

Forum:[Hack Forums](#)

Thread: [The Flagship Red Team & Defense Playbook](#)

??????????? - August 31, 2025, 3:23 am

Credential Access & Dumping (2025 Edition)
The Flagship Red Team & Defense Playbook
Introduction

Credentials are the crown jewels in any environment. Once an attacker dumps them, lateral movement, persistence, and even full domain compromise often follow.

From the days of Mimikatz to modern tools like Nanodump, credential theft techniques have evolved, and so have the defenses.

This flagship thread is designed as a complete playbook for red teamers and defenders.

We will cover 15 credential access methods, each with: Deep technical description (what it is & why it works)

Examples (commands, snippets, variations)

Red Team Use (how attackers apply it in real ops)

Defense (detections, logs, countermeasures)

The goal is to give you a laboratory reference you can return to again and again.

1. Mimikatz (Classic LSASS Dumping)

Description:

Created by Benjamin Delpy (gentilkiwi), Mimikatz changed everything when it revealed Windows stored plaintext credentials, NTLM hashes, and Kerberos tickets in LSASS memory.

It's over a decade old, yet remains one of the most recognized tools in red teaming and APT campaigns.

Examples:

Dump all logon passwords:

Code `mimikatz.exe "privilege::debug" "sekurlsa::logonpasswords" exit`

Dump Kerberos tickets:

Code `mimikatz.exe "privilege::debug" "sekurlsa::tickets" exit`

Pass-the-Hash directly:

Code `mimikatz.exe "sekurlsa::pth /user:admin /domain:corp.local /ntlm:ABCD1234..."`

Red Team Use:

- Quickest method to extract plaintext creds + hashes.
- Still effective against systems without Credential Guard.
- Seen in major attacks like NotPetya, WannaCry, ransomware ops.

Defense:

- Enable Windows Credential Guard (isolates LSASS).
- Sysmon Event ID 10 -> lsass.exe memory access.
- Monitor unsigned binaries reading LSASS.
- EDR signatures on sekurlsa module activity.

2. Nanodump (Modern LSASS Dumper)

Description:

With EDRs blocking raw LSASS access, researchers created Nanodump, a stealth dumper that uses indirect syscalls and avoids detection by patching minidump structures.

It writes a LSASS dump file that many AVs overlook.

Examples:

Dump LSASS to file:

Code `nanodump.exe --write C:\Temp\lsass.dmp`

Run via Cobalt Strike:

Code `beacon> execute-assembly nanodump.exe --handle`

Red Team Use:

- Works where Mimikatz is blocked.
- Runs fully in-memory in C2 frameworks.
- Adopted in modern APT & red team tradecraft.

Defense:

- Monitor minidump file creation events.
- Watch LSASS handle duplication attempts (Event ID 10).
- EDR with syscall inspection can catch Nanodump.
- Restrict SeDebugPrivilege usage to admins only.

3. comsvcs.dll Minidump Trick

Description:

Windows ships with comsvcs.dll, which has a hidden function `MiniDumpW`` that can dump process memory.

Attackers abuse this with `rundll32.exe` to dump LSASS without external tools.

Examples:

Dump LSASS with built-in binary:

Code `rundll32.exe C:\Windows\System32\comsvcs.dll, MiniDump 1234 C:\lsass.dmp full` (where 1234 = LSASS PID)

Red Team Use:

- Abuse Windows' own binaries (LOLBIN).
- Evades signature-based detection.
- Requires no external tools.

Defense:

- Flag `rundll32` calls to comsvcs.dll.
- Detect creation of suspicious *.dmp files.
- Event ID 4688 -> unusual `rundll32` command lines.

4. ProcDump Abuse

Description:

[Procdump], part of Sysinternals, is a legit debugging tool from Microsoft.

Attackers abuse it to dump LSASS memory because it's signed and often trusted.

Examples:

Dump LSASS:

Code `procdump.exe -ma lsass.exe C:\Temp\lsass.dmp`

Silent dumping:

Code `procdump.exe -accepteula -ma -silent lsass.exe lsass.dmp`

Red Team Use:

- Signed by Microsoft -> bypasses basic controls.
- Simple, effective, works in many environments.
- Used by APT32, FIN groups, and ransomware gangs.

Defense:

- Monitor ProcDump execution against LSASS.
- Hunt for dmp file creation in temp/system32.
- Block ProcDump via AppLocker/WDAC if not required.

5. LSASS Handle Duplication

Description:

Instead of calling `OpenProcess` on `lsass.exe` directly, attackers duplicate a handle from another trusted process (like Task Manager) that already has LSASS access.

This bypasses naive handle protections.

