**详细的路由器漏洞分析环境搭建教程**

**2016-8-24 by 伐秦**

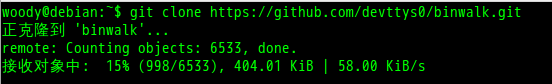
**声明：此教程内容完全是在Debian 8.0稳定版中进行的。添加更新源，系统更新，添加sudo，读者自己搞定!读者当然可以使用任意其他Linux发行版。至于为什么用Debian，最后有笔者的吐槽。**

**0x0 安装Git**

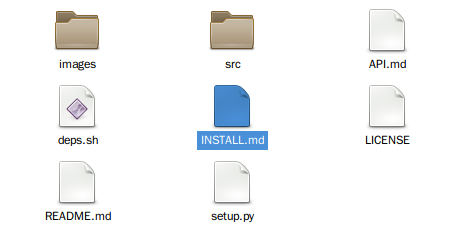
****

**0x1 安装Binwalk**

首先在Github上面把Binwalk clone 下来

****

打开clone下来的文件夹，里面有个install.md这货就是安装binwalk之前要安装的其他一些依赖或者第三方工具。



用gedit打开install.md。按里面说的一个一个安装，但是呢会提示缺少其他的一些东西，所以这里先告诉大家，先装下面这几个东西

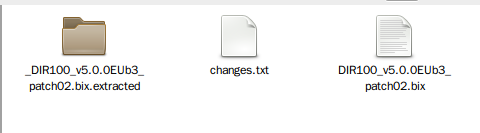
****

再按照install.md里面的内容来搞。你愿意装python2.7.x和3.x都可以，但是我比较喜欢2.7.x，就是喜欢，无理由

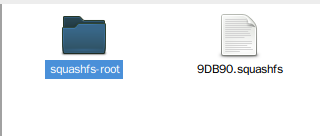
装完以后来试一下，就用论坛里面有前辈分析过的DIR-100\_fw\_reva\_113\_ALL\_en\_20110915.zip，先解压zip，然后得到的文件夹里面有个DIR100\_v5.0.0EUb3\_patch02.bix，用下面这条指令解包。

|  |
| --- |
| binwalk -Me DIR100\_v5.0.0EUb3\_patch02.bix |

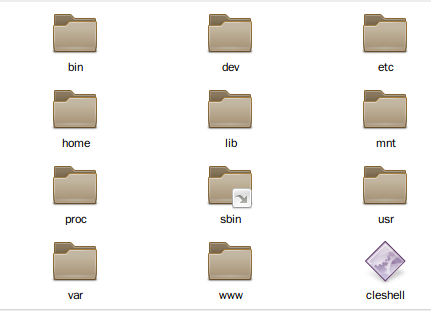
可以看到，多出来一个文件夹



打开它



这个文件夹里面就是我们要的固件。



如果你想试试解包出来的能不能运行，就按下面的这么做：安装这两个东东（当然你也需要按照后面的方法安装QEMU）：

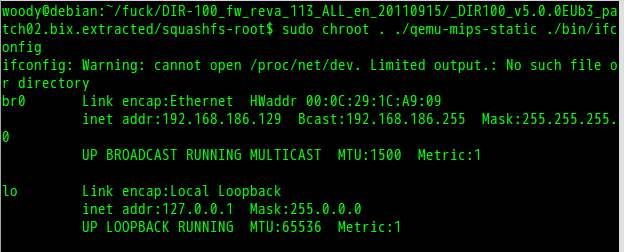
|  |
| --- |
| sudo apt-get install binfmt-support qemu-user-static |

然后在Binwalk解包的固件目录下，执行这条指令

|  |
| --- |
| cp $(which qemu-mips-static) ./ |

接着，就可以执行下面这条指令来运行下固件里面的ifconfig看看

|  |
| --- |
| sudo chroot . ./qemu-mips-static ./bin/ifconfig |



**0x2 安装Buildroot**

Binwalk装完了，接下来转buildroot,到buildroot.org去下载，我下的是buildroot-2016.05.tar.bz2

然后安装这几个包

****

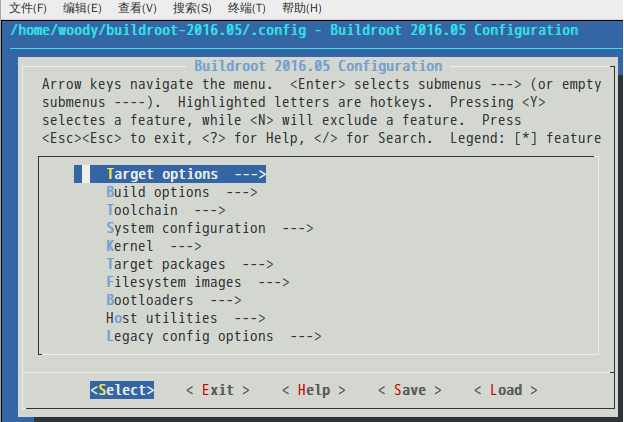
解压buildroot-2016.05.tar.bz2，切换到buildroot-2016.05目录，执行:

