Mimikatz and Active Directory Kerberos Attacks

By Sean Metcalf

Published: 2014-11-22 · Archived: 2026-04-05 13:29:43 UTC

Nov 22 2014

NOTE:
While this page will remain, the majority of the Mimikatz information in this page is now in the "Unofficial Mimikatz Guide & Command Reference" which will be updated on a regular basis.

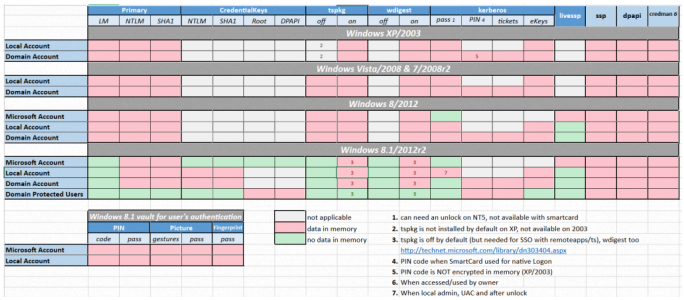
Mimikatz is the latest, and one of the best, tool to gather credential data from Windows systems. In fact I consider Mimikatz to be the "swiss army knife" of Windows credentials – that one tool that can do everything. Since the author of Mimikatz, Benjamin Delpy, is French most of the resources describing Mimikatz usage is in French, at least on his blog. The Mimikatz GitHub repository is in English and includes useful information on command usage.

Mimikatz is a Windows x32/x64 program coded in C by Benjamin Delpy (@gentilkiwi) in 2007 to learn more about Windows credentials (and as a Proof of Concept). There are two optional components that provide additional features, *mimidrv* (driver to interact with the Windows kernal) and *mimilib* (AppLocker bypass, Auth package/SSP, password filter, and sekurlsa for WinDBG). Mimikatz requires administrator or SYSTEM and often debug rights in order to perform certain actions and interact with the LSASS process (depending on the action requested).

After a user logs on, a variety of credentials are generated and stored in the Local Security Authority Subsystem Service, LSASS, process in memory. This is meant to facilitate single sign-on (SSO) ensuring a user isn't prompted each time resource access is requested. The credential data may include NTLM password hashes, LM password hashes (if the password is <15 characters), and even clear-text passwords (to support WDigest and SSP authentication among others. While you can prevent a Windows computer from creating the LM hash in the local computer SAM database (and the AD database), though this doesn't prevent the system from generating the LM hash in memory.

The majority of Mimikatz functionality is available in PowerSploit (PowerShell Post-Exploitation Framework) through the "Invoke-Mimikatz" PowerShell script which "leverages Mimikatz 2.0 and Invoke-ReflectivePEInjection to reflectively load Mimikatz completely in memory. This allows you to do things such as dump credentials without ever writing the mimikatz binary to disk." Mimikatz functionality supported by Invoke-Mimikatz is noted below.

Benjamin Delpy posted an Excel chart on OneDrive (shown below) that shows what type of credential data is available in memory (LSASS), including on Windows 8.1 and Windows 2012 R2 which have enhanced protection mechanisms reducing the amount and type of credentials kept in memory.



(Click image to embiggen)

One of the biggest security concerns with Windows today is “Pass the Hash.” Simply stated, Windows performs a one-way hash function on the user’s password and the result is referred to as a “hash.” The one-way hash algorithm changes the password in expected ways given the input data (the password) with the result being scrambled data that can’t be reverted back to the original input data, the password. Hashing a password into a hash is like putting a steak through a meat grinder to make ground beef – the ground beef can never be put together to be the same steak again. Pass the Hash has many variants, from Pass the Ticket to OverPass the Hash (aka pass the key).

The following quote is a Google Translate English translated version of the Mimikatz website (which is in French).

> Contrary to what could easily be imagined, **Windows does not use the password of the user as a shared secret,** but non-reversible derivatives: LM hash, NTLM, DES keys, AES … According to the protocol, the secret and the algorithms used are different:
>
> | **Protocol** | **Algorithm** | **Secret used** |
> |---|---|---|
> | LM | DES-ECB | LM Hash |
> | NTLMv1 | DES-ECB | NT Hash |
> | NTLMv2 | HMAC-MD5 | NT Hash |

Mimikatz OS support:

- Windows XP
- Windows Vista
- Windows 7
- Windows 8
- Windows Server 2003
- Windows Server 2008 / 2008 R2
- Windows Server 2012 / 2012 R2
- Windows 10

> Since Windows encrypts most credentials in memory (LSASS), they should be protected, but it is a type of reversible encryption (though creds are in clear-text). Encrypt works with LsaProtectMemory and

