Introduction

This guidance – authored by the Australian Signals Directorate (ASD), the Cybersecurity and Infrastructure Security Agency (CISA), the National Security Agency (NSA), the Canadian Centre for Cyber Security (CCCS), the New Zealand National Cyber Security Centre (NCSC-NZ), and the United Kingdom's National Cyber Security Centre (NCSC-UK) – aims to inform organisations about 17 common techniques used to target Active Directory as observed by the authoring agencies. This guidance provides an overview of each technique and how it can be leveraged by malicious actors, as well as recommended strategies to mitigate these techniques. By implementing the recommendations in this guidance, organisations can significantly improve their Active Directory security, and therefore their overall network security, to prevent intrusions by malicious actors.

Microsoft's Active Directory is the most widely used authentication and authorisation solution in enterprise information technology (IT) networks globally. Active Directory provides multiple services, including Active Directory Domain Services (AD DS), Active Directory Federation Services (AD FS) and Active Directory Certificate Services (AD CS). These services provide multiple authentication options, including smart card logon, as well as single sign-on with on-premises and cloud-based services.

Active Directory's pivotal role in authentication and authorisation makes it a valuable target for malicious actors. It is routinely targeted as part of malicious activity on enterprise IT networks. Active Directory is susceptible to compromise due to its permissive default settings, its complex relationships, and permissions; support for legacy protocols and a lack of tooling for diagnosing Active Directory security issues. These issues are commonly exploited by malicious actors to compromise Active Directory.

Specifically, Active Directory's susceptibility to compromise is, in part, because every user in Active Directory has sufficient permission to enable them to both identify and exploit weaknesses. These permissions make Active Directory's attack surface exceptionally large and difficult to defend against. Also contributing to its vulnerability is the complexity and opaqueness of relationships that exist within Active Directory between different users and systems. It is often these hidden relationships, which are overlooked by organisations, that malicious actors exploit, sometimes in trivial ways, to gain complete control over an organisation's enterprise IT network.

Gaining control of Active Directory gives malicious actors privileged access to all systems and users that Active Directory manages. With this privileged access, malicious actors can bypass other controls and access systems, including email and file servers, and critical business applications at will. This privileged access can often be extended to cloud-based systems and services via Microsoft's cloud-based identity and access solution, Microsoft Entra ID (note: Microsoft Entra ID is a paid feature). This allows users to access cloud-based systems and services, however, it can also be exploited by malicious actors to maintain and expand their access. Gaining control of Active Directory can enable malicious actors with a range of intentions, whether they be cyber criminals seeking financial gain or nation states conducting cyber espionage, to obtain the access they need to achieve their malicious objectives in the victim's network.

Active Directory can be misused by malicious actors to establish persistence in organisations. Some persistence techniques allow malicious actors to log in to organisations remotely, even bypassing multi-factor authentication (MFA) controls. Many of these persistence techniques are resistant to cyber security incident response remediation activities intended to evict malicious actors. Additionally, sophisticated malicious actors may persist for months or even years inside Active Directory. Evicting the most determined malicious actors can require drastic action, ranging from resetting all users' passwords to rebuilding Active Directory itself. Responding to and recovering from malicious activity involving Active Directory compromise is often time consuming, costly, and disruptive. Therefore, organisations are encouraged to implement the recommendations within this guidance to better protect Active Directory from malicious actors and prevent them from compromising it.

Understanding Active Directory

For many organisations, Active Directory consists of thousands of objects interacting with each other via a complex set of permissions, configurations and relationships. Understanding object permissions and the relationships between those objects is critical to securing an Active Directory environment.

To gain a better understanding of an organisation's environment, malicious actors commonly enumerate Active Directory for information after gaining initial access to an environment with Active Directory. Using the information gained, they seek to understand the structure, objects, configurations and relationships that are unique to each organisation. By doing this, malicious actors sometimes gain a better understanding of the organisation's Active Directory environment than the organisation itself. This enables them to target Active Directory with increased likelihood of success. Malicious actors use their knowledge of the environment to exploit weakness and misconfigurations to escalate their privileges, move laterally, and gain full control of the Active Directory domain.

To improve Active Directory, organisations must comprehensively understand their own unique configuration of Active Directory. There are numerous commercial and open source tools available to support an organisation's understanding of Active Directory, including the following:

- BloodHound: A tool that provides a graphical user interface to help with understanding Active Directory, as well identifying any misconfigurations and weaknesses that malicious actors may seek to exploit.
- PingCastle: A tool that provides an Active Directory security report.
- Purple Knight: An application that provides information on the security of an Active Directory environment.

Active Directory Objects

Active Directory stores data as objects that represent different resources, such as users, computers, groups and organisational units. The most common objects in an Active Directory domain are user and computer objects. User objects represent real users, service accounts and built-in users such as the Kerberos Ticket Granting Ticket (KRBTGT) user object. Computer objects represent systems, such as servers and workstations in a domain. Every server and workstation that is joined to a domain has a corresponding computer object in Active Directory. These objects are used by Active Directory for authentication, authorisation and policy enforcement.

Detecting and mitigating Active Directory compromises

There are many known and observed techniques used to compromise AD DS, AD CS and AD FS. Malicious actors target these services to escalate their privileges and move laterally across enterprise IT networks. This guidance addresses the most common AD DS, AD CS and AD FS techniques, providing an overview of each technique, as well as how to mitigate it. This guidance organises the outlined compromises in the sequence they are typically executed against Active Directory, beginning with those used to escalate privileges and move laterally, and concluding with compromises aimed at establishing persistence.

Securing privileged access

The immediate objective of malicious activity involving Active Directory is to escalate privileges and gain control of a domain by targeting the highest privileged user objects, such as those in the Domain Admins and Enterprise Admins security groups. Although significant access can often be obtained by targeting other user objects, such as service accounts, preventing malicious actors from acquiring the highest privileges is crucial for limiting their overall access. Therefore, securing privileged access is essential for mitigating Active Directory compromises and should be a top priority for all organisations.

Securing privileged access in Active Directory can be achieved by using a tiered model, such as Microsoft's Enterprise Access Model. This model supersedes and advances the previous tiered model, commonly referred to as the Active Directory Administrative Tier Model. The Enterprise Access Model is intended for hybrid environments – which are increasingly prevalent – where Active Directory is connected to cloud services, such as those in Microsoft Azure, using Microsoft Entra ID through Entra Connect or AD FS.

The following are the key principles of securing privileged access:

- Tier 0 user objects do not expose their credentials to lower tier systems. Tier 0 user objects are any user object that has significant access in a domain (for example, those in the Domain Admins and Enterprise Admins security group, KRBTGT user objects, the AD FS service account, backup administrators and Microsoft Entra Connect user objects). For a full list of typical Tier 0 objects, refer to Microsoft's Privileged Access Security Levels.
- Tier 0 computer objects are only managed by Tier 0 user objects. Tier 0 computer objects include Domain Controllers, the AD FS server, the AD CS root certificate authority, backup servers, and the Microsoft Entra Connect server.
- User and computer objects in lower tiers (such as Tier 1 or Tier 2) can use services provided by higher tiers, but the reverse is not permitted.
- Hierarchy is enforced to prevent control of higher tiers from lower tiers.
- Privileged access pathways are secured by minimising the number of privileged access pathways, implementing protections, and closely monitoring the pathways.

Tier 0 user and computer objects should be given additional security protections, including, but not limited to, phishing-resistant MFA, the use of privileged access workstations, Kerberos armoring and zero trust policy enforcement.

Implementing Microsoft's Enterprise Access Model makes many techniques utilised against Active Directory significantly more difficult to execute and renders some of them impossible. Malicious actors will need to resort to more complex and riskier techniques, thereby increasing the likelihood their activities will be detected. Implementing this model enables organisations to identify Active Directory compromises, significantly reducing response time and limiting overall impact.

Kerberoasting

Kerberoasting exploits user objects configured with a service principal name (SPN). If a user object is configured with an SPN, any other user object – including unprivileged users – can request its ticket granting service (TGS) ticket from a Domain Controller (this functionality is by design allowing user objects to interact with services). The TGS ticket is encrypted with the user object's password hash, which can be cracked to reveal the cleartext password. If malicious actors can crack the TGS ticket, and obtain the cleartext password, they can then authenticate as the user object (see **Figure 1**).

Kerberoasting may be executed by malicious actors shortly after gaining initial access to an Active Directory domain to attempt to escalate privileges and move laterally. The types of user objects configured with SPNs are commonly referred to as service accounts. These are user objects that run services on computer objects, and may have administrative privileges. If one of these service accounts is compromised via Kerberoasting, it often provides additional privileges and access, which malicious actors can use to further compromise an Active Directory environment. In some instances, service accounts may be members of highly privileged security groups, such as the Domain Admins security group, which provides administrative access to all user and computer objects, as well as other objects in Active Directory. The compromise of a service account in the Domain Admins security group, or other highly privileged security group, often results in the complete compromise of a domain.

There are multiple offensive security tools (such as Mimikatz, Rubeus and Impacket) that can perform Kerberoasting. It is also possible to execute Kerberoasting using native PowerShell commands.

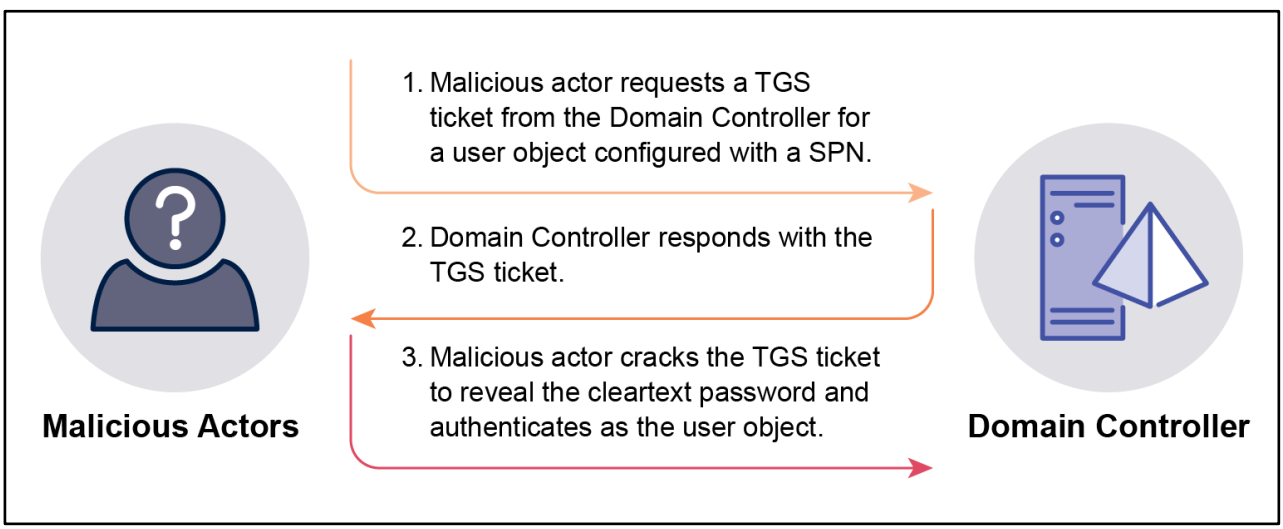

Figure 1: Overview of Kerberoasting

Mitigating Kerberoasting

Kerberoasting mimics normal Active Directory functionality and is a viable technique in any Active Directory environment that has user objects configured with SPNs.

The following security controls should be implemented to mitigate Kerberoasting:

- **Minimise the number of user objects configured with SPNs.** This reduces the attack surface for malicious actors to execute Kerberoasting.
- **Create user objects with SPNs as group Managed Service Accounts (gMSAs).** gMSAs have automatic password rotation, a 120-character password and simplified SPN management. These security features protect the password from being cracked, reducing the likelihood of successful Kerberoasting. However, if creating user objects with SPNs as gMSAs is not feasible, for example, it is a non-Windows based system hosting the service, or the application does not fully support gMSAs, such as Microsoft's System Center Configuration Manager, set a minimum 30-character password that is unique, unpredictable and managed.
- **Assign user objects with SPNs the minimum privileges necessary to perform their functions and make sure they are not members of highly privileged security groups, such as the Domain Admins security group.** If malicious actors successfully execute Kerberoasting and crack the TGS ticket to reveal the cleartext password, minimising the privileges assigned to the user object will reduce the impact and limit the access gained by the malicious actor.

Detecting Kerberoasting

Detecting Kerberoasting can be difficult as this technique mirrors legitimate Active Directory activity. User objects request TGS tickets when accessing services in the domain, and the events generated for this legitimate activity are the same events that are generated by Kerberoasting. This makes it possible for Kerberoasting to be missed amongst the numerous legitimate events that are logged. One method to detect Kerberoasting is to analyse TGS request events (event 4769) and identify instances where TGS requests are made for multiple user objects configured with SPNs within a short timeframe. Kerberoasting typically involves retrieving TGS tickets for all user objects with SPNs simultaneously. The presence of TGS ticket request events (event 4769) for numerous user objects with SPNs in a brief period may indicate that Kerberoasting occurred. Another method for detecting Kerberoasting is to analyse unusual TGS requests for services. For instance, this could include TGS requests for services not typically accessed by the requesting user or computer object, such as a backup server that is usually only accessed by other servers.

The events in **Table 1** should be centrally logged and analysed in a timely manner to identify Kerberoasting.

Table 1. Events that detect Kerberoasting

Event ID	Source	Description
4738, 5136	Domain Controllers	These events are generated when a user account is changed. Malicious actors can modify user objects and add a SPN so they can retrieve their Kerberos service ticket. Once the Kerberos service ticket has been retrieved, the user object is modified again and the SPN removed.
4769	Domain Controllers	This event is generated when a TGS ticket is requested. When malicious actors execute Kerberoasting, event 4769 is generated for each TGS ticket that is requested for a user object. Malicious actors commonly try to retrieve TGS tickets with Rivest Cipher 4 (RC4) encryption as these tickets are easier to crack to reveal their cleartext password. If a TGS is requested with RC4 encryption, then the Ticket Encryption type contains the value '0x17' for event 4769. As this encryption type is less frequently used, there

should be fewer instances of event 4769 with RC4 encryption, making it easier to identify potential Kerberoasting activity.

Common offensive security tools used by malicious actors to perform Kerberoasting will set the Ticket Options value to '0x40800000' or '0x40810000'. These values determine the capabilities of the TGS ticket and how it can be used by malicious actors. As these Ticket Options values are commonly used by offensive security tools to perform Kerberoasting, they can be used to identify Kerberoasting activity.

This technique can be detected with the assistance of Active Directory canaries. For more information, see section Detecting Active Directory Compromises with Canaries.

Authentication Server Response (AS-REP) Roasting

AS-REP Roasting exploits Active Directory user objects that are configured to not require Kerberos pre-authentication. Similar to Kerberoasting, any user object in the domain, including unprivileged user objects, can send an Authentication Server Request (AS-REQ) to retrieve the AS-REP ticket for any user object configured to not require Kerberos pre-authentication. The AS-REP ticket is encrypted with the user object's password hash, which can be cracked to reveal the cleartext password. If malicious actors crack the AS-REP ticket and obtain the cleartext password, then they can authenticate as the user object (see **Figure 2**).

AS-REP Roasting may be executed by malicious actors shortly after they gain initial access to an Active Directory domain to escalate their privileges and move laterally.

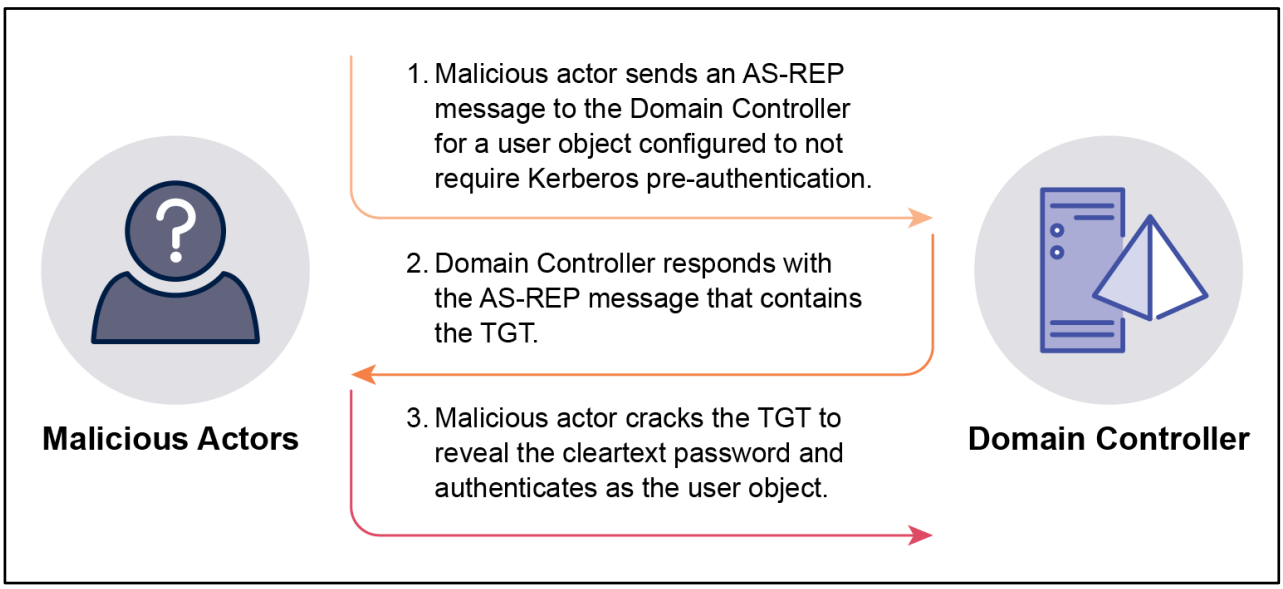

Figure 2: Overview of AS-REP Roasting

Mitigating AS-REP Roasting

Malicious actors have fewer opportunities to perform AS-REP Roasting than Kerberoasting, as user objects are configured by default in AD DS to require Kerberos pre-authentication, and AS-REP Roasting is only possible if user objects are configured to not require Kerberos pre-authentication. The Kerberos authentication protocol, introduced in Kerberos version 5, is the primary authentication protocol used by Active Directory. The configuration to not require Kerberos pre-authentication only exists to support systems that do not support Kerberos, which are typically

considered legacy IT and are less common. If organisations have applications and systems that use a version of Kerberos earlier than version 5, then they may configure user objects to not require Kerberos pre-authentication, leaving the environment vulnerable to AS-REP Roasting.

The following security control should be implemented to mitigate AS-REP Roasting:

- **Ensure user objects require Kerberos pre-authentication.** AS-REP Roasting is mitigated if all user objects require Kerberos pre-authentication. However, if user objects must be configured to bypass Kerberos pre-authentication, then these user objects should be granted the minimum set of privileges required for them to perform their functions and should not be members of highly privileged security groups, such as Domain Admins. Additionally, set a minimum 30-character password for service accounts or a minimum 15-character password for users, and ensure the password is unique, unpredictable and managed.

Detecting AS-REP Roasting

If applications or systems are using a version of Kerberos earlier than version 5, then AS-REP Roasting will mirror legitimate Active Directory activity, including triggering the same events, and may be difficult to detect. AS-REP Roasting typically involves simultaneously retrieving TGT tickets for all users configured to bypass Kerberos pre-authentication. As such, one method to detect AS-REP Roasting (similar to a method to detect Kerberoasting) is to analyse TGT request events (event 4768) and identify instances where TGT ticket requests are made for multiple user objects that have Kerberos pre-authentication disabled within a short timeframe.

The events in **Table 2** should be centrally logged and analysed in a timely manner to identify AS-REP Roasting.

Table 2. Events that detect AS-REP Roasting

Event ID	Source	Description
4625	Domain Controllers	This event is generated when an account fails to log on. AS-REP Roasting can be executed prior to authentication, meaning malicious actors only need to be connected to the domain without needing a valid user object. The AS-REP ticket is still retrieved, but event 4625 is generated as no valid credentials were provided when requesting the ticket. If AS-REP Roasting is executed in the context of a valid user object, then the AS-REP ticket is retrieved, valid credentials are provided and event 4625 is not generated. Event 4625 can be correlated with event 4768 to confirm if AS-REP Roasting was executed in the context of a valid domain user object.
4738, 5136	Domain Controllers	These events are generated when a user account is changed. Malicious actors can modify user objects and configure them to not require Kerberos pre-authentication as a technique to retrieve their AS-REP ticket. Once the AS-REP ticket service ticket has been retrieved, the user object is modified again to require Kerberos pre-authentication. If these events are generated for changes to the Kerberos pre-authentication, it may indicate AS-REP Roasting occurred.
4768	Domain Controllers	This event is generated when a TGT is requested. Malicious actors executing AS-REP Roasting trigger this event as the AS-REP message that is returned from a Domain Controller contains a TGT. If this

event is triggered multiple times in a short timeframe, it may indicate AS-REP Roasting has occurred.

Malicious actors will commonly try to retrieve TGT tickets with Rivest Cipher 4 (RC4) encryption as these TGT tickets are easier to crack to reveal their cleartext password. If a TGT is requested with RC4 encryption, then the Ticket Encryption type will contain the value '0x17' for event 4769. As this encryption type is less frequently used, there should be fewer instances of event 4769 with this encryption type, making it easier to identify potential AS-REP Roasting.

This type of compromise can be detected with the assistance of Active Directory canaries. For more information, see section Detecting Active Directory Compromises with Canaries.

