```

固件类型对应表：

```
sasquatch ->non-standard SquashFS
jefferson ->JFFS2
ubi_reader->UBIFS
yaffshiv->YAFFS
unstuff ->StuffIt
这些文件都存放在user/bin下面
```

仲裁方式：

1.Boost仲裁（makeImage.sh）

总述：

创建分区表-->格式化磁盘-->挂载镜像-->解压提取出的文件系统-->创建FIRMADYNE Directories->找到Init路径boost启动的第一个程序以及所用的服务器版本（inferkernel.sh中找的）->patch image(打补丁)->preInit.sh|network.sh|run_service.sh|injectionChecker.sh传到镜像里面去->接触挂载和分区

关键步骤：

- **boost启动路径仲裁：**（inferFile.sh|inferfile.py）

  inferkernal.py 把images/1.kernal中的linux版本信息和init地址放在工作目录下存储起来；

  在文件系统中找这几个名字"preinitMT" "preinit" "rcS"`
- **服务器版本仲裁：**（makeImage.sh）

```
if (${FIRMAE_ETC}); then
    if [ -e /etc/init.d/uhttpd ]; then
        echo -n "/etc/init.d/uhttpd start" > /firmadyne/service
        echo -n "uhttpd" > /firmadyne/service_name
    elif [ -e /usr/bin/httpd ]; then
        echo -n "/usr/bin/httpd" > /firmadyne/service
        echo -n "httpd" > /firmadyne/service_name
    elif [ -e /usr/sbin/httpd ]; then
        echo -n "/usr/sbin/httpd" > /firmadyne/service
        echo -n "httpd" > /firmadyne/service_name
    elif [ -e /bin/goahead ]; then
        echo -n "/bin/goahead" > /firmadyne/service
        echo -n "goahead" > /firmadyne/service_name
    elif [ -e /bin/alphapd ]; then
        echo -n "/bin/alphapd" > /firmadyne/service
        echo -n "alphapd" > /firmadyne/service_name
    elif [ -e /bin/boa ]; then
        echo -n "/bin/boa" > /firmadyne/service
        echo -n "boa" > /firmadyne/service_name
    elif [ -e /usr/sbin/lighttpd ]; then # for Ubiquiti firmwares
        echo -n "/usr/sbin/lighttpd -f /etc/lighttpd/lighttpd.conf" > /firmadyne/service
        echo -n "lighttpd" > /firmadyne/service_name
```

- **符号链接**

  清理软连接和原始文件

- **给image打补丁Patching Filesystem：**

1.将一些常规的符号执行的目录给创建出来

2.把自带的bin中所有文件提到的路径到挑出来然后进行创建

3./etc时区host passwd文件

4.创建default dev

5.创建a gpio file给watchdog 利用GPIO输出高低电平保证看门狗不重启，停止reboot

6.add some default nvram entries（/nvram在本机上没有所有firmdyea写了一个hook出来模拟nvram,并将其添加在LD_PRELOAD中）

出现 `/dev/nvram: Permission denied`

宿主机上肯定没有 `/dev/nvram` 啊，所以使用 QEMU 进行模拟时使用了 firmadyne/libnvram 这个库 Hook 掉 nvram 的相关操作，然后将这个库添加到 `LD_PRELOAD` 中。在使用时如果发现这个库没有创建 tmpfs 进行存储的话，那就手动用 `sudo mount -t tmpfs -o size=10M tmpfs $挂载路径` 创建一个。可以修改 `config.h` 实现将挂载路径放在别的目录下。例如我将挂载路径改动到了 `/mnt/libnvram`。

2.Network仲裁

- **仲裁过程：**

```
[*] Infer test: /etc_ro/rcS (POSIX shell script, ASCII text executable)
[*] web service: /bin/alphapd
Running firmware 1: terminating after 240 secs...
qemu-system-mipsel: terminating on signal 2 from pid 37390 (timeout)
done!fisrt run!
Infer NVRAM default file!

127.0.0.1
0.0.0.0
2.65.87.200
[*] Interfaces: [('br0', '2.65.87.200')]
[*] ports: []
[*] networkInfo: [('2.65.87.200', 'eth2', None, None, 'br0')]
has ethernet
[*] filter network info: [('2.65.87.200', 'eth2', None, None, 'br0')]
[*] test emulator
```

- 仲裁原理：makeNetwork.py

总述：

挂载镜像->向init.sh中写入->移除挂载

```
out.write('\n/firmadyne/network.sh &\n')
out.write('/firmadyne/run_service.sh &\n')
out.write('/firmadyne/debug.sh\n')
# trendnet TEW-828DRU_1.0.7.2, etc...
out.write('/firmadyne/busybox sleep 36000\n')
```

->运行qemu模拟log到initial.serial.log

->**（关键）**找到NVRAM 配置文件并把文件名保存在nvram_files中把关键词列表放在nvram_kes中

->**（关键）**找到findPorts->findNonLoInterfaces->findMacChanges->getNetworkList网络参数结果保存在

NVRAM 配置文件寻找：（inferDefault.py）

- 通过通过查找qemu.initial.serial.log中的[NVRAM]的关键词可以找到key并把这些keys保存在NVRAM.keys文件当中。
- 通过遍历文件系统下的文件找出含有这些关键词的文件将他们计算成配置文件：
- 通过查找qemu.initial.serial.log中的[NVRAM]的关键词可以找到key并把这些keys保存在NVRAM.keys文件当中
- 找到的配置文件

```
scratch > 1 > nvram_files
1  /etc_ro/Wireless/RT2860AP/RT2860_default_vlan 34 ASCII_text
2  /bin/alphapd 23 ELF_32-bit_LSB_executable,_MIPS,_MIPS-II_version_1_(SYSV),_dynamically_linked,_interpreter_/lib/ld-uClibc.so.0,_stripped
3
```

挽留过

网路信息补全：

1.通过inet_bind查找port

port:
inet_bind[PID: 208 (alphapd)]: proto:SOCK_STREAM, port:443
inet_bind[PID: 271 (ucp)]: proto:SOCK_DGRAM, port:2267
inet_bind[PID: 270 (udev)]: proto:SOCK_STREAM, port:8053
inet_bind[PID: 1138 (alphapd)]: proto:SOCK_STREAM, port:80
inet_bind[PID: 1138 (alphapd)]: proto:SOCK_STREAM, port:443

inet_bind[PID: 270 (udev)]: proto:SOCK_DGRAM, port:1900
似乎他写的正则有问题

2.findNonLoInterfaces：

通过_inet_insert_ifa找非回环地址，和网络接口名称

3.findMacChanges：

通过ioctl_SIOCSIFHWADDR是否修改过mac地址

./clean.sh 的问题注意权限和文档文档的owner

```
chmod -R 777 ./  #权限
chwon -R root ./ #所示
```