### 1) Pass The TGT

In another scenario within the LMDG dataset, a Pass-the-TGT (Ticket Granting Ticket) attack is demonstrated, which aligns with the “Use of Alternate Authentication Material” tactic in the MITRE ATT&CK framework. The attack sequence is outlined in Figure 5. The scenario initiates with an attacker who has access to the SSH server with credentials of the local administrator of subnet 1 (step 1) then gained access to the DC 2 host within subnet 1 via SSH (step 2), where the enterprise administrator presented in subnet 0 recently authenticated via SSH. The attacker begins by dumping the Local Security Authority Subsystem Service (LSASS) memory on the compromised host. This action exposes recent authentication tokens, including the enterprise administrator’s Ticket Granting Ticket (TGT).

Once the TGT is identified and extracted, the attacker injects it directly into memory (step 3 in Figure 5), effectively impersonating the domain administrator. This impersonation grants access to elevated privileges without requiring the administrator’s plaintext credentials. In step 4 (Figure 5), using the injected TGT, the attacker accesses restricted file servers within subnet 6, extracting sensitive information from directories exclusively accessible to domain administrators. This privilege escalation represents a transition in the attacker’s permissions and enables a pivot across different access levels, as outlined in the Discussion section VII.

However, the attack ultimately fails due to several technical barriers:

Replay Protection Mechanisms: Kerberos incorporates replay protection mechanisms, which prevent reused or duplicated tickets from being accepted. If the same TGT is used multiple times in a short period, the KDC or service may detect it as a replay attempt and block authentication.

Session and Context Constraints: In this scenario, the session associated with the TGT was bound to specific network contexts that could not be easily duplicated by the attacker as the ticket was from another subdomain so the connection is inter-domain, limiting the ticket's effectiveness on additional systems.

Domain Policies and Auditing: Advanced domain configurations and monitoring solutions due to the windows server version we are using can detect and block unusual or suspicious ticket requests, especially if they come from unexpected endpoints or involve accounts with sensitive permissions.

Despite these obstacles, the attack was partially successful up to step 2 (as illustrated in Figure 5), where the attacker successfully obtained the TGT. However, due to these practical constraints, the attacker could not advance to step 3 to gain access to the restricted file server within subnet 4 or exfiltrate sensitive information.

The privilege escalation at step 3 exemplifies lateral movement within the network, specifically via authentication token manipulation. This access allows the attacker to move seamlessly through secure environments, exfiltrating data from high-security locations without raising immediate suspicion.

### 2) Asreproastable

In another scenario within the LMDG dataset, an AS-REP Roasting attack demonstrates a technique aligned with the *Credential Access* tactic in the MITRE ATT&CK framework. The attack sequence, as detailed in Figure 5, begins with an attacker who initially gains access to the SSH server using credentials of a standard user (step 1). Using this access, the attacker moves to host 7 within subnet 2 via SSH (step 2). With access established, the attacker downloads necessary tools, including **Rubeus**, to facilitate the AS-REP Roasting attack.

AS-REP Roasting (also known as "ASREPRoast") exploits a vulnerability in Active Directory (AD) accounts configured without requiring Kerberos preauthentication. This lack of preauthentication allows an attacker to request an encrypted Ticket Granting Service (TGS) response directly, exposing the password hash, which can then be extracted and cracked offline to reveal plaintext credentials. In this scenario, the attacker identifies an AS-REP Roasting user within the domain, extracts the hash, and successfully cracks it offline. With the recovered credentials, the attacker moves laterally within the network using the newly obtained user account (step 3).

After establishing access under this new identity, the attacker requests a Ticket Granting Ticket (TGT) with delegation rights. This TGT is saved locally, and the attacker uses PowerShell to extract the Base64-encoded portion of the ticket, storing it as `ticket.base64`. Leveraging Rubeus once again, the attacker initiates Service-for-User (S4U) impersonation to escalate privileges by impersonating the `Administrator` user for a specific service (in this case, HTTP/service within subnet 2). This impersonation uses the S4U2Self and S4U2Proxy Kerberos extensions, allowing the attacker to exploit Kerberos delegation mechanisms. By injecting this ticket into memory, the attacker aims to achieve Pass-the-Ticket (PTT) authentication, gaining unauthorized access to target resources with elevated privileges, an advanced privilege escalation technique in Kerberos-enabled environments.