Password spraying

Password spraying attempts to authenticate to multiple user objects using either a single password or multiple passwords until they successfully authenticate to a user object. These passwords can come from public password wordlists or be derived from the target environment for a higher likelihood of success. For example, malicious actors may identify passwords being reused in the target environment and use these in password spraying to identify if they belong to any user objects. To minimise authentication attempts and the risk of detection, malicious actors can retrieve a list of usernames from Active Directory and attempt to authenticate to each one using a single password. This technique is particularly effective against organisations that reuse passwords. If malicious actors compromise a user object via password spraying, then they control the user object and inherit the user object's access and privileges.

File shares and Active Directory credentials

The authoring agencies have observed malicious actors scanning file shares as part of their efforts to locate insecurely stored secrets, such as credentials for Active Directory user objects. Multiple tools, such as SMBMap and Snaffler, can identify file shares and scan them for credentials (including cleartext passwords), sensitive information, application programming interface (API) keys, digital certificates, standalone password managers, and backups. These tools are commonly used after gaining initial access to an Active Directory domain to try and locate credentials for privileged user objects.

If malicious actors locate credentials, they are likely to use them to escalate their privileges and move laterally. This reduces the likelihood of detecting malicious actors, as they are able to gain control of other user objects without having to attempt riskier techniques, such as **Kerberoasting** and **Password Spraying**.

To reduce the likelihood of malicious actors locating credentials in file shares and using them in password spraying, organisations should use an enterprise-grade password management solution (where possible) to secure store their sensitive information, including credentials for Active Directory user objects.

Additionally, organisations should periodically conduct their own scans to identify any sensitive information that is insecurely stored on file shares.

Many organisations enforce an account lockout threshold policy to lock user objects after a certain number of failed authentication attempts. This is effective at preventing malicious actors from attempting too many different passwords. However, malicious actors can still perform password spraying up to the account lockout threshold without locking out user objects. Different tools exist to perform password spraying that identify the lockout threshold to ensure the threshold is not exceeded, including DomainPasswordSpray and Spray. These tools may also be configured to perform password spraying over a certain time period to limit the number of authentication attempts occurring at any given time and minimise the risk of detection.

While MFA can be effective at mitigating password spraying by malicious actors attempting to gain initial access, it is largely ineffective at mitigating password spraying if malicious actors have already gained initial access. This is because the malicious actors can attempt to authenticate as any user object in the domain directly to a Domain Controller via the New Technology Local Area Network (LAN) Manager (NTLM) protocol, which does not support MFA. To reduce the risk of NTLM-based compromises, disable the NTLM protocol wherever possible. When this is not possible, enable LDAP channel binding, extended protection authentication, and Server Message Block (SMB) signing.

The built-in Administrator account and account lockout threshold

In every Active Directory domain, there is a built-in Administrator account made during the creation of the domain. This account is a default member of the Domain Admins and Administrators security groups, and if the domain is the forest root domain, it is also a member of the Enterprise Admins security group. These security groups make this user object highly privileged as it has administrator access on all objects in the domain.

The account lockout threshold policy that locks accounts after a certain number of failed authentication attempts does not apply to the built-in Administrator account. Even if multiple failed authentication attempts occur and the account reports it is locked out, it can still be authenticated to if the correct password is provided. This automatically removes the locked status and resets the bad password count to zero.

The inability for the built-in Administrator account to be locked out makes it an attractive target for password spraying. Specifically, malicious actors can continually spray this account with multiple passwords knowing that this account will not be locked out, and if the correct password is found, be able to login to the account successfully regardless of how many prior failed authentication attempts were made.

To reduce the risk of password spraying that targets the built-in Administrator user object, set a long (30-character minimum), unique, unpredictable and managed password. Additionally, this account should only be used as an emergency break glass account, and authentication events associated with this account should be monitored for signs of malicious activity such as a Password Spray or other unauthorised access. Organisations can also implement additional protections, including configuring the user object as sensitive to ensure it cannot be delegated and restrict where the user object can be used.

Mitigating password spraying

The following security controls should be implemented to mitigate password spraying:

- **Create passwords for local administrator accounts, service accounts, and break glass accounts that are long (30-character minimum), unique, unpredictable and managed.** Microsoft's Local Administrator Password Solution (LAPS) can be used to achieve this for local administrator accounts. Using strong passwords reduces the likelihood of successful password spraying.

- **Create passwords used for single-factor authentication that consist of at least four random words with a total minimum length of 15-characters** to reduce the likelihood of a successful password spraying.

- **Lock out user objects, except for break glass accounts, after a maximum of five failed logon attempts.** Enforcing an account lock threshold after five failed authentication attempts reduces the number of possible attempts in password spraying.

- **Ensure passwords created for user objects are randomly generated**, such as when a user object is created, or a user requests a password reset. Malicious actors will try to identify reused passwords and use these in password spraying to increase the likelihood of success.

- **Configure the built-in 'Administrator' domain account as sensitive to ensure it cannot be delegated.**

- **Scan networks at least monthly to identify any credentials that are being stored in the clear.** Malicious actors scan networks for cleartext credentials to use in password spraying. Locating and removing these cleartext credentials proactively mitigates this risk.

- **Disable the NTLM protocol.** The NTLM protocol does not support MFA and can be misused by malicious actors to bypass MFA requirements.

Detecting password spraying

Password spraying typically generates an event for each failed authentication attempt. Depending on the number of user objects being targeted, this could result in a large number of events being generated. To effectively detect password spraying, alerts should be generated when there are numerous failed authentication events that occur in a short period of time.

Popular password spraying tools, such as DomainPasswordSpray and CrackMapExec, commonly attempt to authenticate using the SMB protocol. Malicious actors may use another protocol, such as the Lightweight Directory Access Protocol (LDAP), which generates a different event in an attempt to avoid detection. It is important to collect both events to effectively monitor for password spraying.

The events in **Table 3** should be centrally logged and analysed in a timely manner to identify password spraying.

Table 3. Events that detect password spraying

Event ID	Source	Description
2889	Domain Controllers	This event is generated when a computer object tries to make an unsigned LDAP bind. Malicious actors using the LDAP protocol to conduct password spraying generate this event as each password attempt makes an unsigned LDAP bind. If numerous 2889 events occur in a short timeframe, this may indicate password spraying occurred using the LDAP protocol.
4624	Domain Controllers	This event is generated when an object logs on successfully, such as to a user object. If this event occurs near-simultaneously with 4625 events, this can indicate a user object was successfully logged on to as a result of password spraying.
4625	Domain Controllers	This event is generated when an object fails to log on via the SMB protocol. Common password spraying tools default to attempting authentication using the SMB protocol. If numerous 4625 events occur in a short timeframe, this may indicate password spraying occurred using the SMB protocol. Other protocols, such as LDAP, can also be used for password spraying. Malicious actors may choose to use a different protocol to avoid detection. The 'badPasswordTime' user object attribute in Active Directory can be queried to identify the date and time of the last failed authentication attempt. If multiple user objects share the same date and time, or nearly the same date and time, this may indicate password spraying occurred.

4648	Source of Password Spraying, such as a domain joined workstation or server	This event is generated when a logon is attempted using explicit credentials. If password spraying is executed on a domain joined system, this event is generated for each authentication attempt. If numerous 4648 events exist with different usernames in a short timeframe, this can indicate password spraying was executed on the system. Note, if malicious actors have established a tunnel from their infrastructure, they may be able to execute password spraying using their own systems, if this is the case, this event will not be generated.
4740	Domain Controllers	This event is generated when a user object is locked out. Password spraying can cause user objects to be locked out due to the number of failed authentication attempts. If multiple user objects are locked out in a short period of time, this may indicate password spraying occurred. Many password spraying tools check the domain's lockout policy and the number of failed authentication attempts for user objects to avoid lockout as a means to avoid detection.
4771	Domain Controllers	This event is generated when Kerberos pre-authentication fails. In an attempt to evade detection, malicious actors may use the LDAP protocol to execute password spraying. In this case, event 4771 generates the 'Failure Code' property of '0x18'. This value means the incorrect password is the cause for the event. The 'badPasswordTime' user object attribute in Active Directory can be queried to identify the date and time of the last failed authentication attempt. If multiple user objects share the same date and time, or nearly the same date and time, this may indicate password spraying occurred.

MachineAccountQuota compromise

A MachineAccountQuota compromise exploits the default Active Directory setting that allows user objects to create up to ten computer objects in the domain via the 'ms-DS-MachineAccountQuota' attribute. These computer objects are automatically added to the Domain Computers security group and inherit the group's privileges. Most malicious actors, to minimise the risk of detection, set the computer object's name to comply with any domain-specific naming conventions to appear similar to other computer objects. For example, if the Domain Computers security group is overly privileged, is a member of higher privileged security groups, or has privileges to other Active Directory objects, malicious actors can exploit this to escalate their privileges. Malicious actors can achieve this by creating their own computer object, authenticating as this computer object, and inheriting its privileges. This computer object can then be used to interact with the domain, similar to user objects. Computer objects can also access and interact with other systems and services in the domain (see Figure 3).

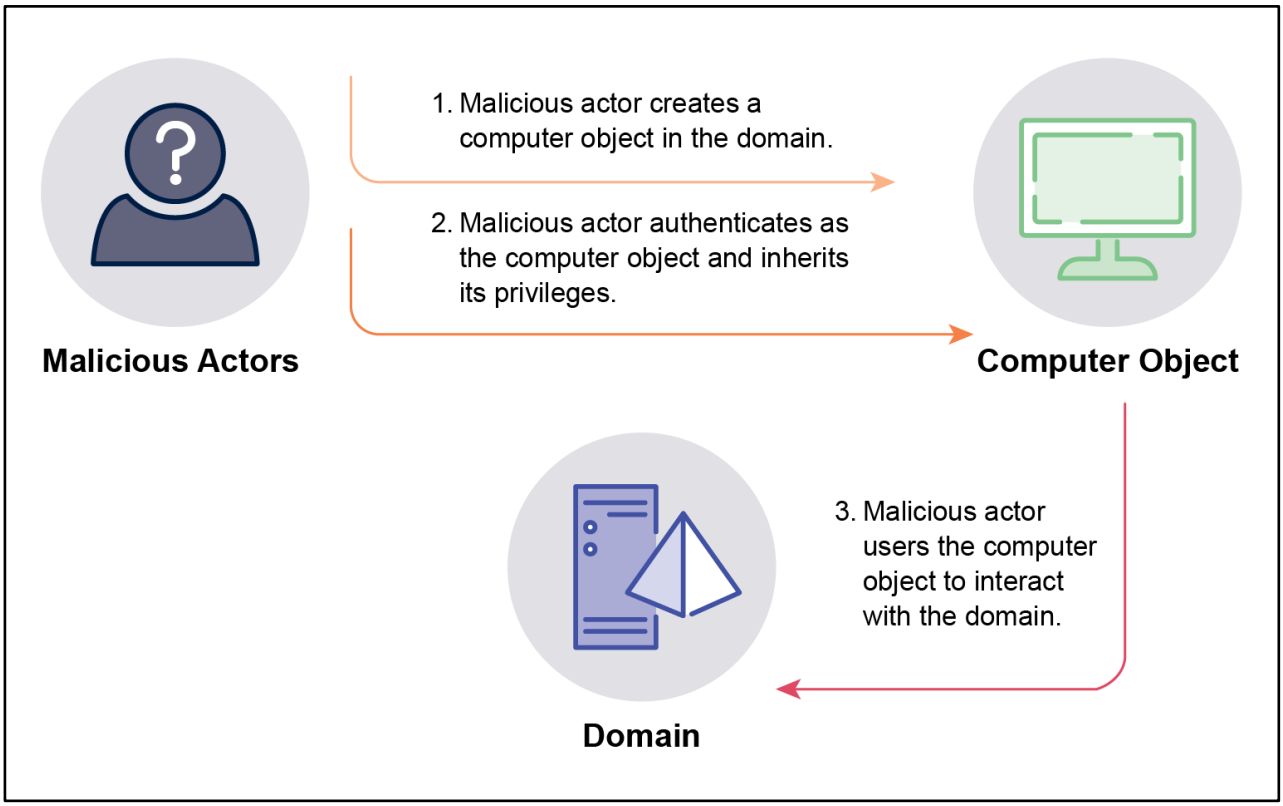

Figure 3: Overview of a MachineAccountQuota compromise

A MachineAccountQuota compromise can also be used as part of another compromise, known as KrbRelayUp. On systems in domains where LDAP signing is not enforced, which is the default in Active Directory setting, malicious actors can execute KrbRelayUp to escalate their privileges to local administrator.

Mitigating a MachineAccountQuota compromise

The following security controls should be implemented to mitigate a MachineAccountQuota compromise:

- **Configure unprivileged user objects so they cannot add computer objects to the domain.** This can be configured by setting the 'MS-DS-MachineAccountQuota' attribute in Active Directory to zero. Typically, only privileged staff, such as system administrators, need to add new computer objects to Active Directory; for example, when a new server or workstation needs to be joined to a domain.

- **Ensure the Domain Computers security group is not a member of privileged security groups.** This prevents malicious actors from escalating their privileges as a result of a MachineAccountQuota compromise.

- **Ensure the Domain Computers security group does not have write privileges to any objects in Active Directory.** This prevents malicious actors from gaining control or access to other Active Directory objects because of a MachineAccountQuota compromise.

- **Enable LDAP signing for Domain Controllers.** LDAP signing provides numerous security protections including user authentication, message signing and encryption. LDAP signing also mitigates against KrbRelayUp.

Detecting a MachineAccountQuota compromise

Every time a computer object is created in Active Directory, event 4741 is generated and includes information about the object's properties and who created it. This event can be analysed to determine whether the computer object was created for legitimate or malicious purposes. Additionally, creating the computer object for MachineAccountQuota requires setting its password. This also generates an event that can be an indicator of a MachineAccountQuota compromise.

The events in **Table 4** should be centrally logged and analysed in a timely manner to identify a MachineAccountQuota compromise.

Table 4. Events that detect a MachineAccountQuota compromise

Event ID	Source	Description
4624	Domain Controllers	This event is generated when an object successfully logs on. This event can be correlated with event 4741 to identify if the computer object created by malicious actors has authenticated to the domain.
4724	Domain Controllers	This event is generated when an attempt is made to reset an object's password. When malicious actors create a new computer object, they set its password so they can subsequently authenticate as the computer object. If this event is generated at the same time (or near the same time) as event 4741, this may indicate a MachineAccountQuota compromise has occurred.
4741	Domain Controllers	This event is generated when a computer object is created in Active Directory. This event can be used to identify a computer object created by malicious actors as part of a MachineAccountQuota compromise. If the computer object is created by user objects that do not normally create computer objects, this may indicate a MachineAccountQuota compromise has occurred.

Unconstrained delegation

Computer objects can be configured for delegation, enabling them to impersonate user objects to access other services on behalf of the user object. There are two types of delegation that can be configured for computer objects: constrained delegation, which limits the impersonation rights to specific services, and unconstrained delegation, which allows a computer object to impersonate a user object to any service. When a computer object is configured for unconstrained delegation, and a user object authenticates to it, a copy of the user object's TGT is stored in the computer's Local Security Authority Subsystem Service (LSASS).

Computer objects configured for unconstrained delegation are targeted by malicious actors to escalate their privileges and move laterally in an environment. If malicious actors successfully compromise one of these computers and gain local administrator access, then they can extract the TGTs from the LSASS process for any user objects that had previously authenticated to the computer object. If a user object with domain administrator privileges had previously authenticated, the malicious actor can extract their TGT, reuse it for their own purposes, and escalate their privileges to that of a domain administrator in the environment. There are also several techniques malicious actors can use to force a user object to authenticate to a computer, thereby storing the user object's TGT in the LSASS process. This allows malicious actors to target any user object in the domain and gain control of it.

Unconstrained delegation and the Domain Controller Print Spooler service

Malicious actors can leverage unconstrained delegation and target the Print Spooler service on Domain Controllers. The Print Spooler service is targeted for misuse to force a system, such as a Domain Controller, to authenticate using its computer account to another system – in this case, with the computer object configured for unconstrained delegation. As a result, malicious actors can retrieve the TGT of the computer account of a Domain Controller. Malicious actors can then use this TGT to authenticate to the Domain Controller and gain administrative access. With administrative access to a Domain Controller, malicious actors can then execute other techniques, such as a **Dumping ntds.dit** and compromising the **Skeleton Key** (discussed below)**.**

Mitigating an unconstrained delegation compromise

The most effective mitigation for unconstrained delegation is to configure computer objects for constrained delegation. User objects can also be configured to not be delegated, meaning a copy of their TGT will not be stored on computers configured for unconstrained delegation.

The following security controls should be implemented to mitigate unconstrained delegation:

- **Ensure computer objects are not configured for unconstrained delegation.** If delegation is required for a computer object, use resource-based constrained delegation instead.
- **Ensure privileged user objects are configured as 'sensitive and cannot be delegated'.** This can be configured by using the 'Account is sensitive and cannot be delegated' option on the user object in Active Directory Users and Computers.
- **Ensure privileged user objects are members of the Protected Users security group.** Members of this security group cannot be delegated.
- **Disable the Print Spooler service on Domain Controllers.** This prevents the Print Spooler service from being used to coerce a Domain Controller into authenticating to another system.

Detecting an unconstrained delegation compromise

Computer objects configured for unconstrained delegation need to be monitored for signs of compromise, such as analysing unusual authentication events; for example, user objects that do not normally authenticate to the system configured with unconstrained delegation or authentication during unusual times of day. Organizations should also log PowerShell activity because malicious actors commonly use PowerShell to leverage unconstrained delegation, and unusual activity involving this tool may indicate an attempted unconstrained delegation compromise. If malicious actors compromise a computer object configured for unconstrained delegation successfully, and extract TGTs from the LSASS process undetected, it will be more difficult to detect the next stage of the compromise that uses the TGTs to impersonate other user objects in the domain. Therefore, organisations should be vigilant in their logging and analysis to detect an attempted unconstrained delegation compromise, as it will be more difficult to detect and mitigate subsequent malicious activities.

The events in **Table 5** should be centrally logged and analysed in a timely manner to identify an unconstrained delegation compromise.

Table 5. Events that detect an unconstrained delegation compromise

Event ID	Source	Description
4103	Computer objects configured for unconstrained delegation	This event is generated when PowerShell executes and logs pipeline execution details. Common malicious tools, such as Rubeus, use PowerShell to leverage unconstrained delegation. Analysing this event for unusual PowerShell executions may indicate an unconstrained delegation compromise has occurred.
4104	Computer objects configured for unconstrained delegation	This event is generated when PowerShell executes code to capture scripts and commands. Analysing this event for unusual PowerShell executions may indicate an unconstrained delegation compromise has occurred.
4624	Computer objects configured for unconstrained delegation Domain Controllers	This event is generated when malicious actors need to authenticate to a computer object configured for unconstrained delegation. This event should be analysed for unusual authentication activity, such as user objects that do not commonly log on and unusual logon times. Separately, this event should be analysed where the Source Network Address matches the internet protocol address of a computer configured for unconstrained delegation. This may indicate the computer object is being used to leverage unconstrained delegation to compromise a Domain Controller.
4688	Computer objects configured for unconstrained delegation	This event is generated when a new process is created, such as extracting TGTs from the LSASS process (this is commonly done using malicious tools). These events can be analysed to determine if the new process is malicious or not. Below are common commands executed by malicious actors to dump the LSASS process: ▪ *procdump.exe -accepteula -ma lsass.exe lsass.dmp* ▪ *.\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump <PID> C:\lsass.dmp full* ▪ *sekurlsa::minidump C:\lsass.DMP.*
4770	Domain Controllers	This event is generated when a TGT is renewed. By default, TGTs have a maximum lifetime of seven days; however, malicious actors may choose to renew a TGT to extend its lifetime. This may indicate a TGT has been compromised as a result of malicious actors leveraging unconstrained delegation.

Password in Group Policy Preferences (GPP) compromise

In 2014, a privilege escalation vulnerability (CVE-2014-1812) was discovered in Microsoft's GPP. This vulnerability allowed malicious actors to decrypt passwords distributed by GPP. Prior to a security patch being released for this vulnerability, GPP could be configured to distribute passwords across a domain and was commonly used to set passwords for local administrator accounts, map network shares and create scheduled tasks.

GPP passwords are known as cpasswords and are stored in the system volume (SYSVOL) directory, which exists on every Domain Controller and is readable by all users in a domain. The passwords are encrypted to protect them from unauthorised disclosure. However, around 2012, the private key used to encrypt cpasswords was made available online. With the private encryption key made public, cpasswords could easily be decrypted to reveal their cleartext passwords. In 2014, Microsoft released a security patch (2962486) to fix this vulnerability, which removed the functionality used by GPP to create cpasswords. However, this security patch did not remove cpasswords that had previously been created; these need to be removed manually. Unfortunately, this manual removal process has not been completed by many organisations, and cpasswords continue to persist.

Due to their susceptibility to discovery and decryption, cpasswords are frequently targeted by malicious actors shortly after gaining initial access to a domain. By accessing files with cpasswords that are accessible by all domain users, malicious actors can rapidly escalate their privileges from a standard user to that of a local administrator or a privileged domain user, typically without being detected.

Mitigating a password in GPP compromise

Microsoft has deprecated the use of cpasswords and now provides more secure methods to configure passwords – for instance, by using Microsoft's LAPS. As other methods exist for configuring passwords via Group Policy, cpasswords should no longer be used and any existing cpasswords should be removed from the SYSVOL directory.

The following security controls should be implemented to mitigate a Password in GPP compromise:

- **Remove all GPP passwords.** This eliminates the risk of a Password in GPP compromise.
- **Apply Microsoft's security patch 2962486 to remove the functionality to create cpasswords.** This security patch prevents the creation of new cpasswords. For more information on the security patch, see Microsoft's Security Bulletin MS14-025.