****

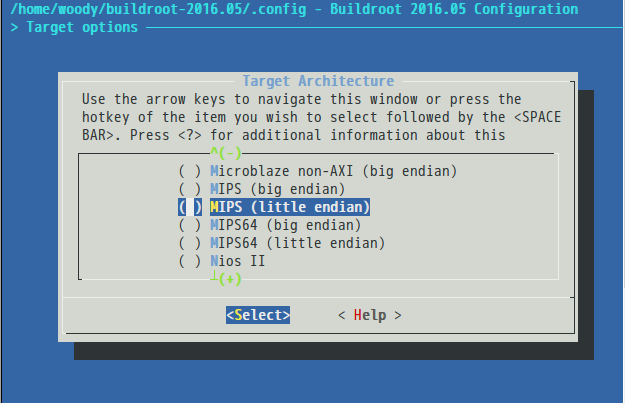
接着执行:

****

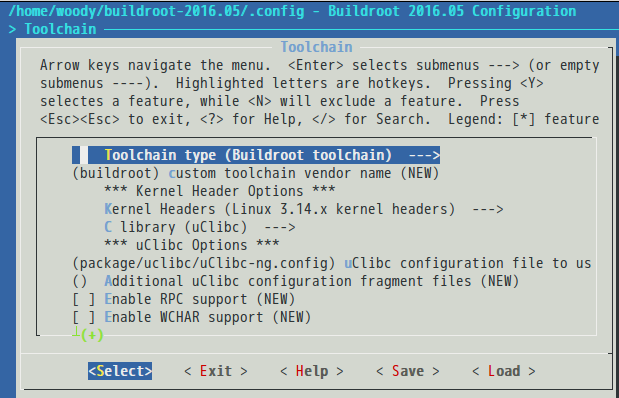
弹出一个丑陋的对话框(嗯，我这么称呼它)

****

选择“Target options中的Target Architecture”，改成MIPS（分别是大端或小端），我这里就选小端了

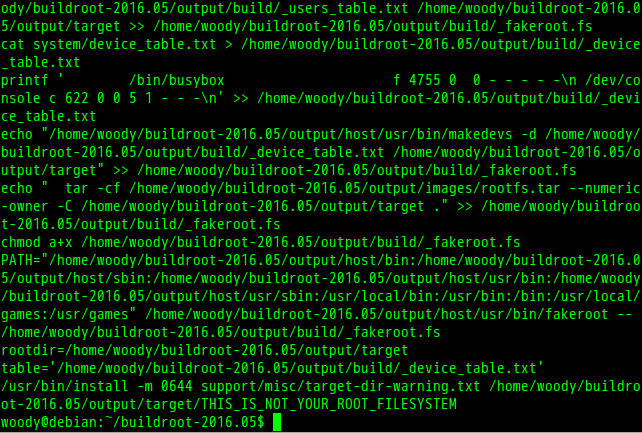
****

然后选择Tollchain中的Kernel Headers 最好选择和你的内核版本一样的，我不知道为啥，有些是这么说，可能是万恶的兼容性吧，这里没有和我一一样的，所以我选择低于我内核版本的，3.14.x



save，退出:

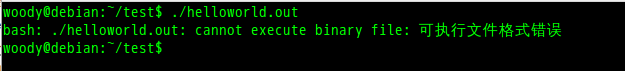
执行 make 命令，接下来是漫长的等待, 你说要多长时间，这说不好，反正我是编译时期又去睡了一觉,编译成功的shell内容大致是这样的:



我们可以来使用编译出来的交叉编译工具来编译一个文件试验一下(说起来好拗口)，就用经典的helloworld



编译得到目标文件运行之，会发现这个错误提示



当然了，因为我们编译出来的是MIPSEL机器的可执行文件呀~这时候就要安装QEMU了



然后可以在MIPSEL环境下执行helloworld.out



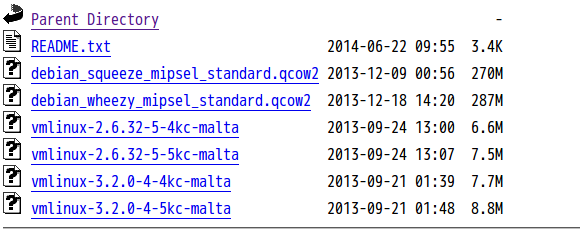
成功了，这说明我们的交叉编译环境和QEMU都是安装成功滴。

**0x3 选择QEMU-MIPS虚拟机映像**

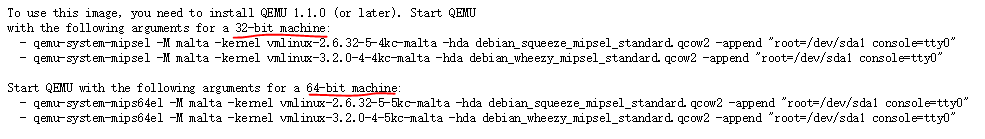
访问 https://people.debian.org/~aurel32/qemu/，下载MIPSEL的系统映像



我选择mipsel(小端)



至于这些文件下载哪个，下面的说明已经很清楚了。另外squeeze和wheezy是说的版本号。



我就下vmlinux-2.6.32-5-4kc-malta和 debian\_squeeze\_mipsel\_standard.qcow2.1

为了能够让QEMU虚拟机和宿主机都上网，并且互通，为动态调试做准备，必须要配置桥接网络<br>  
先装两个东西： bridge-utils和apt-get install uml-utilities  
修改你的/etc/network/interfaces内容为：

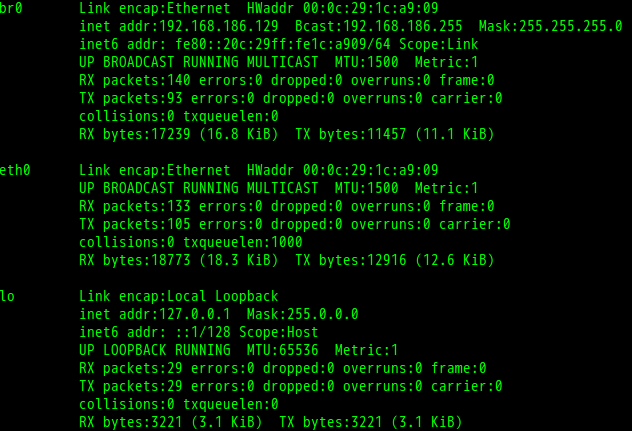
|  |
| --- |
| auto lo     iface lo inet loopback     auto eth0     iface eth0 inet manual     up ifconfig eth0 0.0.0.0 up     auto br0     iface br0 inet dhcp     bridge\_ports eth0     bridge\_stp off     bridge\_maxwait 1 |

编辑/etc/qemu-ifup，  
换成下面的这个

|  |
| --- |
| #!/bin/sh     echo "Executing /etc/qemu-ifup"     echo "Bringing up $1 for bridged mode..."     sudo /sbin/ifconfig $1 0.0.0.0 promisc up     echo "Adding $1 to br0..."     sudo /sbin/brctl addif br0 $1     sleep 3 |

保存，重启。

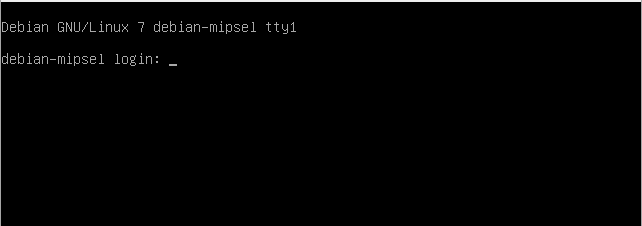
重启之后，你的主机ifconfig是这样的



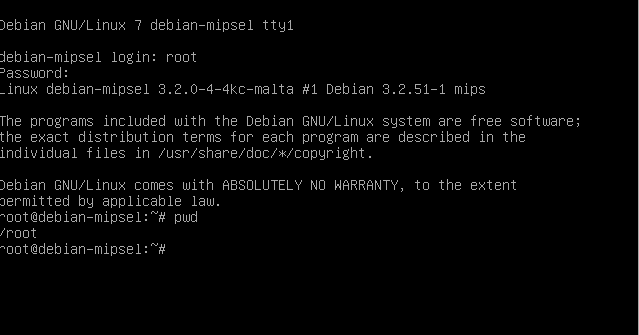
只要没有明显的保存就算是成功了，接下来呢就要用QEMU启动系统映像啦：

|  |
| --- |
| sudo qemu-system-mipsel -M malta -kernel vmlinux-2.6.32-5-4kc-malta     -hda debian\_squeeze\_mipsel\_standard.qcow2.1      -append "root=/dev/sda1 console=tty0" -net nic, -net tap |

