Pentesting Lab

Active Directory

Possegger, Prodinger, Schauklies, Schwarzl, Pongratz

27.05.2024

Summer 2023/24, www.iaik.tugraz.at/ptl

Table of contents

Overview

Overview: What is Active Directory?

- Microsoft Active Directory is a directory service for Windows domain networks.
- It is based on standard technologies
  - LDAP
  - Kerberos
  - DNS
- (You may remember those from previous lectures)
- But there is a lot more...
  - NETBIOS
  - NTLM
  - LLMNR
  - AD Certificate Services

Overview: A brief history

- Released with Windows 2000 Server edition
- Support retrofitted back to Windows 95
- Features and security have been greatly enhanced since then
- Still needs to be backwards compatible
  - That's were the problems start...

Overview: Why (pentest) Active Directory?

- About 90% of Fortune 1000 companies use Active Directory[1]
- You are almost guaranteed to encounter it in an internal pentest
- Active Directory basically does everything
- Very hard to do everything right
  - A single mistake can lead to disaster

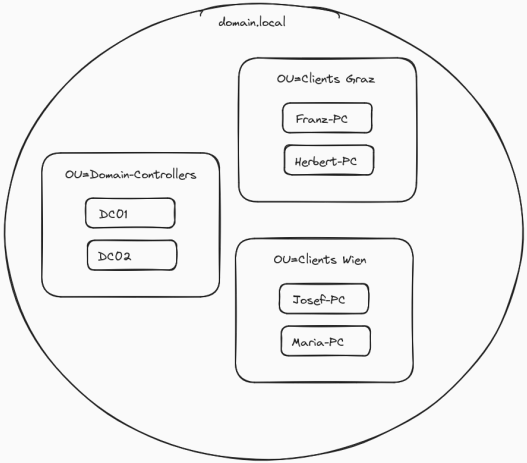
[1] https://www.frost.com/frost-perspectives/active-directory-holds-the-keys-to-your-kingdom-but-is-it-secure/

Structure: The Core Components

- Domain Controller
  - Hosts the Active Directory
    - Users & Groups
    - Group Policies
    - Access Rights
    - Startup-Scripts
    - Name Resolution
    - etc. pp.
  - Verifies credentials and access rights
  - Replicates the database with other Domain Controllers

Domains and Forests

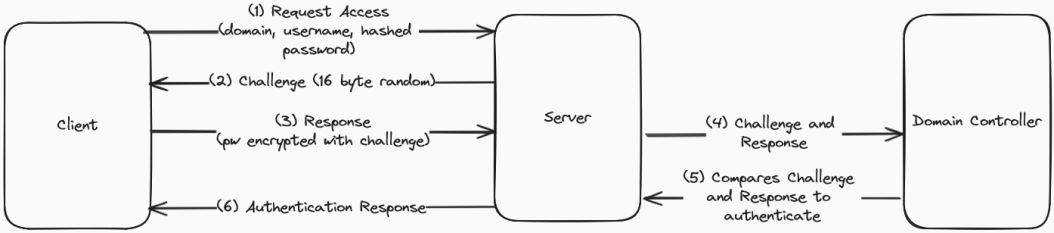
- Domain(s)
  - A logical grouping of network objects (users, computers, groups)
  - Establishes boundaries and ACLs
  - Organizational Units (OUs)
    - Hierarchically managed containers
    - Grouping similar assets together (e.g. Client-Workstations)
- Forest(s)
  - Group of Domains
  - Sharing a common schema and configuration