Detecting a password in GPP compromise

There are no effective techniques to detect malicious actors searching the SYSVOL directory for cpasswords because there are too many methods actors can use to search the SYSVOL directory, including with PowerShell, cmd.exe, and manual browsing using Windows Explorer. The SYSVOL directory is regularly read as part of group policy, further adding to the difficulty of identifying malicious activity.

Detecting a Password in GPP compromise can be achieved by implementing a canary GPP password. A GPP password can be placed in SYSVOL that belongs to a user object that should never be logged into. This user object is then monitored for any authentication events; if the user object is authenticated it may indicate its password has been retrieved from SYSVOL and a password in GPP compromise has occurred.

Active Directory Certificate Services (AD CS) compromise

AD CS implements Microsoft's public key infrastructure (PKI), providing various services including encryption, code signing and authentication. The AD CS Certificate Authority (CA) manages and issues public key certificates. The AD CS CA can be configured with multiple certificate templates, allowing user and computer objects to request certificates for various purposes. Depending on the configuration of the AD CS CA, a range of vulnerabilities can exist which can be exploited by malicious actors to escalate their privileges and move laterally.

A common certificate template vulnerability known as ESC1 allows any user object, regardless of their permissions, to request a certificate on behalf of any other user object (including privileged user objects) in the domain. After obtaining the certificate, it can then be used by malicious actors to authenticate as that user object, allowing for the impersonation of that user object and inheritance of its privileges. This vulnerable certificate template can be

requested using built-in tools, allowing malicious actors to live off the land to minimise the risk of detection. This certificate remains valid even if the user object specified in the certificate changes its password. The certificate is only invalidated when it expires or is revoked by the AD CS CA. Certificates like this can allow malicious actors to persist in an Active Directory domain because certificates are not always revoked as part of cyber security incident response activities, even when a compromise is detected. If the certificate template has the following configuration settings, then it is considered an ESC1 vulnerable certificate:

- Enrolment rights allowing user objects to request the certificate.
- Extended Key Usage (EKU) properties enabling user authentication.
- Subject Alternative Name (SAN) can be supplied.
- CA Certificate Manager approval is not required to approve the certificate request.

There are other types of vulnerable certificate templates and configurations that exist (i.e., ESC2-ESC13) that can be exploited by malicious actors to escalate their privileges and perform lateral movement. For further information, refer to:

- SpecterOps: Certified Pre-Owned: Abusing Active Directory Certificate Services
- Mandiant: Active Directory Certificate Services: Modern Attack Paths, Mitigations, and Hardening
- Microsoft: Securing AD CS: Microsoft Defender for Identity's Sensor Unveiled.

Mitigating AD CS compromise

To mitigate AD CS vulnerabilities and prevent compromises, the vulnerabilities first need to be identified. These vulnerabilities can be identified via several methods, including using the built-in Certificate Manager (certmgr.msc) and Certutil tools, as well as open source tools such as PSPKIAudit and Certify. Certificate Manager displays a warning against any certificate templates that allows a SAN to be supplied. Certutil provides a list of certificate templates that are available to the current user object and identifies any certificate templates that provide 'FullControl' or 'Write' permissions to user objects. PSPKIAudit provides a more comprehensive assessment of AD CS, and can identify if AD CS CAs have the ESC1-ESC8 vulnerabilities. Certify provides similar information to PSPKIAudit and can also request certificates to confirm that templates are vulnerable.

The following security controls should be implemented to mitigate an ESC1 AD CS compromise:

- **Remove the Enrolee Supplies Subject flag.** Do not allow users to provide their own SAN in the certificate signing request for templates configured for client authentication. Templates configured with the Enrolee Supplies Subject flag allow a user to provide their own SAN.
- **Restrict standard user object permissions on certificate templates.** Standard user objects should not have write permissions on certificate templates. User objects with write permissions may be able to change enrolment permissions or configure additional settings to make the certificate template vulnerable.
- **Remove vulnerable AD CS CA configurations.** Ensure that the CA is not configured with the EDITF_ATTRIBUTESUBJECTALTNAME2 flag. When configured, this allows a SAN to be provided on any certificate template.
- **Require CA Certificate Manager approval for certificate templates that allow the SAN to be supplied.** This ensures certificate templates that require CA certificate manager approval are not issued automatically when requested; instead, they must be approved using certificate manager before the certificate is issued.

- **Remove EKUs that enable user authentication.** This prevents malicious actors from exploiting the certificate to authenticate as other users.

- **Limit access to AD CS CA servers to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group and reduces the number of opportunities for malicious actors to gain access to CA servers.

- **Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.** AD CS servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.** This reduces the attack surface of AD CS CA servers as there are fewer services, ports and applications that may be vulnerable and used to compromise an AD CS CA server.

- **Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.** Backups of AD CS CA servers need to be afforded the same security as the actual AD CS CA servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as AD CS CA servers.

- **Centrally log and analyse AD CS CA server logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to a CA server, this activity should be identified as soon as possible to respond and limit the impact.

Conditions may exist within complex AD CS configurations that introduce pathways that allow malicious actors to target AD CS through more sophisticated techniques. For example, a certificate template may be configured to allow enrolment by a particular security group rather than all user objects. Malicious actors may target members of this security group to gain control of one user object and then continue to target certificate templates to escalate their privileges. Alternatively, some certificate templates may be configured to allow members of Domain Computers to enrol, which can be exploited if malicious actors are able to escalate their privileges to local administrator on any domain joined computer or if they are able to create a computer account within the domain.

Detecting an AD CS compromise

Detection of AD CS compromises requires logging and analysing events from multiple sources, including Domain Controllers and root and subordinate CAs. AD CS compromises may blend in with normal activity and further analysis of events may be required to identify misuse of SANs in certificate requests, the addition of user objects to certificate templates and the removal of security settings.

An administrator with access to the CA can audit issued certificates using the built-in certificate management tools on the CA. AD CS compromises may be detected by observing any certificates issued for Client Authentication that have a mismatch between the requester and the subject name. A mismatch may indicate that a malicious actor has requested a certificate for another user.

AD CS event auditing is not enabled by default. Follow these steps to configure audit logging for AD CS:

- **Enable 'Audit object access' for Certificate Services in Group Policy for AD CS CAs.** This can be found within the 'Advanced Audit Policy Configuration' within Security Settings.

- **Within the CA properties, the Auditing tab shows configurations of events to log. Enable all available options.**

The events in **Table 6** should be centrally logged and analysed in a timely manner to identify an AD CS compromise.

Table 6. Events that detect an AD CS compromise

Event ID	Source	Description
39	Domain Controllers	This event is generated when no strong certificate mappings can be found, and the certificate does not have a new Security Identifier (SID) extension that the Key Distribution Centre (KDC) could validate. This event is logged in the 'Kerberos-Key-Distribution-Center' log.
40	Domain Controllers	This event is generated when a certificate is supplied that was issued to the user before the user existed in Active Directory and no strong mapping is found.
41	Domain Controllers	This event is generated when a certificate is supplied where the SID contained in the new extension of the user's certificate does not match the user's SID, implying that the certificate was issued to another user. This may indicate that malicious actors are attempting to authenticate with a certificate with a SAN that does not match their current account.
1102	Root and subordinate CAs	This event is generated when the Security audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if an AD CS CA has been compromised.
4674	Domain Controllers	This event is generated when an attempt is made to perform privileged operations on a protected subsystem object after the object is already opened. This may be triggered when malicious actors attempt to change security descriptors of a certificate template. The 'Object Name' field lists the certificate template name as the value that can determine which template was changed.
4768	Domain Controllers	This event is generated when a TGT is requested. The 'PreAuthType' of '16' indicates that a certificate was used in the TGT request.
4886	Root and subordinate CAs	This event is generated when AD CS receives a certificate request. This may indicate if malicious actors attempted to elevate privileges by requesting an authentication certificate for a privileged user.
4887	Root and subordinate CAs	This event is generated when AD CS approves a certificate request and issues a certificate. This may be used to indicate when malicious actors successfully escalated privileges using AD CS.
4899	Root and subordinate CAs	This event is generated when a certificate template is updated. This may occur when malicious actors attempt to modify a certificate template to introduce additional features that may make it vulnerable to privilege escalation.
4900	Root and subordinate CAs	This event is generated when security settings on a Certificate Services template are updated. This may occur when the Access Control List on the template has been modified to potentially

introduce vulnerable conditions, such as modification of enrolment rights to a certificate template.

Golden Certificate

A Golden Certificate is a persistence technique that expands upon an AD CS compromise. If malicious actors obtain administrative access to a CA, they can extract a CA certificate and private key. Once obtained, these can be used to forge valid certificates for client authentication to impersonate any other user object in the domain. The CA certificate and private key from either a root or subordinate CA can be retrieved using built-in management tools designed for backup purposes, or by using open-source tools such as Mimikatz, Seatbelt and SharpDPAPI. Mimikatz can also be used to forge a new certificate, as can ForgeCert. Certificates created with these tools are signed by the private key of the extracted CA certificate, allowing them to be used within the domain. Certificates remain valid until they are revoked, which if not done periodically may result in perpetually valid certificates that enables the malicious actor to persist on the network.

Mitigating a Golden Certificate

Mitigating a Golden Certificate requires securing both root and subordinate CAs. Due to their critical role, CAs need to be afforded the same security as other critical servers, such as Domain Controllers. This includes minimising the number of user objects with privileged access to CAs, not using CAs for any other purposes except for AD CS and monitoring CAs for signs of compromise.

The following security controls should be implemented to mitigate a Golden Certificate:

- **Use MFA to authenticate privileged users of systems.** MFA for privileged users can hinder malicious actors from gaining access to a CA using stolen credentials, thus preventing the extraction of a CA certificate and private key.
- **Implement application control on AD CS CAs.** An effective application control configuration on CAs prevents the execution of malicious executables such as Mimikatz.
- **Use a Hardware Security Module (HSM) to protect key material for AD CS CAs.** Protect private keys by using a HSM with CAs. If a HSM is used, the private key for CAs cannot be backed up and exfiltrated by malicious actors.
- **Limit access to AD CS CAs to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group and reduces the number of opportunities for malicious actors to gain access to a CA.
- **Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.** AD CS servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.
- **Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.** This reduces the attack surface of AD CS CA servers as there are fewer services, ports and applications that may be vulnerable and used to compromise an AD CS CA server.
- **Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.** Backups of AD CS CA servers need to be afforded the same security as the actual AD CS CA servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as AD CS CA servers.

- **Centrally log and analyse AD CS CA logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to a CA, this activity should be identified as soon as possible to respond and limit the impact.

Detecting a Golden Certificate

A Golden Certificate is difficult to detect as it requires detection of the initial backup and exfiltration of a CA certificate and private key. AD CS CAs can be configured to enable audit logging of some events; however, visibility of CA certificate backups is still difficult.

AD CS CA event auditing is not enabled by default. To configure audit logging for AD CS CAs:

- **Enable 'Audit object access' for Certificate Services in Group Policy for CAs.** This can be found within the 'Advanced Audit Policy Configuration' within Security Settings.
- **Enable 'Backup and restore the CA database' as events to audit in the Auditing tab within the properties for CAs.**

Event 4876 is triggered when a complete backup of the CA database is requested. This only occurs if the 'Certificate database and certificate database log' option is selected in the backup wizard. If only the 'Private key and CA certificate' option is selected, this event is not generated. As such, this cannot be relied upon to detect all backup attempts.

Windows CAPI2 logs can capture certificate export events. This log would need to be enabled within Event Viewer on CAs. When enabled, any backup of a CA certificate and private key generates event 70, which is labelled as 'Acquire Certificate Private Key'.

The events in **Table 7** should be centrally logged and analysed in a timely manner to identify a Golden Certificate.

Table 7. Events that detect a Golden Certificate

Event ID	Source	Description
70	CAPI2 logs on the root and subordinate CAs	This event is generated when a certificate is exported. This event should be filtered to check that the 'subjectName' field matches that of a CA certificate.
1102	Root and subordinate CAs	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if an AD CS CA has been compromised.
4103	Root and subordinate CAs	This event is generated when PowerShell executes and logs pipeline execution details. Common tools such as Certutil and Mimikatz use PowerShell. Analysing this event for PowerShell execution relating to these tools may indicate a Golden Certificate.
4104	Root and subordinate CAs	This event is generated when PowerShell executes code to capture scripts and commands. Common tools such as Certutil and Mimikatz use PowerShell. Analysing this event for PowerShell

		execution relating to these tools may indicate a Golden Certificate.
4876	Root and subordinate CAs	This event is triggered when a backup of the CA database is started. This does not return any logs for exporting the private key, but may be an indicator of other potentially suspicious activity occurring on a CA.

DCSync

DCSync replicates information from Active Directory, including password hashes. This requires 'Replicating Directory Changes', 'Replicating Directory Changes All' or 'Replicating Directory Changes in Filtered Set' privileges – or either 'GenericAll' or 'AllExtendedRights' permissions – on the domain root object in Active Directory. By default, these permissions and rights are granted to members of the Enterprise Admins and Domain Admins security groups, as well as the Administrators security group on Domain Controllers.

By gaining access to a security group or a user object that has the above privileges or permissions, malicious actors can then execute a DCSync (see **Figure 4**). In doing so, malicious actors may choose to retrieve all user and computer object password hashes or target individual objects, such as the KRBTGT user object, which can be used for other compromise techniques, such as a This technique can be detected with the assistance of Active Directory canaries. For more information, see section Detecting Active Directory Compromises with Canaries.

Dumping ntds.dit

The New Technology Directory Services Directory Information Tree (ntds.dit) is the AD DS database file which stores information about all objects in the domain. This information includes the password hashes for user and computer objects. Due to this, it is frequently targeted by malicious actors when compromising AD DS. A copy of the ntds.dit file is stored on every Domain Controller (except read-only Domain Controllers) in the domain. Any user object that can log on to Domain Controllers, such as members of the Domain Admins security group, can access the ntds.dit file.

The ntds.dit file is constantly updated as changes are made in the domain. For this reason, the file is locked and unable to be copied using standard techniques. To bypass the file locking mechanism and make a copy of the ntds.dit file, malicious actors can use native tools, such as the Volume Shadow Copy Service and Ntdsutil. Some of the information stored in the ntds.dit file is encrypted. To decrypt all of this information, malicious actors need to retrieve the SYSTEM hive from the registry of the same Domain Controller where they obtained the ntds.dit file. The SYSTEM hive is retrieved using a single command executed in PowerShell or cmd.exe.

Backups of Domain Controllers

Malicious actors may target backups of Domain Controllers to try to retrieve a copy of Active Directory's database file (ntds.dit); backups may be an easier way for a malicious actor to access the ntds.dit file if the backups of Domain Controllers are not as secure as the Domain Controllers themselves. For example, backups of Domain Controllers may be stored on Windows file shares as part of the backup process, and if these backups are not removed, malicious actors may be able to retrieve them simply by accessing the file share.

Malicious actors that escalate their privileges and gain access to a Domain Controller commonly attempt to access and exfiltrate the ntds.dit file and the SYSTEM hive. After exfiltrating these files to their own system, they can decrypt the ntds.dit file and attempt to crack the password hashes for every user and computer object. For any password hashes that are successfully cracked to reveal the cleartext password, malicious actors can be authenticated as these user and computer objects and gain control of them. Additionally, malicious actors attempting to persist in a domain may

continue to copy and exfiltrate the ntds.dit file on a regular basis. This way, they can retrieve password hashes that have changed, such as when a user object changes its password, and retain their access.

The unauthorised copying and exfiltration of the ntds.dit file signifies the complete compromise of an Active Directory domain. The loss of all sensitive information from Active Directory – including user and computer object password hashes, the KRBTGT password hash, trusted domain object (TDO) password hash, and the data protection API (DPAPI) backup key – is significant. Recovering from the loss of the ntds.dit file requires resetting all user and computer object passwords, as well as the TDO password, in a coordinated manner. Full recovery may require building a new Active Directory domain with new user and computer objects and destroying the compromised domain. For many organisations, these recovery activities constitute a significant, costly and disruptive effort.

Active Directory Data Protection Application Programming Interface (DPAPI) backup keys

The DPAPI backup keys in Active Directory are considered one of the most highly sensitive pieces of information in the entire Active Directory domain. These DPAPI backup keys are used to secure other data, such as user object DPAPI keys, which are used to encrypt sensitive information such as passwords.

Every user object in a domain has their own DPAPI key that is encrypted with their password. Each user object DPAPI key has a copy encrypted with the Active Directory backup DPAPI key. This DPAPI key copy exists for recovery purposes – if a user object resets their password, then the previously encrypted data cannot be decrypted. The DPAPI key copy is used to decrypt the data and is then encrypted using the user object's new DPAPI key.

Due to the role and function of Active Directory DPAPI backup keys, these keys are immutable and cannot be reset. If malicious actors obtain these keys, they can use them indefinitely to decrypt user objects' sensitive data. Microsoft's only recommended and supported recovery option is to create a new domain and migrate all users to the new domain. For more information, see Microsoft's DPAPI Backup Keys on Active Directory Domain Controllers.

Mitigating dumping ntds.dit

Mitigating techniques targeting the ntds.dit file begins with hardening Domain Controllers by restricting privileged access pathways, disabling unused services and ports, not installing additional features or applications, using antivirus and endpoint detection and response solutions, and monitoring for signs of compromise. These mitigations reduce the attack surface of Domain Controllers and increase the likelihood of detecting malicious activity.

The following security controls should be implemented to mitigate dumping ntds.dit:

- **Limit access to Domain Controllers to only privileged users that require access.** This reduces the number of opportunities for malicious actors to gain access to Domain Controllers.
- **Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.** Domain Controllers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.
- **Encrypt and securely store backups of Domain Controllers and limit access to only Backup Administrators.** Backups of Domain Controllers need to be afforded the same security as the actual Domain Controllers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as Domain Controllers.
- **Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.** This reduces the attack surface of Domain Controllers as there are fewer services, ports and applications that may be vulnerable and used to compromise a Domain Controller.

- **Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.** Domain Controller logs provide a rich source of information that is important for investigating potentially malicious activity on Domain Controllers and in the domain.

- **Disable the Print Spooler service on Domain Controllers.** For example, malicious actors have targeted the Print Spooler service on Domain Controllers as a technique to authenticate to a system they control to collect the Domain Controllers computer object password hash or TGT. Malicious actors can then use this to authenticate to the Domain Controller they coerced and gain administrative access.

- **Disable the Server Message Block (SMB) version 1 protocol on Domain Controllers.** There are multiple Active Directory compromises that leverage weaknesses in the SMBv1 protocol to gain access to systems, including Domain Controllers. Disabling SMBv1 on Domain Controllers and on all systems in a domain mitigates compromises that leverage the SMBv1 protocol.

Detecting dumping ntds.dit

Tools such as Volume Shadow Copy Service and Ntdsutil are commonly used by malicious actors to dump the ntds.dit file and the SYSTEM hive from Domain Controllers. These tools can be executed using PowerShell. If PowerShell logging is enabled, these tool names and their parameters are recorded, which can help identify if an attempt was made to compromise the ntds.dit file. Additionally, monitoring for signs of compromise by analysing events for unusual authentication events, such as objects that do not normally authenticate or authentication during unusual times of the day, can assist in identifying malicious activity.

The events in **Table 9** should be centrally logged and analysed in a timely manner to identify dumping ntds.dit.

Table 9. Events that detect dumping ntds.dit

Event ID	Source	Description
1102	Domain Controllers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Domain Controller has been compromised.
4103	Domain Controllers	This event is generated when PowerShell executes and logs pipeline execution details. Malicious actors commonly leverage PowerShell in their compromises. Analysing this event for PowerShell execution relating to the ntds.dit file may indicate dumping of the ntds.dit file.
4104	Domain Controllers	This event is generated when PowerShell executes code to capture scripts and commands. Malicious actors commonly leverage PowerShell in their compromises. Analysing this event for PowerShell execution relating to the ntds.dit file may indicate dumping of the ntds.dit file.
4656	Domain Controllers	This event is generated when a handle to an object has been requested, such as a file: for example, when malicious actors attempt to access the ntds.dit file in any way (e.g., read, write or delete). If the 'Object Name' value in the event matches the ntds.dit file, this may indicate the ntds.dit file has been compromised.

4663	Domain Controllers	This event is generated when the System Access Control List (SACL) is enabled for the ntds.dit file and an attempt is made to access, read, write, or modify an object, such as a file. If the 'Object Name' value in the event matches the ntds.dit file, this may indicate the ntds.dit file has been compromised.
4688	Domain Controllers	This event is generated when a new process has been created. This event provides context of the commands and parameters that are executed when a new process is created. Malicious actors are likely to create a new process when dumping the ntds.dit file, such as via PowerShell, Volume Shadow Copy Service or Ntdsutil.
8222	Domain Controllers	This event is generated when a shadow copy is made. Making a shadow copy of the ntds.dit file is a common way to bypass file lock restrictions. This event can be analysed to determine if the shadow copy was legitimate or not.

Golden Ticket. After retrieving the password hashes from a Domain Controller, malicious actors can either attempt to crack them to reveal the cleartext passwords or use a password hash in a **Pass-the-Hash** .

