SIGPwny

Purple Team FA2025 • 2025-10-14

Active Directory I

Ronan Boyarski

ctf.sigpwny.com

sigpwny{Domain Expansion}

Overview

- Active Directory Overview
  - Why does it exist
  - How does it work
  - Why it's a good target
  - Domain Controllers
- Kerberos Protocol
  - Kerberoasting & AS-REProasting
- NTLM & Net-NTLMv2
  - Pass-the-hash revisited
  - Responder & Hash Relaying
- Offensive Flow
  - Recon
  - Lateral movement
  - Domain Dominance

Why & How

Why have AD?

- Think about the system of University of Illinois
- 3 campuses, in total 100k+ students, so 25k+ students every year
- They all need
    - An email account
    - Access to school computers (Windows and Linux)
    - Everyone needs a different “view” of the computer
    - Access to different locations

- UIUC has 35k+ computers alone

How do we manage them?

Why have AD?

- Built to centralize authentication, authorization, and user management in enterprise Windows environments
- Introduced by Microsoft in 1999, built to merge LDAP (directory access), Kerberos (authentication) and DNS (name resolution) into one monolithic enterprise management solution
- LDAP is the Active Directory protocol (like HTTP is for the web)
- If we didn't have this, we would be connecting computers in an ad-hoc fashion (decentralized)

Why have AD?

- Built to centralize authentication, authorization, and user management in enterprise Windows environments
- Introduced by Microsoft in 1999, built to merge LDAP (directory access), Kerberos (authentication) and DNS (name resolution) into one monolithic enterprise management solution
- LDAP is the Active Directory protocol (like HTTP is for the web)
- If we didn't have this, we would be connecting computers in an ad-hoc fashion (decentralized)
- **Putting all of your trust in one place is dangerous…**

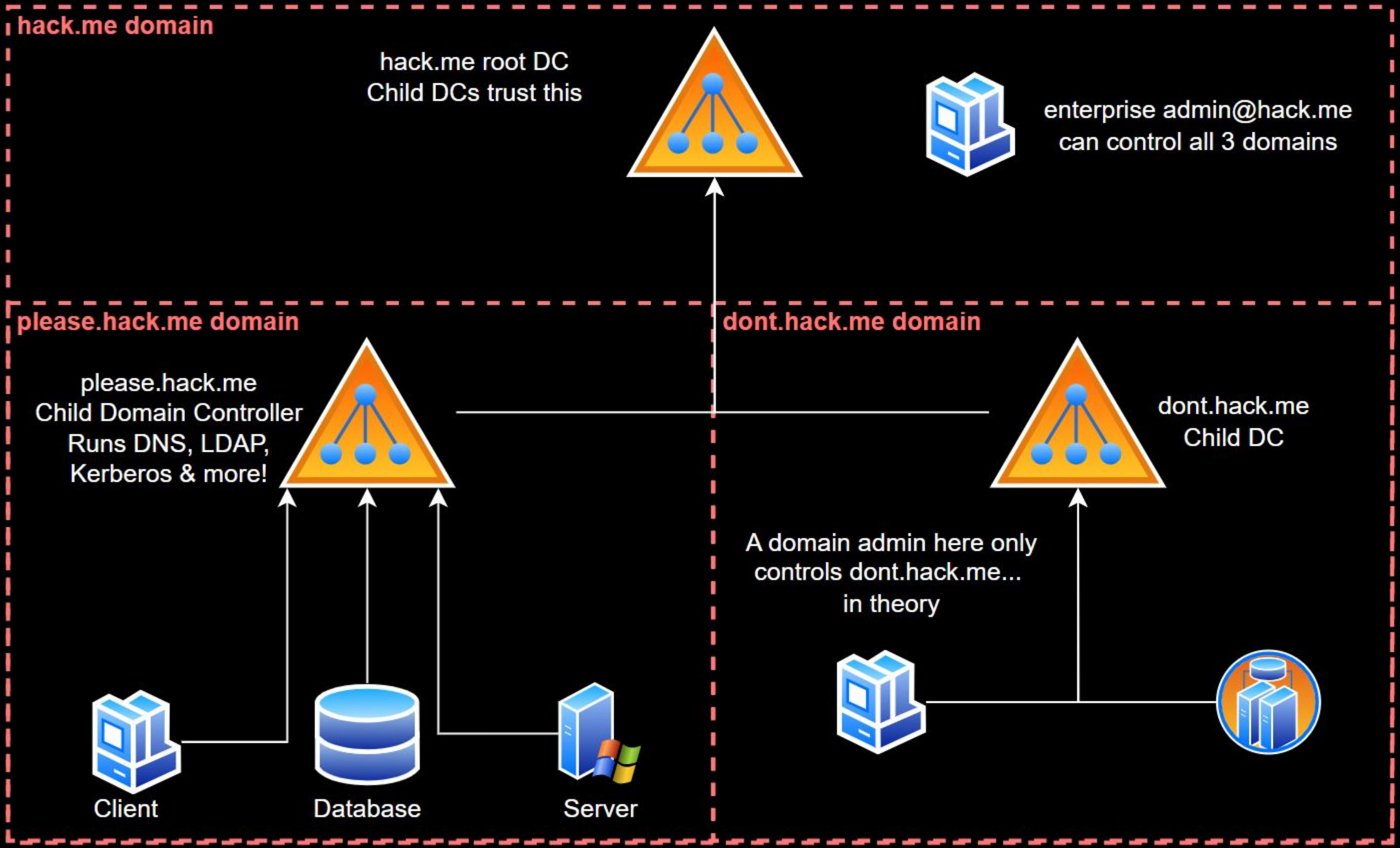
What is Active Directory?

- Basically a system for a bunch of computers to interact with one another in a work setting with configurable privileges and remote access
- Machines will be joined to an Active Directory Domain (illinois.edu)
- Each domain will *at least* have a Domain Controller
  - This is your centralized source of truth for every record in LDAP
  - When you request something over LDAP / DNS / Kerberos, it goes here
- It's also possible to have parent/child domains and other Forests
  - LDAP is set up like a tree data structure - each child domain requires a child domain controller
  - Crossing domains is not a security boundary but crossing Forests is

What is Active Directory?

- Active Directory is very permissive by default and changes some default settings to allow additional remote access
- This is by design. We can focus on targeting features of Active Directory and attacking misconfigurations to abuse trust relationships rather than traditional vulnerabilities
- You can log in to other domain-joined computers using NTLM (remember Pass-the-Hash from last time!) as well as Kerberos

Toy Domain Example



How does AD work?

- The key service that runs AD is "Active Directory Domain Services"
  - The domain controller is the server for this
  - This stores information about domain users and computers, as well as their access rights
  - This is contacted whenever a user logs into a machine
  - All of the user credentials are usually stored on the Domain Controller in a database called **NTDS.dit**
- AD manages your ability to do things to certain objects, e.g. create a user, remote log in to a computer, manage a service
  - The security check for this is SDDLs, everyone's favorite!
  - Each user and group will have a SID that the domain recognizes
- **Owning the domain controller means owning the target domain, effectively forever**

Why target Active Directory?

- Something like >95% of the Fortune 500 use Active Directory
- Permissive by default and extremely difficult to configure securely
- Tons of niche or poorly documented features that are often misconfigured (ADCS, SCCM)
- You likely don't need ANY vulnerabilities to get domain admin, meaning you can chain a phish or data breach with features to achieve complete compromise
- A domain compromise is game over for defenders and will almost guarantee you access to your objective
- **If you can hack this one piece of software, you can hack 95% of companies & organizations**

Kerberos

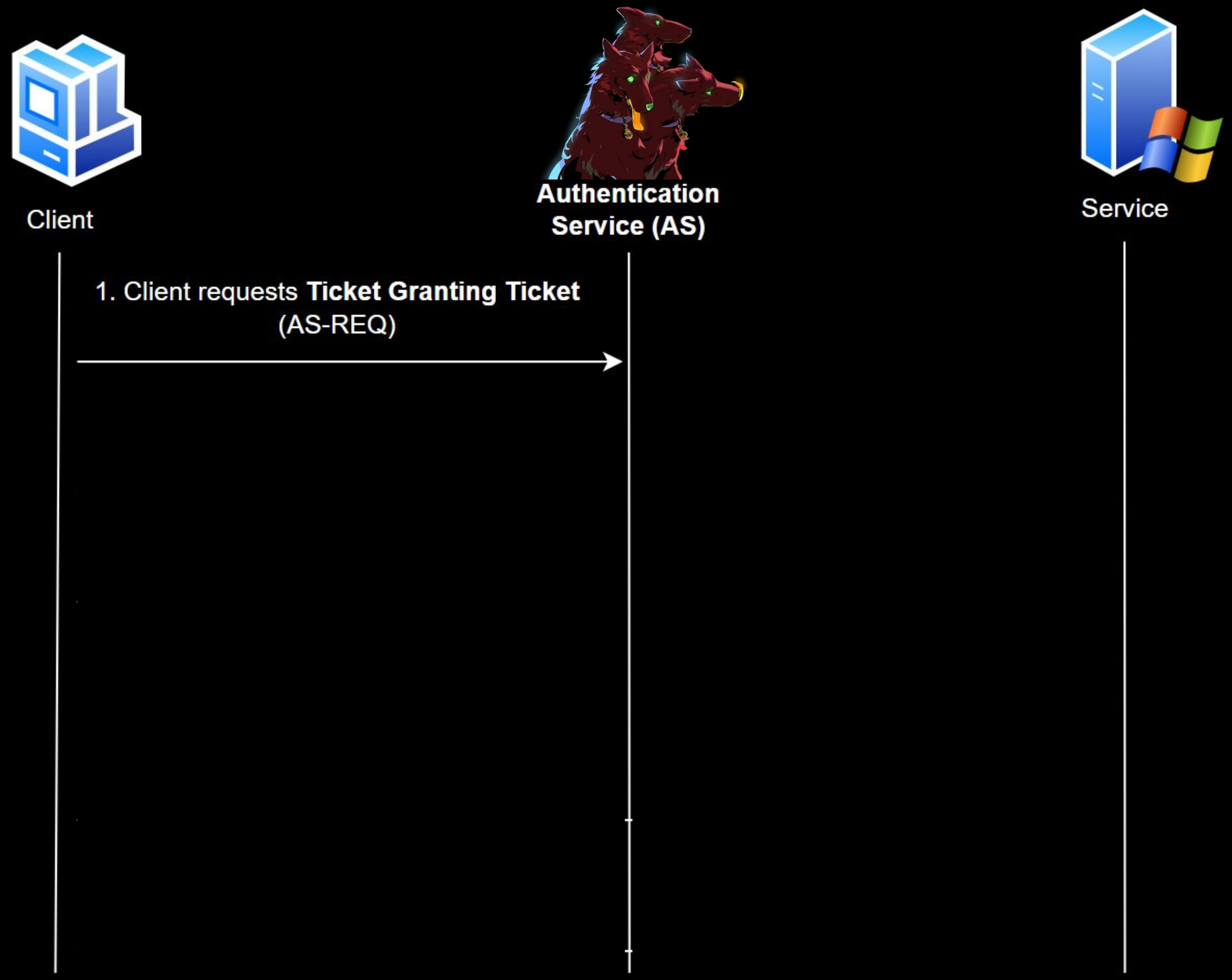
Guardian of the Domain

Kerberos

- Kerberos does authentication differently than NTLM
  - Goal was to provide mutual authentication between clients and services without transmitting passwords over the network
  - Proves identity via cryptographic keys derived from passwords instead
  - Functions similarly to a zero-knowledge proof where the client proves knowledge of a secret (their password) without revealing the secret to the verifier
- You will have **tickets** that let you do **stuff** to **things**
  - Surprisingly good mnemonic to remember Kerberos versus NTLM

