**漏洞背景**：

Samba是在Linux和UNIX系统上实现SMB协议的一个软件，2017年5月24日Samba发布了4.6.4版本，中间修复了一个严重的远程代码执行漏洞，漏洞编号CVE-2017-7494，漏洞影响了Samba 3.5.0 之后到4.6.4/4.5.10/4.4.14中间的所有版本,，确认属于严重漏洞，可以造成远程代码执行。

**环境搭建**：

靶机环境    ：                Linux-mint 17 镜像网上随便下内核版本3.13.0-24-generic x86\_64

samba版本：                4.1.6-Ubuntu

然后配置smb服务配置  
vim /etc/samba/smb.confg

参考其他人分析在末行加入

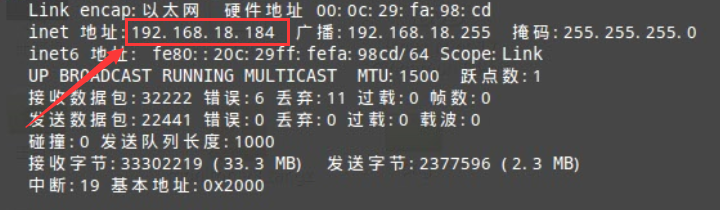


但是我自己测试 只需要就可以

  
重新reload smb的配置文件 重启服务  
/etc/init.d/samba restart  
/etc/init.d/samba reload  
有的机器上需要关闭防火墙

/etc/init.d/iptables stop

看当前靶机ip



攻击机环境： kali2017-amd64 .iso         ：[https://www.kali.org/downloads/](https://www.kali.org/downloads/" \t "_self)  直接安装镜像即可，linux kali 自带metasploit框架，利用起来非常的方便

Exploit\_CVE-2017-7494(msf-ruby)        ：[https://github.com/hdm/metasploit-framework/blob/0520d7cf76f8e5e654cb60f157772200c1b9e230/modules/exploits/linux/samba/is\_known\_pipename.rb](https://github.com/hdm/metasploit-framework/blob/0520d7cf76f8e5e654cb60f157772200c1b9e230/modules/exploits/linux/samba/is_known_pipename.rb" \t "_self)

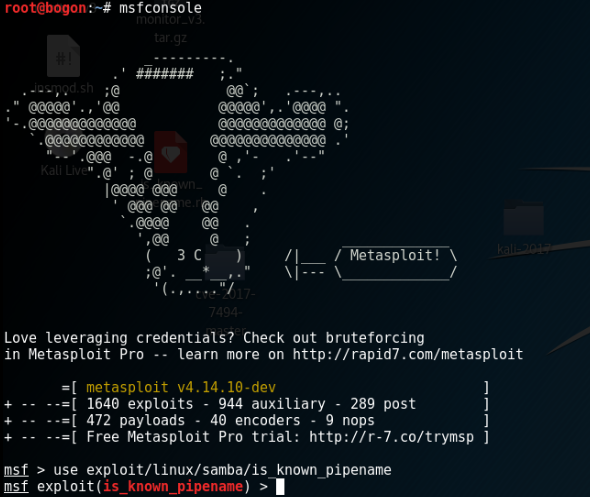
在攻击机上扫描目标靶机是否445 端口是否打开，因为samba服务开启需要开启445端口