Examples:

Duplicate LSASS handle in C#:

Code `DuplicateHandle(hSourceProc, hLsass, hDestProc, out hDup, 0, false, DUPLICATE_SAME_ACCESS);`

Nanodump handle stealing variant:

Code `nanodump.exe --duplicate --target-pid 1234`

Red Team Use:

- Evades EDR hooks that watch for direct LSASS access.
- Used in modern malware for stealth dumping.
- Effective when combined with syscall injection.

Defense:

- Monitor `DuplicateHandle` API use against LSASS.
- Sysmon Event ID 10 -> unusual handle requests.
- Track processes inheriting handles unexpectedly.

6. SAM & SYSTEM Hive Extraction

Description:

Windows stores local account hashes in the SAM (Security Account Manager). These are protected with keys from the SYSTEM hive.

By extracting both, attackers can dump NTLM hashes for offline cracking.

Examples:

Dump with `reg.exe`:

Code reg save HKLM\SAM C:\Temp\SAM.save
reg save HKLM\SYSTEM C:\Temp\SYSTEM.save

Extract with Mimikatz:

Code mimikatz.exe "lsadump::sam /system:SYSTEM.save /sam:SAM.save"

Red Team Use:

- Offline hash extraction -> safe from live EDR hooks.
- Common for privilege escalation from local admin to domain lateral movement.
- Works even when LSASS dumping is blocked.

Defense:

- Monitor reg.exe saving SAM or SYSTEM.
- Event ID 4656 -> handle access to sensitive hives.
- EDR alerts on unauthorized hive dumping.

7. DPAPI Abuse (Decrypting Secrets)

Description:

Windows uses DPAPI (Data Protection API) to encrypt credentials (e.g., browser passwords, Wi-Fi keys, RDP creds).

If attackers steal the master key, they can decrypt DPAPI-protected secrets.

Examples:

Export DPAPI masterkeys:

Code mimikatz.exe "dpapi::masterkey /in:C:\Users\User\AppData\Roaming\Microsoft\Protect\..."

Decrypt Chrome logs:

Code mimikatz.exe "dpapi::chrome /in:C:\Users\User\AppData\Local\Google\Chrome\User Data\Default\Login Data"

Red Team Use:

- Extract saved browser credentials.
- Decrypt RDP and Wi-Fi stored passwords.
- Persist access to cloud accounts even after password resets.

Defense:

- Monitor file access to DPAPI masterkeys.
- Harden with Windows Hello / Credential Guard.
- Restrict local admin -> prevents DPAPI key theft.

8. Credential Manager Dump

Description:

Windows Credential Manager stores web and system credentials. Attackers extract them with Mimikatz or PowerShell.

Examples:

Dump with cmdkey:

Code cmdkey /list

Dump with Mimikatz:

Code mimikatz.exe "vault::cred /patch"

PowerShell method:

Code Get-StoredCredential -Target "Domain:target.com"

Red Team Use:

- Steal saved RDP/SMB credentials.
- Use in privilege escalation and lateral movement.
- Often overlooked by defenders.

Defense:

- Monitor vaultsvc service activity.
- Event ID 4672 (special privileges assigned).
- Disable unnecessary credential storage.

9. Browser Credential Theft

Description:

Browsers (Chrome, Edge, Firefox) store saved logins locally, often protected by DPAPI. Attackers target these SQLite/JSON files and decrypt them.

Examples:

Chrome login DB:

Code C:\Users\User\AppData\Local\Google\Chrome\User Data\Default>Login Data

Dump with LaZagne:

Code lazagne.exe browsers

PowerShell script to extract:

Code \$path = "\$env:USERPROFILE\AppData\Local\Google\Chrome\User Data\Default>Login Data"

Get-ChromePasswords -Database \$path

Red Team Use:

- Extract SaaS/cloud credentials (Google, AWS, O365).
- Useful for phishing escalation.
- Often yields high-value tokens.

Defense:

- Monitor access to browser login DBs.
- Force enterprise password managers (don't allow browser saves).
- Alert on LaZagne/SharpChrome activity.

10. LSA Secrets Extraction

Description:

The LSA (Local Security Authority) stores cached service account passwords, RDP creds, and trust secrets.

Attackers dump these to pivot into service accounts or domain trust exploitation.

Examples:

Dump LSA with Mimikatz:

Code mimikatz.exe "lsadump::secrets"

Extract with secretsdump.py (Impacket):

Code secretsdump.py -sam SAM -system SYSTEM -security SECURITY LOCAL

Red Team Use:

- Extract service account creds.
- Dump cached RDP passwords.
- Abuse domain trust secrets for forest pivoting.

Defense:

- Event ID 4611 -> suspicious LSA plug-in load.
- Monitor registry access to SECURITY hive.
- EDR alerts on lsadump::secrets usage.

11. Kerberos Ticket Dumping

Description:

Kerberos tickets (TGTs & TGSs) are stored in memory and can be stolen for impersonation. Tools like Mimikatz and Rubeus allow attackers to extract them and perform Pass-the-Ticket.