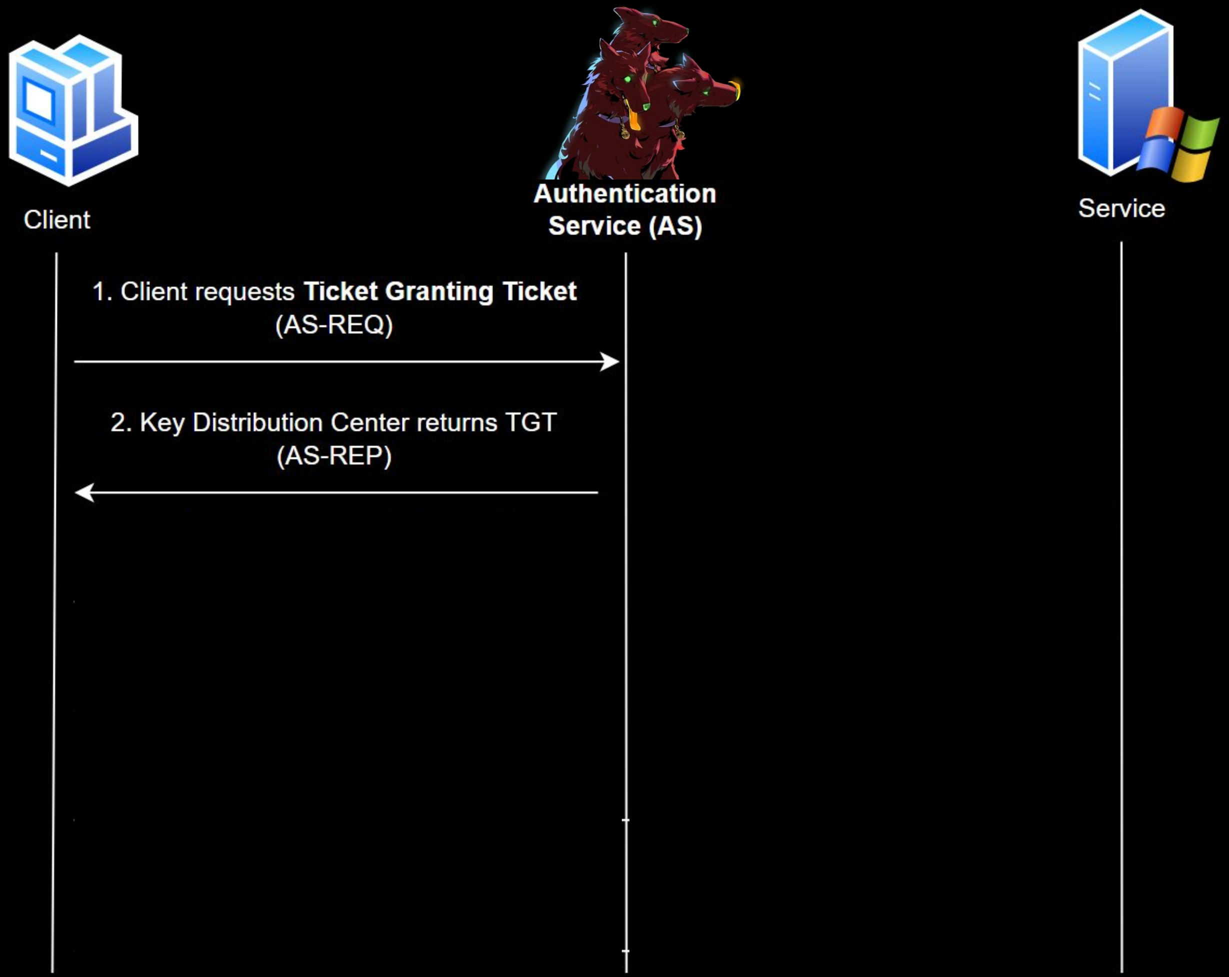
Kerberos

- The KDC knows all passwords
- AS-REQ: Client sends username and realm to the server, and optionally a pre-auth to proof their identity


Client
Authentication Service (AS)
Service
1. Client requests Ticket Granting Ticket (AS-REQ)

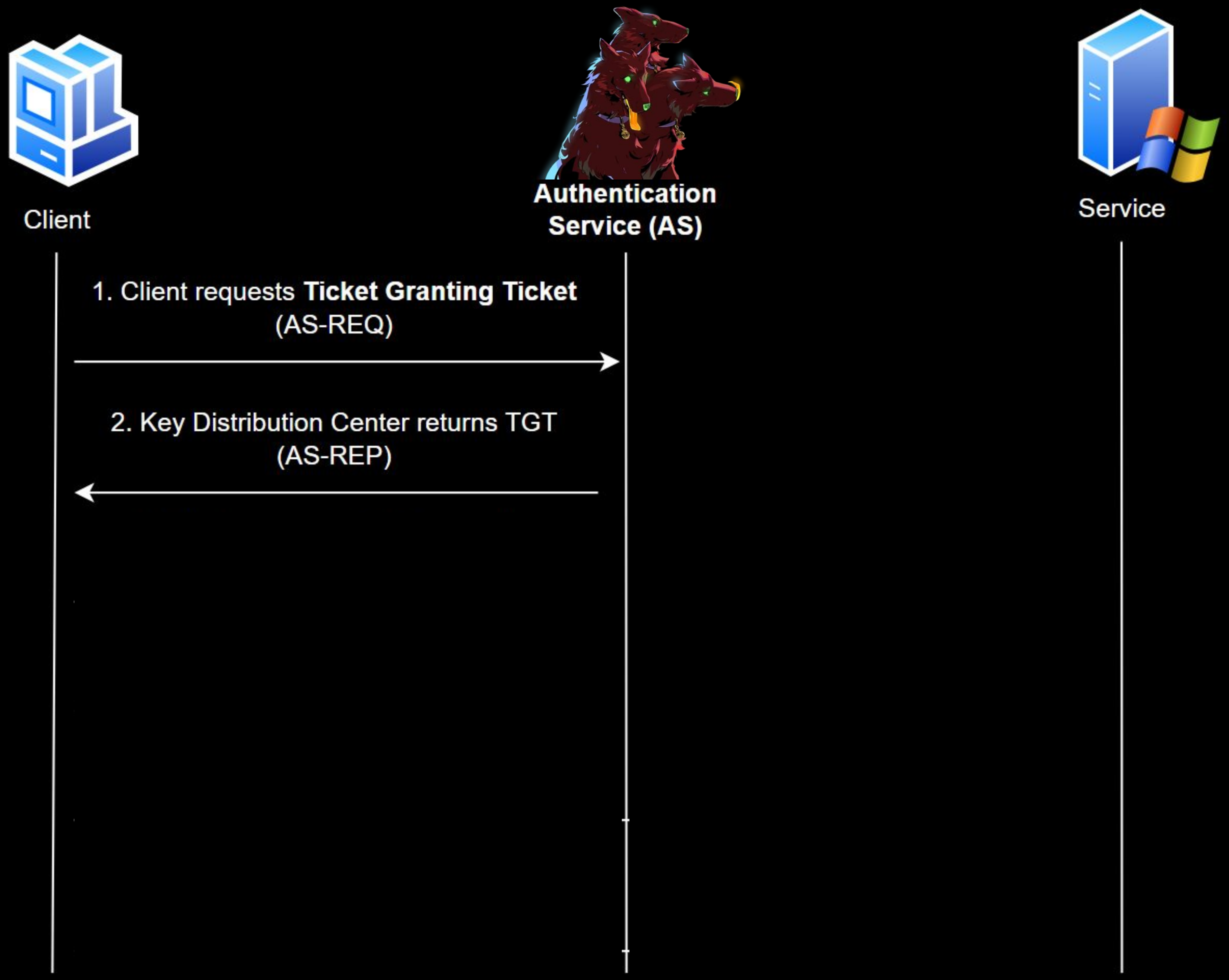
Kerberos

- The KDC knows all passwords
- AS-REQ: Client sends username and realm to the server, and optionally a pre-auth to proof their identity
- AS-REP: Server then sends a ticket (TGT) that is encrypted with server's hash, and a session key that the client can decrypt


Client
Authentication Service (AS)
Service
1. Client requests Ticket Granting Ticket (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)

Kerberos

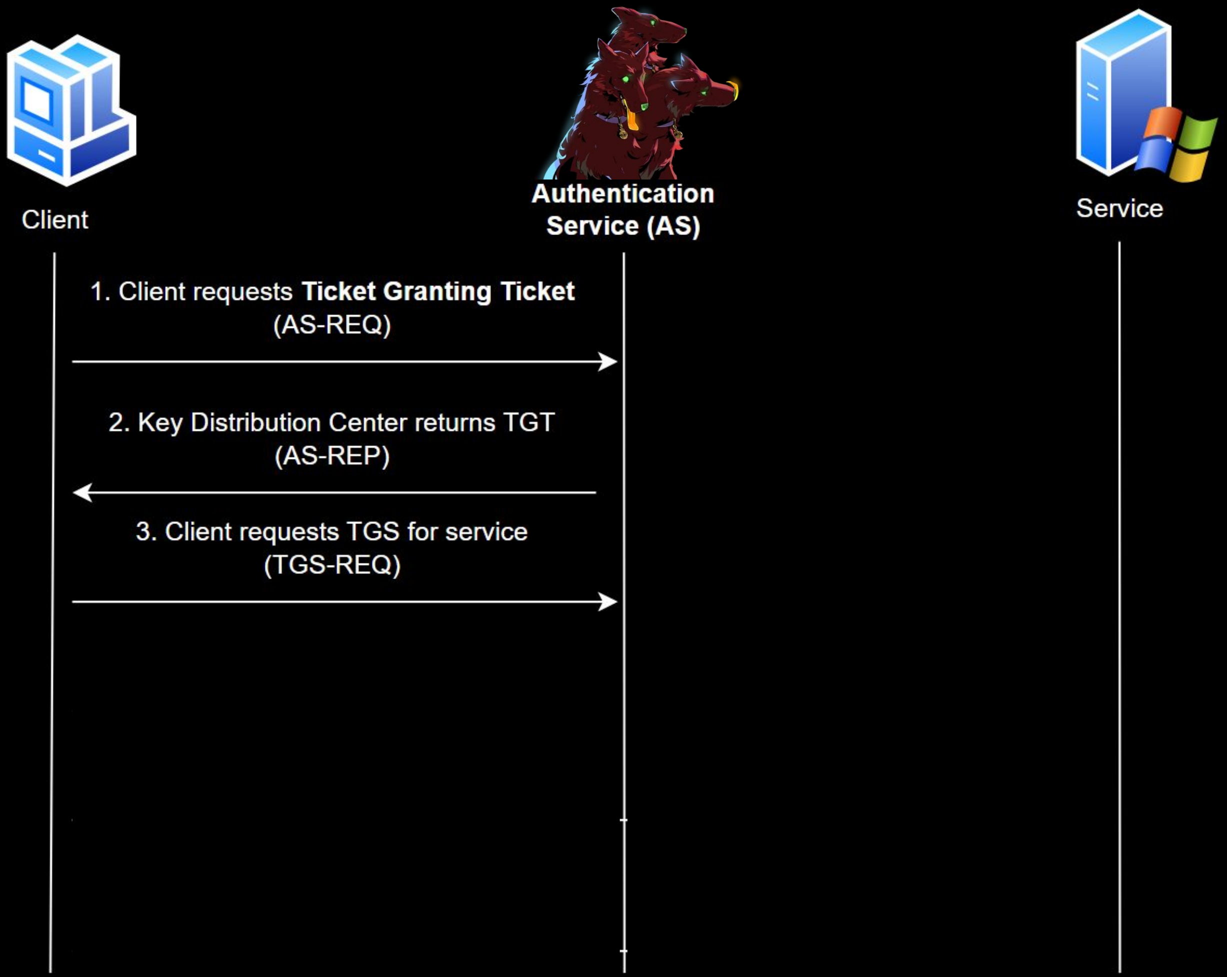
- This provides mutual(?) authentication between the client and server
- The client now has a ticket (TGT), that can be used to request service access (TGS)
- Since TGT is service-independence, TGT = NTLM hash in terms of power


Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)

Kerberos

- With TGT, the client now asks the server for ~~a candy because it's halloween~~ a *specific* ticket to access the service
- TGS-REQ: the client sends the previous TGT and pre-authentication data to prove its identity

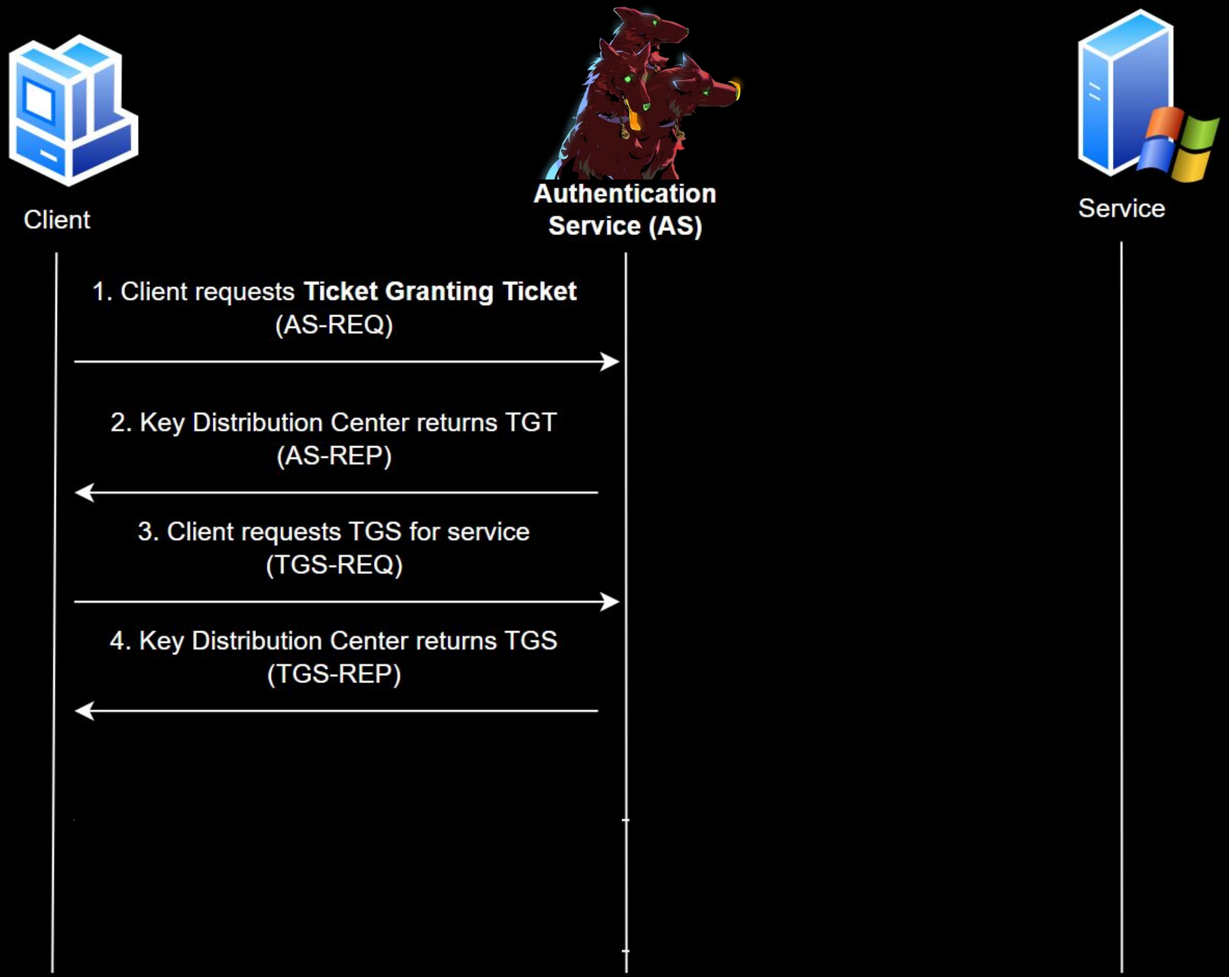
Side note: TGT is like a TGS to kerberos itself - thus it is encrypted **with the kerberos service hash**



Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)

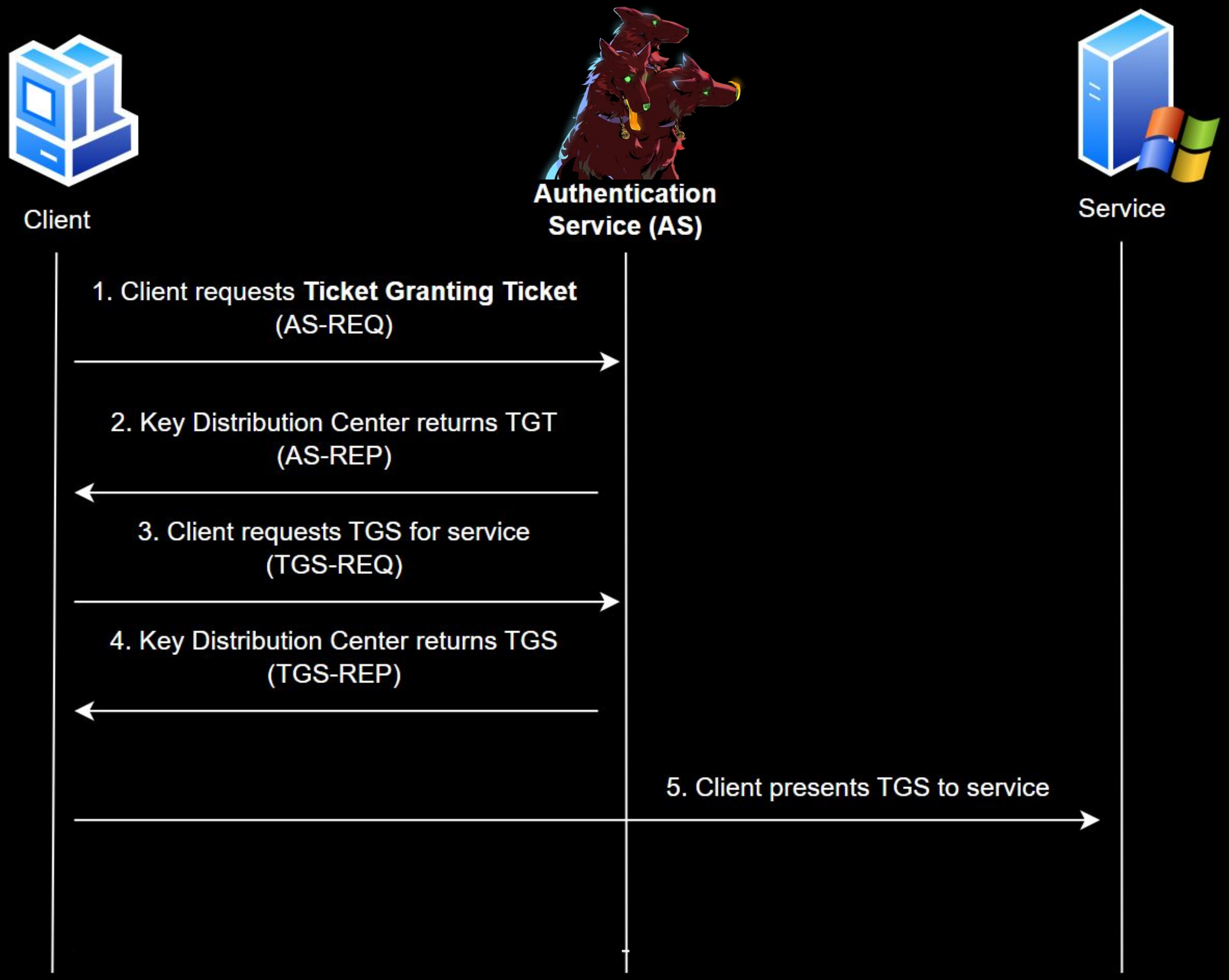
Kerberos

- With TGT, the client now asks the server for ~~a candy because it's halloween~~ a *specific* ticket to access the service
- TGS-REQ: the client sends the previous TGT and pre-authentication data to prove its identity
- TGS-REP: the server sends a TGS that is encrypted with the service's hash, as well as a session key that the client can decrypt


Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)