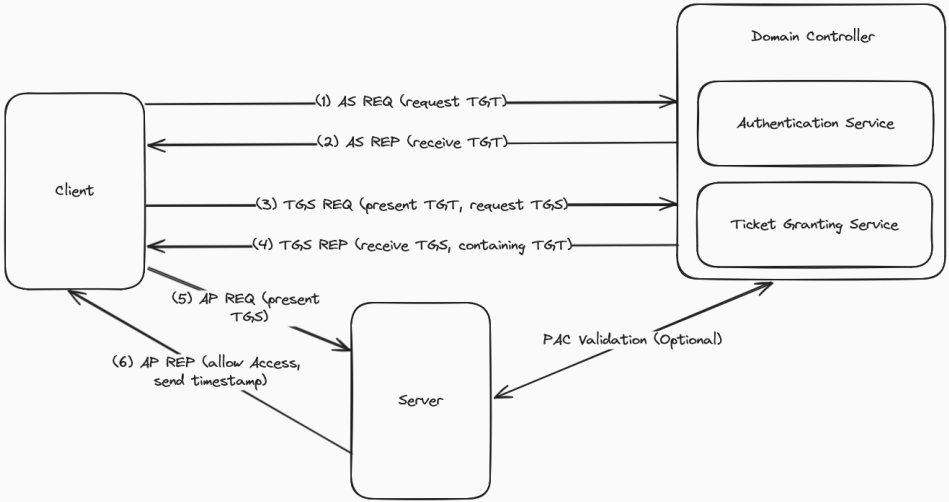

domain.local
OU=Clients Graz
Franz-PC
Herbert-PC
OU=Domain-Controllers
DC01
DC02
OU=Clients Wien
Josef-PC
Maria-PC

- DNS
  - Resolving Domain Names to IP Addresses
  - Crucial for a working environment (Kerberos, Certificates etc.)
  - But there is a fallback mechanism
- Active Directory Certificate Services
  - Certificate Management
  - Issuing certs based on templates
  - Certificates are used for Encryption, Signing and Authentication

- Special Guest: SMB
  - Used for fileshares and remote administration
  - Not technically part of AD
  - Tightly connected and required for Group Policies and Startup Scripts

- Either via NTLM (legacy)
- Or Kerberos (modern)
- NTLM is vastly insecure and allows lots of attacks
- Kerberos is more modern but has some problems too


(1) Request Access
(domain, username, hashed password)
(2) Challenge (16 byte random)
(3) Response
(pw encrypted with challenge)
(6) Authentication Response
Client
Server
(4) Challenge and Response
(5) Compares Challenge and Response to authenticate
Domain Controller


Domain Controller
Authentication Service
Ticket Granting Service
Client
Server
(1) AS REQ (request TGT)
(2) AS REP (receive TGT)
(3) TGS REQ (present TGT, request TGS)
(4) TGS REP (receive TGS, containing TGT)
(5) AP REQ (present TGS)
(6) AP REP (allow Access, send timestamp)
PAC Validation (Optional)

Tooling

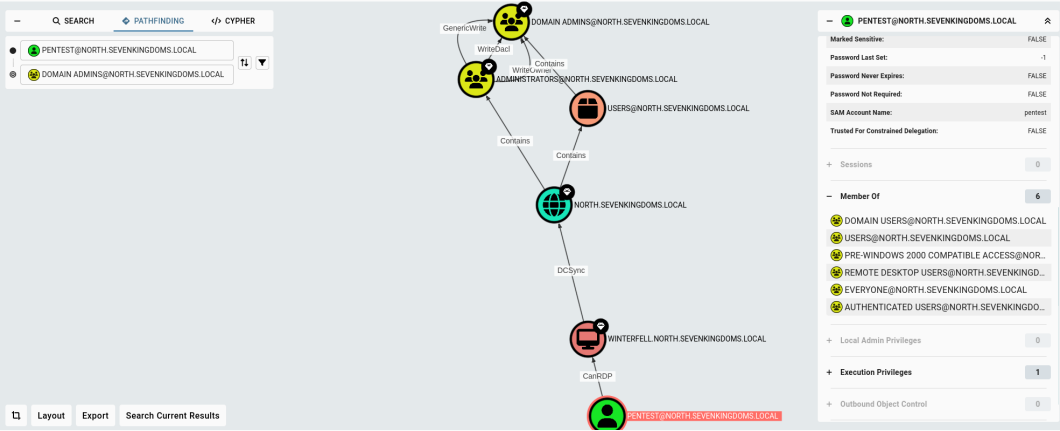
- There are some tools you will always need to be familiar with when offensively working with AD
- You will need them on EVERY engagement
- Know how they work, their output and their limitations!