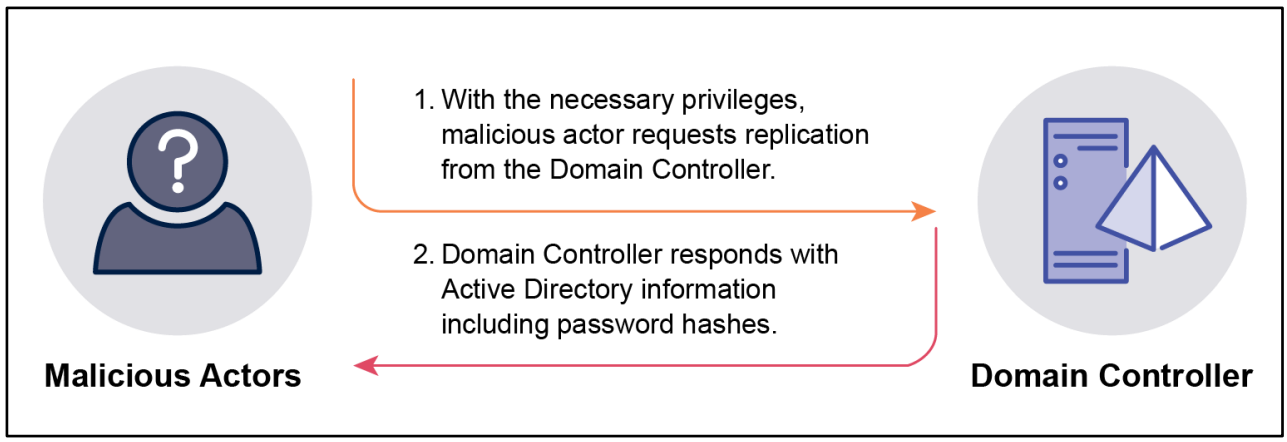

Figure 4: Overview of DCSync

Pass-the-Hash (PtH)

PtH is a technique that exploits a weakness in NTLM version 1 and 2 protocols. Active Directory stores passwords as NTLM hashes for every user and computer object and accepts them as valid authentication tokens. Malicious actors can use them to authenticate to systems and services that use Active Directory, without the need to crack them to reveal their cleartext passwords. If malicious actors are able to obtain NTLM password hashes, it negates the security benefits of long, unpredictable and unique passwords.

DCSync retrieves password hashes from a Domain Controller for all user and computer objects. If the NTLM protocol is still enabled, malicious actors can retrieve NTLM password hashes. These NTLM password hashes can then be immediately used to authenticate as user and computer objects as there is no need to crack the password hashes to reveal the cleartext passwords.

The successful execution of DCSync by malicious actors signifies the complete compromise of an Active Directory domain. The loss of all user and computer object password hashes, and the KRBTGT password hash, can be difficult to recover from as it requires resetting all user and computer object password hashes in the domain, as well as resetting

the KRBTGT password twice in a coordinated manner. For many organisations, these recovery activities are significant, costly and disruptive.

DCSync and reversible encryption setting

User objects in AD DS can be configured to store their password using reversible encryption. This enables AD DS to store user object passwords in cleartext and is primarily used to support legacy applications that still require cleartext passwords. Malicious actors can exploit this configuration setting prior to executing DCSync. For example, malicious actors enabling the 'store passwords using reversible encryption' setting on user objects they want to control. Subsequently, the next time the password is changed for these user objects, AD DS will store a copy of the cleartext password. Malicious actors then perform DCSync targeting user objects with this setting enabled to retrieve their cleartext passwords. This technique negates password complexity requirements and bypasses the requirement for password cracking as the cleartext password is obtained directly from AD DS.

Mitigating DCSync

For the proper functioning of Active Directory, specific user and computer objects are configured with the privileges or permissions that make it possible for malicious actors to execute DCSync. Therefore, it is not possible to completely eliminate the risk of DCSync. However, it is possible to reduce the likelihood of DCSync by minimising the number of user and computer objects with the necessary privileges or permissions that allow malicious actors to perform DCSync. Doing so focusses protection measures on these user and computer objects, preventing their compromise. User and computer objects with these privileges and permissions are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

The following security controls should be implemented to mitigate DCSync:

- **Minimise the number of user objects with DCSync permissions.** By default, members of the Enterprise Admins, Domain Admins and Administrators security group have permissions to perform DCSync. Therefore, the number of user objects in these security groups should be minimised and direct assignment of these permissions to other user objects should be limited.

- **Ensure user objects that are configured with a SPN do not have DCSync permissions.** This is to reduce the risk of a user object with a SPN being compromised as the result of a successful Kerberoasting and then being used by malicious actors to execute DCSync.

- **Ensure user objects with DCSync permissions cannot log on to unprivileged operating environments.** Lower privileged operating environments, such as those used by internet-facing systems and user workstations, are often exploited by malicious actors to gain initial access and to pivot to higher privileged operating environments. Preventing privileged user objects from logging into these lower privileged operating environments reduces the risk of these user objects being compromised and subsequently used to pivot to higher privileged operating environments. This is a key protection in the tiered administrative model.

- **Review user objects with DCSync permissions every 12 months to determine if these permissions are still required.** Regularly reviewing permissions, and removing them when no longer required, reduces the attack surface that malicious actors can target.

- **Disable the NTLMv1 protocol.** This prevents NTLM password hashes from being retrieved by DCSync and then being either cracked or used as part of PtH.

- **Ensure LAN Manager (LM) password hashes are not used.** This can be enforced by requiring and updating passwords to be a minimum of 15-characters. LM only supports passwords up to 14-characters in length and

passwords that are 15-characters or more will not be stored as a LM hash. LM password hashes can be quickly cracked to reveal cleartext passwords and are not considered secure.

Detecting DCSync

Domain Controllers routinely replicate changes to each other for each Domain Controller to maintain an up-to-date record of all objects and their assigned properties within a domain. When this replication is triggered, an event is generated. This same event is generated when DCSync occurs, but the user object name is the hostname of a Domain Controller rather than the user object name. If this event is generated by anything other than a Domain Controller, it may be indicative of DCSync. **Note:** Sophisticated malicious actors may be able to impersonate a Domain Controller so the account name appears legitimate to evade detection.

The event in **Table 8** should be centrally logged and analysed in a timely manner to identify a DCSync.

Table 8. Event that detects a DCSync

Event ID	Source	Description
4662	Domain Controllers	This event is generated when an operation is performed on an object. When DCSync is executed, this event is generated on the targeted Domain Controller, and the event properties contain the following values: ▪ 1131f6ad-9c07-11d1-f79f-00c04fc2dcd2 (DS-Replication-Get-Changes-All) ▪ 19195a5b-6da0-11d0-afd3-00c04fd930c9 (Domain-DNS class WRITE_DAC) ▪ 89e95b76-444d-4c62-991a-0facbeda640c (DS-Replication-Get-Changes-In-Filtered-Set) If this event is not generated by a Domain Controller, it may indicate a DCSync has occurred.

This technique can be detected with the assistance of Active Directory canaries. For more information, see section Detecting Active Directory Compromises with Canaries.

Dumping ntds.dit

The New Technology Directory Services Directory Information Tree (ntds.dit) is the AD DS database file which stores information about all objects in the domain. This information includes the password hashes for user and computer objects. Due to this, it is frequently targeted by malicious actors when compromising AD DS. A copy of the ntds.dit file is stored on every Domain Controller (except read-only Domain Controllers) in the domain. Any user object that can log on to Domain Controllers, such as members of the Domain Admins security group, can access the ntds.dit file.

The ntds.dit file is constantly updated as changes are made in the domain. For this reason, the file is locked and unable to be copied using standard techniques. To bypass the file locking mechanism and make a copy of the ntds.dit file, malicious actors can use native tools, such as the Volume Shadow Copy Service and Ntdsutil. Some of the information stored in the ntds.dit file is encrypted. To decrypt all of this information, malicious actors need to retrieve

the SYSTEM hive from the registry of the same Domain Controller where they obtained the ntds.dit file. The SYSTEM hive is retrieved using a single command executed in PowerShell or cmd.exe.

Backups of Domain Controllers

Malicious actors may target backups of Domain Controllers to try to retrieve a copy of Active Directory's database file (ntds.dit); backups may be an easier way for a malicious actor to access the ntds.dit file if the backups of Domain Controllers are not as secure as the Domain Controllers themselves. For example, backups of Domain Controllers may be stored on Windows file shares as part of the backup process, and if these backups are not removed, malicious actors may be able to retrieve them simply by accessing the file share.

Malicious actors that escalate their privileges and gain access to a Domain Controller commonly attempt to access and exfiltrate the ntds.dit file and the SYSTEM hive. After exfiltrating these files to their own system, they can decrypt the ntds.dit file and attempt to crack the password hashes for every user and computer object. For any password hashes that are successfully cracked to reveal the cleartext password, malicious actors can be authenticated as these user and computer objects and gain control of them. Additionally, malicious actors attempting to persist in a domain may continue to copy and exfiltrate the ntds.dit file on a regular basis. This way, they can retrieve password hashes that have changed, such as when a user object changes its password, and retain their access.

The unauthorised copying and exfiltration of the ntds.dit file signifies the complete compromise of an Active Directory domain. The loss of all sensitive information from Active Directory – including user and computer object password hashes, the KRBTGT password hash, trusted domain object (TDO) password hash, and the data protection API (DPAPI) backup key – is significant. Recovering from the loss of the ntds.dit file requires resetting all user and computer object passwords, as well as the TDO password, in a coordinated manner. Full recovery may require building a new Active Directory domain with new user and computer objects and destroying the compromised domain. For many organisations, these recovery activities constitute a significant, costly and disruptive effort.

Active Directory Data Protection Application Programming Interface (DPAPI) backup keys

The DPAPI backup keys in Active Directory are considered one of the most highly sensitive pieces of information in the entire Active Directory domain. These DPAPI backup keys are used to secure other data, such as user object DPAPI keys, which are used to encrypt sensitive information such as passwords.

Every user object in a domain has their own DPAPI key that is encrypted with their password. Each user object DPAPI key has a copy encrypted with the Active Directory backup DPAPI key. This DPAPI key copy exists for recovery purposes – if a user object resets their password, then the previously encrypted data cannot be decrypted. The DPAPI key copy is used to decrypt the data and is then encrypted using the user object's new DPAPI key.

Due to the role and function of Active Directory DPAPI backup keys, these keys are immutable and cannot be reset. If malicious actors obtain these keys, they can use them indefinitely to decrypt user objects' sensitive data. Microsoft's only recommended and supported recovery option is to create a new domain and migrate all users to the new domain. For more information, see Microsoft's DPAPI Backup Keys on Active Directory Domain Controllers.

Mitigating dumping ntds.dit

Mitigating techniques targeting the ntds.dit file begins with hardening Domain Controllers by restricting privileged access pathways, disabling unused services and ports, not installing additional features or applications, using antivirus and endpoint detection and response solutions, and monitoring for signs of compromise. These mitigations reduce the attack surface of Domain Controllers and increase the likelihood of detecting malicious activity.

The following security controls should be implemented to mitigate dumping ntds.dit:

- **Limit access to Domain Controllers to only privileged users that require access.** This reduces the number of opportunities for malicious actors to gain access to Domain Controllers.

- **Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.** Domain Controllers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Encrypt and securely store backups of Domain Controllers and limit access to only Backup Administrators.** Backups of Domain Controllers need to be afforded the same security as the actual Domain Controllers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as Domain Controllers.

- **Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.** This reduces the attack surface of Domain Controllers as there are fewer services, ports and applications that may be vulnerable and used to compromise a Domain Controller.

- **Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.** Domain Controller logs provide a rich source of information that is important for investigating potentially malicious activity on Domain Controllers and in the domain.

- **Disable the Print Spooler service on Domain Controllers.** For example, malicious actors have targeted the Print Spooler service on Domain Controllers as a technique to authenticate to a system they control to collect the Domain Controllers computer object password hash or TGT. Malicious actors can then use this to authenticate to the Domain Controller they coerced and gain administrative access.

- **Disable the Server Message Block (SMB) version 1 protocol on Domain Controllers.** There are multiple Active Directory compromises that leverage weaknesses in the SMBv1 protocol to gain access to systems, including Domain Controllers. Disabling SMBv1 on Domain Controllers and on all systems in a domain mitigates compromises that leverage the SMBv1 protocol.

Detecting dumping ntds.dit

Tools such as Volume Shadow Copy Service and Ntdsutil are commonly used by malicious actors to dump the ntds.dit file and the SYSTEM hive from Domain Controllers. These tools can be executed using PowerShell. If PowerShell logging is enabled, these tool names and their parameters are recorded, which can help identify if an attempt was made to compromise the ntds.dit file. Additionally, monitoring for signs of compromise by analysing events for unusual authentication events, such as objects that do not normally authenticate or authentication during unusual times of the day, can assist in identifying malicious activity.

The events in **Table 9** should be centrally logged and analysed in a timely manner to identify dumping ntds.dit.

Table 9. Events that detect dumping ntds.dit

Event ID	Source	Description
1102	Domain Controllers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Domain Controller has been compromised.
4103	Domain Controllers	This event is generated when PowerShell executes and logs pipeline execution details. Malicious actors commonly leverage PowerShell in

		their compromises. Analysing this event for PowerShell execution relating to the ntds.dit file may indicate dumping of the ntds.dit file.
4104	Domain Controllers	This event is generated when PowerShell executes code to capture scripts and commands. Malicious actors commonly leverage PowerShell in their compromises. Analysing this event for PowerShell execution relating to the ntds.dit file may indicate dumping of the ntds.dit file.
4656	Domain Controllers	This event is generated when a handle to an object has been requested, such as a file: for example, when malicious actors attempt to access the ntds.dit file in any way (e.g., read, write or delete). If the 'Object Name' value in the event matches the ntds.dit file, this may indicate the ntds.dit file has been compromised.
4663	Domain Controllers	This event is generated when the System Access Control List (SACL) is enabled for the ntds.dit file and an attempt is made to access, read, write, or modify an object, such as a file. If the 'Object Name' value in the event matches the ntds.dit file, this may indicate the ntds.dit file has been compromised.
4688	Domain Controllers	This event is generated when a new process has been created. This event provides context of the commands and parameters that are executed when a new process is created. Malicious actors are likely to create a new process when dumping the ntds.dit file, such as via PowerShell, Volume Shadow Copy Service or Ntdsutil.
8222	Domain Controllers	This event is generated when a shadow copy is made. Making a shadow copy of the ntds.dit file is a common way to bypass file lock restrictions. This event can be analysed to determine if the shadow copy was legitimate or not.

Golden Ticket

A Golden Ticket misuses the KRBTGT user object's password hash to forge TGTs. With the KRBTGT user object's password hash, malicious actors can forge their own TGTs to impersonate any user object and subsequently request a TGS ticket from a Domain Controller. The TGS ticket can then be used to access other Active Directory systems as the impersonated user object, including any privileges they have (see **Figure 5**). This enables privilege escalation and lateral movement while minimising the risk of detection.

The KRBTGT user object's password hash is commonly obtained via a This type of compromise can be detected with the assistance of Active Directory canaries. For more information, see section Detecting Active Directory Compromises with Canaries.

Password spraying

Password spraying attempts to authenticate to multiple user objects using either a single password or multiple passwords until they successfully authenticate to a user object. These passwords can come from public password wordlists or be derived from the target environment for a higher likelihood of success. For example, malicious actors may identify passwords being reused in the target environment and use these in password spraying to identify if they belong to any user objects. To minimise authentication attempts and the risk of detection, malicious actors can

retrieve a list of usernames from Active Directory and attempt to authenticate to each one using a single password. This technique is particularly effective against organisations that reuse passwords. If malicious actors compromise a user object via password spraying, then they control the user object and inherit the user object's access and privileges.

File shares and Active Directory credentials

The authoring agencies have observed malicious actors scanning file shares as part of their efforts to locate insecurely stored secrets, such as credentials for Active Directory user objects. Multiple tools, such as SMBMap and Snaffler, can identify file shares and scan them for credentials (including cleartext passwords), sensitive information, application programming interface (API) keys, digital certificates, standalone password managers, and backups. These tools are commonly used after gaining initial access to an Active Directory domain to try and locate credentials for privileged user objects.

If malicious actors locate credentials, they are likely to use them to escalate their privileges and move laterally. This reduces the likelihood of detecting malicious actors, as they are able to gain control of other user objects without having to attempt riskier techniques, such as **Kerberoasting** and **Password Spraying**.

To reduce the likelihood of malicious actors locating credentials in file shares and using them in password spraying, organisations should use an enterprise-grade password management solution (where possible) to secure store their sensitive information, including credentials for Active Directory user objects.

Additionally, organisations should periodically conduct their own scans to identify any sensitive information that is insecurely stored on file shares.

Many organisations enforce an account lockout threshold policy to lock user objects after a certain number of failed authentication attempts. This is effective at preventing malicious actors from attempting too many different passwords. However, malicious actors can still perform password spraying up to the account lockout threshold without locking out user objects. Different tools exist to perform password spraying that identify the lockout threshold to ensure the threshold is not exceeded, including DomainPasswordSpray and Spray. These tools may also be configured to perform password spraying over a certain time period to limit the number of authentication attempts occurring at any given time and minimise the risk of detection.

While MFA can be effective at mitigating password spraying by malicious actors attempting to gain initial access, it is largely ineffective at mitigating password spraying if malicious actors have already gained initial access. This is because the malicious actors can attempt to authenticate as any user object in the domain directly to a Domain Controller via the New Technology Local Area Network (LAN) Manager (NTLM) protocol, which does not support MFA. To reduce the risk of NTLM-based compromises, disable the NTLM protocol wherever possible. When this is not possible, enable LDAP channel binding, extended protection authentication, and Server Message Block (SMB) signing.

The built-in Administrator account and account lockout threshold

In every Active Directory domain, there is a built-in Administrator account made during the creation of the domain. This account is a default member of the Domain Admins and Administrators security groups, and if the domain is the forest root domain, it is also a member of the Enterprise Admins security group. These security groups make this user object highly privileged as it has administrator access on all objects in the domain.

The account lockout threshold policy that locks accounts after a certain number of failed authentication attempts does not apply to the built-in Administrator account. Even if multiple failed authentication attempts occur and the account reports it is locked out, it can still be authenticated to if the correct password is provided. This automatically removes the locked status and resets the bad password count to zero.

The inability for the built-in Administrator account to be locked out makes it an attractive target for password spraying. Specifically, malicious actors can continually spray this account with multiple passwords knowing that this account will not be locked out, and if the correct password is found, be able to login to the account successfully regardless of how many prior failed authentication attempts were made.

To reduce the risk of password spraying that targets the built-in Administrator user object, set a long (30-character minimum), unique, unpredictable and managed password. Additionally, this account should only be used as an emergency break glass account, and authentication events associated with this account should be monitored for signs of malicious activity such as a Password Spray or other unauthorised access. Organisations can also implement additional protections, including configuring the user object as sensitive to ensure it cannot be delegated and restrict where the user object can be used.

Mitigating password spraying

The following security controls should be implemented to mitigate password spraying:

- **Create passwords for local administrator accounts, service accounts, and break glass accounts that are long (30-character minimum), unique, unpredictable and managed.** Microsoft's Local Administrator Password Solution (LAPS) can be used to achieve this for local administrator accounts. Using strong passwords reduces the likelihood of successful password spraying.

- **Create passwords used for single-factor authentication that consist of at least four random words with a total minimum length of 15-characters** to reduce the likelihood of a successful password spraying.

- **Lock out user objects, except for break glass accounts, after a maximum of five failed logon attempts.** Enforcing an account lock threshold after five failed authentication attempts reduces the number of possible attempts in password spraying.

- **Ensure passwords created for user objects are randomly generated**, such as when a user object is created, or a user requests a password reset. Malicious actors will try to identify reused passwords and use these in password spraying to increase the likelihood of success.

- **Configure the built-in 'Administrator' domain account as sensitive to ensure it cannot be delegated.**

- **Scan networks at least monthly to identify any credentials that are being stored in the clear.** Malicious actors scan networks for cleartext credentials to use in password spraying. Locating and removing these cleartext credentials proactively mitigates this risk.

- **Disable the NTLM protocol.** The NTLM protocol does not support MFA and can be misused by malicious actors to bypass MFA requirements.

Detecting password spraying

Password spraying typically generates an event for each failed authentication attempt. Depending on the number of user objects being targeted, this could result in a large number of events being generated. To effectively detect password spraying, alerts should be generated when there are numerous failed authentication events that occur in a short period of time.

Popular password spraying tools, such as DomainPasswordSpray and CrackMapExec, commonly attempt to authenticate using the SMB protocol. Malicious actors may use another protocol, such as the Lightweight Directory Access Protocol (LDAP), which generates a different event in an attempt to avoid detection. It is important to collect both events to effectively monitor for password spraying.

The events in **Table 3** should be centrally logged and analysed in a timely manner to identify password spraying.

Table 3. Events that detect password spraying

Event ID	Source	Description
2889	Domain Controllers	This event is generated when a computer object tries to make an unsigned LDAP bind. Malicious actors using the LDAP protocol to conduct password spraying generate this event as each password attempt makes an unsigned LDAP bind. If numerous 2889 events occur in a short timeframe, this may indicate password spraying occurred using the LDAP protocol.
4624	Domain Controllers	This event is generated when an object logs on successfully, such as to a user object. If this event occurs near-simultaneously with 4625 events, this can indicate a user object was successfully logged on to as a result of password spraying.
4625	Domain Controllers	This event is generated when an object fails to log on via the SMB protocol. Common password spraying tools default to attempting authentication using the SMB protocol. If numerous 4625 events occur in a short timeframe, this may indicate password spraying occurred using the SMB protocol. Other protocols, such as LDAP, can also be used for password spraying. Malicious actors may choose to use a different protocol to avoid detection. The 'badPasswordTime' user object attribute in Active Directory can be queried to identify the date and time of the last failed authentication attempt. If multiple user objects share the same date and time, or nearly the same date and time, this may indicate password spraying occurred.
4648	Source of Password Spraying, such as a domain joined workstation or server	This event is generated when a logon is attempted using explicit credentials. If password spraying is executed on a domain joined system, this event is generated for each authentication attempt. If numerous 4648 events exist with different usernames in a short timeframe, this can indicate password spraying was executed on the system. Note, if malicious actors have established a tunnel from their infrastructure, they may be able to execute password spraying using their own systems, if this is the case, this event will not be generated.
4740	Domain Controllers	This event is generated when a user object is locked out. Password spraying can cause user objects to be locked out due to the number of failed authentication attempts. If multiple user objects are locked out in a short period of time, this may indicate password spraying occurred. Many password spraying tools check the domain's lockout policy and the number of failed authentication attempts for user objects to avoid lockout as a means to avoid detection.