> decrypt with LsaUnprotectMemory.
>
> NT5 encryption types: RC4 & DESx
>
> NT6 encryption types: 3DES & AES

Mimikatz capabilities:

- **Dump credentials from LSASS** (Windows Local Security Account database) [sekurlsa module]
  - MSV1.0: hashes & keys (dpapi)
  - Kerberos password, ekeys, tickets, & PIN
  - TsPkg (password)
  - WDigest (clear-text password)
  - LiveSSP (clear-text password)
  - SSP (clear-text password)
- Generate Kerberos Golden Tickets (Kerberos TGT logon token ticket attack)
- Generate Kerberos Silver Tickets (Kerberos TGS service ticket attack)
- **Export certificates and keys** (even those not normally exportable).
- Dump cached credentials
- Stop event monitoring.
- Bypass Microsoft AppLocker / Software Restriction Polcies
- Patch Terminal Server
- Basic GPO bypass

*Items in **bold** denotes functionality provided by the PowerSploit Invoke-Mimikatz module with built-in parameters. Other mimikatz commands may work using the command parameter.*

Mimikatz Command Overview:

The primary command components are sekurlsa, kerberos, crypto, vault, and lsadump.
Sekurlsa interacts with the LSASS process in memory to gather credential data and provides enhanced capability over kerberos.

The Mimikatz kerberos command set enables modification of Kerberos tickets and interacts with the official Microsoft Kerberos API. This is the command that creates Golden Tickets. Pass the ticket is also possible with this command since it can inject Kerberos ticket(s) (TGT or TGS) into the current session. External Kerberos tools may be used for session injection, but they must follow the Kerberos credential format (KRB_CRED).
Mimikatz kerberos also enables the creation of Silver Tickets which are Kerberos tickets (TGT or TGS) with arbitrary data enabling AD user/ group impersonation.

The key required for ticket creation depends on the type of ticket being generated:

- Golden tickets require the KRBTGT account NTLM password hash.
- Silver tickets require the computer or service account’s NTLM password hash.

Crypto enables export of certificates on the system that are not marked exportable since it bypasses the standard export process.

Vault enables dumping data from the Windows vault.

Lsadump enables dumping credential data from the Security Account Manager (SAM) database which contains the NTLM (sometimes LM hash) and supports online and offline mode as well as dumping credential data from the LSASS process in memory. Lsadump can also be used to dump cached credentials. In a Windows domain, credentials are cached (up to 10) in case a Domain Controller is unavailable for authentication. However, these credentials are stored on the computer.

> These caches are located in the registry at the location `HKEY_LOCAL_MACHINE\SECURITY\Cache` (accessible SYSTEM).
> These entries are encrypted symmetrically, but we find some information about the user, as well as sufficient to verify the hash authentication.

Further down is a more detailed list of mimikatz command functionality.

Common Kerberos Attacks:

- **Pass The Hash**
  - On Windows, a user provides the userid and password and the password is hashed, creating the password hash. When the user on one Windows system wants to access another, the user's password hash is sent (passed) to the destination's resource to authenticate. This means there is no need to crack the user's password since the user's password hash is all that's needed to gain access.

> Contrary to what could easily be imagined, **Windows does not use the password of the user as a shared secret,** but non-reversible derivatives: LM hash, NTLM, DES keys, AES …

- **Pass the Ticket (Google Translation)**
  - Extract an existing, valid Kerberos ticket from one machine and pass it to another one to gain access to resoiurces as that user.

- **Over-Pass The Hash (aka Pass the Key) (Google Translation)**
  - Use the NTLM hash to obtain a valid user Kerberos ticket request.
  - The user key (NTLM hash when using RC4) is used to encrypt the Pre-Authentication & first data requests.
  - The following quote is a Google Translate English translated version of the Mimikatz website (which is in French):

> Authentication via Kerberos is a tad different. The client encrypts a `timestamp` from its user secret, possibly with parameters realm and iteration number sent from the server.
> If the secret is correct, the server can decrypt the `timestamp` (and the passage verify that the clocks are not too time-shifted).