NetExec (formerly CrackMapExec)

- NetExec[a] is the replacement for CrackMapExec[b]
- It's the swiss army tool for pentesting Active Directory Environments
- It allows authentication, information gathering and code execution over multiple channels
- It can run vulnerability scans, enumerate targets, dump credentials and deploy your C2
- https://www.netexec.wiki/getting-started/installation/installation-on-unix

[a] https://github.com/Pennyw0rth/NetExec
[b] https://github.com/byt3bl33d3r/CrackMapExec

- It's the de-facto standard for (offensively) enumerating AD relationships
- Writes all necessary information into a neo4j DB...
- ... Which makes it extremely easy to search
- https://github.com/SpecterOps/BloodHound
- https://github.com/BloodHoundAD/SharpHound/releases/


SEARCH
PATHFINDING
CYPHER
PENTEST@NORTH.SEVENKINGDOMS.LOCAL
DOMAIN ADMINS@NORTH.SEVENKINGDOMS.LOCAL
Layout
Export
Search Current Results
DOMAIN ADMINS@NORTH.SEVENKINGDOMS.LOCAL
GenericWrite
WriteDacl
Contains
WriteOwner
ADMINISTRATORS@NORTH.SEVENKINGDOMS.LOCAL
USERS@NORTH.SEVENKINGDOMS.LOCAL
Contains
Contains
NORTH.SEVENKINGDOMS.LOCAL
DCSync
WINTERFELL.NORTH.SEVENKINGDOMS.LOCAL
CanRDP
PENTEST@NORTH.SEVENKINGDOMS.LOCAL
PENTEST@NORTH.SEVENKINGDOMS.LOCAL
Marked Sensitive: FALSE
Password Last Set: -1
Password Never Expires: FALSE
Password Not Required: FALSE
SAM Account Name: pentest
Trusted For Constrained Delegation: FALSE
Sessions 0
Member Of 6
DOMAIN USERS@NORTH.SEVENKINGDOMS.LOCAL
USERS@NORTH.SEVENKINGDOMS.LOCAL
PRE-WINDOWS 2000 COMPATIBLE ACCESS@NOR...
REMOTE DESKTOP USERS@NORTH.SEVENKINGD...
EVERYONE@NORTH.SEVENKINGDOMS.LOCAL
AUTHENTICATED USERS@NORTH.SEVENKINGDO...
Local Admin Privileges 0
Execution Privileges 1
Outbound Object Control 0

- Active-Directory Auditing Tool
- Shows you a health score of the general AD Environment
- Very useful for identifying major misconfigurations
- https://github.com/vletoux/pingcastle

Indicators

Domain Risk Level: 65 / 100

It is the maximum score of the 4 indicators and one score cannot be higher than 100. The lower the better

Compare with statistics

Privacy notice

Stale Object : 31 /100 — 6 rules matched

It is about operations related to user or computer objects

Trusts : 1 /100 — 1 rules matched

It is about connections between two Active Directories

Privileged Accounts : 40 /100 — 4 rules matched

It is about administrators of the Active Directory

Anomalies : 65 /100 — 14 rules matched

It is about specific security control points

Reconnaissance

Overview

- Goal: Find as much information as possible
- Tooling
  - BloodhoundAD[2]
  - PowerView[3]
  - Certipy[4]
  - Snaffler[5]
  - Kerbrute[6]
  - Get-GPPPassword[7]

[2] https://github.com/SpecterOps/BloodHound
[3] https://github.com/PowerShellMafia/PowerSploit/blob/master/Recon/PowerView.ps1
[4] https://github.com/ly4k/Certipy
[5] https://github.com/SnaffCon/Snaffler
[6] https://github.com/ropnop/kerbrute
[7] https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Get-GPPPassword.ps1

- Does our user have local admin rights?
- Can we connect via RDP to another machine?
- Passwords where they shouldn't be?
  - Group Policies
  - User / Computer descriptions
  - File Shares
  - AD-Attributes
- Users with weak passwords?
- Any old systems with known vulnerabilities?