4771	Domain Controllers	This event is generated when Kerberos pre-authentication fails. In an attempt to evade detection, malicious actors may use the LDAP protocol to execute password spraying. In this case, event 4771 generates the 'Failure Code' property of '0x18'. This value means the incorrect password is the cause for the event. The 'badPasswordTime' user object attribute in Active Directory can be queried to identify the date and time of the last failed authentication attempt. If multiple user objects share the same date and time, or nearly the same date and time, this may indicate password spraying occurred.

MachineAccountQuota compromise

A MachineAccountQuota compromise exploits the default Active Directory setting that allows user objects to create up to ten computer objects in the domain via the 'ms-DS-MachineAccountQuota' attribute. These computer objects are automatically added to the Domain Computers security group and inherit the group's privileges. Most malicious actors, to minimise the risk of detection, set the computer object's name to comply with any domain-specific naming conventions to appear similar to other computer objects. For example, if the Domain Computers security group is overly privileged, is a member of higher privileged security groups, or has privileges to other Active Directory objects, malicious actors can exploit this to escalate their privileges. Malicious actors can achieve this by creating their own computer object, authenticating as this computer object, and inheriting its privileges. This computer object can then be used to interact with the domain, similar to user objects. Computer objects can also access and interact with other systems and services in the domain (see Figure 3).

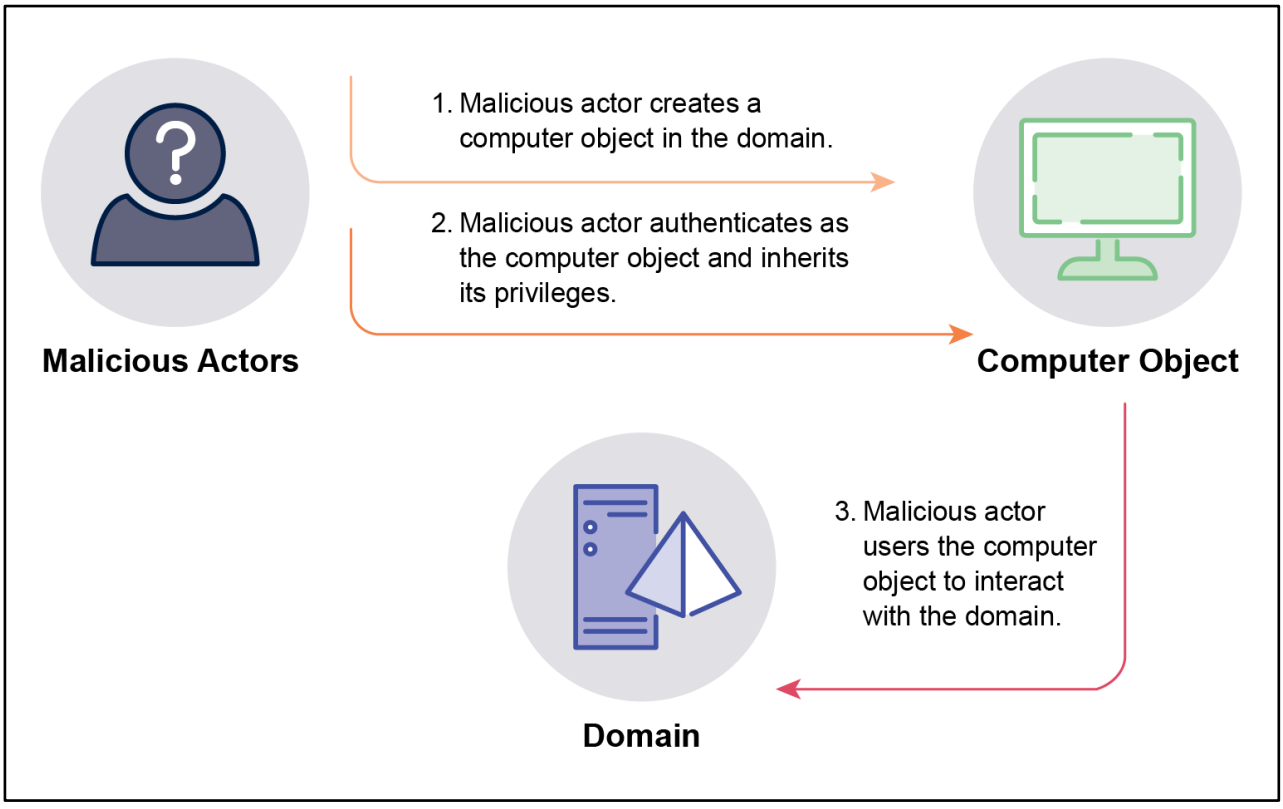

Figure 3: Overview of a MachineAccountQuota compromise

A MachineAccountQuota compromise can also be used as part of another compromise, known as KrbRelayUp. On systems in domains where LDAP signing is not enforced, which is the default in Active Directory setting, malicious actors can execute KrbRelayUp to escalate their privileges to local administrator.

Mitigating a MachineAccountQuota compromise

The following security controls should be implemented to mitigate a MachineAccountQuota compromise:

- **Configure unprivileged user objects so they cannot add computer objects to the domain.** This can be configured by setting the 'MS-DS-MachineAccountQuota' attribute in Active Directory to zero. Typically, only privileged staff, such as system administrators, need to add new computer objects to Active Directory; for example, when a new server or workstation needs to be joined to a domain.
- **Ensure the Domain Computers security group is not a member of privileged security groups.** This prevents malicious actors from escalating their privileges as a result of a MachineAccountQuota compromise.
- **Ensure the Domain Computers security group does not have write privileges to any objects in Active Directory.** This prevents malicious actors from gaining control or access to other Active Directory objects because of a MachineAccountQuota compromise.
- **Enable LDAP signing for Domain Controllers.** LDAP signing provides numerous security protections including user authentication, message signing and encryption. LDAP signing also mitigates against KrbRelayUp.

Detecting a MachineAccountQuota compromise

Every time a computer object is created in Active Directory, event 4741 is generated and includes information about the object's properties and who created it. This event can be analysed to determine whether the computer object was created for legitimate or malicious purposes. Additionally, creating the computer object for MachineAccountQuota requires setting its password. This also generates an event that can be an indicator of a MachineAccountQuota compromise.

The events in **Table 4** should be centrally logged and analysed in a timely manner to identify a MachineAccountQuota compromise.

Table 4. Events that detect a MachineAccountQuota compromise

Event ID	Source	Description
4624	Domain Controllers	This event is generated when an object successfully logs on. This event can be correlated with event 4741 to identify if the computer object created by malicious actors has authenticated to the domain.
4724	Domain Controllers	This event is generated when an attempt is made to reset an object's password. When malicious actors create a new computer object, they set its password so they can subsequently authenticate as the computer object. If this event is generated at the same time (or near the same time) as event 4741, this may indicate a MachineAccountQuota compromise has occurred.
4741	Domain Controllers	This event is generated when a computer object is created in Active Directory. This event can be used to identify a computer object created by malicious actors as part of a MachineAccountQuota

compromise. If the computer object is created by user objects that do not normally create computer objects, this may indicate a MachineAccountQuota compromise has occurred.

Unconstrained delegation

Computer objects can be configured for delegation, enabling them to impersonate user objects to access other services on behalf of the user object. There are two types of delegation that can be configured for computer objects: constrained delegation, which limits the impersonation rights to specific services, and unconstrained delegation, which allows a computer object to impersonate a user object to any service. When a computer object is configured for unconstrained delegation, and a user object authenticates to it, a copy of the user object's TGT is stored in the computer's Local Security Authority Subsystem Service (LSASS).

Computer objects configured for unconstrained delegation are targeted by malicious actors to escalate their privileges and move laterally in an environment. If malicious actors successfully compromise one of these computers and gain local administrator access, then they can extract the TGTs from the LSASS process for any user objects that had previously authenticated to the computer object. If a user object with domain administrator privileges had previously authenticated, the malicious actor can extract their TGT, reuse it for their own purposes, and escalate their privileges to that of a domain administrator in the environment. There are also several techniques malicious actors can use to force a user object to authenticate to a computer, thereby storing the user object's TGT in the LSASS process. This allows malicious actors to target any user object in the domain and gain control of it.

Unconstrained delegation and the Domain Controller Print Spooler service

Malicious actors can leverage unconstrained delegation and target the Print Spooler service on Domain Controllers. The Print Spooler service is targeted for misuse to force a system, such as a Domain Controller, to authenticate using its computer account to another system – in this case, with the computer object configured for unconstrained delegation. As a result, malicious actors can retrieve the TGT of the computer account of a Domain Controller. Malicious actors can then use this TGT to authenticate to the Domain Controller and gain administrative access. With administrative access to a Domain Controller, malicious actors can then execute other techniques, such as a **Dumping ntds.dit** and compromising the **Skeleton Key** (discussed below)**.**

Mitigating an unconstrained delegation compromise

The most effective mitigation for unconstrained delegation is to configure computer objects for constrained delegation. User objects can also be configured to not be delegated, meaning a copy of their TGT will not be stored on computers configured for unconstrained delegation.

The following security controls should be implemented to mitigate unconstrained delegation:

- **Ensure computer objects are not configured for unconstrained delegation.** If delegation is required for a computer object, use resource-based constrained delegation instead.
- **Ensure privileged user objects are configured as 'sensitive and cannot be delegated'.** This can be configured by using the 'Account is sensitive and cannot be delegated' option on the user object in Active Directory Users and Computers.
- **Ensure privileged user objects are members of the Protected Users security group.** Members of this security group cannot be delegated.
- **Disable the Print Spooler service on Domain Controllers.** This prevents the Print Spooler service from being used to coerce a Domain Controller into authenticating to another system.

Detecting an unconstrained delegation compromise

Computer objects configured for unconstrained delegation need to be monitored for signs of compromise, such as analysing unusual authentication events; for example, user objects that do not normally authenticate to the system configured with unconstrained delegation or authentication during unusual times of day. Organizations should also log PowerShell activity because malicious actors commonly use PowerShell to leverage unconstrained delegation, and unusual activity involving this tool may indicate an attempted unconstrained delegation compromise. If malicious actors compromise a computer object configured for unconstrained delegation successfully, and extract TGTs from the LSASS process undetected, it will be more difficult to detect the next stage of the compromise that uses the TGTs to impersonate other user objects in the domain. Therefore, organisations should be vigilant in their logging and analysis to detect an attempted unconstrained delegation compromise, as it will be more difficult to detect and mitigate subsequent malicious activities.

The events in **Table 5** should be centrally logged and analysed in a timely manner to identify an unconstrained delegation compromise.

Table 5. Events that detect an unconstrained delegation compromise

Event ID	Source	Description
4103	Computer objects configured for unconstrained delegation	This event is generated when PowerShell executes and logs pipeline execution details. Common malicious tools, such as Rubeus, use PowerShell to leverage unconstrained delegation. Analysing this event for unusual PowerShell executions may indicate an unconstrained delegation compromise has occurred.
4104	Computer objects configured for unconstrained delegation	This event is generated when PowerShell executes code to capture scripts and commands. Analysing this event for unusual PowerShell executions may indicate an unconstrained delegation compromise has occurred.
4624	Computer objects configured for unconstrained delegation Domain Controllers	This event is generated when malicious actors need to authenticate to a computer object configured for unconstrained delegation. This event should be analysed for unusual authentication activity, such as user objects that do not commonly log on and unusual logon times. Separately, this event should be analysed where the Source Network Address matches the internet protocol address of a computer configured for unconstrained delegation. This may indicate the computer object is being used to leverage unconstrained delegation to compromise a Domain Controller.
4688	Computer objects configured for unconstrained delegation	This event is generated when a new process is created, such as extracting TGTs from the LSASS process (this is commonly done using malicious tools). These events can be analysed to determine if the new process is malicious or not. Below are common commands executed by malicious actors to dump the LSASS process: ▪ *procdump.exe -accepteula -ma lsass.exe lsass.dmp* ▪ *.\rundll32.exe C:\windows\System32\comsvcs.dll, MiniDump <PID> C:\lsass.dmp full* ▪ *sekurlsa::minidump C:\lsass.DMP.*
4770	Domain Controllers	This event is generated when a TGT is renewed. By default, TGTs have a maximum lifetime of seven days; however, malicious actors may choose to renew a TGT to extend its lifetime. This may indicate a TGT has been compromised as a result of malicious actors leveraging unconstrained delegation.

Password in Group Policy Preferences (GPP) compromise

In 2014, a privilege escalation vulnerability (CVE-2014-1812) was discovered in Microsoft's GPP. This vulnerability allowed malicious actors to decrypt passwords distributed by GPP. Prior to a security patch being released for this vulnerability, GPP could be configured to distribute passwords across a domain and was commonly used to set passwords for local administrator accounts, map network shares and create scheduled tasks.

GPP passwords are known as cpasswords and are stored in the system volume (SYSVOL) directory, which exists on every Domain Controller and is readable by all users in a domain. The passwords are encrypted to protect them from unauthorised disclosure. However, around 2012, the private key used to encrypt cpasswords was made available online. With the private encryption key made public, cpasswords could easily be decrypted to reveal their cleartext passwords. In 2014, Microsoft released a security patch (2962486) to fix this vulnerability, which removed the functionality used by GPP to create cpasswords. However, this security patch did not remove cpasswords that had previously been created; these need to be removed manually. Unfortunately, this manual removal process has not been completed by many organisations, and cpasswords continue to persist.

Due to their susceptibility to discovery and decryption, cpasswords are frequently targeted by malicious actors shortly after gaining initial access to a domain. By accessing files with cpasswords that are accessible by all domain users, malicious actors can rapidly escalate their privileges from a standard user to that of a local administrator or a privileged domain user, typically without being detected.

Mitigating a password in GPP compromise

Microsoft has deprecated the use of cpasswords and now provides more secure methods to configure passwords – for instance, by using Microsoft's LAPS. As other methods exist for configuring passwords via Group Policy, cpasswords should no longer be used and any existing cpasswords should be removed from the SYSVOL directory.

The following security controls should be implemented to mitigate a Password in GPP compromise:

- **Remove all GPP passwords.** This eliminates the risk of a Password in GPP compromise.
- **Apply Microsoft's security patch 2962486 to remove the functionality to create cpasswords.** This security patch prevents the creation of new cpasswords. For more information on the security patch, see Microsoft's Security Bulletin MS14-025.

Detecting a password in GPP compromise

There are no effective techniques to detect malicious actors searching the SYSVOL directory for cpasswords because there are too many methods actors can use to search the SYSVOL directory, including with PowerShell, cmd.exe, and manual browsing using Windows Explorer. The SYSVOL directory is regularly read as part of group policy, further adding to the difficulty of identifying malicious activity.

Detecting a Password in GPP compromise can be achieved by implementing a canary GPP password. A GPP password can be placed in SYSVOL that belongs to a user object that should never be logged into. This user object is then monitored for any authentication events; if the user object is authenticated it may indicate its password has been retrieved from SYSVOL and a password in GPP compromise has occurred.

Active Directory Certificate Services (AD CS) compromise

AD CS implements Microsoft's public key infrastructure (PKI), providing various services including encryption, code signing and authentication. The AD CS Certificate Authority (CA) manages and issues public key certificates. The AD CS CA can be configured with multiple certificate templates, allowing user and computer objects to request certificates for various purposes. Depending on the configuration of the AD CS CA, a range of vulnerabilities can exist which can be exploited by malicious actors to escalate their privileges and move laterally.

A common certificate template vulnerability known as ESC1 allows any user object, regardless of their permissions, to request a certificate on behalf of any other user object (including privileged user objects) in the domain. After obtaining the certificate, it can then be used by malicious actors to authenticate as that user object, allowing for the impersonation of that user object and inheritance of its privileges. This vulnerable certificate template can be

requested using built-in tools, allowing malicious actors to live off the land to minimise the risk of detection. This certificate remains valid even if the user object specified in the certificate changes its password. The certificate is only invalidated when it expires or is revoked by the AD CS CA. Certificates like this can allow malicious actors to persist in an Active Directory domain because certificates are not always revoked as part of cyber security incident response activities, even when a compromise is detected. If the certificate template has the following configuration settings, then it is considered an ESC1 vulnerable certificate:

- Enrolment rights allowing user objects to request the certificate.
- Extended Key Usage (EKU) properties enabling user authentication.
- Subject Alternative Name (SAN) can be supplied.
- CA Certificate Manager approval is not required to approve the certificate request.

There are other types of vulnerable certificate templates and configurations that exist (i.e., ESC2-ESC13) that can be exploited by malicious actors to escalate their privileges and perform lateral movement. For further information, refer to:

- SpecterOps: Certified Pre-Owned: Abusing Active Directory Certificate Services
- Mandiant: Active Directory Certificate Services: Modern Attack Paths, Mitigations, and Hardening
- Microsoft: Securing AD CS: Microsoft Defender for Identity's Sensor Unveiled.

Mitigating AD CS compromise

To mitigate AD CS vulnerabilities and prevent compromises, the vulnerabilities first need to be identified. These vulnerabilities can be identified via several methods, including using the built-in Certificate Manager (certmgr.msc) and Certutil tools, as well as open source tools such as PSPKIAudit and Certify. Certificate Manager displays a warning against any certificate templates that allows a SAN to be supplied. Certutil provides a list of certificate templates that are available to the current user object and identifies any certificate templates that provide 'FullControl' or 'Write' permissions to user objects. PSPKIAudit provides a more comprehensive assessment of AD CS, and can identify if AD CS CAs have the ESC1-ESC8 vulnerabilities. Certify provides similar information to PSPKIAudit and can also request certificates to confirm that templates are vulnerable.

The following security controls should be implemented to mitigate an ESC1 AD CS compromise:

- **Remove the Enrolee Supplies Subject flag.** Do not allow users to provide their own SAN in the certificate signing request for templates configured for client authentication. Templates configured with the Enrolee Supplies Subject flag allow a user to provide their own SAN.
- **Restrict standard user object permissions on certificate templates.** Standard user objects should not have write permissions on certificate templates. User objects with write permissions may be able to change enrolment permissions or configure additional settings to make the certificate template vulnerable.
- **Remove vulnerable AD CS CA configurations.** Ensure that the CA is not configured with the EDITF_ATTRIBUTESUBJECTALTNAME2 flag. When configured, this allows a SAN to be provided on any certificate template.
- **Require CA Certificate Manager approval for certificate templates that allow the SAN to be supplied.** This ensures certificate templates that require CA certificate manager approval are not issued automatically when requested; instead, they must be approved using certificate manager before the certificate is issued.

- **Remove EKUs that enable user authentication.** This prevents malicious actors from exploiting the certificate to authenticate as other users.

- **Limit access to AD CS CA servers to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group and reduces the number of opportunities for malicious actors to gain access to CA servers.

- **Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.** AD CS servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.** This reduces the attack surface of AD CS CA servers as there are fewer services, ports and applications that may be vulnerable and used to compromise an AD CS CA server.

- **Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.** Backups of AD CS CA servers need to be afforded the same security as the actual AD CS CA servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as AD CS CA servers.

- **Centrally log and analyse AD CS CA server logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to a CA server, this activity should be identified as soon as possible to respond and limit the impact.

Conditions may exist within complex AD CS configurations that introduce pathways that allow malicious actors to target AD CS through more sophisticated techniques. For example, a certificate template may be configured to allow enrolment by a particular security group rather than all user objects. Malicious actors may target members of this security group to gain control of one user object and then continue to target certificate templates to escalate their privileges. Alternatively, some certificate templates may be configured to allow members of Domain Computers to enrol, which can be exploited if malicious actors are able to escalate their privileges to local administrator on any domain joined computer or if they are able to create a computer account within the domain.

Detecting an AD CS compromise

Detection of AD CS compromises requires logging and analysing events from multiple sources, including Domain Controllers and root and subordinate CAs. AD CS compromises may blend in with normal activity and further analysis of events may be required to identify misuse of SANs in certificate requests, the addition of user objects to certificate templates and the removal of security settings.

An administrator with access to the CA can audit issued certificates using the built-in certificate management tools on the CA. AD CS compromises may be detected by observing any certificates issued for Client Authentication that have a mismatch between the requester and the subject name. A mismatch may indicate that a malicious actor has requested a certificate for another user.

AD CS event auditing is not enabled by default. Follow these steps to configure audit logging for AD CS:

- **Enable 'Audit object access' for Certificate Services in Group Policy for AD CS CAs.** This can be found within the 'Advanced Audit Policy Configuration' within Security Settings.

- **Within the CA properties, the Auditing tab shows configurations of events to log. Enable all available options.**

The events in **Table 6** should be centrally logged and analysed in a timely manner to identify an AD CS compromise.

