# Mimikatz: The Finest in Post-Exploitation

## Overview

The MS-ISAC continuously observes attacks using the post-exploitation credential stealing tool Mimikatz. Many cyber threat actors (CTAs) use this open source tool to escalate privileges and move laterally to maximize their attacks against state, local, tribal, and territorial (SLTT) governments. In one incident, a local government reached out to the MS-ISAC Cyber Incident Response Team (CIRT) for assistance on antivirus detection and network scanning alerts. MS-ISAC CIRT investigations found a network intrusion via a compromised third-party vendor. The CTA used Mimikatz to dump passwords, gain access to accounts with administrative privileges, and laterally move across the network with the harvested credentials. After compromising additional user accounts and systems, the CTA exfiltrated data from affected file shares.

Security Researcher, Benjamin Delpy initially created Mimikatz in 2007 to demonstrate how the Microsoft Windows operating system's handling of credentials and associated authentication protocols were vulnerable to attack. Delpy decided to publicly release Mimikatz to prove Microsoft needed to change the way they secured credentials. With constant updates, 16 modules, and ease of use, Mimikatz is popular with both penetration testers and CTAs.

Mimikatz is also often used in attacks because it can extract plaintext passwords, hashes, pin codes, and Kerberos tickets from memory. Additionally, the tool uses these credentials for pass-the-hash[1] and pass-the-ticket[2] attacks, as well as to build Kerberos Golden Tickets and Kerberos Silver Tickets.[3] It can also be used to extract certificates and private keys. Furthermore, many current malware variants use Mimikatz in their attack sequences. For example, TrickBot uses Mimikatz to scrape credentials from LSASS.exe via the lsadump module, which is used to escalate privileges and spread laterally across the network. CTAs also use Mimikatz's lsadump module to carry out other attacks, such as DCSync, DCShadow, and the Kerberos Golden Ticket compromise.

## DCSync

A DCSync attack simulates domain controller (DC) behavior to retrieve password data through domain replication. A successful DCSync attack provides a CTA with administrative access to all information from the DC. This effectively compromises the entire active directory (AD) forest. To accomplish this, the CTA needs access to a privileged account with domain replication rights to start replication protocols mimicking the DC. This can be accomplished with Mimikatz's lsadump module. Mimikatz uses Microsoft's Directory Replication Service Remote Protocol and the GetNCChanges application programming interface (API) function to mimic the behavior of the DC and ask other DCs for copies of information.

## DCShadow

[1]Pass-the-hash involves extracting the Widows NTLM hash string from a target and using it to login as that user. CTAs use this because they do not need to crack passwords and only need the hash string for authentication.

[2]Pass-the-ticket involves extracting a user's Kerberos ticket and using it for authentication. It is basically the same as pass-the-hash, except it works for newer versions of Windows.

[3]CTAs use Kerberos Silver Tickets to exploit a Windows feature, which allows them to use network services. Kerberos uses the ticket granting service to issue tickets to access services on a network, but Windows does not always check these tickets so CTAs can use them to gain further network access.

In a DCShadow attack, the CTA creates a fake DC to replicate malicious changes to the legitimate DC. It uses legitimate APIs to modify existing data in the AD which the DC then uses for maintaining persistence and obfuscation. Often times, the CTA's main goal is to avoid detection while compromising the AD infrastructure. Many attacks that compromise the AD infrastructure create a lot of noise and are easily detected. Unlike these options, a DCShadow attack creates a fake DC and uses legitimate APIs to prevent detection during log analysis, helping the CTA bypass security controls and maintain network persistence.

The CTA needs to obtain domain admin privileges for this attack, which can be done through Mimikatz's lsadump module. During this attack, the lsadump module registers the compromised workstation as a DC in the AD. This is done through changing the AD's configuration schema and the workstation's Kerberos service principal names (SPNs) values. Ultimately, the CTA leverages the compromised system to effectively create a legitimate DC via replication.

## Kerberos Golden Ticket Compromise

A successful Kerberos Golden Ticket compromise gives the CTA access to all AD-connected systems. Mimikatz is effective at compromising single and multi-forest environments. When creating a Golden Ticket, Mimikatz includes SID history, which allows CTAs to expand their access and work across trusts. The ability to compromise all forests and any AD-connected system makes the Kerberos Golden Ticket Compromise one of the most dangerous abilities Mimikatz possesses.

To start this compromise, the CTA needs access to a privileged account that can access the AD DC. Second, they need access to the Kerberos Ticket Generating Ticket (KRBTGT) account. This account is used for encrypting all Kerberos authentication tokens for the DC. The DC uses this account to decrypt Kerberos tickets for validation and the password and username for this account never changes. Upon leveraging these accounts, CTAs use the Mimikatz lsadump module to extract password hashes, the domain name, and domain security identifier (SID). The CTA uses these components to create a Kerberos Golden Ticket for the account to perform a pass-the-ticket attack. This pass-the-ticket attack provides access to the entire DC.

### Recommendations

SLTT governments should review the best practices outlined in the CIS Controls and CIS Benchmarks, which are part of CIS SecureSuite. The MS-ISAC also recommends the following actions to help prevent and detect compromises leveraging Mimikatz.