Despite these efforts, the attack ultimately fails due to technical constraints:

- **Strict KDC and Domain Policies**: The environment's Windows Server version enforces stringent Kerberos and domain policies, which include restrictions on TGT delegation and impose limitations on service accounts performing impersonation. For instance, if Service-for-User-to-Proxy (S4U2Proxy) functionality is disabled, the attacker's attempt to impersonate the Administrator fails. Additionally, advanced domain monitoring and auditing configurations may detect abnormal Kerberos requests, flagging or blocking them in real-time.

- **Kerberos Constrained Delegation (KCD) Restrictions**: The network employs Kerberos Constrained Delegation (KCD), requiring explicit permissions for delegation to specific services. In this scenario, the attacker's compromised account lacks authorization to delegate to the HTTP service within subnet 2, preventing the impersonation attempt from succeeding.

These protective measures prevent the attacker from advancing past step 3 and achieving full privilege escalation, thereby restricting lateral movement and mitigating unauthorized access to sensitive resources.

### 3) Attack delegation

In another scenario from the LMDG dataset, an advanced delegation attack demonstrates multiple techniques spanning *Credential Access*, *Persistence*, and *Privilege Escalation* tactics within the MITRE ATT&CK framework. The sequence, detailed in Figure X, begins with an attacker who initially gains access to an SSH server using the credentials of a standard user (step 1). The attacker then transitions laterally to host 9 within subnet 4 via SSH (step 2). With access to this host, the attacker downloads key tools, such as *Rubeus*, necessary to facilitate an AS-REP Roasting attack.

In this scenario, the attacker identifies a user in the domain who is vulnerable to AS-REP Roasting, extracts the hash, and cracks it offline. With the recovered credentials, the attacker conducts lateral movement by using this new user account to further their access within the network (step 3).

Upon acquiring additional access, the attacker identifies a misconfiguration involving the `AddSelf` permission under the current user account. This permission enables the attacker to assign themselves to AD groups with higher privileges. Exploiting this, the attacker leverages the `AddSelf` permission to add their account to the *Exchange Windows Permissions* group, which has broad rights within AD (step 4???). The attacker now has a foothold to manipulate permissions further and moves to escalate privileges.

With these elevated permissions, the attacker then uses *PowerView* (from the PowerSploit toolkit) to modify AD object permissions. By executing the `Add-DomainObjectAcl` command, the attacker assigns themselves *DCSync* rights over the *Domain Admins* group by altering the Access Control List (ACL) (step 5???). The DCSync right is a highly privileged permission that allows the account to perform directory replication activities, effectively emulating domain controller operations. This privilege enables the attacker to request password hashes and other sensitive data directly from the domain controller, granting access to high-privilege accounts, including domain administrators. This escalation step provides the attacker with control over the domain without direct access to the domain controller.

Following this setup, the attacker performs a credential theft attack using *Mimikatz*, invoking the `lsadump::dcsync` command to impersonate a domain controller and retrieve sensitive authentication data, specifically the NTLM hash of the domain administrator within subnet 4. This process leverages the granted DCSync rights to replicate AD data without compromising the domain controller itself. By capturing the NTLM hash, the attacker gains the ability to impersonate the administrator, potentially carrying out *Pass-the-Hash (PtH)* attacks to gain unauthorised access across the domain.

In step 6???, the attacker conducts the PtH attack to gain domain administrator access.

However, due to the security restrictions related to *double-hop authentication*, the attack fails to propagate the administrator's full rights across machines. To overcome this, the attacker reinitiates the attack in step 6 within the same machine context, renewing the Ticket Granting Ticket (TGT) to ensure persistence and full administrator rights. Finally, in step 7, with administrator privileges secured, the attacker is able to access sensitive data on restricted file servers within subnet 6, demonstrating a complete compromise and data exfiltration pathway within the network.

This scenario highlights the critical risk posed by unauthorised delegation and improperly configured permissions, especially when *AddSelf* and DCSync rights are exploited in combination to achieve unauthorised privilege escalation and persistence within an AD environment.

### 4) Password spray

In another scenario from the LMDG dataset, a password spray attack demonstrates multiple techniques spanning *Credential Access*, *Persistence*, and *Privilege Escalation and Exfiltration* tactics within the MITRE ATT&CK framework. The sequence, detailed in Figure X, begins with an attacker who initially gains access to an SSH server using the credentials of a standard user (step 1). The attacker then transitions laterally to host 7 within subnet 3 via SSH using the same credentials (step 2). With access to this host, the attacker finds a zip file protected with a password. The attacker will then download a wordlist that he will need to perform his attack of brute forcing the zip file using an automated custom script where he succeeds. The attacker identifies the users in the domain and then performs password spraying attack where he successfully finds a valid combination of credentials to be the domain administrator on subnet 3.