Table 6. Events that detect an AD CS compromise

Event ID	Source	Description
39	Domain Controllers	This event is generated when no strong certificate mappings can be found, and the certificate does not have a new Security Identifier (SID) extension that the Key Distribution Centre (KDC) could validate. This event is logged in the 'Kerberos-Key-Distribution-Center' log.
40	Domain Controllers	This event is generated when a certificate is supplied that was issued to the user before the user existed in Active Directory and no strong mapping is found.
41	Domain Controllers	This event is generated when a certificate is supplied where the SID contained in the new extension of the user's certificate does not match the user's SID, implying that the certificate was issued to another user. This may indicate that malicious actors are attempting to authenticate with a certificate with a SAN that does not match their current account.
1102	Root and subordinate CAs	This event is generated when the Security audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if an AD CS CA has been compromised.
4674	Domain Controllers	This event is generated when an attempt is made to perform privileged operations on a protected subsystem object after the object is already opened. This may be triggered when malicious actors attempt to change security descriptors of a certificate template. The 'Object Name' field lists the certificate template name as the value that can determine which template was changed.
4768	Domain Controllers	This event is generated when a TGT is requested. The 'PreAuthType' of '16' indicates that a certificate was used in the TGT request.
4886	Root and subordinate CAs	This event is generated when AD CS receives a certificate request. This may indicate if malicious actors attempted to elevate privileges by requesting an authentication certificate for a privileged user.
4887	Root and subordinate CAs	This event is generated when AD CS approves a certificate request and issues a certificate. This may be used to indicate when malicious actors successfully escalated privileges using AD CS.
4899	Root and subordinate CAs	This event is generated when a certificate template is updated. This may occur when malicious actors attempt to modify a certificate template to introduce additional features that may make it vulnerable to privilege escalation.
4900	Root and subordinate CAs	This event is generated when security settings on a Certificate Services template are updated. This may occur when the Access Control List on the template has been modified to potentially

introduce vulnerable conditions, such as modification of enrolment rights to a certificate template.

Golden Certificate

A Golden Certificate is a persistence technique that expands upon an AD CS compromise. If malicious actors obtain administrative access to a CA, they can extract a CA certificate and private key. Once obtained, these can be used to forge valid certificates for client authentication to impersonate any other user object in the domain. The CA certificate and private key from either a root or subordinate CA can be retrieved using built-in management tools designed for backup purposes, or by using open-source tools such as Mimikatz, Seatbelt and SharpDPAPI. Mimikatz can also be used to forge a new certificate, as can ForgeCert. Certificates created with these tools are signed by the private key of the extracted CA certificate, allowing them to be used within the domain. Certificates remain valid until they are revoked, which if not done periodically may result in perpetually valid certificates that enables the malicious actor to persist on the network.

Mitigating a Golden Certificate

Mitigating a Golden Certificate requires securing both root and subordinate CAs. Due to their critical role, CAs need to be afforded the same security as other critical servers, such as Domain Controllers. This includes minimising the number of user objects with privileged access to CAs, not using CAs for any other purposes except for AD CS and monitoring CAs for signs of compromise.

The following security controls should be implemented to mitigate a Golden Certificate:

- **Use MFA to authenticate privileged users of systems.** MFA for privileged users can hinder malicious actors from gaining access to a CA using stolen credentials, thus preventing the extraction of a CA certificate and private key.
- **Implement application control on AD CS CAs.** An effective application control configuration on CAs prevents the execution of malicious executables such as Mimikatz.
- **Use a Hardware Security Module (HSM) to protect key material for AD CS CAs.** Protect private keys by using a HSM with CAs. If a HSM is used, the private key for CAs cannot be backed up and exfiltrated by malicious actors.
- **Limit access to AD CS CAs to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group and reduces the number of opportunities for malicious actors to gain access to a CA.
- **Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.** AD CS servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.
- **Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.** This reduces the attack surface of AD CS CA servers as there are fewer services, ports and applications that may be vulnerable and used to compromise an AD CS CA server.
- **Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.** Backups of AD CS CA servers need to be afforded the same security as the actual AD CS CA servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as AD CS CA servers.

- **Centrally log and analyse AD CS CA logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to a CA, this activity should be identified as soon as possible to respond and limit the impact.

Detecting a Golden Certificate

A Golden Certificate is difficult to detect as it requires detection of the initial backup and exfiltration of a CA certificate and private key. AD CS CAs can be configured to enable audit logging of some events; however, visibility of CA certificate backups is still difficult.

AD CS CA event auditing is not enabled by default. To configure audit logging for AD CS CAs:

- **Enable 'Audit object access' for Certificate Services in Group Policy for CAs.** This can be found within the 'Advanced Audit Policy Configuration' within Security Settings.
- **Enable 'Backup and restore the CA database' as events to audit in the Auditing tab within the properties for CAs.**

Event 4876 is triggered when a complete backup of the CA database is requested. This only occurs if the 'Certificate database and certificate database log' option is selected in the backup wizard. If only the 'Private key and CA certificate' option is selected, this event is not generated. As such, this cannot be relied upon to detect all backup attempts.

Windows CAPI2 logs can capture certificate export events. This log would need to be enabled within Event Viewer on CAs. When enabled, any backup of a CA certificate and private key generates event 70, which is labelled as 'Acquire Certificate Private Key'.

The events in **Table 7** should be centrally logged and analysed in a timely manner to identify a Golden Certificate.

Table 7. Events that detect a Golden Certificate

Event ID	Source	Description
70	CAPI2 logs on the root and subordinate CAs	This event is generated when a certificate is exported. This event should be filtered to check that the 'subjectName' field matches that of a CA certificate.
1102	Root and subordinate CAs	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if an AD CS CA has been compromised.
4103	Root and subordinate CAs	This event is generated when PowerShell executes and logs pipeline execution details. Common tools such as Certutil and Mimikatz use PowerShell. Analysing this event for PowerShell execution relating to these tools may indicate a Golden Certificate.
4104	Root and subordinate CAs	This event is generated when PowerShell executes code to capture scripts and commands. Common tools such as Certutil and Mimikatz use PowerShell. Analysing this event for PowerShell

		execution relating to these tools may indicate a Golden Certificate.
4876	Root and subordinate CAs	This event is triggered when a backup of the CA database is started. This does not return any logs for exporting the private key, but may be an indicator of other potentially suspicious activity occurring on a CA.

DCSync or Dumping ntds.dit, which directly compromises Active Directory's ntds.dit file on a Domain Controller. The KRBTGT user object's password hash does not need to be cracked to reveal the cleartext password before it can be used. This is because the password hash is used to encrypt the TGTs, not the cleartext password.

A successful Golden Ticket signifies the complete compromise of an Active Directory domain. The loss of all secrets from Active Directory, including user and computer object password hashes, is significant and difficult to recover from. The KRBTGT is the root of trust for a domain, and its compromise requires resetting all user and computer object passwords, including the KRBTGT user object, in a coordinated manner. To fully recover, it may require building a new Active Directory domain with new user and computer objects and destroying the old, compromised domain. For many organisations, these recovery activities would be significant, costly and disruptive.

The KRBTGT user object

The KRBTGT user object is created automatically when a new AD DS domain is created. It is a domain user object and exists on all Domain Controllers. When the KRBTGT user object is created, a random password is set by the Domain Controller, and the user object is disabled.

The KRBTGT user object is the root of trust for the domain and is used by the KDC to as part of the authentication process for all user and computer objects in a domain. The KDC uses the KRBTGT user object's password hash to encrypt TGTs which provides proof that the object has successfully authenticated to the domain. TGTs are used to request TGS tickets which are used to access specific services and systems in AD DS.

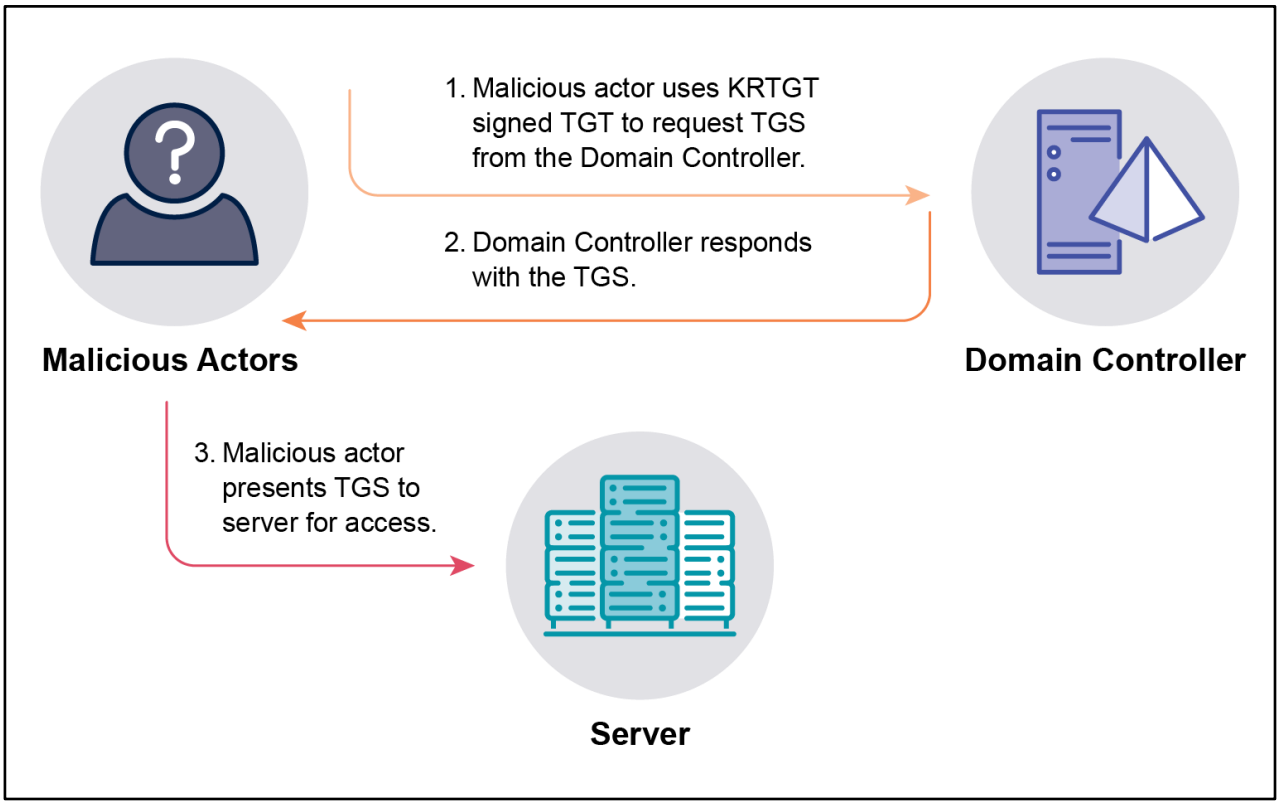

Figure 5: Overview of a Golden Ticket

Mitigating a Golden Ticket

The most effective mitigation for a Golden Ticket is to prevent the KRBTGT user object's password hash from being compromised. As DCSync and dumping ntds.dit are commonly used to compromise user object password hashes, mitigating these two techniques are the most effective way to prevent a Golden Ticket. Refer to Mitigating DCSync and Mitigating Dumping ntds.dit sections for further information.

The following security control should be implemented to mitigate a Golden Ticket:

- **Change the KRBTGT password every 12 months, or when the domain has been compromised or suspected to have been compromised.** Changing the KRBTGT password will invalidate any existing Golden Tickets that are being used.

    - To effectively change the KRBTGT user object's password hash, and invalidate any Golden Tickets, the KRBTGT password must be reset twice. This is because both the new and old KRBTGT passwords are stored by Domain Controllers such that authentication in the domain is not disrupted during a KRBTGT password change. When resetting the KRBTGT password, it is important to ensure that sufficient time is allowed between password resets to ensure the new password has had time to replicate to all Domain Controllers. For more information, see Microsoft's guidance and PowerShell script to assist with resetting the KRBTGT password.

Detecting a Golden Ticket

Similar to a Golden Certificate, detecting a Golden Ticket is difficult as it relies on analysing events to detect TGT manipulation. As a Golden Ticket enables malicious actors to forge their own TGTs, malicious actors can provide any information in the TGT to advance their objectives, including:

- Usernames for user objects that do not exist in the domain
- Mismatched security group memberships; for example, specifying in the TGT the user object is a member of the Domain Admins security group when the user object is not a member
- Using weaker cryptographic algorithms, such as RC4 instead of the Advanced Encryption Standard (AES)
- Changing the TGT lifetime from the default ten hours to a different duration (e.g. Mimikatz, a popular tool for executing a Golden Ticket, by default sets the TGT lifetime to ten years instead of ten hours).

When a Golden Ticket occurs, the events generated on a Domain Controller deviate from normal authentication activity. Normal Kerberos authentication should generate both event 4768, when a TGT ticket is requested, and event 4769, when a TGS ticket is requested. These events should correspond to each other. As a Golden Ticket forges its own TGT, it bypasses the initial step of requesting a TGT. Consequently, only event 4769 is generated, and event 4768 is absent. Detecting 4769 events without corresponding 4768 events can indicate that a Golden Ticket has occurred.

Malicious actors trying to evade detection attempt to match legitimate TGTs as close as possible to minimise the risk of detection. As such, relying on detecting unusual TGTs is not guaranteed to detect Golden Tickets. This is why if a domain has been compromised, or suspected to have been compromised, the password for the KRBTGT user object needs to be changed twice.

The following events in **Table 10** should be centrally logged and analysed in a timely manner to identify a Golden Ticket.

Table 10. Events that detect a Golden Ticket

Event ID	Source	Description
4768	Domain Controllers	This event is generated when a TGT is requested. This event, and event 4769, can be correlated to identify a potential Golden Ticket. Specifically, Kerberos authentication starts with an object requesting a TGT and subsequently providing this TGT to request a TGS ticket to access a specific service or resource. Both the TGT and TGS ticket requests generate events, 4768 and 4769. If event 4769 exists, but a corresponding event 4768 does not, this is indicative that a TGT has been forged and a Golden Ticket may have occurred. If the TGT has been forged offline, event 4768 will not exist as it was never requested from the KDC on a Domain Controller.
4769	Domain Controllers	This event is generated when a TGS ticket is requested. This event can be checked for inconsistent information, such as a weaker cryptographic algorithm than the default for the domain. This event can also be correlated with event 4768 to identify the potential use of a forged TGT.

Silver Ticket

A Silver Ticket exploits specific services running on computers by compromising the password hash of the user object running a service, or a computer object's password hash. Just like user objects, computer objects are also configured with passwords and is how they authenticate to a domain. If a computer object's password hash is compromised, malicious actors can authenticate to the computer, or authenticate as the computer object itself, gaining access to its domain privileges. Malicious actors with either a computer object's password hash or the password hash of a user object running a service can forge their own valid TGS tickets. With a valid TGS ticket, malicious actors can

authenticate directly to a computer object or via a service running on the computer object, without interacting with a Domain Controller, (see **Figure 6**).

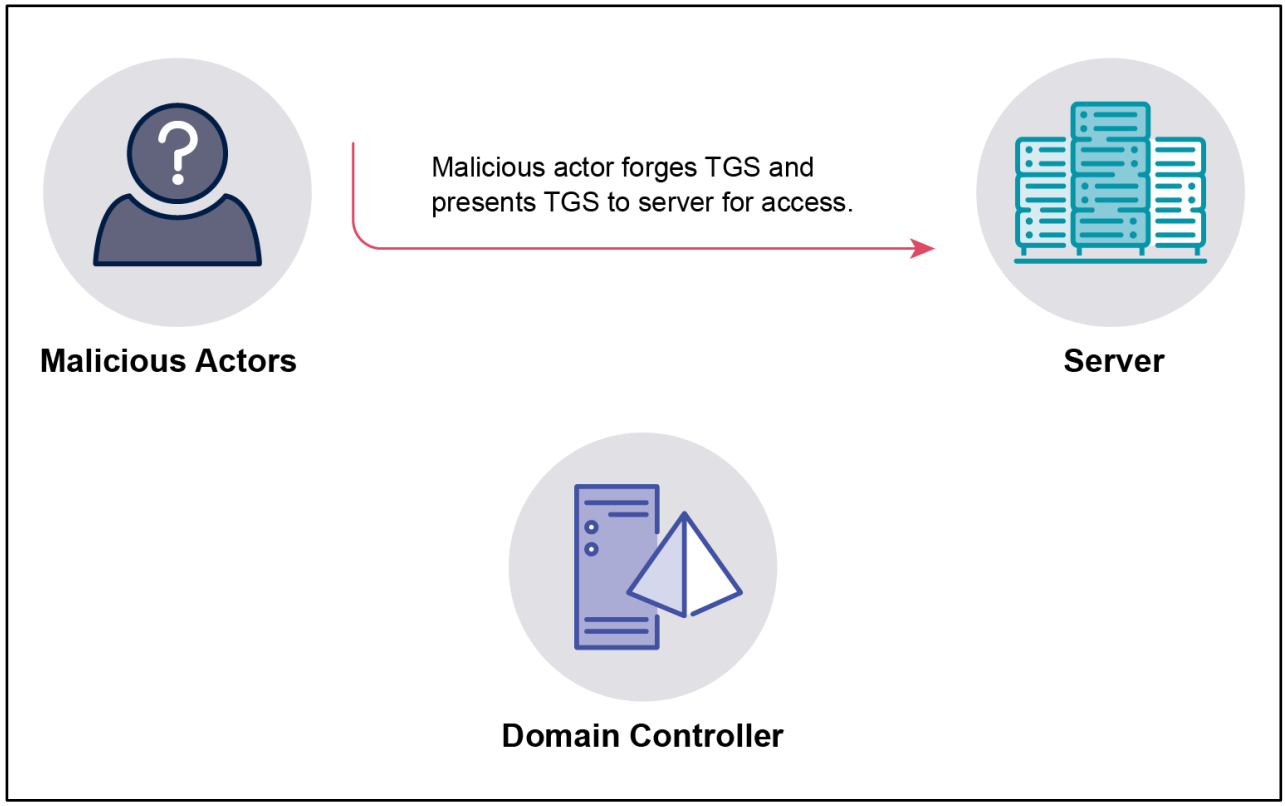

Figure 6: Overview of a Silver Ticket

Silver Tickets are used as an evasion technique by malicious actors; specifically, the authentication process is isolated to malicious actors and the computer object being targeted. This minimises the risk of detection as authentication events are more commonly sourced from Domain Controllers rather than from individual computer objects, such as servers.

Services that can be exploited by a Silver Ticket include the Common Internet File System service (which provides access to the computer object's file system), LDAP (which is used to query Active Directory), the Microsoft Structured Query Language (SQL) service (which provides access to databases) and the HOST service (which can be used to modify scheduled tasks and access the computer object with PowerShell Remoting). There are also a number of other services that can be exploited using Silver Tickets.

After successfully executing a Silver Ticket, malicious actors may also use their access to a computer object to establish persistence in an Active Directory environment and potentially retain access indefinitely. Active Directory, by default, changes computer object passwords every 30 days, but this password change process is initiated by computer objects and not Domain Controllers. This means Active Directory does not block computer objects with passwords older than 30 days from accessing resources in a domain. Even computer objects with passwords that have not been changed in years are still allowed to authenticate and access resources in a domain. Malicious actors with a computer object's password hash could tamper with this password change process, allowing them to continue authenticating as a computer object to indefinitely access the domain.

Mitigating a Silver Ticket

Mitigating a Silver Ticket requires protecting user objects that run services on computer objects, as well as the computer objects themselves. Additionally, reducing the privileges of computer objects in Active Directory can reduce the impact of a successful Silver Ticket executed in a domain.

The following security controls should be implemented to mitigate a Silver Ticket:

- **Create User objects with SPNs as group Managed Service Accounts (gMSAs).** gMSAs have automatic password rotation, a 120-character password and simplified SPN management. These security features protect the password from being cracked, reducing the likelihood of a successful Silver Ticket. However, **if creating user objects with SPNs as gMSAs is not feasible, set a minimum 30-character password that is unique, unpredictable and managed is set.**

- **Change all computer object (including Domain Controller) passwords every 30 days.** Malicious actors can establish persistence in Active Directory using a computer object's password; ensuring all computer object passwords (including Domain Controller passwords) are changed every 30 days can mitigate this persistence technique.

- **Ensure computer objects are not members of privileged security groups**, such as the Domain Admins security group. If malicious actors obtain a computer object's password hash, then they gain any privileges the computer object has in the domain.

- **Ensure the Domain Computers security group does not have write or modify permissions to any objects in Active Directory.** All computer objects are members of the Domain Computers security group. If this security group has rights over other objects, then malicious actors can use these rights to compromise other objects and potentially escalate their privileges and perform lateral movement.

Detecting a Silver Ticket

Detecting a Silver Ticket is especially difficult as it is commonly used by malicious actors to avoid detection. With a forged TGS, malicious actors can authenticate directly to a computer object without interacting with a Domain Controller, thereby avoiding any events being logged on a Domain Controller. To detect a Silver Ticket, events from the targeted computer object need to be analysed. It is common for organisations to log authentication events on Domain Controllers, and less so from other computer objects in the domain. Malicious actors are aware of this and may use a Silver Ticket to avoid detection.

The events in **Table 11** should be centrally logged and analysed in a timely manner to identify a Silver Ticket.

Table 11. Events that detect a Silver Ticket

Event ID	Source	Description
4624	Target computer	This event is generated when an account is logged into a computer. It can be correlated and analysed with event 4627 for signs of a potential Silver Ticket.
4627	Target computer	This event is generated alongside event 4624 and provides additional information regarding the group membership of the account that logged in. This event can be analysed for discrepancies, such as mismatching SID and group membership information, associated with the user object that logged on. Note

that a Silver Ticket forges the TGS, which can contain false information, such as a different SID to the user object logging on and different group memberships. Malicious actors falsify this information to escalate their privileges on the target computer object.