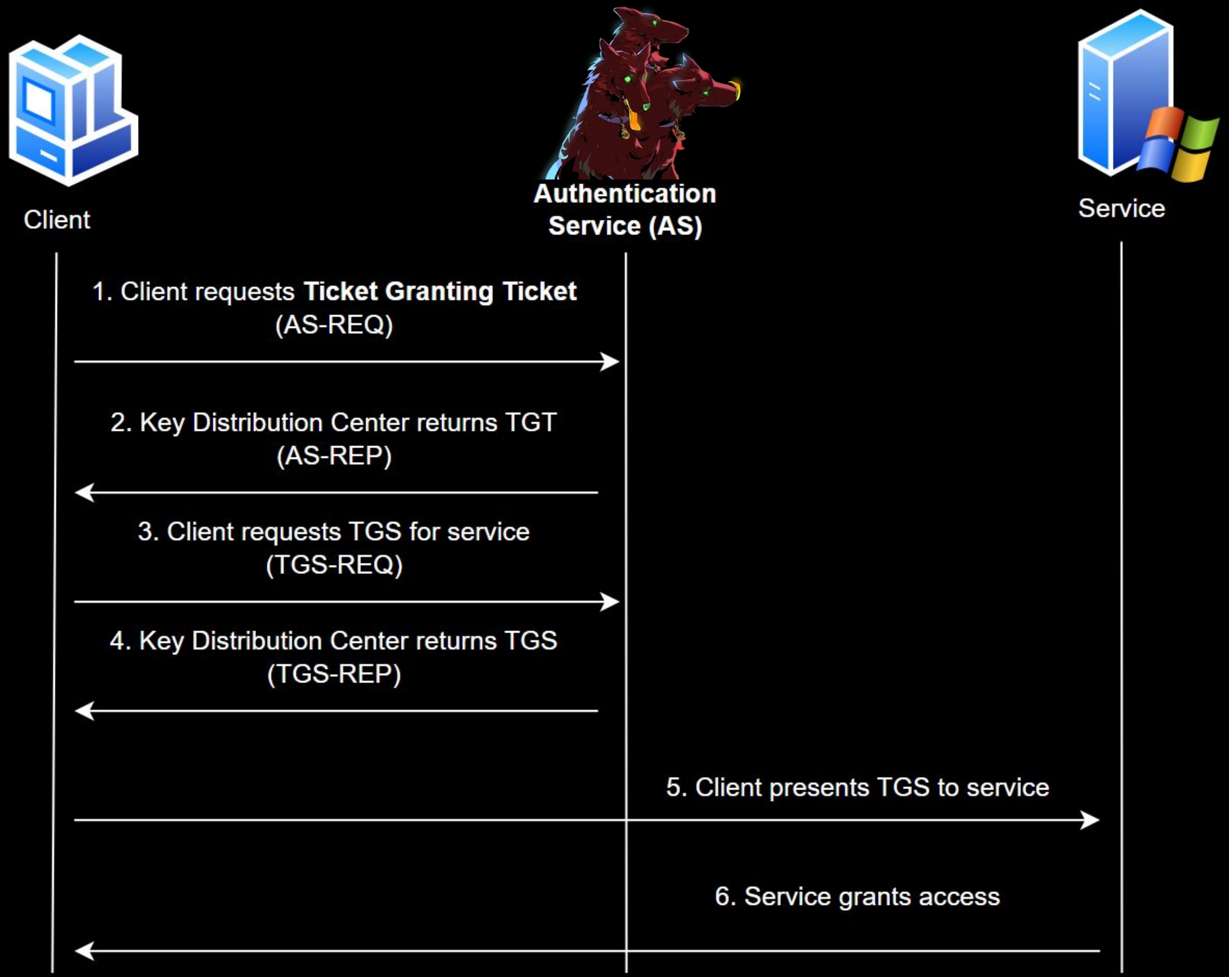
Kerberos

- Client now presents TGS and its own session key to service. TGS contains many client information and the session key.


Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service

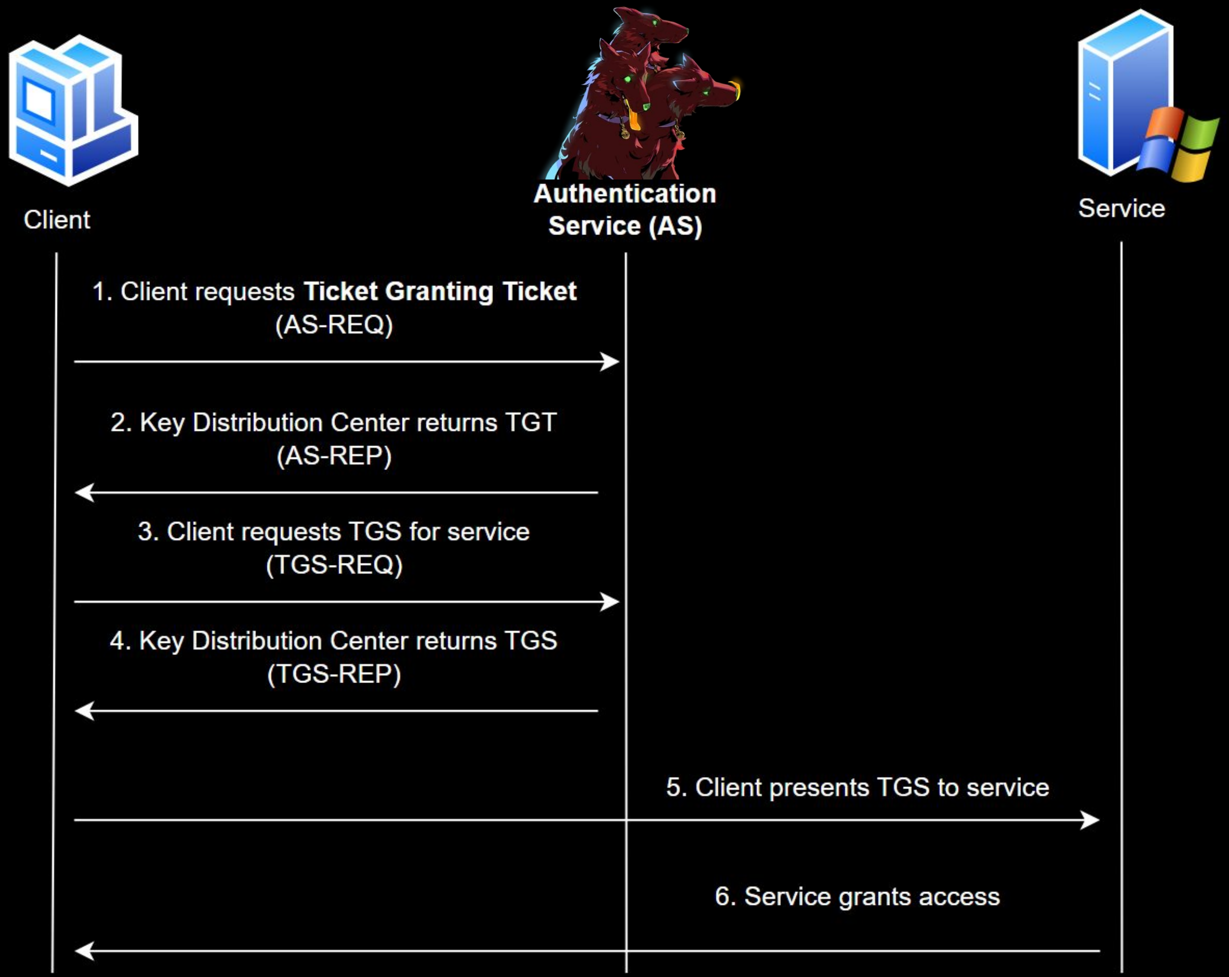
Kerberos

- Client now presents TGS and its own session key to service. TGS contains many client information and the session key.
- Service decrypts the TGS, reads the client name and privileges, and then grants or denies access.


Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service
6. Service grants access

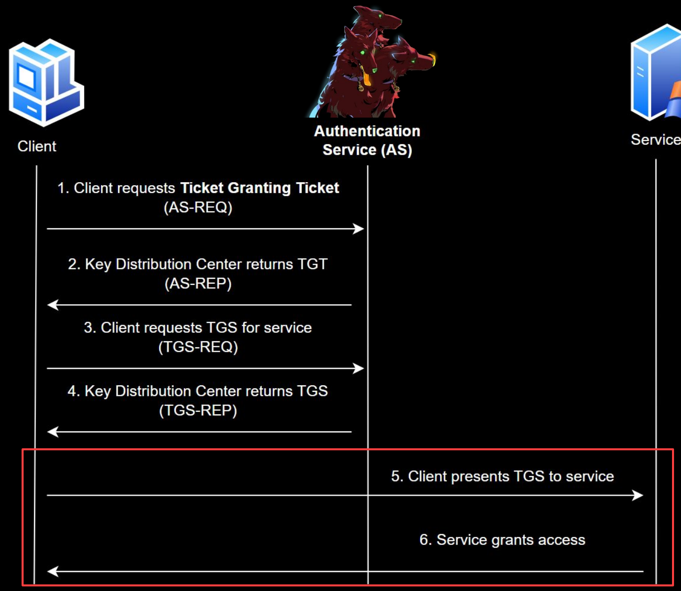
Kerberos

- Client now presents TGS and its own session key to service. TGS contains many client information and the session key.
- Service decrypts the TGS, reads the client name and privileges, and then grants or denies access.
- **Can you find any problems with this flow?**


Client
Authentication Service (AS)
Service
1. Client requests **Ticket Granting Ticket** (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service
6. Service grants access

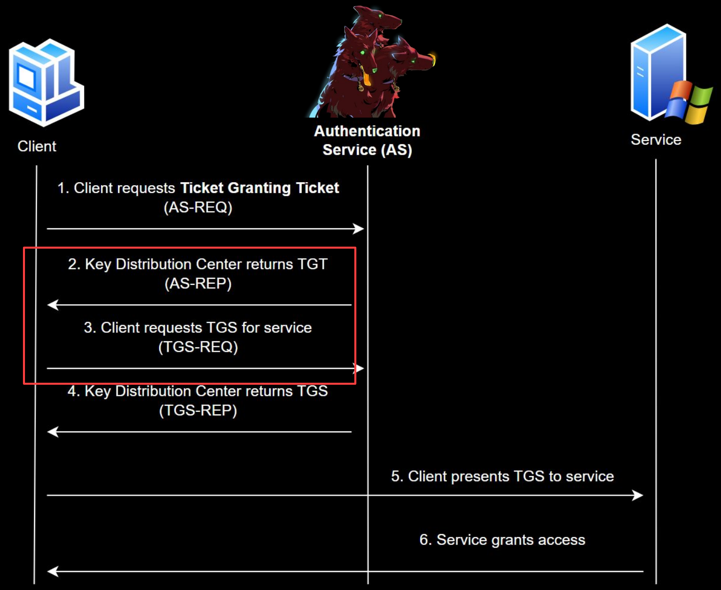
Kerberos Points of Failure: TGS

- Working as normal, we can abuse **where** secrets come from
  - Part of the TGS returned by the KDC is encrypted with a secret derived from the password of the user account running that service
  - We can request TGS's for services running under domain accounts and crack them offline to recover their password! (**Kerberoasting**)


Client
Authentication Service (AS)
Service
1. Client requests Ticket Granting Ticket (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service
6. Service grants access

Kerberos Points of Failure: TGT

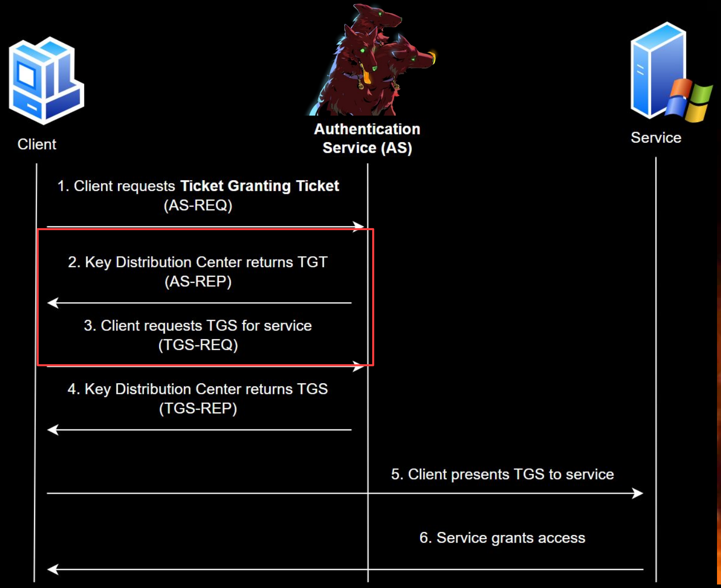
- For all things that require a TGT (Kerberos pre-auth), they all rely on **password** of a special account (**krbtgt**)!
  - If I were able to obtain the Kerberos account hash, then I could forge TGTs and therefore TGSs that give me access to anything
  - This requires a domain controller compromise, but if you can do it, you now own everything in the domain (called a **Golden Ticket**)


Client
Authentication Service (AS)
Service
1. Client requests Ticket Granting Ticket (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service
6. Service grants access