In step 3, the attacker will login with the found credentials performing privilege escalation gaining all the rights on the subnet 3 domain with administrator account on the DC 4 machine. The attacker looks for writable shares where he finds one owned by the enterprise administrator where a script is present and is being executed by the enterprise administrator periodically. The attacker abuses the privilege of the owned account to write into this script adding a malicious payload to take ownership of the enterprise admin by a reverse shell payload.

A reverse shell in  cybersecurity is a network connection in which a compromised system initiates an outbound connection to a remote attacker-controlled machine, creating a shell access session. Unlike a standard shell, where an attacker attempts to connect directly to the target system, a reverse shell "reverses" the connection flow.

When the reverse shell is executed in step 4, the compromised system opens a specific port and connects to the attacker’s machine, typically via protocols like TCP or HTTP gaining access over the enterprise administrator account compromising the whole domain.

Finally in step 5, the attacker is now able to access the forbidden data being present in the file server in subnet 6.

In another scenario within the LMDG dataset, a password spray attack showcases a coordinated series of techniques that span *Credential Access*, *Persistence*, *Privilege Escalation*, and *Exfiltration* tactics as outlined in the MITRE ATT&CK framework. The sequence, as shown in Figure 5, begins when an attacker gains initial access to an SSH server using the credentials of a standard user (Step 1). Leveraging this initial foothold, the attacker moves laterally to host 7 in subnet 2 via SSH (Step 2). On this host, the attacker encounters a password-protected ZIP file and, after downloading a wordlist, executes a brute-force attack using a custom automated script to extract the file's contents.

Following the successful decryption of the ZIP file, the attacker enumerates domain users and proceeds with a password spray attack to discover a valid credential set for a high-privilege domain account. This tactic is effective due to its lower probability of triggering account lockouts, and, in this case, leads to the discovery of the domain administrator’s credentials in subnet 2.

In Step 3, the attacker uses the compromised credentials to escalate privileges by logging in as the domain administrator on DC 3, thus gaining elevated rights within subnet 2. While exploring the network, the attacker locates a writable network share owned by the enterprise administrator. This share contains a script that is regularly executed under the enterprise administrator’s context. By modifying this script, the attacker injects a malicious payload, specifically a reverse shell, to enable remote command execution under elevated privileges.

A reverse shell is a network connection in which the compromised system initiates an outbound connection to the attacker’s machine, effectively “reversing” the typical client-server model to bypass network restrictions. Unlike a traditional shell, where an attacker directly connects to a target system, a reverse shell enables the target system to connect to the attacker, often bypassing firewall constraints on inbound connections. When the reverse shell executes (Step 4), it establishes a connection over a specific port, often through TCP or HTTP, to the attacker’s remote machine, providing remote shell access with the privileges of the enterprise administrator.

With this level of access, the attacker compromises the enterprise administrator account, effectively gaining control over the entire domain. In the final step (Step 5), the attacker leverages these privileges to access sensitive and restricted files located on a file server within subnet 6, completing the exfiltration phase of the attack. This scenario exemplifies the integration of privilege escalation, persistence, and exfiltration tactics to achieve domain-wide compromise and data theft in a structured multi-step attack.

## 5) Silver ticket

In another scenario within the LMDG dataset, a silver ticket attack is demonstrated, which aligns with the Privilege Escalation and Persistence tactics in the MITRE ATT&CK framework. The attack sequence is outlined in Figure X. The scenario initiates with an attacker who has access to the SSH server with credentials of the local administrator of subnet 5 (step 1) then gained access to the DC 6 host within subnet 5 via SSH (step 2), where the enterprise administrator presented in subnet 0 recently authenticated via SSH.

The attacker begins by downloading the tools that he will be needing in the attack like mimikatz. The attacker begins by dumping the Local Security Authority Subsystem Service (LSASS) memory on the compromised host using mimikatz. This action exposes recent authentication tokens, including the enterprise administrator’s NTLM hash. In this attack,the attacker  utilises Mimikatz to perform a **Silver Ticket** attack against a specific service within a Kerberos-enabled network. The attacker forges a **service ticket** for a service on the target machine, using the credentials of the `enterprise administrator`. The forged ticket is created by specifying the user, domain, service, and the NTLM hash of the administrator's hash. This service ticket is injected into the current session using the (Pass-the-Ticket) option, allowing the attacker to authenticate as the enterprise `Administrator` account to access the service without needing a valid Ticket Granting Ticket (TGT) (step 3).

