总体说明：

仿真主程序围绕**run.sh展开**，在run.sh 中有很多模式-d(debug) -a(analyse) -r(run) -c(check),个人推荐在-d模型把仿真程序跑起来手动复现漏洞,-a模式下进行自动化漏洞挖掘；

推荐启动方式：

```
root@ubuntu:/home/dilision/FirmAE# ./run.sh -a D-link DIR868L_B1_FW205WWb02.bin
root@ubuntu:/home/dilision/FirmAE# ./run.sh -d D-link DIR868L_B1_FW205WWb02.bin
command: ./run.sh -模式 -brand firmdir||firmdir
```

目录说明：

最外层文件：

- install.sh 启动postgresql数据库
- run.sh 主程序
- firmae.config 全局变量和函数
- dockerinit|helper 待研究
- debug.py在debug模式下，simulation完成之后用与固件相连见下debug说明；
- clear.sh 本人加的，简单四行代码，用于上一次意外退出之后，接口和虚拟网络没有删除造成的设备冲突现象，在每次simulation之前可以先用./clean.sh 防止冲突

script:存放一下会到的脚本

scratch:工作目录，每次仿真之后他会单独创建出来，不会删除

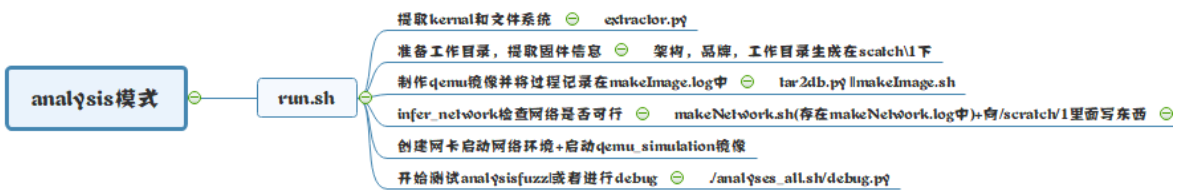
images：在createimage时遗留下的目录，其中有很多文件在image创建之后被删除

binaries：存放createimage需要向image中添加的二进制文件，方便仲裁和debug时使用

analyses：在-a模式下使用用fuzzer，initilizer，nmap.sh 是三个功能，其中nmap随便扫一下网络设备的端口，initilizer用selenium随便测试了一下能不能联通，而nmap对web端口进行了扫面，fuzzer进行fuzz测试

run.sh分析：

这里有对run流程的详细解释，以下是我个人的理解



对makeimage的分析（用于boost仲裁机制的核心部分qume-image创建过程中）：

- IMAGE image.raw镜像文件 qemu-img create -f raw "${IMAGE}" 1G format：raw size：1G || IMAGE_DIR 挂在镜像的目录 注意此时的image为空！！！
- 创建device并分区(磁盘)->格式化磁盘mkfs.ext2->挂载image目录到device->将提取出的固件放在imagedir
- 并在imagedir其中加入/firmadyne工具和仲裁引导的初始化工具 preInit.sh,network.sh,run_service.sh的信息对image进行仲裁修复。
- 取消挂载删除分区表

  注：进行分区这里有对df查看磁盘,fdisk对磁盘分区,mount挂载解释

  https://people.debian.org/~aurel32/qemu/ qemu的镜像储藏（不同固件的qemu镜像）

  https://en.wikibooks.org/wiki/QEMU/Images qemuimage的资料

makeNetwork.sh在打包/image前做最后的网络测试网络检测仲裁：

fileter筛选出合理的IP和端口号防止冲突-->qemu模拟起来->测试，用curl和ping的方式进行测试网是否真的运行起来。

run_analyze：把环境启动起来+ananlyses_all.sh对web服务器进行分析

主要功能把网络的接口虚拟网络绑定并启动，quem指令运行qemuimge

其中有些命令的解释：

- 对目录或者文件夹的权限进行调整：

```
chmod是文件权限而chown是文件所属
chmod -R 777 path/
chown -R dilison path/
```

- 关于虚拟网络|虚拟网卡创建网络设置：

```
#https://linuxize.com/post/linux-ip-command/  接口
#接口
ip link set xxx up #创建
ip addr add 192.168.XXXX dev xxx #分配ip地址
ip link set xxx down #删除

#网络设备
tunctl -t xxx -u user #添加创建tap（三层设备）
```

```
ip link add link 接口 name 子接口 type vlan id 1 #绑定在指定网络
ip link set xxx up  #启动网卡
tunctl -d xxx #删除网卡

#流程
创建接口网卡，启动设备，启动接口分配ip
关闭网卡和删除网络和设备

#clear.sh
umount ./scratch/1/image > /dev/null
sudo ip link set tap0_2 down
sudo ip link delete tap0_2.1
sudo tunctl tap0_1
```

-d，功能debug:

1.里面可以用telnet对仿真对象进行远程连接；

2.可以用tcpdump来监视固件的网卡数据；

3.可以把把文件仿真的固件上去，传到/firmdyre目录下

4.gdbserver对固件上的程序进行远程调试；