GOLDEN TICKET
TO AN ACTIVE DIRECTORY DOMAIN
A GOLDEN TICKET TO TOTALLY COMPROMISING THE CONFIDENTIALITY, INTEGRITY, AND AVAILABILITY OF YOUR DOMAIN

Kerberos Points of Failure: Pre-Auth

- Remember when I said "optionally a pre-auth to proof their identity"?
- What if we don't require the TGT?
  - If a user or service does NOT require the TGT, then anyone can request an AS-REP for that user, which uses their password to encrypt some secret data, which again we can crack offline
  - This technique is called **AS-REP roasting**

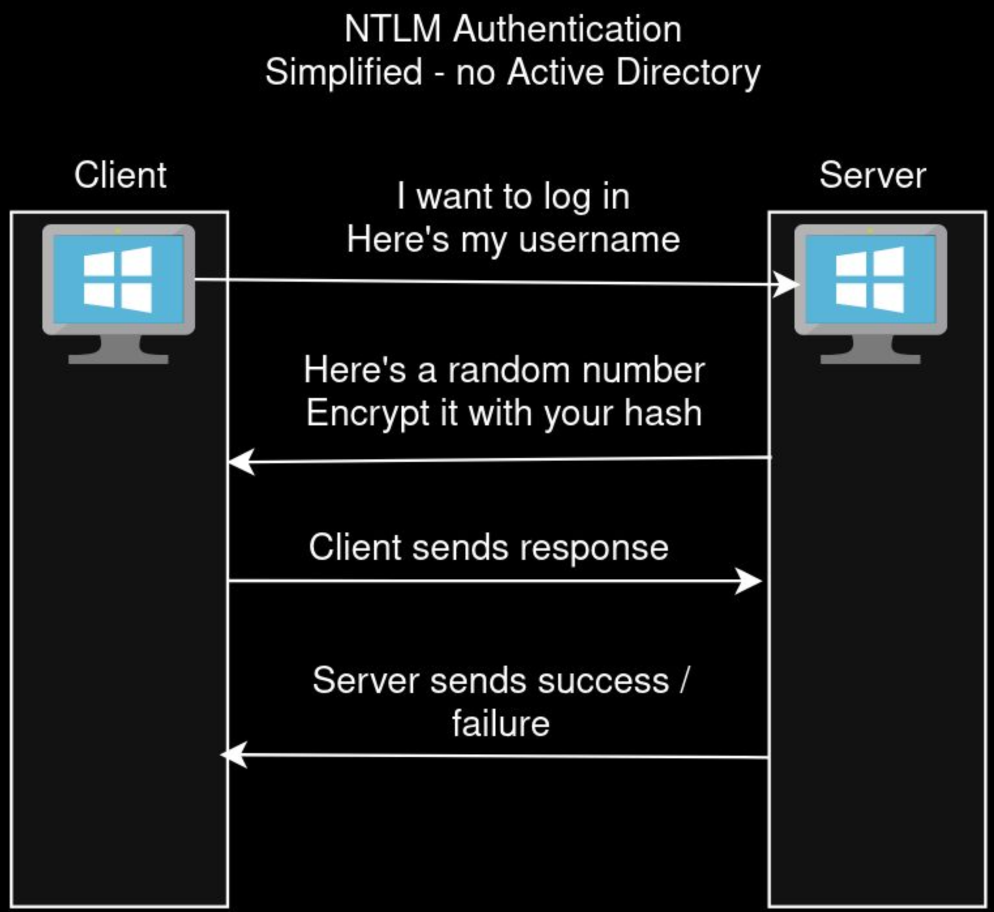

Client
Authentication Service (AS)
Service
1. Client requests Ticket Granting Ticket (AS-REQ)
2. Key Distribution Center returns TGT (AS-REP)
3. Client requests TGS for service (TGS-REQ)
4. Key Distribution Center returns TGS (TGS-REP)
5. Client presents TGS to service
6. Service grants access

NTLM

"*I roll all my own crypto*" - Bill Gates, probably

NTLM Authentication

- NTLM authentication functions as a zero knowledge proof where the secret is the password hash
- The auth mechanism is challenge / response
- Key point is that **the hash is the authentication material, not the password**
- Stealing the hash **allows authentication**



Hash Theft

- AD user hashes are in LSASS process memory
  - This is generally not possible to access if Credential Guard is enabled
  - Different threat model - an AD user could have access to other boxes!
- Before Windows 11, all you needed to do was get SYSTEM, dump LSASS process memory, and get hashes

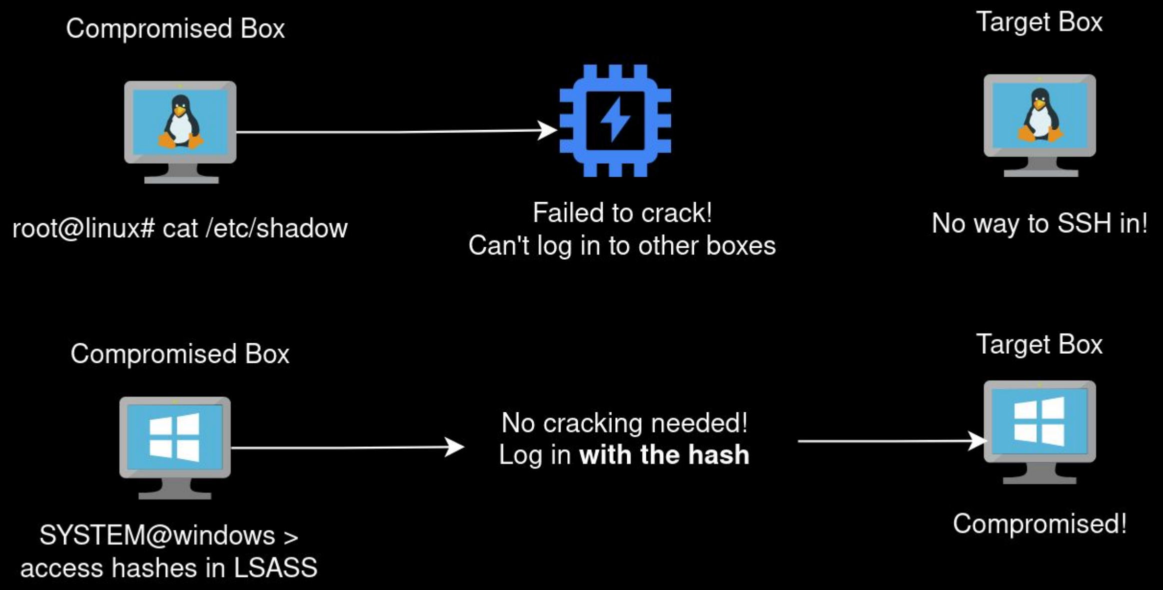
```
beacon> mimikatz !sekurlsa::logonpasswords

Authentication Id : 0 ; 579458 (00000000:0008d782)
Session           : Batch from 0
User Name         : jking
Domain            : DEV
Logon Server      : DC-2
Logon Time        : 8/31/2022 11:49:48 AM
SID               : S-1-5-21-569305411-121244042-2357301523-1105
        msv :
         [00000003] Primary
         * Username: jking
         * Domain   : DEV
         * NTLM     : 59fc0f884922b4ce376051134c71e22c
         * SHA1     : 74fa9854d529092b92e0d9ebef7ce3d065027f45
```

The Pass-the-Hash attack

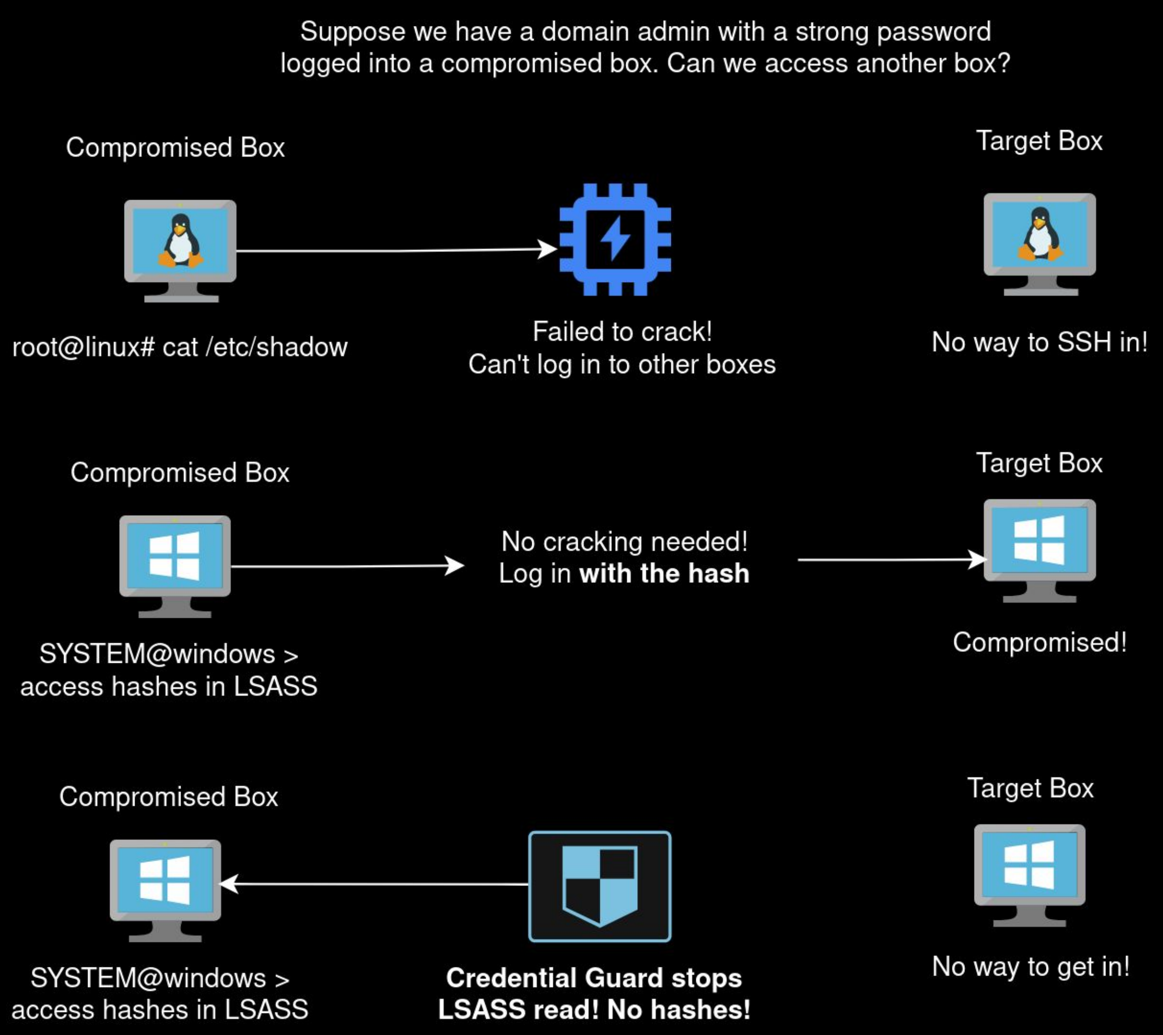
- Suppose we have compromised a user's NTLM hash, but not their password
- Since NTLM is like a proof, we can prove to anyone else that we are the user in question by presenting a permutation of their hash
- So, we can obtain persistent access as any user whose hash we have obtained
- The usual flow for this would be to compromise a box, dump hashes, and log into everything we can with the new credentials
  - You could rinse-repeat this however many degrees of separation it would take to become Domain Admin
- Hash theft on Windows 11 requires a kernel and hypervisor exploit, or to just lie in wait by backdooring LSASS

Pass-the-Hash Example Scenario

Suppose we have a domain admin with a strong password logged into a compromised box. Can we access another box?