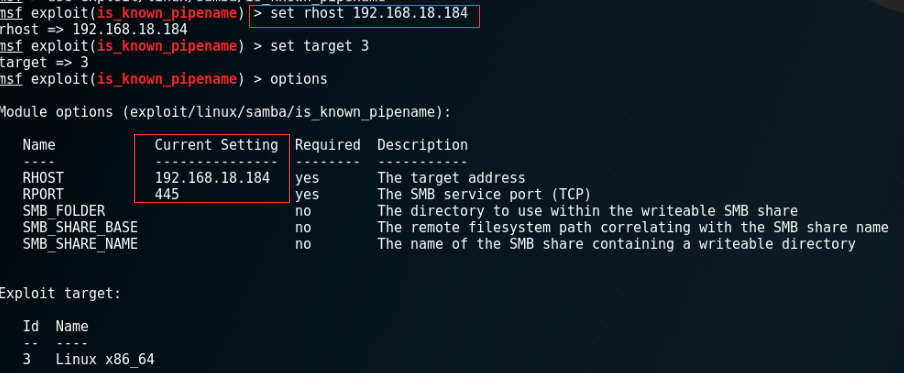
可以看到445和139端口open开启

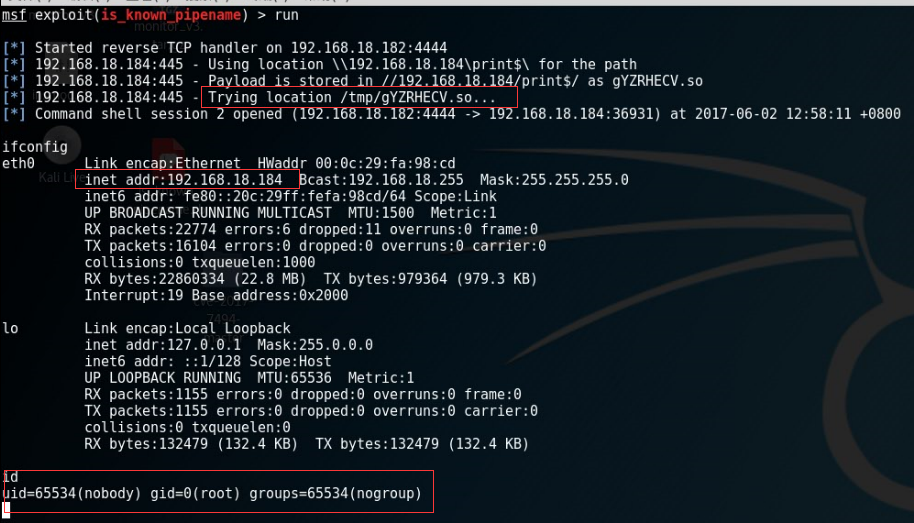
把down下来的ruby脚本is\_known\_pipename.rb拷贝至kali机器上的/usr/share/metasploit-framework/modules/exploits/linux/samba/目录下

启动启动msfconsole，并选择使用is\_known\_pipename模块：



设置目标机ip等，看下配置选项：

[](http://upload/tmp/707728_npkbi9liqfbdcz9.png)  
开始进行攻击：

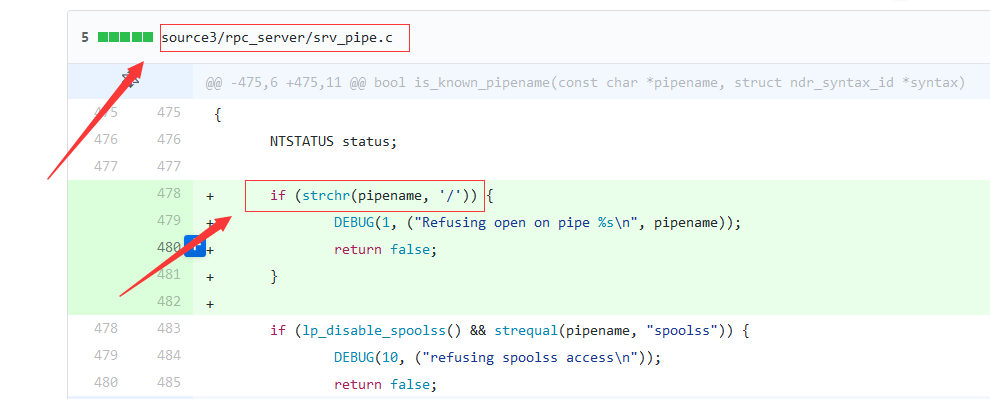
[](https://bbs.pediy.com/upload/attach/201706/707728_lm2ob7yxgvcmf94.png)

可以看到尝试着在靶机的共享目录下上传一个随机字符串生成的so文件，然后就会回连一个shell，看出ip就是靶机的ip

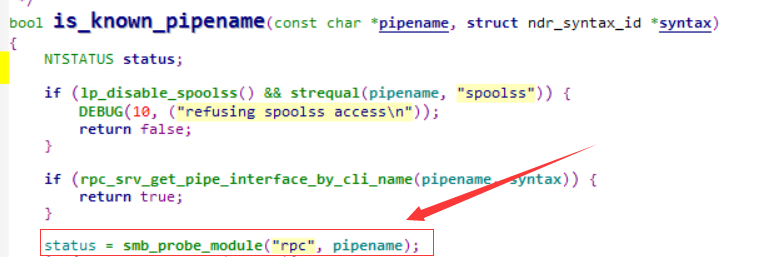
获取一个组ID为root的shell终端。

**漏洞代码分析：**

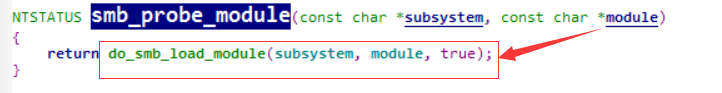
官方给出的漏洞补丁可以看出**：**

**[](http://upload/tmp/707728_3jovedinb0kxu9e.png)**

漏洞出现在source3/rpc\_server/srv\_pipe.c 的475行中也就是发生在is\_known\_pipename函数中对传进来的管道名称进行了过滤的处理，如果管道名称中含有了“/”的字符进行了过滤，其实更严格的话应该是判断首个字符串是不是‘/’来判断是否需要过滤，后续会有介绍，那么根据补丁来进行分析肯定管道名称带有'/'会触发漏洞，所以跟进代码看一下