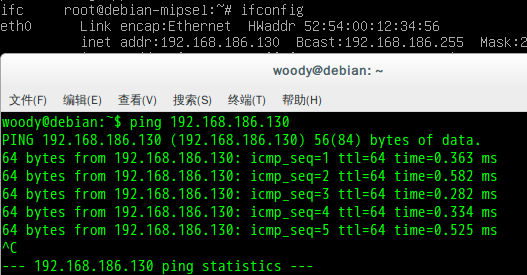
执行以后，会弹出一个窗口，过一会儿系统的登陆界面就出来了



用户名和密码全是root,登陆以后是这样的



查看虚拟机IP，在ping一下虚拟机



**0x5 配置IDA动态调试**

然后安装IDA Pro 我用的是论坛提供的6.8 在安装之前要先安装wine，wine的安装就不说了。很简单

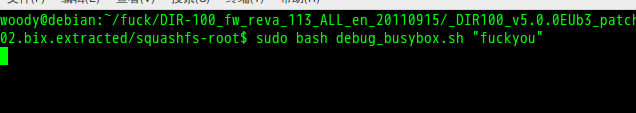
然后：

|  |
| --- |
| sudo wine <ida安装程序路径> |

会提示你装Mono装了吧，反正我装了。像0D一样，可以直接调试，也可以附加调试，我们以之前binwalk解包的bin目录下的busybox为例，，需要用到远程调试，所以我们需要建立一个脚本

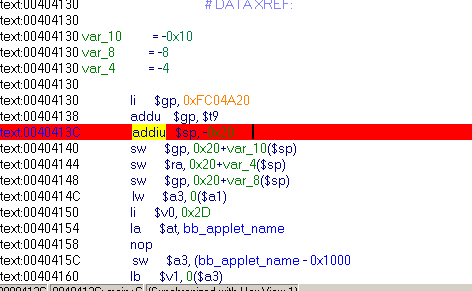
|  |
| --- |
| #! /bin/bash      INPUT="$1"     LEN=$(echo -n "$INPUT" | wc -c)     PORT="1234"      if [ "$LEN" == "0" ] || [ "$INPUT" == "-h" ] || [ "$UID" != "0" ]         then         echo -e "\nUsage: sudo $0 \n"     exit 1     fi     cp $(which qemu-mips-static) ./qemu     echo "$INPUT" | chroot . /qemu -E CONTENT\_LENGTH=$LEN -g $PORT /bin/busybox 2>/dev/null |

这个脚本如果稍微学一下linux shell就可以看懂，所以我不在赘述。  
运行之

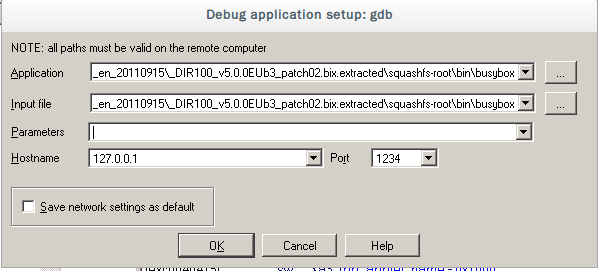


**直接调试**

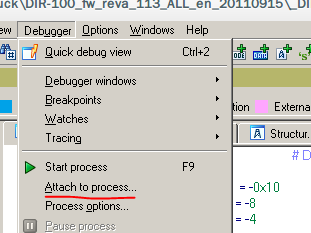
先用ida加载busybox,反汇编完成后，在main函数处插入断点

****

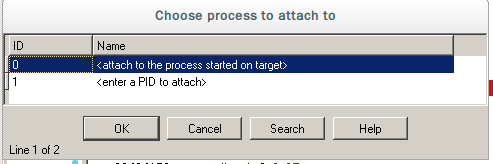
然后在Debuger中选择 Process Options,填上IP和端口，其他不变