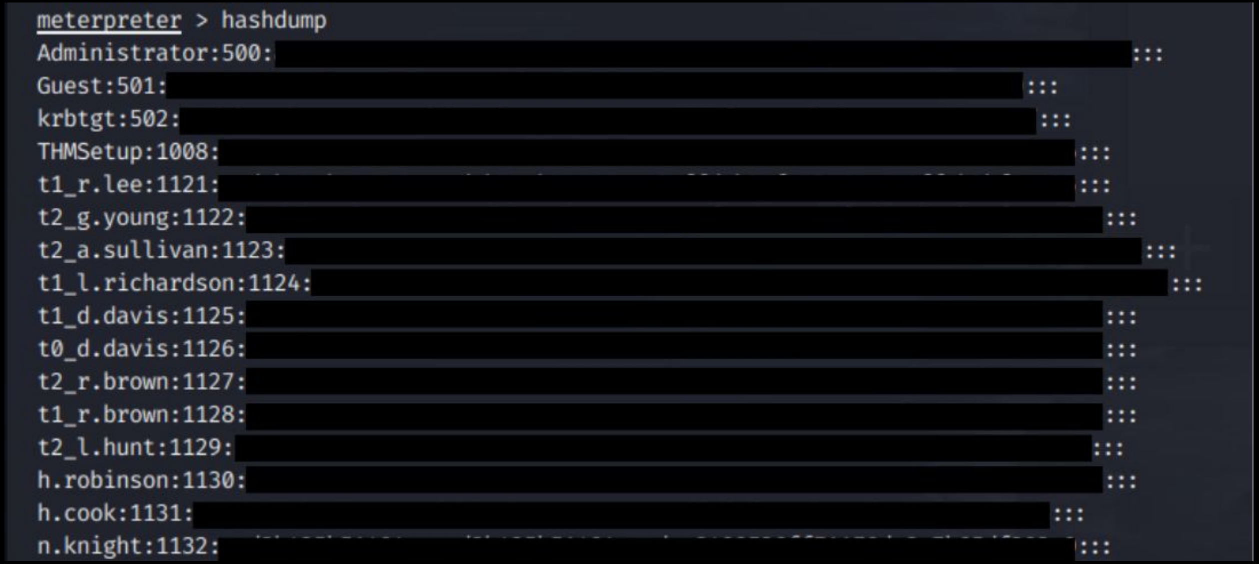


Credential Guard


Suppose we have a domain admin with a strong password logged into a compromised box. Can we access another box?
Compromised Box
root@linux# cat /etc/shadow
Failed to crack!
Can't log in to other boxes
Target Box
No way to SSH in!
Compromised Box
SYSTEM@windows > access hashes in LSASS
No cracking needed!
Log in **with the hash**
Target Box
Compromised!
Compromised Box
SYSTEM@windows > access hashes in LSASS
Credential Guard stops LSASS read! No hashes!
Target Box
No way to get in!

Practical Uses

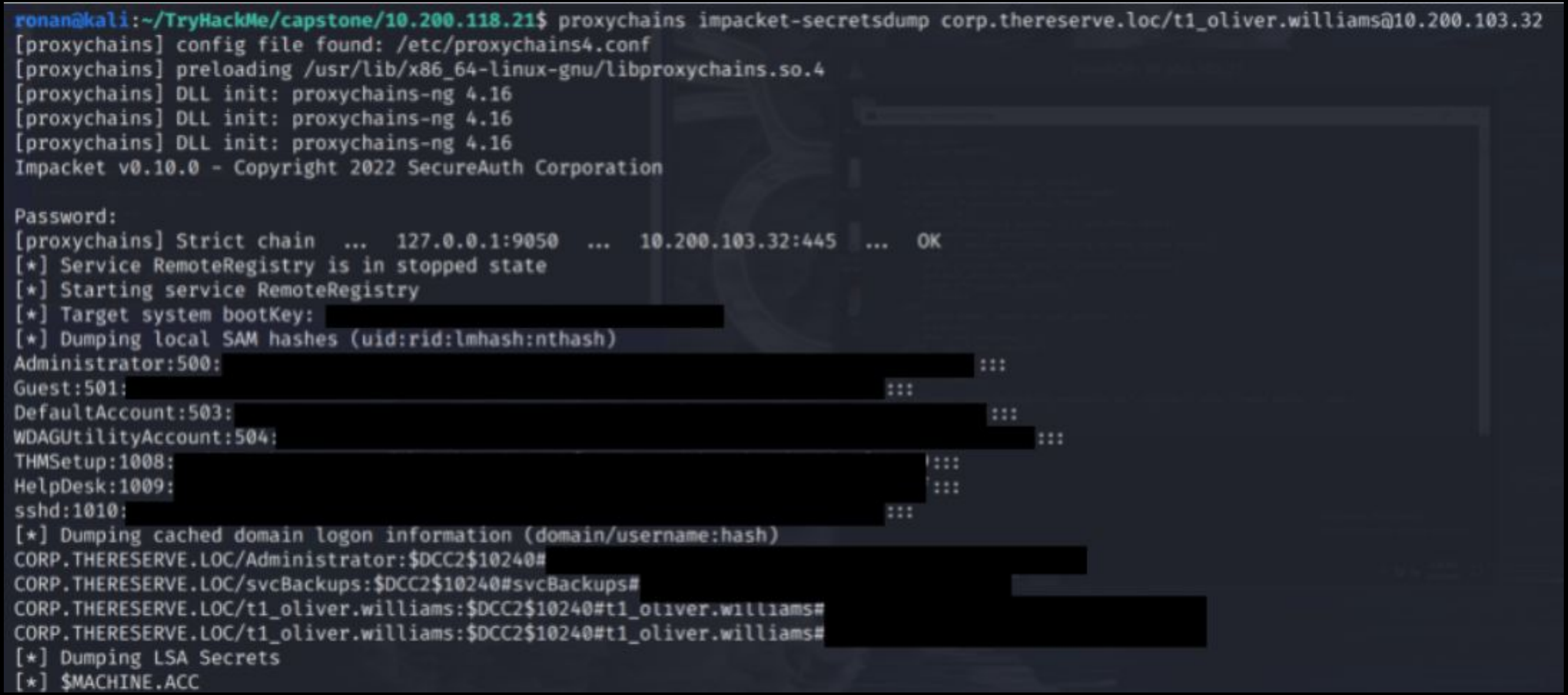
- **Mimikatz**
  - Does a variety of things to access confidential information
  - The most signatured piece of malware in existence
  - Can steal everything stored in LSASS & registry
  - Actual EXE dropped on-target
  - Built in to meterpreter as an extension (kiwi)



```
meterpreter > hashdump
Administrator:500:[redacted]:::
Guest:501:[redacted]:::
krbtgt:502:[redacted]:::
THMSetup:1008:[redacted]:::
t1_r.lee:1121:[redacted]:::
t2_g.young:1122:[redacted]:::
t2_a.sullivan:1123:[redacted]:::
t1_l.richardson:1124:[redacted]:::
t1_d.davis:1125:[redacted]:::
t0_d.davis:1126:[redacted]:::
t2_r.brown:1127:[redacted]:::
t1_r.brown:1128:[redacted]:::
t2_l.hunt:1129:[redacted]:::
h.robinson:1130:[redacted]:::
h.cook:1131:[redacted]:::
n.knight:1132:[redacted]:::
```

Practical Uses

- **Impacket-Secretsdump**
  - Steals as much as possible while executing no agent (network only)
  - Does not access LSASS but accesses everything in registry



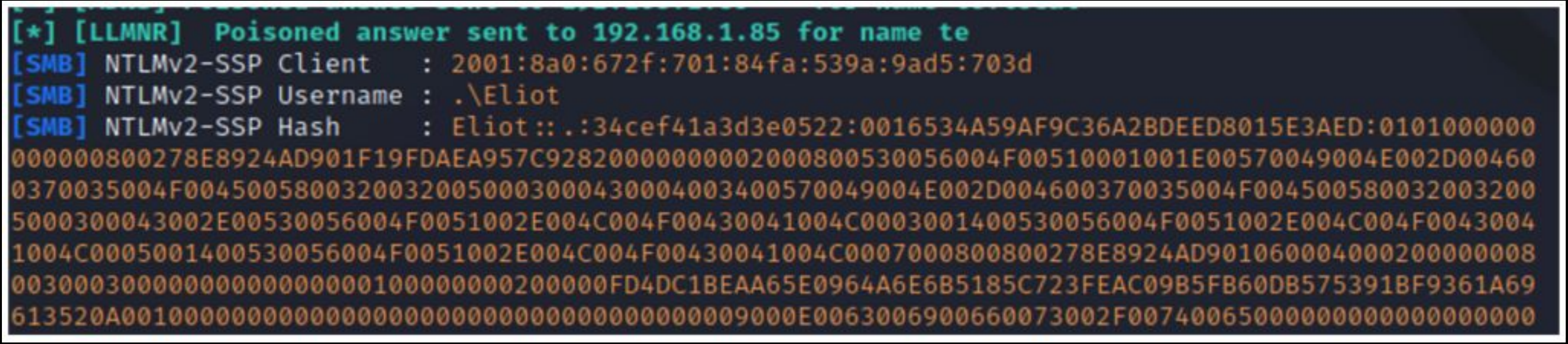
```
ronan@kali:~/TryHackMe/capstone/10.200.118.21$ proxychains impacket-secretsdump corp.thereserve.loc/t1_oliver.williams@10.200.103.32
[proxychains] config file found: /etc/proxychains4.conf
[proxychains] preloading /usr/lib/x86_64-linux-gnu/libproxychains.so.4
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
[proxychains] DLL init: proxychains-ng 4.16
Impacket v0.10.0 - Copyright 2022 SecureAuth Corporation

Password:
[proxychains] Strict chain  ...  127.0.0.1:9050  ...  10.200.103.32:445  ...  OK
[*] Service RemoteRegistry is in stopped state
[*] Starting service RemoteRegistry
[*] Target system bootKey: [redacted]
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:[redacted]:::
Guest:501:[redacted]:::
DefaultAccount:503:[redacted]:::
WDAGUtilityAccount:504:[redacted]:::
THMSetup:1008:[redacted]:::
HelpDesk:1009:[redacted]:::
sshd:1010:[redacted]:::
[*] Dumping cached domain logon information (domain/username:hash)
CORP.THERESERVE.LOC/Administrator:$DCC2$10240#[redacted]
CORP.THERESERVE.LOC/svcBackups:$DCC2$10240#svcBackups#[redacted]
CORP.THERESERVE.LOC/t1_oliver.williams:$DCC2$10240#t1_oliver.williams#[redacted]
CORP.THERESERVE.LOC/t1_oliver.williams:$DCC2$10240#t1_oliver.williams#[redacted]
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
```