- BloodhoundAD is perfect for this!
- You can find out quickly if your user indirect control of another object
- Example:
  - You just pwned a helpdesk user
  - Helpdesk users have the ability to reset passwords of other users
  - Now you can reset the password of an IT-Administrator
  - Use their account to connect to a server and run mimikatz to gather even more passwords
  - If a Domain-Admin had a session on this server, you just pwned everything

- There are a lot of places for (almost) cleartext passwords to be stored in AD
- Group Policies can store AutoLogon passwords which can be decrypted
  - Use Get-GPPPassword.ps1[8]
- Some administrators are not aware that descriptions can be read by everyone

```
nxc ldap <hostname> -u <user> -p <pass> -M get-desc-users
```

- Other attributes can store passwords as well:

```
nxc ldap <hostname> -u <user> -p <pass> -M get-unixUserPassword -M
    getUserPassword
```

[8] https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Get-GPPPassword.ps1

- Usually, there is a lockout policy for number of password attempts
- Instead of trying many passwords for one user...
- we are going to use one password for many users!
- Users (and Admins!) tend do use guessable passwords
  - username = password
  - Summer2024!
  - 'Company'1234!
  - Init01!
  - etc.

Spraying Passwords

- Start by acquiring a list of domain users:

```
nxc ldap <hostname> -u <user> -p <pass> --active-users > active.txt
tail active.txt -n+5 | awk -F ' ' '{ print $5 }' > domain_users.txt
```

- Highly recommended: read password policy:

```
nxc smb <hostname> -u <user> -p <pass> --pass-pol
```

- Then, use kerbrute to spray your passwords:

```
./kerbrute passwordspray -d <domain> domain_users.txt Winter2022
./kerbrute passwordspray -d <domain> domain_users.txt --user-as-pass
```

- Most companies use some sort of knowledgebase
- Searching through those is recommended for every engagement
- Snaffler[9] automates this process for shares
- But sometimes you may want to do it more manually
- From PowerView.ps1[10], you can use Check-ShareAccess:

```
Find-DomainShare -CheckShareAccess
```

- And then search through them manually
- Also search local filesystems of servers / workstations (e.g. C:\tmp)

[9] https://github.com/SnaffCon/Snaffler
[10] https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

Spoofing / Coercion

- Goal: Get other systems to authenticate to us
- Tooling
  - Responder[11]
  - Inveigh[12]
  - mitm6[13]
  - Powermad[14]
  - Coercer[15]

[11] https://github.com/lgandx/Responder
[12] https://github.com/Kevin-Robertson/Inveigh
[13] https://github.com/dirkjanm/mitm6
[14] https://github.com/Kevin-Robertson/Powermad
[15] https://github.com/p0dalirius/Coercer

- LLMNR / NBNS Spoofing
- Adding a DNS Wildcard
- Create a fake DHCPv6 Server that provides a fake DNS Server
- Use Print Spooler / other RPC calls to force remote authentication
- Crack the captures hashes or relay them

- Lots of legacy protocols still in use
- LLMNR / NBNS are multicast without any authentication
- Windows queries various protocols for name resolution:
  - Local hosts file
  - DNS-Server
  - LLMNR / NBNS
- Anyone can answer!