****

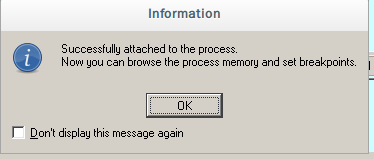
然后再Debuger中选择附加进程

****

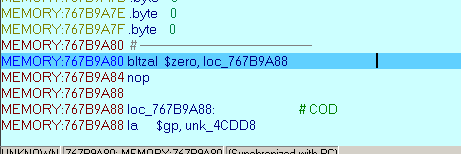
弹出一个对话框，然我们选择是附加到进程，还是选择，当然选第一个

****

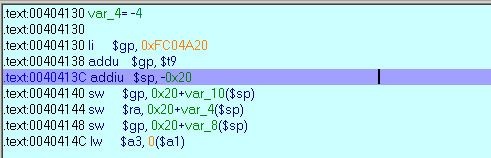
弹出一个对话框提示成功

****

点击OK，你发现没有断在我们之前设置的断点

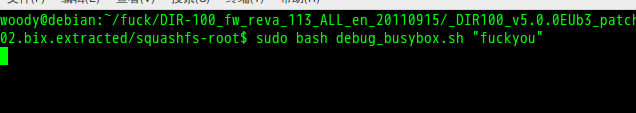
****

没关系，直接点运行，顺利中断在我们设置的断点处

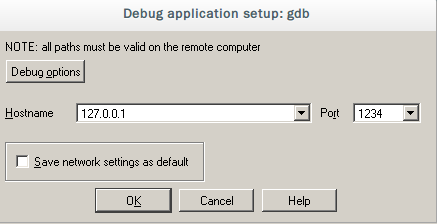
****

**附加调试**

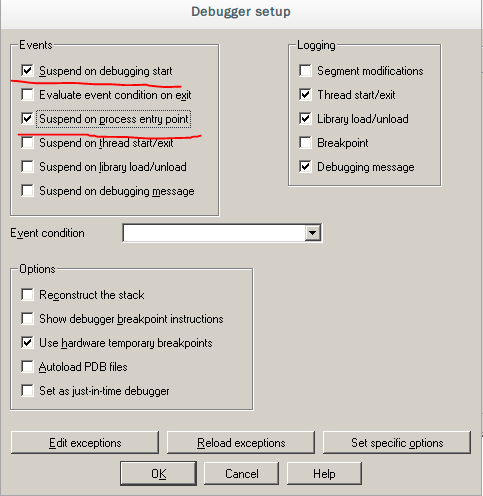
还是需要先运行这个脚本

****

打开IDA，Debugger->attach选择GDB Romote Debugger, 还是一样输入IP和端口

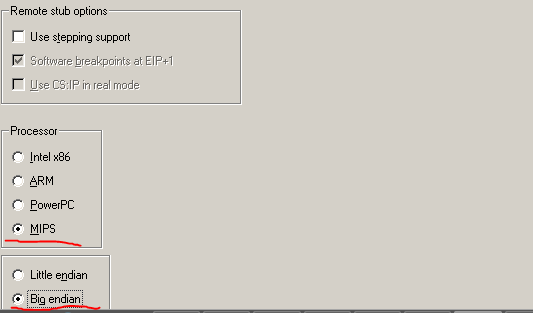
****

选择上面的Debug options

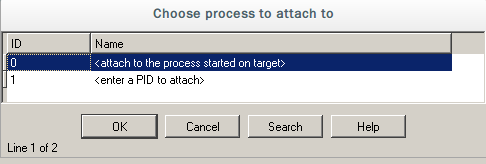
****

勾选如图两个事件，为啥呢，因为你附加的总要有中断的位置吧，就尝试在调试开始和程序入口点<br>

在点Set specific options设置一下处理器为啥要自己设置？因为IDA没有事先分析代码呀，所以只能自己来啦，具体如红线所指



然后还是熟悉的提示



点OK 中断到调试器



**吐槽与总结:**

1. **Debian是我认为适合技术人员使用的发行版，没有之一，我还记得乌版图没完的bug,Arch升级之后各种程序挂了一片，gentoo繁琐的安装，等等等**
2. **binwalk，很好用，嗯，我第一点没有提Kali,现在说说，Kali默认安装了binwalk,但是始终不能把固件完全解包，出来的使各种html或者狗血文件，原因是什么?Kali中的binwalk也就是个binwalk，没有其他INSTALL.md中的各种模块，这点严重的坑到了我，之前用的kali 1.0，装模块是还会出现依赖问题，现在新版出来不知道解决没有，反正我用它祖宗Debian了。**
3. **linux坑太多，一定要慎重选择发行版，就算你选择Debian，也一定选择stable，testing对人的品格是一种拷问。**
4. **善于翻文档，文档中有绝大部分问题的说明与解决办法。**
5. **善于使用Google,尤其是硬件安全这种新兴方向，百度你能度出个鸡毛来。啥都别想，果断Google。**
6. **英语，使用Google时经常出来的是英文页面，甚至是日文，日文怎么会，但是英文一定要看懂意思，不行就直接上Google翻译，虽然蛋疼点，总比一点都不懂强**
7. **关于调试，很明显直接调试比附加调试好一点，直接调试断点随便设置，还能利用IDA强大的符号分析，岂不美哉！**