Net-NTLMv2

- For designated remote logins, there's Net-NTLMv2
- Net-NTLMv2 is NOT NTLM!
- **Net-NTLMv2 cannot log in to the computer that sent the hash**, but it can log in to anywhere else
- We can attempt to crack this to recover the plaintext password
- There are some things we can do to get Net-NTLMv2 hashes over a network
  - If we can trick someone into click on a .lnk or similar, that can log them into our SMB server
  - When you access an SMB server in windows, you will **automatically** log in via Net-NTLMv2
  - So, a common phish would be to host shortcut files that point to a legitimate file hosting on an attacker's SMB server

Net-NTLMv2

- We can also try to MITM Net-NTLMv2 instead of phishing
  - You can use a tool called Responder, which will leverage (among many other techniques) Link Local Multicast Name Resolution to say that your attacker share corresponds to certain hostnames
  - They then visit it and you get their Net-NTLMv2 hash
- **Using responder in poisoning mode on a public network is super illegal (and very effective 😉)**



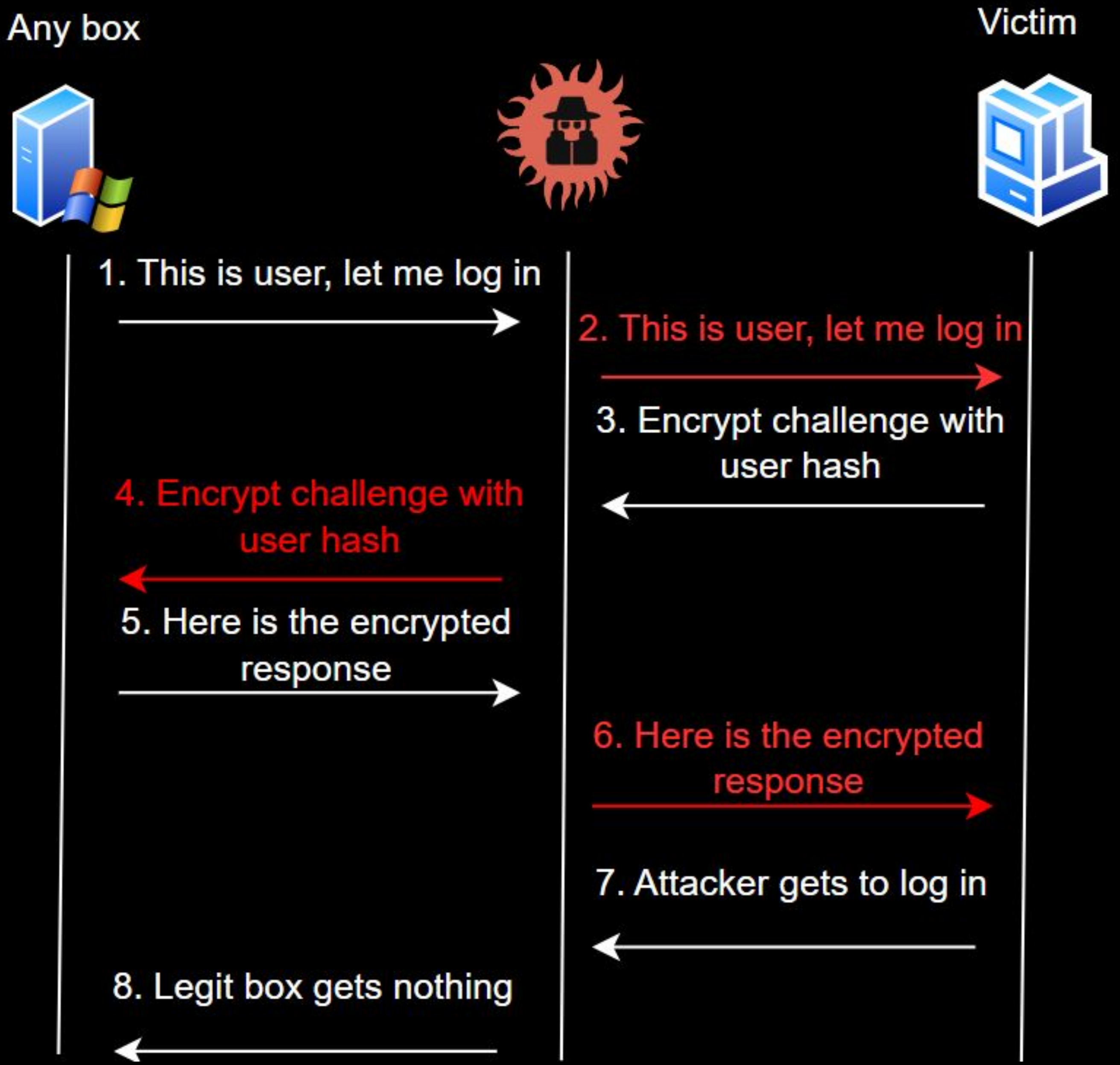
```
[*] [LLMNR]  Poisoned answer sent to 192.168.1.85 for name te
[SMB] NTLMv2-SSP Client   : 2001:8a0:672f:701:84fa:539a:9ad5:703d
[SMB] NTLMv2-SSP Username : .\Eliot
[SMB] NTLMv2-SSP Hash     : Eliot::.:34cef41a3d3e0522:0016534A59AF9C36A2BDEED8015E3AED:0101000000
000000800278E8924AD901F19FDAEA957C928200000000002000800530056004F00510001001E00570049004E002D00460
0370035004F0045005800320032005000300043000400340057004900 4E002D004600370035004F0045005800320032 00
5000300043002E00530056004F0051002E004C004F00430041004C000300140053005600 4F0051002E004C004F0043004
1004C0005001400530056004F0051002E004C004F00430041004C00070008000800278E8924AD901060004000200000008
0030003000000000000000010000000020000 0FD4DC1BEAA65E0964A6E6B5185C723FEAC09B5FB60DB575391BF9361A69
613520A001000000000000000000000000000000000009000E00630069006600730 02F0074006500000000000000000000
```

Authentication Coercion

- We do not need to MITM or phish if the target is vulnerable to authentication coercion
- Many of these bugs still exist
- There are a number of authentication coercion "**features**" like the infamous Printer Bug, which, under certain circumstances, **will force the target machine to authenticate to an attacker-controlled host**
  - For the Printer Bug, the Print Spooler must be running on the target
- So, there are some circumstances where we can disclose a Net-NTLMv2 hash **at will** (google PetitPotam, Printer Bug)
- Can we use any of these bugs (or combine them) to compromise the sender machine?

Hash Relay Attack

- Recall that *everyone else* respects the Net-NTLM hash, so we can just be a man in the middle and bounce the hash back and forth
- Any time we can force a Net-NTLMv2 login in an AD environment (auth coercion, .lnk file, xp_dirtree, responder), we can pull this trick


Any box
Victim
1. This is user, let me log in
2. This is user, let me log in
3. Encrypt challenge with user hash
4. Encrypt challenge with user hash
5. Here is the encrypted response
6. Here is the encrypted response
7. Attacker gets to log in
8. Legit box gets nothing

Attacking Active Directory

Domain Recon

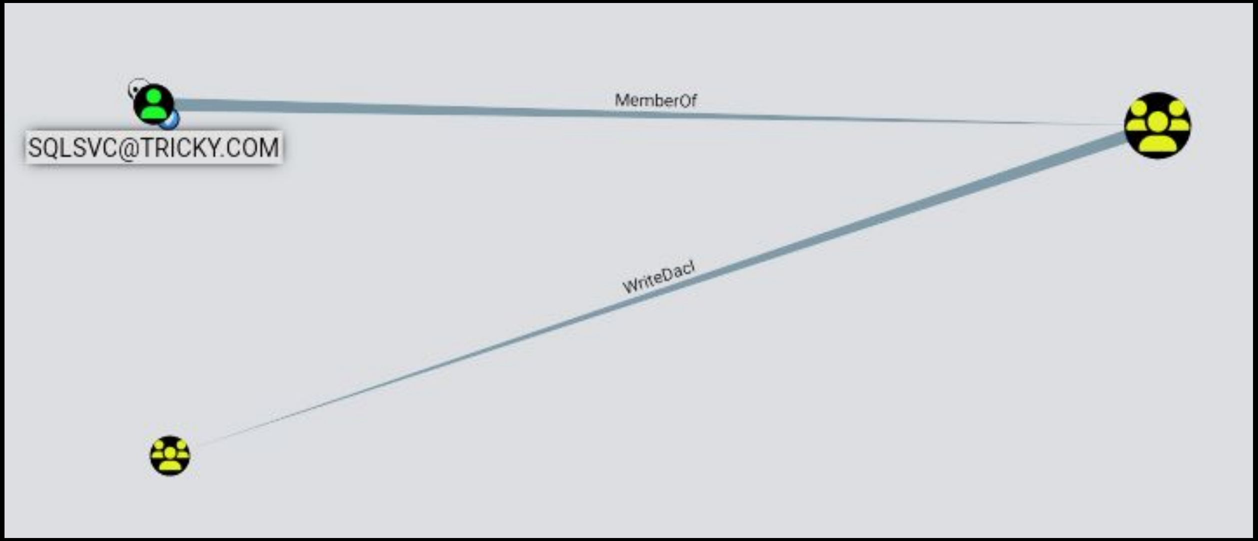
- Like NetSec and WebSec, we need to recon our target before we do anything
- For example, it is **trivial** to detect Kerberoasting via Honeypots
  - We want to do manual recon to figure out the lay of the land and avoid active defense and deception measures
  - Use this information to do selective, targeted attacks
- Like PrivEsc, we can do either manual or fully automated enumeration - full auto has poor stealth but some amazing tooling
- We want to look up users, computers, groups, and privileged relationships
- A common tool for this would be **SharpView**

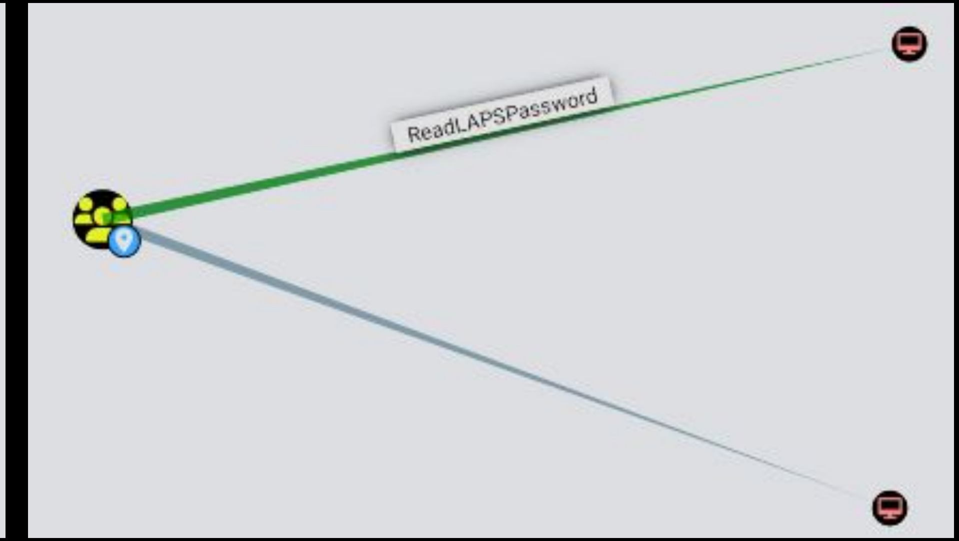
SharpView Cheat Sheet

Command	Description
Get-Domain	Returns information about the current domain or the domain specified with -Domain
Get-DomainController	Returns information about the domain controller for the current or specified domain
Get-ForestDomain	Returns all domains for the current or specified forest
Get-DomainPolicyData	Returns the default domain policy, which can reveal things like the password policy
Get-DomainUser	Returns all users in the domain
Get-DomainComputer	Returns all computers in the domain
Get-DomainOU	Search for all Organizational Units or specific ones
Get-DomainGroup	Returns all groups on the domain
Get-DomainGroupMember	Returns all members of a given group on the domain
Get-DomainGPO	Returns all GPO objects on the domain
Get-DomainGPOLocalGroup	Returns all GPOs that modify local group membership through restricted groups or group policy preferences.
Get-DomainGPOUserLocalGroupMapping	This enumerates the machines where a specific domain user / group is a member of a specific local group. This can be used to cross reference to find administrative privileges.
Get-DomainTrust	Returns all domain trusts for the current or specified domain