- Sometimes, the AD-DNS Server allows creation of DNS Records
- This is useful for machines to add their own name
- Sometimes, all users or even ”Anonymous” can add records
- You can add wildcard entries
- Further reading: https://www.netspi.com/blog/technical-blog/network-pentesting/exploiting-adidns/

- By default, Windows (since Vista) prefers IPv6 to IPv4
- If a network does not provide a DHCPv6 server...
- ...become one yourself!
- By becoming the preferred DHCP, you can set a preferred DNS server too
- Further reading: https://redfoxsec.com/blog/ipv6-dns-takeover/

- There are several methods you can use to get a Server (or DC) to connect to another system
  - PetitPotam[16]
  - PrinterBug[17]
  - DFSCoerce[18]
  - There are many more!
- You'll usually want the target to connect to you though...

[16] https://www.prosec-networks.com/blog/petit-potam-ntlm-relay-angriff/
[17] https://www.thehacker.recipes/ad/movement/mitm-and-coerced-authentications/ms-rprn
[18] https://www.bleepingcomputer.com/news/microsoft/new-dfscoerce-ntlm-relay-attack-allows-windows-domain-takeover/

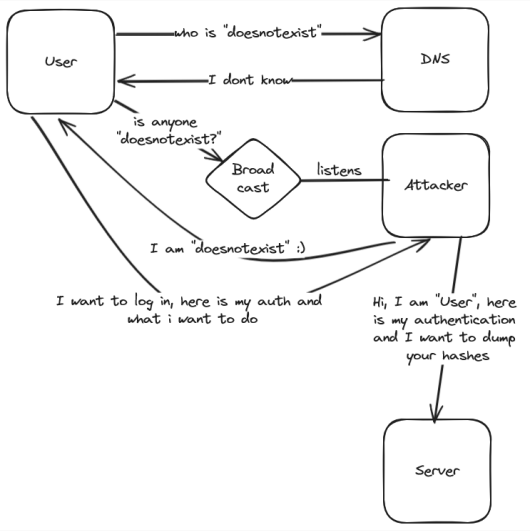
NTLM Relaying

- Goal: Don't want to crack Net-NTLMv2 hashes? Relay them!
- Tooling
  - ntlmrelayx[19]
  - LdapRelayScan[20]
  - Inveigh[21]

[19] https://github.com/fortra/impacket/blob/master/examples/ntlmrelayx.py
[20] https://github.com/zyn3rgy/LdapRelayScan
[21] https://github.com/Kevin-Robertson/Inveigh

- If you manage to get an authentication request from another system / user, you can use the ”authentication” part with a different ”payload”
- Relay authentication from higher-privileged accounts
- You can relay to many services:
  - SMB: Allows code execution if account is an administrator
  - LDAP: Allows reading / writing LDAP Attributes
  - HTTP: Attack Certificate Services
- Further Reading:
  https://trustedsec.com/blog/a-comprehensive-guide-on-relaying-anno-2022

The Big Picture



Active Directory Certificate Services

- Goal: Exploit various misconfigurations in ADCS to gain elevated privileges
- Tooling
  - certipy[22]
  - Certify[23]

[22] https://github.com/ly4k/Certipy
[23] https://github.com/GhostPack/Certify

Attack Scenarios

- There are various documented misconfigurations in Certificate Templates that allow attacks
- They are dubbed ESC1 - ESC14, some are easy to exploit, some pretty hard
- ESC1
  - A user can enroll a certificate and specify a custom UPN
  - This allows them to create a certificate which is valid for anyone they choose (like ”Administrator@domain.local”)
  - You can authenticate with a valid certificate, giving you instant Domain-Admin rights
- ESC4
  - A user can edit a Certificate Template, allowing them to enable ESC1
- ESC8
  - NTLM Relay to the Certificate Service HTTP endpoint