### Preventing Initial Compromise:

- Provide end-user training to help users identify suspicious emails or links. Ensure users are aware of support policies and procedures for assistance.
- Implement filters at the email gateway to filter out emails with known malspam indicators, such as known malicious subject lines senders, or IP addresses.
    - If you do not have a policy regarding suspicious emails, consider creating one. Part of this policy should specify that all suspicious emails should be reported to the security and/or IT departments.
- Mark external emails with a banner denoting it is from an external source. This will assist users in detecting spoofed emails.
- To lower the chance of spoofed or modified emails, implement Domain Message Authentication Reporting and Conformance (DMARC) policy and verification, starting by implementing the Sender Policy Framework (SPF) and the Domain Keys Identified Mail (DKIM) standards.

- Adhere to the principal of least privilege, ensuring that users have the minimum level of access required to accomplish their duties. Limit administrative credentials to designated administrators.
- Keep operating systems and software up-to-date, including the organization's antimalware solutions.

## Preventing Mimikatz Attacks:

- Utilize application allow-listing technology on all assets to ensure that only authorized software executes and all unauthorized software, such as Mimikatz, is blocked from executing on assets.
- Debugger privileges should only be given to those who need it, such as system programmers, and should not be left as default for all local admins. Debugger privileges should be disabled for local admins on all servers and workstations. An attacker needs this privilege while running Mimikatz in order to interact with specific processes.
- Make sure WDigest protocol is disabled. This ensures that plain text passwords are not stored in the Local Security Authority Subsystem Service (LSASS).
- Enable Restricted Admin Mode. This makes sure Remote Desktop Protocol (RDP) sessions are not storing credentials in the memory of the host.
    - Additionally, enable this as the default for all RDP sessions initiated by domain members.
- Enable the security group “Protect Users” to protect local admins and prevent NT LAN Manager (NTLM) password hashes or plain-text credentials in LSASS from leaking. This enforces Kerberos authentication.
- Restrict admin accounts and services to specific workstations and servers.
- Disable local password caching. By default, Windows systems will cache the last 10 password hashes used for easy authentication. Preventing this ensures that CTAs will not have easy access to these password hashes.
- Disable storage of plaintext passwords in the active directory. Unless absolutely needed, disable the Reversible Encryption setting in account policies on Windows systems.
- Enforce Network Level Authentication for RDP sessions to ensure authentication is over Transport Layer Security (TLS). This helps prevent password sniffing, which would allow a CTA access to a privileged user’s password.
- In Windows systems older than Server 2012 R2 and Windows 8.1, LSA protection should be enabled because it enforces local security policies as well as validates local and remote sign-ins for users.
- Ensure local admin accounts have complex and unique passwords.
- Limit domain admin account permissions to DCs and limited servers.
- Do not allow a user to be a local administrator for multiple systems.

## Detecting DCSync Compromises:

- Use network monitoring by adding all DC IP addresses to the Replication Allow list. Then, configure an IDS to alert on DsGetNCChange requests that are not from IP addresses on the Replication Allow list.

## Detecting DCShadow Compromises:

- Monitor network traffic associated with data replication between DCs and from DCs to or from normal hosts for analysis, such as API functions DrsAddEntry, DrsReplicaAdd, and GetNCChanges.
- If possible, consider alerting on replications of the AD objects (Audit Detailed Directory Service Replication Events 4928 and 4929).
- Monitor for Event ID 5137 and 5141. These events can help you detect the creation of rogue DCs.

- Monitor SPNs associated with services not in the DC organizational unit. CTAs will set SPNs for Directory Replication Service and Remote Protocol interface because this can be done without being logged. The CTA's rogue DC has to use these two SPNs for the replication process to authenticate as a service and complete the compromise. You can use this to investigate whether computers that have these services are on the DC organizational unit. If they aren't, it may be a sign there is a compromise.
- Consider using the open source tool UncoverDCShadow. This tool is a PowerShell utility designed to discover DCShadow attacks.

**Detecting Kerberos Golden Ticket Compromises:**

- Alert on known behavior that may indicate a Golden Ticket attack is taking place. This can be done by auditing all Kerberos authentication and credential use events.
- If previous Kerberos Golden Tickets have been reset and a CTA attempts to use an invalid ticket, the Domain Controller will generate an event, specifically Event ID 4769.

**Mitigation of the DCSync and Kerberos Golden Ticket Compromises:**

- Change local admin account passwords and ensure they have complex, unique passwords.
- Limit domain admin account permissions to DCs and limited servers.
- Do not allow a user to be a local administrator for multiple systems.
- Specifically, for Kerberos Golden Ticket Compromises:
  - To invalidate compromised Golden Tickets, reset the built-in KRBTGT account password twice. This invalidates any previous golden tickets that have been created.

**Mitigation of DCShadow Compromises:**

Mitigating DCShadow compromises is difficult since the attack abuses legitimate system features. However, there are a few ways to mitigate this attack.

- Monitor and alert on Event ID 5137 and 5141. These events can help you detect the creation of rogue DCs.
- Once rogue DCs are identified, delete them in the AD site container, specifically the Configuration Namespace.