Protocol	Secret (key) used

Kerberos	OF
	RC4 = **NT Hash!**
	AES128
	AES256

Yes, the RC4 key type available and enabled by default in XP 8.1 is our **NT hash!**

- **Kerberos Golden Ticket (Google Translation)**
  - The **Kerberos Golden Ticket** is a valid TGT Kerberos ticket since it is encrypted/signed by the domain Kerberos account (KRBTGT). The TGT is only used to prove to the KDC service on the Domain Controller that the user was authenticated by another Domain Controller. The fact that the TGT is encrypted by the KRBTGT password hash and can be decrypted by any KDC service in the domain proves it is valid.
  - Golden Ticket Requirements:
    * **Domain Name** [AD PowerShell module: (Get-ADDomain).DNSRoot]
    * **Domain SID** [AD PowerShell module: (Get-ADDomain).DomainSID.Value]
    * **Domain KRBTGT Account NTLM password hash**
    * **UserID for impersonation**.
  - The Domain Controller KDC service doesn't perform validate the user account until the TGT is older than 20 minutes old, which means the attacker can use a disabled/deleted user account or even a fictional user that doesn't exist in AD!

    Microsoft's MS-KILE specification (section 5.1.3):

    "Kerberos V5 does not provide account revocation checking for TGS requests, which allows TGT renewals and service tickets to be issued as long as the TGT is valid even if the account has been revoked. KILE provides a check account policy (section 3.3.5.7.1) that limits the exposure to a shorter time. KILE KDCs in the account domain are required to check accounts when the TGT is older than 20 minutes. This limits the period that a client can get a ticket with a revoked account while limiting the performance cost for AD queries."

  - Since the domain Kerberos policy is set on the ticket when generated by the KDC service on the Domain Controller, when the ticket is provided, systems trust the ticket validity. This means that even if the domain policy states a Kerberos logon ticket (TGT) is only valid for 10 hours, if the ticket states it is valid for 10 *years*, it is accepted as such.
  - The KRBTGT account password is never changed* and the attacker can create Golden Tickets until the KRBTGT password is changed (twice). Note that a Golden Ticket created to impersonate a user persists even if the impersonated user changes their password.
  - This crafted TGT requires an attacker to have the Active Directory domain's KRBTGT password hash (typically dumped from a Domain Controller).
  - The KRBTGT NTLM hash can be used to generate a valid TGT (using RC4) to impersonate any user with access to any resource in Active Directory.

    - The Golden Ticket (TGT) be generated and used on any machine, even one not domain-joined. The created TGT can be used without requiring Debug rights.
    - **Mitigation**: Limit Domain Admins from logging on to any other computers other than Domain Controllers and a handful of Admin servers (don't let other admins log on to these servers) Delegate all other rights to custom admin groups. This greatly reduces the ability of an attacker to gain access to a Domain Controller's Active Directory database. If the attacker can't access the AD database (ntds.dit file), they can't get the KRBTGT account NTLM password hash. Configuring Active Directory Kerberos to only allow AES may prevent Golden Tickets from being created. Another mitigation option is Microsoft KB2871997 which back-ports some of the enhanced security in Windows 8.1 and Windows 2012 R2.

- **Kerberos Silver Ticket**
    - The **Kerberos Silver Ticket** is a valid Ticket Granting Service (TGS) Kerberos ticket since it is encrypted/signed by the service account configured with a Service Principal Name for each server the Kerberos-authenticating service runs on.
    - While a Golden ticket is encrypted/signed with the KRBTGT, a Silver Ticket is encrypted/signed by the service account (computer account credential extracted from the computer's local SAM or service account credential).
    - Most services don't validate the PAC (by sending the PAC checksum to the Domain Controller for PAC validation), so a valid TGS generated with the service account password hash can include a PAC that is entirely fictitious – even claiming the user is a Domain Admin without challenge or correction.

> Since service tickets are identical in format to TGTs albeit with a different service name, all you need to do is specify a different service name and use the RC4 (NTLM hash) of the account password (either the computer account for default services or the actual account) and you can now issue service tickets for the requested service. Note: You can also use the AES keys if you happen to have them instead of the NTLM key and it will still work 😉
>
> It is worth noting, that services like MSSQL, Sharepoint, etc will only allow you to play with those services. The computer account will allow access to CIFS, service creation, and a whole host of other activities on the targeted computer. You can leverage the computer account into a shell with PSEXEC and you will be running as system on that particular computer. Lateral movement is then a matter of doing whatever you need to do from there 🙂

- **Service Account Password Cracking by attacking the Kerberos Session Ticket (TGS)**
    - NOTE: This attack does NOT require hacking tools on the network since it can be performed offline.
    - The Kerberos session ticket (TGS) has a component that is encrypted with the service's (either computer account or service account) password hash. The TGS for the service is generated and delivered to the user after the user's TGT is presented to the KDC service on the Domain Controller.
    - Since the service account's password hash is used to encrypt the server component, it is possible to request the service TGS and perform an offline password attack.