、

再跟：

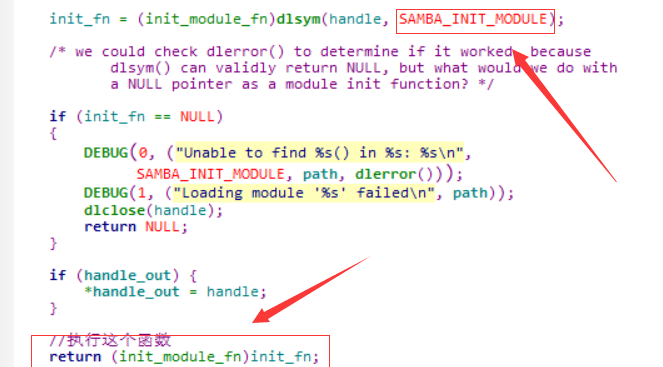


继续：

[](https://bbs.pediy.com/upload/attach/201706/707728_c4zav5zwkhlmnn6.png)

以samba-4.5.9.tar.gz源码为例分析：

可以看出参数pipename参数传递过来就是module\_name,如果首字符是'/'的话也就是管道的名称的首字符是'/'也就是传过来的是绝对路径的时候，就会跳到load\_module函数中，在跟进去就是会加载上传上来的so，而且导出函数



会以SAMBA\_INIT\_MODULE的函数名进行导出，然后进行执行导出函数，在代码中搜一下可以

#define SAMBA\_INIT\_MODULE "samba\_init\_module"

所以上传的so必须要是以samba\_init\_module函数名导出才可以加载，（但是后续分析ruby exp上传的so在dlopen的时候就会结束，但是会得到反弹shell不知道为什么）

**动态gdb跟踪调试：**

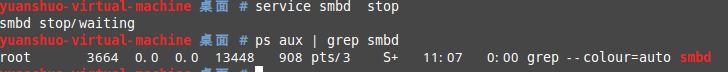
下载 samba-4.5.9.tar.gz    链接：[https://download.samba.org/pub/samba/stable/samba-4.5.9.tar.gz](https://download.samba.org/pub/samba/stable/samba-4.5.9.tar.gz" \t "_self)

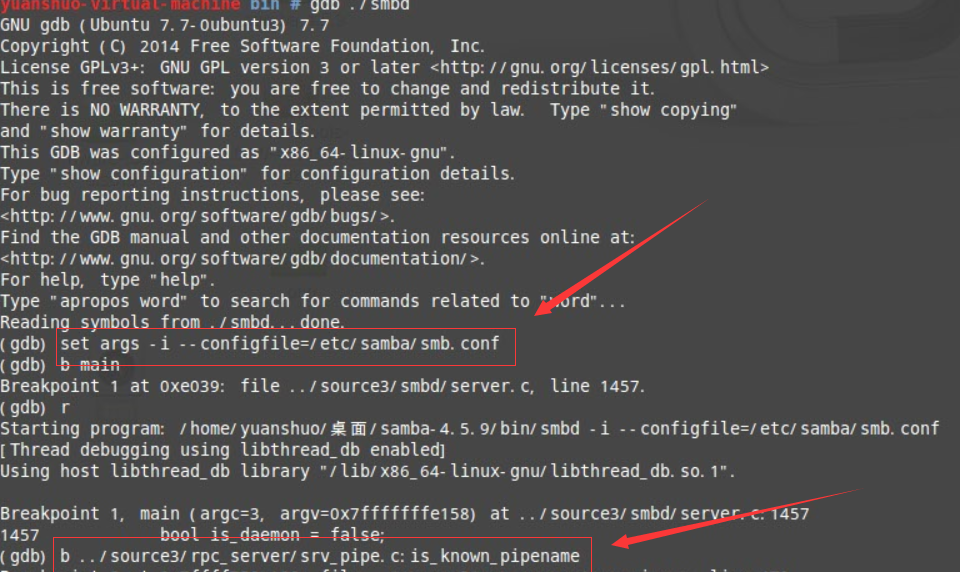
[./configure ... --enable-debug](https://download.samba.org/pub/samba/stable/samba-4.5.9.tar.gz)