Attack Scenarios

- The other ESCs are a little more advanced and would take quite some time to explain
- ESC1 - ESC8: https://posts.specterops.io/certified-pre-owned-d95910965cd2
- ESC9 and ESC10: https://research.ifcr.dk/certipy-4-0-esc9-esc10-bloodhound-gui-new-authentication-and-request-methods-and-more-7237d88061f7
- ESC11: https://blog.compass-security.com/2022/11/relaying-to-ad-certificate-services-over-rpc/
- ESC12: https://pkiblog.knobloch.info/esc12-shell-access-to-adcs-ca-with-yubihsm
- ESC13: https://posts.specterops.io/adcs-esc13-abuse-technique-fda4272fbd53
- ESC14: https://posts.specterops.io/adcs-esc14-abuse-technique-333a004dc2b9
- You can find a good overview here: https://www.thehacker.recipes/ad/movement/ad-cs

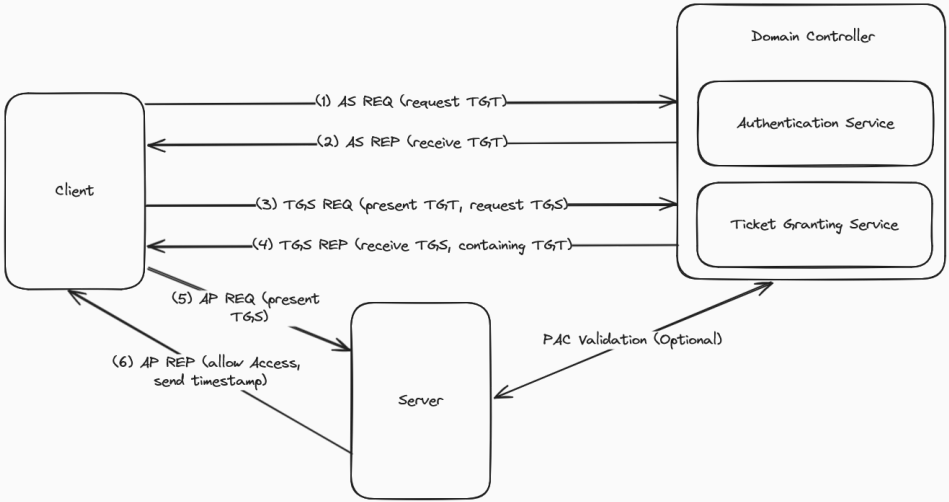
Kerberos

- Goal: Exploit features in NTLM's successor, Kerberos
- Tooling
  - Rubeus[24]
  - Mimikatz[25]

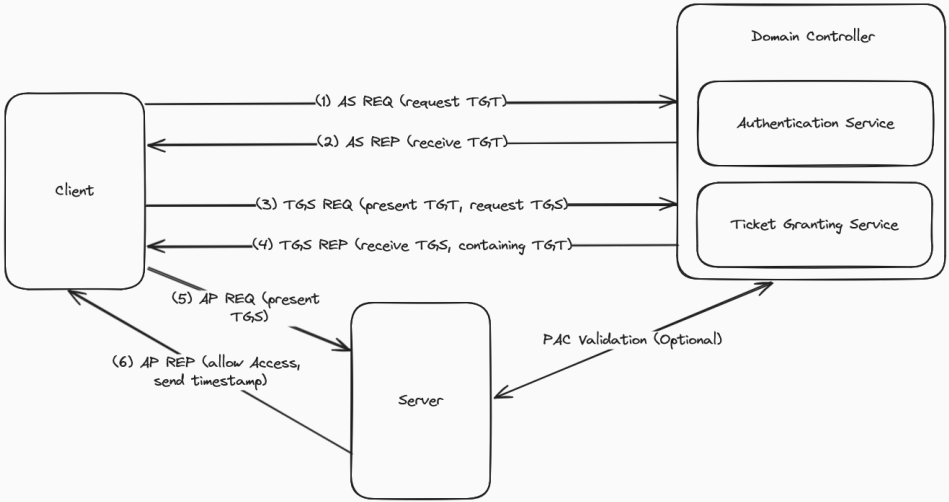
[24] https://github.com/GhostPack/Rubeus
[25] https://github.com/gentilkiwi/mimikatz