Unlike a **Golden Ticket** which grants broad access across the domain, the **Silver Ticket** is restricted to the targeted service , but still allows the attacker to bypass normal authentication mechanisms and gain unauthorized access to the service, potentially enabling the exfiltration of sensitive data or further exploitation of the network.

After performing this attack in step 3, the attacker will now move to step 4 to try to access the forbidden data only accessible by the enterprise administrator on the file server on subnet 6.

However, the attack ultimately fails due to several technical barriers:

Replay Protection Mechanisms: Kerberos incorporates replay protection mechanisms, which prevent reused or duplicated tickets from being accepted. If the same TGT is used multiple times in a short period, the KDC or service may detect it as a replay attempt and block authentication.

Session and Context Constraints: In this scenario, the session associated with the TGT was bound to specific network contexts that could not be easily duplicated by the attacker as the ticket was from another subdomain so the connection is inter-domain, limiting the ticket's effectiveness on additional systems.

Domain Policies and Auditing: Advanced domain configurations and monitoring solutions due to the windows server version we are using can detect and block unusual or suspicious ticket requests, especially if they come from unexpected endpoints or involve accounts with sensitive permissions.

**Service Ticket Signature Validation**: Kerberos uses encryption and integrity checks to validate service tickets. If the service on the target server is properly configured to check the integrity of incoming service tickets (by validating the ticket’s signature against the server’s key), the forged Silver Ticket will be rejected.

Despite these obstacles, the attack was partially successful up to step 2 (as illustrated in Figure x), where the attacker successfully obtained the Ticket. However, due to these practical constraints, the attacker could not advance to step 3 to gain access to the restricted file server within subnet 4 or exfiltrate sensitive information.

### 6) Golden Ticket

In another scenario within the LMDG dataset, a **Golden Ticket** attack is demonstrated, aligning with the **Privilege Escalation** and **Persistence** tactics in the MITRE ATT&CK framework. The attack sequence, as detailed in Figure X, initiates with an attacker who initially gains access to the SSH server using the credentials of a local administrator in subnet 5 (step 1) and subsequently gains access to the DC 6 host within subnet 5 via SSH (step 2), where the enterprise administrator in subnet 0 recently authenticated to the domain controller.

Upon accessing the domain controller, the attacker downloads necessary tools such as Mimikatz. Using Mimikatz, the attacker dumps the memory of the **Local Security Authority Subsystem Service (LSASS)**, revealing cached authentication tokens, including the NTLM hash and AES key of the enterprise administrator's account. Leveraging this information, the attacker generates a **Golden Ticket**—a forged Kerberos Ticket Granting Ticket (TGT) for the domain administrator account, utilising the `/aes256` parameter to specify the administrator's AES key.

The attacker then forges the Golden Ticket with Mimikatz. This crafted ticket allows the attacker to impersonate the enterprise administrator across the entire domain (step 3). Unlike a **Silver Ticket** which limits access to a specific service, a Golden Ticket provides domain-wide access to any Kerberos-enabled resource without needing a valid initial TGT, enabling full administrative privileges and sustained persistence within the network.

After creating and injecting the Golden Ticket in step 3, the attacker attempts to use it in step 4 to access sensitive files exclusive to the enterprise administrator on a file server within subnet 6.

However, despite the attack’s potential, it fails to achieve complete success due to several key security constraints:

1. **Replay Protection Mechanisms**: Kerberos includes replay protection that prevents the reuse of tickets within short intervals. If the Golden Ticket is reused frequently in a short span, detection mechanisms may flag it as a replay attempt and block access.
2. **Session and Context Constraints**: In this environment, network security policies tie sessions to specific subdomain contexts, which limit the attacker's ability to reuse the Golden Ticket across different network contexts (particularly in inter-domain scenarios).
3. **Domain Policies and Auditing**: Advanced configurations and monitoring policies in the Windows Server environment detect and log suspicious ticket activities, particularly if high-privilege tickets appear unexpectedly on new endpoints or subdomains.
4. **Service Ticket Signature Validation**: Although the Golden Ticket bypasses typical TGT verification, service tickets themselves are often validated by services before granting access. If the server receiving the Golden Ticket is configured to verify ticket signatures, it can detect and reject unauthorised tickets.

These security measures hinder the attacker’s ability to execute step 3 and gain access to the protected file server within subnet 6. Although the Golden Ticket allowed the attacker to escalate privileges and bypass certain security controls, these defences prevented data exfiltration and other unauthorised activities, demonstrating the importance of robust Kerberos policy enforcement and monitoring in mitigating advanced persistent threats.