[make](https://download.samba.org/pub/samba/stable/samba-4.5.9.tar.gz)

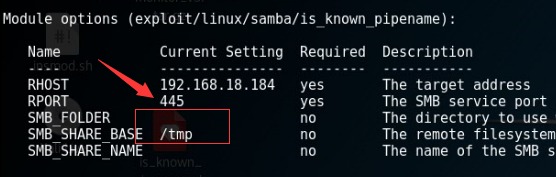
安装好对应的依赖，进行编译，因为本机的smbd服务正在运行，要首先stop系统对应的smbd的服务，然后在调试debug的smbd进程

关掉本机的smbd服务

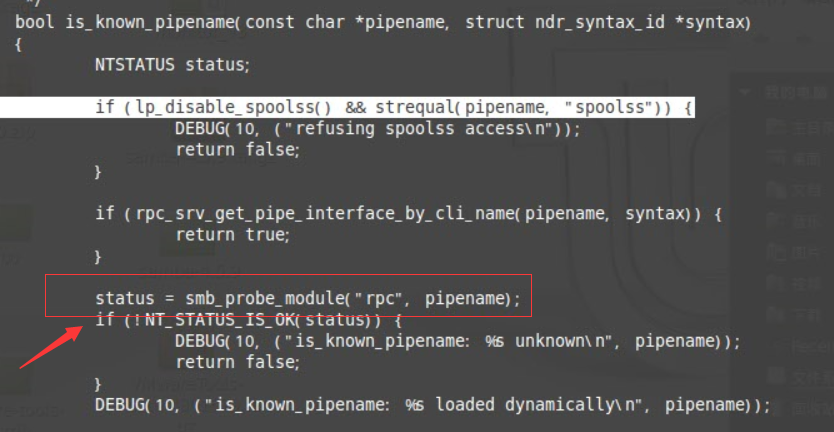


然后进行调试debug版的smbd程序，在源码目录中的bin中smbd,以交互模式和设置配置文件路径进行启动[](https://bbs.pediy.com/upload/attach/201706/707728_bmfvpmy2wh3x97x.png)  
已经知道是在srv\_pipe.c:is\_known\_pipename函数中，下断点，攻击机这时候发起攻击，因为为了调试所以直接就指定靶机的共享目录  /tmp

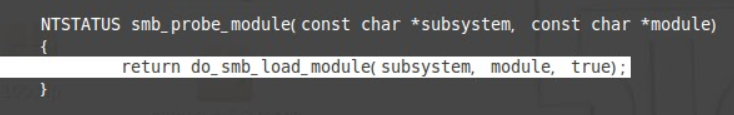
在exp  set SMB\_SHARE\_BASE /tmp



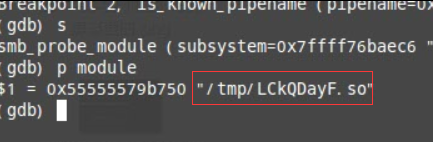
靶机这gdb 跟进：

[[](https://bbs.pediy.com/upload/attach/201706/707728_lu3pvtqa8o965np.png)](https://bbs.pediy.com/upload/attach/201706/707728_lu3pvtqa8o965np.png)

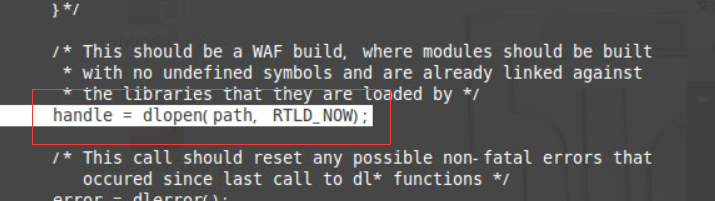
[然后跟进函数中：](https://bbs.pediy.com/upload/attach/201706/707728_lu3pvtqa8o965np.png)

[](https://bbs.pediy.com/upload/attach/201706/707728_lu3pvtqa8o965np.png)

打印module名称就是从攻击机上上传上来的恶意的so的名称，然后进入到load\_module函数中：



然后通过do\_smb\_load\_module->load\_module函数中，最后在load\_module中加载so，并且导出函数并且执行下图为加载上传的so文件：



然后加载执行，但是这个恶意的so中无论是gdb调试还是正常的运行中都会在dlopen之后退出，然后回弹一个远程shell，不再执行导出函数和执行函数的步骤，这块还不懂怎么回事，希望厉害的人可以解答一下，so就在附件中。

临时解决方案：

在samba的配置文件global节点中添加nt pipe support = 0，即关闭pipe支持（依赖pipe的功能受影响），然后重启服务。