- Only normal Kerberos traffic is observed on the wire: the TGT is delivered to the Domain Controller along with a TGS request and response. At this point, no further network traffic is required.
- Service accounts typically have weak passwords and are rarely changed making these excellent targets. Computer account passwords are changed about every 30 days and are extremely complex making them virtually uncrackable.
- Finding interesting service accounts is as simple as sending a Service Principal Name query to the Global Catalog.
- Service accounts often have elevated rights in Active Directory and since only a Kerberos service ticket (TGS) is required to attack the service account's password, getting a TGS and saving it to another system to crack the password means this is a difficult attack to stop.
- **Mitigation**: Ensure all service accounts have long (>25 characters), complex passwords and only have the exact rights required (ensure the principle of least privilege).
- Tim Medin (@timmedin) describes this attack at his "Attacking Microsoft Kerberos: Kicking the Guard Dog of Hades" presentation at DerbyCon 2014.
  [Slides: https://www.dropbox.com/s/1j6v6zbtsdg1kam/Kerberoast.pdf?dl=0]
- In his DerbyCon2014 presentation, Tim Medin provided PowerShell code examples for requesting a TGS. I have modified it slightly to add the $SPN variable.

```
$SPN = "HTTP/sharepoint.adsecurity.org"
Add-Type -AssemblyNAme System.IdentityModel
New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "$SPN"
```

- **Pass the Cache** (*nix systems)
  - Linux/Unix systems (Mac OSX) store Kerberos credentials in a cache file. As of 11/23/2014, Mimikatz supports extracting the credential data for passing to Active Directory in a similar manner to the Pass the Hash/ Pass the Ticket method.

Mimikatz Commands:

- logonpasswords: `mimikatz # sekurlsa::logonpasswords`)
  - Extracts passwords in memory
- pth (pass the hash):

```
mimikatz # sekurlsa::pth /user:Administrateur /domain:chocolate.local /ntlm:cc36cf7a8514893efccd3324461
```

  - A fake identity is created and the faske identitt's NTLM hash is replaced with the real one.
  - "ntlm hash is mandatory on XP/2003/Vista/2008 and before 7/2008r2/8/2012 kb2871997 (AES not available or replaceable)"
  - "AES keys can be replaced only on 8.1/2012r2 or 7/2008r2/8/2012 with kb2871997, in this case you can avoid ntlm hash."
- ptt (pass the ticket):

```
mimikatz # kerberos::ptt
```

  - Enables Kerberos ticket (TGT or TGS) injection into the current session.
- tickets: `mimikatz # sekurlsa::tickets /export`
  - Identifies all session Kerberos tickets and lists/exports them.
  - sekurlsa pulls the Kerberos data from memory and can access all user session tickets on the computer.
- ekeys: `mimikatz # sekurlsa::ekeys`
  - Extract the Kerberos ekeys from memory. Provides theft of a user account until the password is changed (which may be never for a Smartcard/PKI user).
- dpapi: `mimikatz # sekurlsa::dpapi`
- minidump:

```
mimikatz # sekurlsa::minidump lsass.dmp
```

  - Perform a minidump of the LSASS process and extract credential data from the lsass.dmp. A minidump can be saved off the computer for credential extraction later, but the major version of Windows must match (you can't open the dump file from Windows 2012 on a Windows 2008 system).
- kerberos:

```
mimikatz # sekurlsa::kerberos
```

  - ```
    Extracts the smartcad/PIV PIN from memory (cached in LSASS when using a smartcard).
    ```
- debug:

```
mimikatz # privilege::debug
```
```

    - Sets debug mode for current mimikatz session enabling LSASS access.
- lsadump cache: (requires token::elevate to be SYSTEM)

```
mimikatz # lsadump::cache
```

    - Dumps cached Windows domain credentials from HKEY_LOCAL_MACHINE\SECURITY\Cache (accessible SYSTEM).

References:

- Benjamin Delpy's blog (Google Translate English translated version)
- Mimikatz GitHub repository
- Mimikatz Github wiki
- Mimikatz 2 Presentation Slides (Benjamin Delpy, July 2014)
- All Mimikatz Presentation resources on blog.gentilkiwi.com
- Excel chart on OneDrive that shows what type of credential data is available in memory (LSASS), including on Windows 8.1 and Windows 2012 R2 which have enhanced protection mechanisms.
- PAC Validation issue aka the Silver Ticket description from the Passing the Hash Blog

Sean Metcalf

I improve security for enterprises around the world working for TrustedSec & I am @PyroTek3 on Twitter.
Read the About page (top left) for information about me. :)
https://adsecurity.org/?page_id=8

Comments have been disabled.