Golden Security Assertion Markup Language (SAML)

AD FS enables the secure sharing of verified identity information across security and enterprise boundaries. It is commonly used to extend authentication from an AD DS domain to cloud-based resources and services. AD FS supports SAML, an authentication standard that enables single sign-on. AD FS can be configured as an identity provider for different services (i.e. service providers such as Azure, AWS and Microsoft 365) that it can securely share identity information with, acting as an authentication broker. AD FS uses a private key to sign SAML responses, and these tokens are used to identify and authenticate users to the services for which AD FS acts as an identity provider.

A Golden SAML compromises the AD FS private key to enable the forging of SAML responses. It is similar to a Golden Ticket, but instead of forging tickets for accessing on-premises systems, a Golden SAML forges SAML responses to access cloud-based resources and services. SAML responses can be forged to impersonate any user and to obtain access to any service for which AD FS acts as an identity provider (see **Figure 7**).

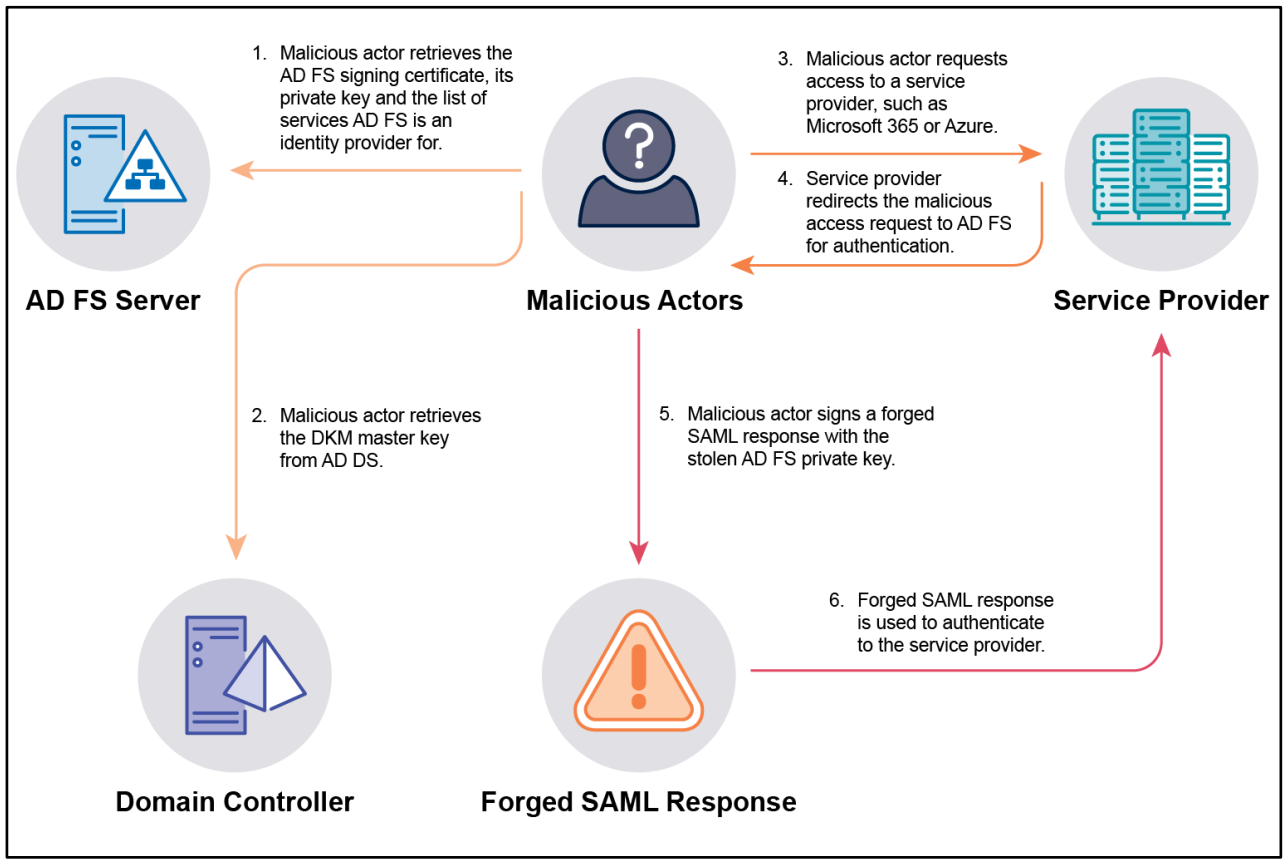

Figure 7: Overview of a Golden SAML

If a Relying Party (RP) such as Microsoft Entra ID trusts the MFA claims of the identity provider, such as an AD FS server, then a Golden SAML can bypass any MFA controls and allow malicious actors to persist even if the compromised user object changes its password. This technique was used as part of a cyber security supply chain attack involving SolarWinds that occurred in 2019 and has been used in many cyber security compromises since. Many

RPs can be configured to not accept the MFA claims from identity providers. For example, Microsoft Entra ID can be configured to require MFA even if the identity provider has an MFA claim.

To successfully execute a Golden SAML, malicious actors require administrative access to an AD FS server. This access is commonly obtained by compromising the AD FS service account that runs AD FS, or by compromising another privileged user object, such as a user object in the Domain Admins security group. Malicious actors can create their own AD FS server after gaining sufficient privileges to bypass common Golden SAML detection techniques. Once malicious actors gain privileged access to an AD FS server, the following information is retrieved and used to forge SAML responses (which is the last step in a Golden SAML):

- Token signing certificate and its private key.
- Distributed Key Manager (DKM) master key from AD DS (the DKM master key decrypts the AD FS certificate).
- List of services for which AD FS is an identity provider.

A Golden SAML is a post-exploitation and persistence technique. It is often used by malicious actors to move laterally to cloud services and to evade detection. It can be difficult to detect as the forged SAML responses appear legitimate and can be used to mimic normal user login times and activity.

Mitigating a Golden SAML

Mitigating a Golden SAML requires protecting the AD FS service account, securing access to AD FS servers, implementing system hardening, and conducting effective logging and analysis. The AD FS service account is critical to the operation of AD FS and is commonly targeted by malicious actors to gain access to the AD FS private key and other information to forge their own SAML responses. To mitigate the risk of the AD FS service account being compromised, it should be created as a gMSA; this sets a 120-character password that is automatically rotated every 30 days by Active Directory. Other privileged user objects – such as those belonging by default to the Domain Admins security group and other built-in privileged security groups – also have access to AD FS servers. For this reason, only a small subset of privileged user objects (even smaller than the Domain Admins security group) should have access. This reduces the overall attack surface that malicious actors can target.

AD FS servers should be afforded the same security as other critical servers, such as Domain Controllers. This includes implementing restricted privileged access pathways, like limiting access from jump servers and secure access workstations to only the specific ports required for administrative activities. AD FS servers should also be hardened by disabling unused services and ensuring that security products, such as antivirus and endpoint detection and response solutions, are running and up to date.

The following security controls should be implemented to mitigate a Golden SAML:

- **Ensure the AD FS service account is a gMSA.** This minimises the likelihood of the account being compromised via other techniques, such as Kerberoasting or DCSync.
- **Ensure the AD FS service account is used only for AD FS and no other purpose.** By using the AD FS service account only for AD FS, and no other purpose, it reduces its attack surface by not exposing its credentials to other systems.
- **Ensure passwords for AD FS server local administrator accounts are long (30-character minimum), unique, unpredictable and managed.** Microsoft's Local Administrator Password Solution (LAPS) can be used to achieve this for local administrator accounts. Local administrator accounts can be targeted by malicious actors to gain access to AD FS servers. For this reason, these accounts need to be protected from compromise.

- **Limit access to AD FS servers to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group. This reduces the number of opportunities for malicious actors to gain access to AD FS servers.

- **Restrict privileged access pathways to AD FS servers to jump servers and secure admin workstations using only the ports and services that are required.** AD FS servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Only use AD FS servers for AD FS and ensure no other non-security-related services or applications are installed.** This reduces the attack surface of AD FS servers as there are fewer services, ports and applications that may be vulnerable and used to compromise an AD FS server.

- **Centrally log and analyse AD FS server logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to AD FS servers, this activity should be identified as soon as possible to respond and limit the impact.

- **Encrypt and securely store backups of AD FS servers and limit access to only Backup Administrators.** Backups of AD FS servers need to be afforded the same security as the actual AD FS servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as AD FS servers.

- **Rotate AD FS token-signing and encryption certificates every 12 months, or sooner if an AD FS server has been compromised or suspected to have been compromised.** Both certificates need to be rotated twice in rapid succession to revoke all existing AD FS tokens.

Detecting a Golden SAML

Detecting a Golden SAML can be difficult, especially after malicious actors have successfully executed the compromise and are using forged SAML responses to access service providers like Microsoft 365 and Azure. The first opportunity to detect a Golden SAML is the generation of event 70, created by the compromise of an AD FS server and the export of the private key. Event 70 can be analysed to determine if the export was authorised or not. If malicious actors successfully execute a Golden SAML and forge SAML responses to authenticate to service providers, then the AD FS and service providers authentication events can be correlated to identify inconsistencies that may indicate the use of forged SAML responses.

The events in **Table 12** should be centrally logged and analysed in a timely manner to identify a Golden SAML.

Table 12. Events that detect a Golden SAML

Event ID	Source	Description
70	AD FS Servers	This event is generated when a certificate's private key is exported. Extracting the private key is the first step in a Golden SAML.
307	AD FS Servers	This event is generated when there is a change to the AD FS configuration. Malicious actors may add a new trusted AD FS server they can control instead of extracting the certificate and other information from an existing AD FS server.
510	AD FS Servers	This event provides additional information and can be correlated with event 307 with the same instance ID. Any events generated for changes to AD FS should be investigated to confirm if the changes were authorised or not.

1007	AD FS Servers	This event is generated when a certificate is exported. The first step of a Golden SAML is to export the signing certificate from an AD FS server.
1102	AD FS Servers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if an AD FS server has been compromised.
1200	AD FS Servers	This event is generated when AD FS issues a valid token as part of the authentication process with a service provider, such as Microsoft 365 or Azure. A Golden SAML bypass AD FS servers, resulting in the absence of this event (and event 1202). This event can be correlated with authentication events from service providers to identify the absence of AD FS authentication events, which may be a sign that a forged SAML response was used.
1202	AD FS Servers	This event is generated when AD FS validates a new credential as part of the authentication process with a service provider, such as Microsoft 365 or Azure. A Golden SAML bypasses AD FS servers, resulting in the absence of this event (and event 1200). This event can be correlated with authentication events from service providers to identify the absence of AD FS authentication events, which may be a sign that a forged SAML response was used.
4662	Domain Controllers	This event is generated when the AD FS DKM container in Active Directory is accessed. The 'Active Directory Service Access' setting needs to be configured for auditing with 'Read All Properties' configured for the AD FS parent and child containers in Active Directory. This event should be monitored for the 'thumbnailPhoto' attribute with a Globally Unique Identifier (GUID) value matching '{8d3bca50-1d7e-11d0-a081-00aa006c33ed}'. This attributed GUID stores the DKM master key and should only be periodically accessed by the AD FS service account. Each time this event is generated, it should be analysed to determine if the activity was authorised.

Microsoft Entra Connect Compromise

Microsoft Entra Connect is an on-premises application that enables hybrid identity management by synchronising Active Directory with its cloud-based counterpart Microsoft Entra ID (note: Microsoft Entra ID is a paid feature). Microsoft Entra Connect allows Active Directory objects, such as user objects, to seamlessly access cloud-based resources and services, such as Microsoft 365 and Azure, using their Active Directory credentials and single sign-on.

Microsoft Entra Connect supports multiple authentication configurations to synchronise identities between Active Directory and Microsoft Entra ID. The default and most used configuration is Password Hash Synchronisation (PHS). This synchronises password hashes from Active Directory to Microsoft Entra ID, so that password hashes are stored in both Active Directory and Microsoft Entra ID. The second most common configuration is Pass-Through Authentication (PTA) which allows Microsoft Entra ID to forward authentication requests via Microsoft Entra Connect to Active Directory to validate whether the credentials are correct. This configuration means password hashes are only stored in Active Directory, not in Microsoft Entra ID.

Both PHS and PTA have been known to be exploited by malicious actors to escalate their privileges, move laterally, and to establish persistence in Active Directory and Microsoft Entra ID. The PHS configuration creates two new user objects, one in Active Directory with the username prefix MSOL and another in Microsoft Entra ID with a username prefix Sync. The Active Directory MSOL user has permissions to replicate information, including password hashes from Domain Controllers, which are the same permissions required for DCSync. The Microsoft Entra ID Sync user object has the Directory Synchronisation Accounts role; this allows it to create, modify and delete user objects and set passwords. If malicious actors gain administrative access to a Microsoft Entra Connect server, they can use malicious tools such as AADInternals to extract the cleartext password for both the MSOL and Sync user objects. After retrieving the cleartext passwords for these user objects, malicious actors can use the Active Directory MSOL user object to retrieve the password hash for any user object in Active Directory. The Microsoft Entra ID Sync user object could also be used to set the password for a user object with the Global Administrator role. This role can manage all aspects of Microsoft Entra ID and Microsoft services that use Microsoft Entra identities, granting malicious actors complete control of all cloud-based resources and services in an organisation's Azure subscription.

The PTA Microsoft Entra Connect configuration is susceptible to compromise if malicious actors can obtain administrative access to a Microsoft Entra Connect server. Instead of retrieving cleartext passwords, like in a PHS, compromising PTA involves overriding the authentication process between Microsoft Entra ID and Active Directory. Malicious actors can override the PTA authentication process to allow themselves to authenticate as any user object, regardless of whether they know the password or not. The PTA authentication process can also be compromised to save a copy of a user object's cleartext password anytime they authenticate using PTA. These techniques can be effective for maintaining persistence, as they allow malicious actors to impersonate other users and minimise the risk of detection.

Mitigating a Microsoft Entra Connect compromise

Mitigating compromises against Microsoft Entra Connect requires protecting Microsoft Entra Connect servers. These servers need to be afforded the same security as Domain Controllers as they are also part of the authentication flow between Active Directory and Microsoft Entra ID.

The following security controls should be implemented to mitigate a Microsoft Entra Connect compromise:

- **Disable hard match takeover.** This prevents the source of authority for objects in Microsoft Entra ID from changing to Active Directory. If the source of authority for a Microsoft Entra ID object is changed to Active Directory, then changes made to the Active Directory object overwrite the object's properties in Microsoft Entra ID, including the password hash. If this setting is not disabled, and PHS is enabled, malicious actors can use this feature to take control of Microsoft Entra ID objects and gain privileged access to cloud-based resources and services.

- **Disable soft matching.** After initial synchronisation between Active Directory and Microsoft Entra ID, there is no requirement to keep soft matching enabled. If soft matching is enabled, it attempts to match new Active Directory objects with existing Microsoft Entra ID objects. If no match is found, then a new Microsoft Entra ID object is provisioned. Malicious actors can use this feature to provision a new user object they control in Microsoft Entra ID and gain privileged access to cloud-based resources and services.

- **Do not synchronise privileged user objects from AD DS to Microsoft Entra ID. Use separate privileged accounts for AD DS and Microsoft Entra ID.** If malicious actors compromise an AD DS domain and gain access to a privileged user object that synchronises with Microsoft Entra ID, then this gives them access to Microsoft Entra ID and they can quickly expand the compromise from AD DS systems to cloud-based services and resources.

- **Enable MFA for all privileged users in Microsoft Entra ID.** This makes it harder for malicious actors to take control of a privileged user object in Microsoft Entra ID as they need the additional authentication factor required by MFA.

- **Limit access to Microsoft Entra Connect servers to only privileged users that require access.** This may be a smaller subset of privileged users than the Domain Admins security group, which reduces the number of user objects malicious actors can target to gain access to Microsoft Entra Connect servers.

- **Restrict privileged access pathways to Microsoft Entra Connect servers to jump servers and secure admin workstations using only the ports and services that are required for administration.** Microsoft Entra Connect servers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Ensure passwords for Microsoft Entra Connect server local administrator accounts are long (30-character minimum), unique, unpredictable and managed.** Microsoft's Local Administrator Password Solution (LAPS) can be used to achieve this for local administrator accounts. Local administrator accounts can be targeted by malicious actors to gain access to Microsoft Entra Connect servers. For this reason, these accounts need to be protected from compromise.

- **Only use Microsoft Entra Connect servers for Microsoft Entra Connect and ensure no other non-security-related services or applications are installed.** This reduces the attack surface of Microsoft Entra Connect servers as there are fewer services, ports and applications that may be vulnerable and used to compromise a Microsoft Entra Connect server.

- **Encrypt and securely store backups of Microsoft Entra Connect and limit access to only Backup Administrators.** Backups of Microsoft Entra Connect servers need to be afforded the same security as the actual Microsoft Entra Connect servers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as Microsoft Entra Connect servers.

- **Centrally log and analyse Microsoft Entra Connect server logs in a timely manner to identify malicious activity.** If malicious actors gain privileged access to Microsoft Entra Connect servers, this activity should be identified as soon as possible, increasing response time and limiting impact.

Detecting a Microsoft Entra Connect compromise

Detecting Microsoft Entra Connect compromises requires analysing event logs from Microsoft Entra Connect servers, Microsoft Entra ID, and other associated systems and services, such as Microsoft Entra Connect Health and Microsoft Sentinel (note: Microsoft Sentinel is a paid feature). Event logs from Microsoft Entra Connect servers need to be monitored for signs of unauthorised access and malicious activity. This includes analysing event logs for unusual authentication events, malicious PowerShell activity and unusual Microsoft Entra Connect activity, such as password synchronisation events and the stopping and starting of services. Each of these events can indicate a compromise targeting a Microsoft Entra Connect server.

Microsoft Entra Connect Health cloud service provides a centralised view of Microsoft Entra Connect synchronisation operations and errors, including changes to the Microsoft Entra ID Sync user object. This service generates alerts based on these events which can be indicative of a compromise against Microsoft Entra Connect.

Similar to detecting a Golden SAML, correlating event logs from different sources can help identify authentication discrepancies between Active Directory and Microsoft Entra ID. For example, if there is a PTA authentication event, and there is no corresponding authentication event on a Domain Controller, this can be indicative of a PTA configuration exploit against Microsoft Entra Connect.

The events in **Table 13** should be centrally logged and analysed in a timely manner to identify a Microsoft Entra Connect compromise.

Table 13. Events that detect a Microsoft Entra Connect compromise

Event ID	Source	Description

611	Microsoft Entra Connect Servers	This event is generated when the PHS has failed. This event can be analysed to identify unusual password synchronisation activity that could indicate a compromise against Microsoft Entra Connect.
650	Microsoft Entra Connect Servers	This event is generated when password synchronisation starts retrieving updated passwords from Active Directory. This event can be analysed to identify unusual password synchronisation activity that could indicate a compromise against Microsoft Entra Connect.
651	Microsoft Entra Connect Servers	This event is generated when password synchronisation finishes retrieving updated passwords from Active Directory. This event can be analysed to identify unusual password synchronisation activity that could indicate a compromise against Microsoft Entra Connect.
656	Microsoft Entra Connect Servers	This event is generated when password synchronisation indicates that a password change occurred and there was an attempt to sync this password to Microsoft Entra ID. This event can be analysed to identify unusual password synchronisation activity that could indicate a compromise against Microsoft Entra Connect.
657	Microsoft Entra Connect Servers	This event is generated when a password change request is successfully sent to Microsoft Entra ID. This event can be analysed to identify unusual password synchronisation activity that could indicate a compromise against Microsoft Entra Connect.
1102	Microsoft Entra Connect Servers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Microsoft Entra Connect server has been compromised.
4103	Microsoft Entra Connect Servers	This event is generated when PowerShell executes and logs pipeline execution details. AADInternals, a popular toolkit used for exploiting Microsoft Entra Connect, uses PowerShell for its execution. This event can indicate the use of PowerShell-based malicious tools, which may assist in identifying if a malicious actor attempted to exploit Microsoft Entra Connect.
4104	Microsoft Entra Connect Servers	This event is generated when PowerShell executes code to capture scripts and commands. AADInternals, a popular toolkit used for exploiting Microsoft Entra Connect, uses PowerShell for its execution. This event can indicate the use of PowerShell-based malicious tools, which may assist in identifying if a malicious actor attempted to exploit Microsoft Entra Connect.

One-way domain trust bypass

Active Directory supports trusts between domains to allow users from one domain to be authenticated in another domain and access its resources. Trust relationships are either one-way or two-way, and transitive or non-transitive. In a one-way trust, users in Domain B (trusted) can access resources in Domain A (trusting), but users in Domain A cannot access resources in Domain B (see **Figure 8**). If a trust is transitive, then trust can be extended to other

domains beyond the two domains that established it, while a non-transitive trust can be used to deny trust relationships with other domains.