- Kerberoasting / AS-REP Roasting
- Constrained / Unconstrained Delegation

- You can send a TGS-REQ (Service Ticket Request) for any Service-Account (SPN) in the domain
- You need a TGT first, so you need to do the AS-REQ stuff as a Domain-User first
- The TGS-REP you get back is encrypted with the SPN's hash
- This hash can be cracked offline, although it is pretty hard


Domain Controller
Authentication Service
Ticket Granting Service
Client
Server
(1) AS REQ (request TGT)
(2) AS REP (receive TGT)
(3) TGS REQ (present TGT, request TGS)
(4) TGS REP (receive TGS, containing TGT)
(5) AP REQ (present TGS)
(6) AP REP (allow Access, send timestamp)
PAC Validation (Optional)

- This works very similar to Kerberoasting
- The AS-REQ contains a username, the desired service to access, and a timestamp encrypted with the users password
- The Authentication Service then checks if it can decrypt the timestamp using the password hash the Domain-Controller has stored
- For users that have a special flag (do not require pre-authentication) set however, you can skip the whole timestamp stuff
- Which means, you do NOT need a valid domain user password for this attack!
- However, you do need to know the username of the account you want to request a TGT for
- If you manage to do that, you can try to crack the encrypted password stored in the TGT you received


Domain Controller
Authentication Service
Ticket Granting Service
Client
Server
(1) AS REQ (request TGT)
(2) AS REP (receive TGT)
(3) TGS REQ (present TGT, request TGS)
(4) TGS REP (receive TGS, containing TGT)
(5) AP REQ (present TGS)
(6) AP REP (allow Access, send timestamp)
PAC Validation (Optional)

- Thankfully, you don't have to actually understand it:

```
nxc ldap <hostname> -u <user> -p <pass> --kerberoasting output.txt
nxc ldap <hostname> -u <user> -p '' --asreproast output.txt
```

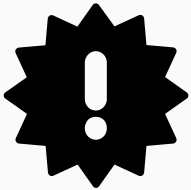
- A system with ”Unconstrained Delegation” enabled will store Tickets in memory
- Which means, if you gain administrative rights on such a system, you can dump and use saved tickets
- You can now Coerce another System to authenticate to this Unconstrained Delegation System, and use its ticket
- (This is very similar to NTLM relaying attacks)
- Further reading: https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/domain-compromise-via-unrestricted-kerberos-delegation

Try it yourself

- You have just been booked to conduct an internal pentest!
- They have not yet heard of ”security”...
- ...but you are here to change that!
- They do not believe they can be hacked (three totally unrelated ransomware incidents in the past did not change that stance)
- Show them your mad 1337 skills and pwn their whole infrastructure!
- But make sure you don’t break anything. They have VERY IMPORTANT BUSINESS to do.

- Connect to your WireGuard VPN
- You will find the GOAD-Light Environment[a]
- Try to do the common Enumeration / Recon / Exploitation Tasks we discussed
- You are allowed to read and follow walkthroughs, but I want you to understand and document what you did
- **LLMNR/NBNS Poisoning does not work in this environment**
- Due to the nature of the VPN connection, also coerce attacks to your system will fail
- However, you could theoretically still do it...

[a] https://github.com/Orange-Cyberdefense/GOAD?tab=readme-ov-file



- You are in a **shared** environment
- Treat this environment as a ”production” setup (just like a real pentest!):
  - Do not change any passwords of users or machines you did not create
  - Do not (willingly) destroy or block services
  - RTFM of any tools you use
  - notify your customer (me) immediately if you suspect you broke something
  - Attacking other VPN users is forbidden.
- The WireGuard connection, as well as the jumphost itself are strictly out of scope

- You will start with a normal domain user for both domains:
  - Username: north.sevenkingdoms.local\pentest
  - Password: TestMe2024!
  - Username: sevenkingdoms.local\renly.baratheon
  - Password: lorastyrell
- Good Luck finding the flags :)