Examples:

Dump Kerberos tickets with Mimikatz:

Code mimikatz.exe "sekurlsa::tickets"

Extract with Rubeus:

Code Rubeus dump /nowrap

Inject TGT into current session:

Code Rubeus ptt /ticket:ticket.kirbi

Red Team Use:

- Persist as a domain user without needing plaintext password.
- Pivot across systems with valid Kerberos tickets.
- Combine with Golden Ticket attacks for domain dominance.

Defense:

- Monitor unusual Kerberos ticket requests (Event ID 4769).
- Detect long-lived tickets or non-standard lifetimes.
- Deploy Kerberos Armoring (FAST).

12. DCSync Attack

Description:

Instead of dumping LSASS, attackers can impersonate a domain controller and request replication via DCSync.

This yields NTLM hashes for all domain accounts.

Examples:

Run DCSync with Mimikatz:

Code mimikatz.exe "lsadump::dcsync /domain:corp.local /user:krbtgt"

Impacket secretsdump:

Code secretsdump.py domain/admin:pass@DC-IP

Red Team Use:

- Full domain compromise without touching LSASS.
- Stealthy, works remotely.
- Used in high-profile APT attacks.

Defense:

- Event ID 4662 -> Replication requests.
- Alert when non-DCs request replication.
- Restrict Replicating Directory Changes permissions.

13. Token Impersonation

Description:

Windows creates tokens for logged-in users. Attackers can steal or duplicate tokens to impersonate other accounts (including SYSTEM).

Examples:

Impersonate token with Mimikatz:

Code `mimikatz.exe "token::elevate"`

Steal token in PowerShell:

Code `Invoke-TokenManipulation -ImpersonateUser "Administrator"`

Red Team Use:

- Privilege escalation to SYSTEM.
- Lateral movement by reusing tokens.
- Combine with pass-the-hash/ticket for chaining.

Defense:

- Event ID 4624 (logons) with unusual token usage.
- Monitor for `SeImpersonatePrivilege`.
- Harden services against token theft.

14. Pass-the-Hash (PtH)

Description:

Instead of cracking NTLM hashes, attackers reuse them directly to authenticate.

This bypasses the need for plaintext passwords.

Examples:

PtH with Mimikatz:

Code `mimikatz.exe "sekurlsa::pth /user:admin /domain:corp.local /ntlm:ABCD1234..."`

Using CrackMapExec:

Code `cme smb 10.0.0.5 -u admin -H ABCD1234...`

Red Team Use:

- Move laterally in Windows environments.
- Often combined with DCSync hashes.
- Still works if SMB signing is disabled.

Defense:

- Enable SMB signing & Kerberos only.
- Monitor NTLM authentication attempts.
- Disable NTLM where possible.

15. Cloud Credential Theft

Description:

With enterprises moving to the cloud, attackers target Azure AD, AWS, and Google Cloud credentials.

These are often stored locally in config files, CLI caches, or environment variables.

Examples:

Dump AWS keys from CLI config:

Code `cat %USERPROFILE%\aws\credentials`

Extract Azure tokens with AADInternals:

Code `Get-AADIntAccessTokenForMSGraph -Credentials $creds`

Red Team Use:

- Cloud persistence after on-prem compromise.
- Pivot into SaaS resources.
- Weaponized in ransomware double-extortion.

Defense:

- Monitor CLI tools (aws.exe, az.exe) for abnormal usage.
- Enforce MFA + conditional access policies.
- Scan endpoints for leaked tokens/keys.

Final Defense Playbook

Enable Credential Guard to isolate LSASS.

Block vulnerable tools (ProcDump, comsvcs abuse) with AppLocker/WDAC.

Hunt for SAM/System hive exports.

Monitor for DCSync attempts (Event 4662).

Detect Kerberos ticket anomalies (Event 4769).

Use Sysmon for process injection, handle duplication, and file creation events.

Rotate service account creds regularly.

Monitor cloud key usage & enforce MFA.

Conclusion

Credential access remains the number one goal for attackers.

From classic LSASS dumping with Mimikatz to modern cloud token theft, the game keeps evolving.

Red teamers must master these techniques for realistic ops, and defenders must learn to detect them to stay ahead.

Which of these credential attacks have you tested in your own labs?

Do you think cloud key abuse will fully replace LSASS dumping in the future? Share your thoughts below!

CL1B - August 31, 2025, 4:12 am

Spoiler ([Click to View](#))

migalo57x - August 31, 2025, 6:33 am

dm now

num3riz - September 29, 2025, 4:01 am

Thank uuu

???????????? - October 18, 2025, 2:03 pm

(Aug 31, 2025 06:33 AM)migalo57x Wrote: [CITE](#) dm now
Regarding?

(Sep 29, 2025 04:01 AM)num3riz Wrote: [\[CITE\]](#) Thank uuu
You are always welcome.