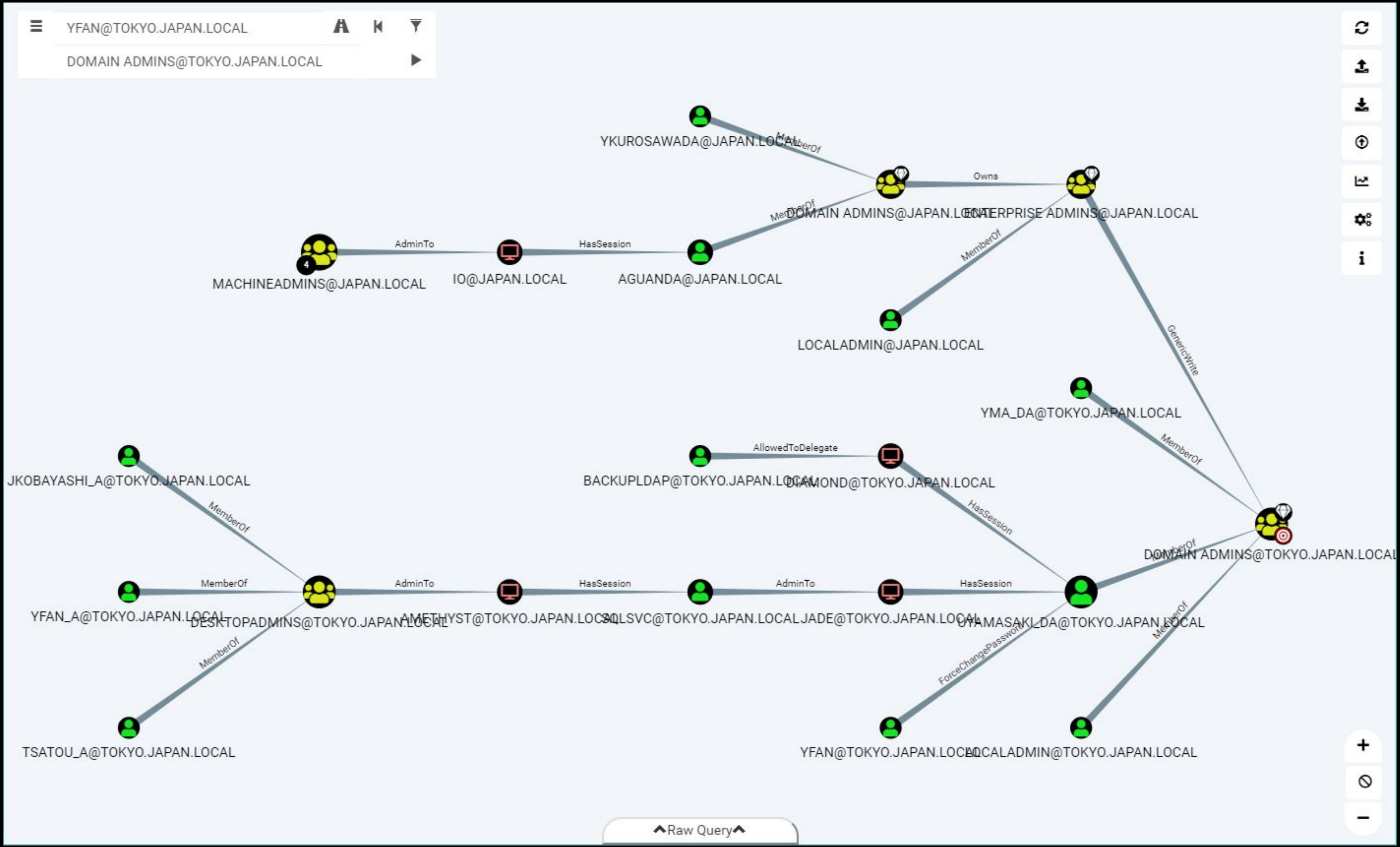
BloodHound

- Keeping users and relationships in your head is very difficult
- It is **substantially** more difficult when there are thousands of them
- What if we map out all people and computers and their relationships, then run Dijkstra's algorithm to speedrun Domain Admin?
    - Auto ingest data with SharpHound or Bloodhound-Python





BloodHound


YFAN@TOKYO.JAPAN.LOCAL
DOMAIN ADMINS@TOKYO.JAPAN.LOCAL
YKUROSAWADA@JAPAN.LOCAL
MemberOf
DOMAIN ADMINS@JAPAN.LOCAL
Owns
ENTERPRISE ADMINS@JAPAN.LOCAL
MACHINEADMINS@JAPAN.LOCAL
AdminTo
IO@JAPAN.LOCAL
HasSession
AGUANDA@JAPAN.LOCAL
MemberOf
LOCALADMIN@JAPAN.LOCAL
GenericWrite
YMA_DA@TOKYO.JAPAN.LOCAL
JKOBAYASHI_A@TOKYO.JAPAN.LOCAL
BACKUPLDAP@TOKYO.JAPAN.LOCAL
AllowedToDelegate
DIAMOND@TOKYO.JAPAN.LOCAL
HasSession
DOMAIN ADMINS@TOKYO.JAPAN.LOCAL
YFAN_A@TOKYO.JAPAN.LOCAL
DESKTOPADMINS@TOKYO.JAPAN.LOCAL
AMETHYST@TOKYO.JAPAN.LOCAL
SQLSVC@TOKYO.JAPAN.LOCAL
JADE@TOKYO.JAPAN.LOCAL
OYAMASAKI_DA@TOKYO.JAPAN.LOCAL
ForceChangePassword
TSATOU_A@TOKYO.JAPAN.LOCAL
YFAN@TOKYO.JAPAN.LOCAL
LOCALADMIN@TOKYO.JAPAN.LOCAL
Raw Query

Lateral Movement

- Let's say we have a credential onto some machine (TGT / NTLM)
- How do I turn that into Remote Code Execution?
- There are many techniques, since we can essentially arbitrarily read and write files via authenticated SMB
  - Write a file over SMB -> start a service over SMB (PSEXEC)
  - Use WMI's Win32_Process class to run commands (WMIEXEC)
    - A favorite of APT29!
  - Modify a service to point to cmd.exe and pipe stdout (SMBEXEC)
  - Utilize SCM manipulation for fileless lateral movement (SCSHELL)
    - A favorite of mine, defeats CrowdStrike Falcon & MDE with ease
- Check impacket / github to run these via command line

Practical Kerberos Attacks

- There are many great tools to do this, but the standard is **Rubeus**
- Kerberoast with `rubeus.exe kerberoast`
- Asreproast with `rubeus.exe asreproast`
- Create a golden ticket with `rubeus.exe golden /ldap /user:targetuser /aes256:<KRBTGT AES256 hash>`
- We can do Overpass the Hash, which is where we use an NTLM hash to get a Kerberos ticket
- `rubeus.exe asktgt /user:targetuser /ntlm:theirntlmhash /nowrap`
- We can use **ptt** (pass the ticket) to paste in a ticket and use it in-memory

Network Detections

- Kerberoasting will generate a 4769 event, which is normal for requesting TGS
  - However, a skilled defender could create a honeypot and monitor for 4769 events, so if stealth is a priority, enumerate first, and kerberoast targets one-by-one
  - Tools like Rubeus will automatically kerberoast every SPN. Very bad!
- AS-REP roasting will generate a 4768 event with an RC4 (!!!) encryption type and a preauth type of zero
  - There are some instances where RC4 is acceptable, but you generally want to be using AES256 whenever appropriate (mismatching encryption types stick out from normal activity)

OPSEC Considerations

- Rubeus will make a request with **randomly generated domain info** if it is not specified. It is trivial to identify ticket requests that go out to something like AqMvbnZ.local
- If your process shouldn't be making Kerberos requests (and you have Rubeus injected into it), you will generate an event for "**Kerberos activity from an anomalous process**". If you instead use Mimikatz, you will touch LSASS, which is even worse.

Domain Dominance

- Scenario: you've compromised a Domain Admin account and are now ready to own all the things
- First step: use your credentials to dump the Domain Controller's NTDS.dit remotely
  - There are many ways of doing this, including Mimikatz, NetExec, impacket-secretsdump
  - If you have a domain admin account, disabling all associated security software will be a walk in the park (if you can write malware, that is)
- Next, take the KRBTGT NTLM hash and use it to forge a Golden Ticket
- We can set the golden ticket expiration time to be 10 years or so

Golden Tickets

- It's what it sounds like - a magical skeleton key that lets you log into anywhere in the domain with all of the privileges, and, by default, *it works forever* (KRBTGT password is not rotated)
- Once generated, just pass-the-ticket with Rubeus or Impacket
- Make sure to specify the Domain SID (use SharpView etc.)

```
Rubeus.exe golden /aes256:51d7…4e7e /user:nlamb /domain:dev.cyberbotic.io
/sid:S-1-5-21-569305411-121244042-2357301523 /nowrap
```

```
Rubeus.exe createnetonly /program:C:\Windows\System32\cmd.exe /domain:DEV
/username:nlamb /password:FakePass /ticket:doIFLz[...snip...]MuaW8
```

Domain Dominance

- This is just the tip of the iceberg, it only gets worse than this
- Often times, acquiring domain admin means that recovering the domain for the defense will require a full domain rebuild, and you will have power over everything in the domain
- Getting domain admin is usually the last step before you can start actually acting on your objectives
- Try to avoid noisy techniques like creating new domain admins unless it is absolutely necessary

Review

- This is only the most basic of AD attacks
  - The more complex the system becomes, the more vulnerable it becomes
  - Future topics: Delegation, DACL abuse, S4U, ADCS, MSSQL, GPO, SCCM
- Don't forget to chain this with other Windows vulnerabilities
- While you're learning, lean heavily on BloodHound, but also do manual as well so you can see what manual query corresponds to what relationship in BloodHound
  - Stealthy red teamers don't get to use BloodHound much because it makes a ton of LDAP queries
- There are tons of AD practice resources out there!

Next Meetings

2025-10-16 • This Thursday

- Detecting Windows Attacks
- Learn to do forensics on Native Windows Hosts & Kerberos

2025-10-21 • Next Tuesday

- Active Directory II
- Delegation, LAPS, DACLs, S4U, and more!

2025-10-28 • Next Next Tuesday

- Active Directory III
- Asymmetric Cryptography, MSSQL, Smart Cards, cross-protocol attacks, and SCCM

ctf.sigpwny.com

sigpwny{Domain Expansion}

Meeting content can be found at sigpwny.com/meetings.

SIGPwny