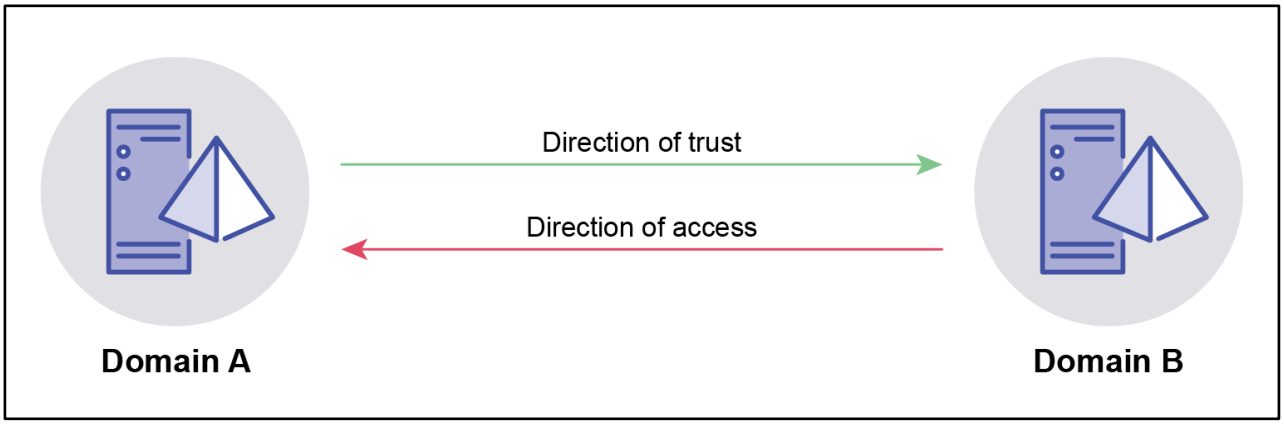

Figure 8: Overview of One-way domain trust and direction of access

When a trust is established between two domains, a Trusted Domain Object (TDO) is created in Active Directory. The TDO has a password that is shared between both domains in a trust relationship. Additionally, the password is stored in Active Directory and can be retrieved. Malicious actors with administrator access to a Domain Controller in Domain A (the trusting domain) can retrieve the TDO's password hash from the System Container in Active Directory. With the password hash, malicious actors can request a TGT from Domain B (the trusted domain) by supplying the TDO's password hash. Domain B responds with a TGT; this TGT can then be used to authenticate from Domain A to Domain B, bypassing the trust relationship (see **Figure 9**). As trust relationships between domains can be bypassed, regardless of the direction, Active Directory domains do not act as security boundaries.

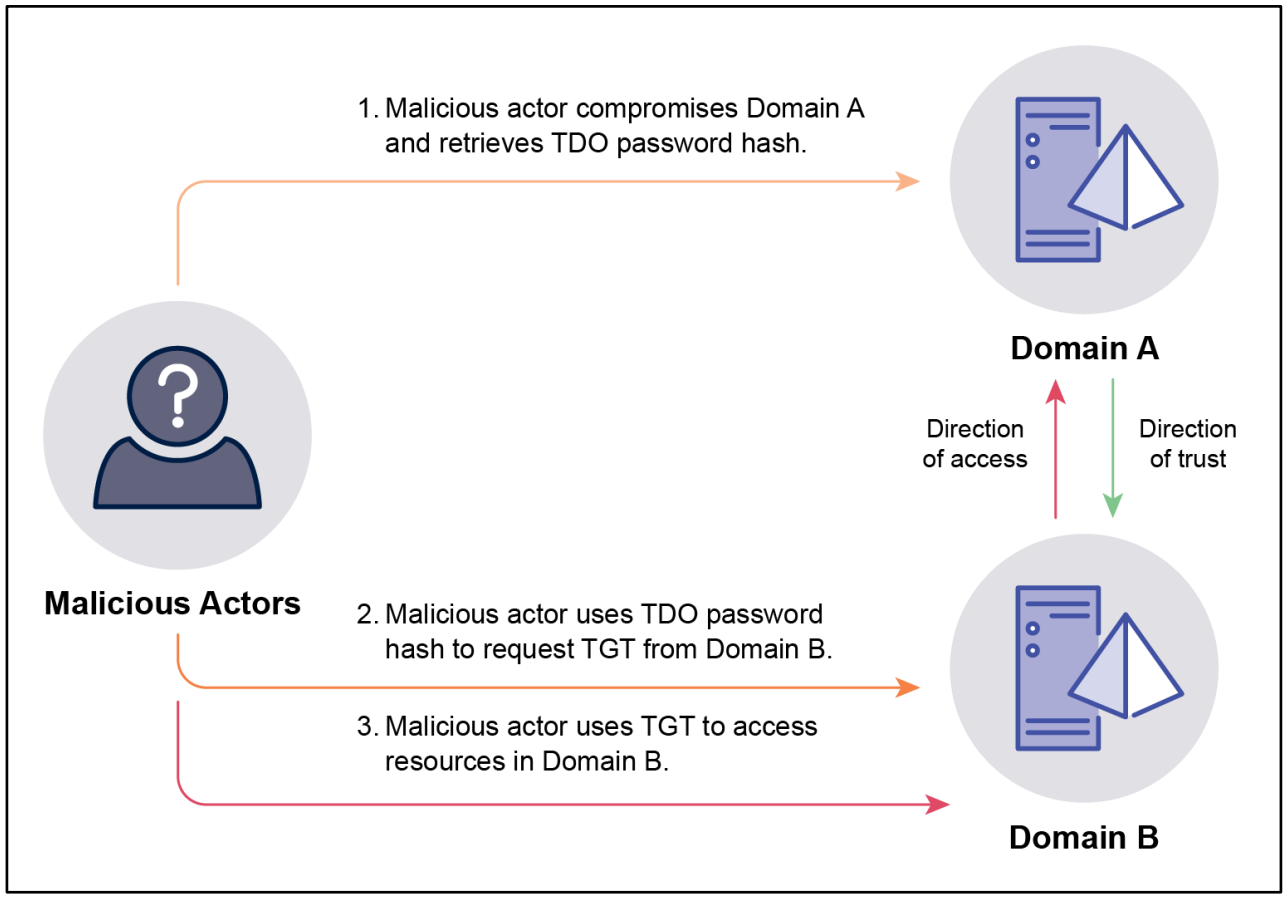

Figure 9: Overview of one-way domain trust bypass

If malicious actors gain administrator access to a Domain Controller, and a trust relationship exists with one or more other domains, they can move laterally to these domains. Cyber security incident response activities should include all other domains where a trust relationship exists if one domain is compromised. If an Active Directory domain has been compromised, or suspected to have been compromised, the TDO password should first be reset in the trusting domain (Domain A) and then the same password reset in the trusted domain (Domain B).

Mitigating a one-way domain trust bypass

Active Directory domain trust relationships should be carefully considered before being implemented. Domain trusts should not be implemented to establish security boundaries between different domains, as they can be bypassed if malicious actors can gain access to a Domain Controller. For this reason, the most effective mitigation to prevent domain trust bypasses is to secure privileged access to Domain Controllers. This is best achieved by minimising the number of user objects with access to Domain Controllers and restricting which systems privileged users can connect from to access Domain Controllers. Monitoring Domain Controllers for unusual authentication events and system activity is also important.

The following security controls should be implemented to mitigate a one-way domain trust bypass:

- **Limit access to Domain Controllers to only privileged users that require access.** This reduces the number of opportunities for malicious actors to gain access to Domain Controllers.

- **Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.** Domain Controllers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.

- **Encrypt and securely store backups of Domain Controllers and limit access to only Backup Administrators.** Backups of Domain Controllers need to be afforded the same security as the actual Domain Controllers. Malicious actors may target backup systems to gain access to critical and sensitive computer objects, such as Domain Controllers.

- **Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.** This reduces the attack surface of Domain Controllers as there are fewer services, ports and applications that may be vulnerable and used to compromise a Domain Controller.

- **Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.** Domain Controller logs provide a rich source of information that is important for investigating potentially malicious activity on Domain Controllers and in the domain.

- **Disable the Print Spooler service on Domain Controllers.** For example, malicious actors have targeted the Print Spooler service on Domain Controllers as a technique to authenticate to a system they control to collect the Domain Controllers computer object password hash or TGT. Malicious actors can then use this to authenticate to the Domain Controller they coerced and gain administrative access.

Detecting a one-way domain trust bypass

Detecting a one-way domain trust bypass requires monitoring authentication related events and analysing the User ID value for matches to the TDO. The TDO should only be used to communicate with the other trusting or trusted domain it was created for. Any other activity associated with the TDO – such as unusual LDAP queries or authentication events, should be analysed to determine if the activity is malicious or not.

The events in **Table 14** should be centrally logged and analysed in a timely manner to identify a one-way domain trust bypass.

Table 14. Events that detect a one-way domain trust bypass

Event ID	Source	Description
1102	Domain Controllers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Domain Controller has been compromised.
4103	Domain Controllers	This event is generated when PowerShell executes and logs pipeline execution details. Common malicious tools used to retrieve the TDO password hash, like Mimikatz, use PowerShell. Analysing this event for unusual PowerShell executions on Domain Controllers may indicate the TDO has been compromised.
4104	Domain Controllers	This event is generated when PowerShell executes code to capture scripts and commands. Common malicious tools used to retrieve the TDO password hash, such as Mimikatz, use PowerShell. Analysing this event for unusual PowerShell executions on Domain Controllers may indicate the TDO has been compromised.

4768	Domain Controllers in the trusted domain	This event is generated when a TGT is requested. After the TDO password hash has been retrieved, it is commonly used to request a TGT in the trusted domain. If the User ID value matches the TDO username, this may indicate the TDO has been compromised and a one-way domain trust bypass has occurred.

Security Identifier (SID) History compromise

Every object in AD DS has a unique and immutable SID that is used by AD DS to identify the object and determine the privileges it has when accessing systems, services and resources. As usernames can be changed, AD DS relies on the SID to distinguish between objects to ensure that the correct access is provided to an object. In addition to the 'SID' attribute, there is the 'sIDHistory' attribute which stores previous SIDs. If a SID is changed, for example, when an object is migrated from one domain to another, the object will be given a new SID and its previous SID will be stored in the 'sIDHistory' attribute.

Malicious actors can exploit the SID History functionality to establish persistence and hide in an AD DS environment. This technique is performed after malicious actors achieve initial access and privilege escalation, and requires administrator privileges to a Domain Controller. With this access, malicious actors can add a SID to the sIDHistory of an object they control. The SID added to the 'sIDHistory' attribute is typically from a privileged user object or security group, such as the default administrator user object or the Domain Admins security group. After adding the SID of a privileged user object or security group to another user object's 'sIDHistory' attribute, this user object then inherits that SID's privileges. For example, adding the Domain Admins SID to another user object will grant the user object domain administrator privileges without the user object appearing in the Domain Admins security group membership. This helps malicious actors persist in environments by allowing them to leverage user objects that appear to be standard users (i.e. not privileged) but are still able to perform privileged actions.

Domain hopping with Golden Tickets and SID History

By combining a Golden Ticket with SID History, malicious actors can forge a TGT to access other domains in the same forest—or even other forests, if inter-forest trusts exist. When forging a TGT as part of a Golden Ticket, malicious actors can add a SID for a security group from another domain. For example, if malicious actors compromise Domain A, they then forge a TGT and include the SID of a security group from Domain B. The TGT is then sent to a Domain Controller in Domain B, which validates it and sends back a TGS. The TGS includes a Privileged Attribute Certificate that includes the Domain B security group. The TGS can then be used to access any systems and resources in Domain B that the security group has access to. As the Active Directory security boundary is at the forest level and not the domain level, malicious actors who compromise one domain can use their access to access any other domain in the same forest.

Mitigating a SID History compromise

A SID History compromise happens in the post-exploitation phase and is used as a way for malicious actors to persist in an AD DS environment and evade detection. Mitigating this requires removing values from the 'sIDHistory' attribute on objects. This also applies to user objects that have been migrated from one domain to another. In such cases, once user objects are migrated and appropriate accesses configured, the 'sIDHistory' attribute should be cleared.

The following security controls should be implemented to mitigate a SID History compromise:

- **Ensure the 'sIDHistory' attribute is not used.** Unless migrating user objects from one domain to another, the 'sIDHistory' attribute should not be required. If no user objects are configured with this attribute, then a SID History compromise is not possible.

- **Ensure the 'sIDHistory' attribute is checked weekly.** Malicious actors may add a value to the 'sIDHistory' attribute of a user object they control to establish persistence. Regularly checking for this attribute on Active Directory objects may increase detection of this persistence strategy.

- **Enable SID Filtering for domain and forest trusts.** This prevents SIDs of built-in security groups, such as Domain Admins and Enterprise Admins, being used in TGTs across domains. However, malicious actors can still use the SIDs of other security groups if the Relative Identifier is greater than 1000.

Detecting a SID History compromise

A SID History compromise can be detected by monitoring for changes to the 'sIDHistory' attribute on Active Directory objects. As this attribute should only be used for domain migration purposes, it should be rare for this attribute to be modified for most organisations. Moreover, logging PowerShell activity on Domain Controllers can help detect this compromise as common malicious tools use PowerShell to execute a SID History compromise.

The events in **Table 15** should be centrally logged and analysed in a timely manner to identify a SID History compromise.

Table 15. Events that detect a SID History compromise

Event ID	Source	Description
1102	Domain Controllers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Domain Controller has been compromised.
4103	Domain Controllers	This event is generated when PowerShell executes and logs pipeline execution details. Common malicious tools used to execute a SID History compromise, such as Mimikatz, use PowerShell. Analysing this event for PowerShell execution relating to SID History may indicate dumping of the ntds.dit file.
4104	Domain Controllers	This event is generated when PowerShell executes code to capture scripts and commands. Common malicious tools used to execute a SID History compromise, such as Mimikatz, use PowerShell. Analysing this event for PowerShell execution relating to SID History may indicate dumping of the ntds.dit file.
4675	Domain Controllers	This event is generated when SIDs are filtered. Domain hopping with Golden Tickets and SID History may use SIDs that get filtered. If this event is generated, it may indicate a SID History compromise has been attempted.
4738	Domain Controllers	This event is generated when the 'sIDHistory' attribute is modified for a user object.

Skeleton Key

Skeleton Key is malware that overrides the NTLM and Kerberos authentication process and sets a password – called the Skeleton Key – to authenticate as any user object in a domain. This compromises the LSASS process on a Domain Controller and requires administrative privileges to execute. This malware is used by malicious actors to establish

persistence and evade detection. After overriding the authentication process and injecting the Skeleton Key, malicious authentications are virtually indistinguishable from legitimate authentications, making it difficult to identify malicious activity.

Skeleton Key does not interrupt legitimate authentication attempts in the domain. It achieves this by modifying the authentication process to do the following:

- When a user object authenticates with their correct and legitimate password, the authentication attempt succeeds.
- When malicious actors authenticate with the Skeleton Key, the authentication process checks if it is the correct and legitimate password. If not, it compares the Skeleton Key to the one stored in memory on the Domain Controller. If they match, the authentication succeeds.

To achieve this, Skeleton Key malware downgrades the cryptographic algorithm used by Kerberos to RC4, even if a stronger cryptographic algorithm, such as AES, is available. This can cause protected accounts, or others configured to not support RC4, to fail to authenticate to an infected Domain Controller. A version of Skeleton Key which does not perform this downgrade is available, although it can cause noticeable performance or memory issues on infected Domain Controllers.

Skeleton Key can be executed on every Domain Controller in a domain to ensure all malicious authentication attempts succeed. It can also be executed on a single Domain Controller. However, this may cause malicious authentication attempts to fail as they may be sent to a Domain Controller whose authentication process is functioning correctly, instead of being sent to the compromised Domain Controller.

The compromised NTLM, Kerberos authentication process and Skeleton Key reside in memory and can be removed by restarting the compromised Domain Controller. However, if malicious actors maintain administrative privileges to the Domain Controller, they may execute Skeleton Key again to regain their persistence in the domain.

Running the LSASS process in Protected Mode

The LSASS process is responsible for validating users for local and remote sign-ins and enforcing security policy. It is commonly targeted by malicious actors to extract credentials from memory or inject code to modify the authentication flow, such as with Skeleton Key. Local administrator privileges are required for compromises against the LSASS process.

When in protected mode, any standard (i.e. non-protected) processes cannot access or modify the memory of the LSASS process, providing some protection against such compromises. Furthermore, any plugins or drivers loaded into the Local Security Authority (LSA) (generally used for authentication extensions, smart cards, etc.) are required to meet Microsoft's signing requirements.

Running the LSASS process in protected mode (forming part of LSA protection) can be configured through group policy. Auditing can be enabled to log which drivers do not meet Microsoft's signing requirements and are prevented from loading.

This protection can be bypassed by leveraging vulnerable drivers or other means to execute malicious code in kernel mode, which ignores such restrictions and allows the modification of protection levels. Malicious actors can leverage this to remove the protection of the LSASS process itself or to elevate their own process to protected mode, either of which would bypass this control.

Mitigating Skeleton Key

Mitigating Skeleton Key requires reducing the likelihood of malicious actors gaining administrative access to a Domain Controller by minimising the number of user objects with administrative access, securing the user objects that require access and hardening Domain Controllers. Running the LSASS process in protected mode makes it more difficult for malicious actors to override this process if they gain administrative access to a Domain Controller. However, the LSASS protected mode can be bypassed using a malicious or vulnerable kernel mode driver – though implementing Microsoft's vulnerable driver blocklist can help mitigate this. Additionally, not running any additional services, ports or applications on Domain Controllers reduces its attack surface.

The following security controls should be implemented to mitigate Skeleton Key:

- **Limit access to Domain Controllers to only privileged users that require access.** This reduces the number of opportunities for malicious actors to gain access to Domain Controllers.
- **Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.** Domain Controllers are classified as 'Tier 0' assets within Microsoft's 'Enterprise Access Model'.
- **Run the LSASS process in protected mode.** This makes it more difficult to override the LSASS process, which is required for Skeleton Key to succeed.
- **Implement Microsoft's vulnerable driver blocklist.** Restricting known malicious or vulnerable drivers on Domain Controllers makes it more difficult for malicious actors to bypass LSASS protection.
- **Restrict driver execution to an approved set.** Restricting the drivers that can be loaded on Domain Controllers to an approved set hardens it against attempts to bypass LSASS protection. This can be achieved through application control solutions, including Microsoft's Windows Defender Application Control.
- **Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.** This reduces the attack surface of Domain Controllers as there are fewer services, ports and applications that may be vulnerable and used to compromise a Domain Controller.
- **Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.** Domain Controller logs provide a rich source of information that is important for investigating potentially malicious activity on Domain Controllers and in the domain.
- **Disable the Print Spooler service on Domain Controllers.** For example, malicious actors have targeted the Print Spooler service on Domain Controllers as a technique to authenticate to a system they control to collect the Domain Controllers computer object password hash or TGT. Malicious actors can then use this to authenticate to the Domain Controller they coerced and gain administrative access.

Detecting Skeleton Key

Detecting Skeleton Key requires monitoring the LSASS process on a Domain Controller. Typically, a Skeleton Key will be performed on every Domain Controller in a domain to ensure all malicious authentication attempts are successful. It must also be performed each time a Domain Controller is restarted as the Skeleton Key is in memory only. These characteristics of a Skeleton Key provide opportunities for its detection.

Note: Some of the indicators below rely on LSASS protection being enabled, as otherwise the use of kernel mode drivers and other bypasses may not be required by malicious actors.

The following audit measures should be implemented to assist in detecting Skeleton Key:

- **Enable audit mode for the LSASS process.** This generates an event for each driver that fails to load if LSASS protection is enabled. This can indicate malicious actors attempting to load a malicious or vulnerable driver to compromise the LSASS process.

- **Enable 'Audit Kernel Object' on Domain Controllers of Microsoft Windows Server 2016 or higher.** This logs attempts to access secure objects with an appropriate System Access Control List (SACL) configured. From Microsoft Windows Server 2016 onwards, there is a default SACL for the LSASS process.

The events in **Table 16** should be centrally logged and analysed in a timely manner to identify Skeleton Key.

Table 16. Events that detect a Skeleton Key

Event ID	Source	Description
1102	Domain Controllers	This event is generated when the 'Security' audit log is cleared. To avoid detection, malicious actors may clear this audit log to remove any evidence of their activities. Analysing this event can assist in identifying if a Domain Controller has been compromised.
3033	Domain Controllers	This event is generated when a driver fails to load because it does not meet Microsoft's signing requirements. This indicates that a code integrity check determined that a process, usually LSASS.exe, attempted to load a driver that did not meet the Microsoft signing level requirements. These drivers fail to load if LSASS protection is enabled and should be audited prior to enabling protection. Furthermore, an unknown driver or plugin may indicate attempted tampering with the LSASS process.
3063	Domain Controllers	This event is generated when a driver failed to load because it did not meet the security requirements for shared sections. This indicates a code integrity check determined that a process, usually lsass.exe, attempted to load a driver that did not meet the security requirements for shared sections. These drivers will fail to load if LSASS protection is enabled, and should be audited, prior to enabling protection. An unknown driver or plugin may also indicate attempted tampering with the LSASS process.
4103	Domain Controllers	This event is generated when PowerShell executes and logs pipeline execution details. Common malicious tools used to execute a Skeleton Key, such as Mimikatz, use PowerShell. Analysing this event for PowerShell execution relating to a Skeleton Key may indicate a compromise.
4104	Domain Controllers	This event is generated when code is executed by PowerShell, capturing scripts and the commands run. Abnormal script execution should be investigated, noting that PowerShell-based tools such as Invoke-Mimikatz can be utilised to deploy a Skeleton Key without having to copy any files onto the Domain Controller.

4663	Domain Controllers	This event is generated when an attempt was made to access an object. If 'Kernel Object Auditing' is enabled, this will include logging when a process attempts to access the memory of the LSASS process. This is the most direct indicator of tampering with the LSASS process. Any event with the object as 'lsass.exe' from an unexpected process (including remote administrative tools such as PowerShell Remoting [wsmprovhost.exe]), could indicate the deployment of a Skeleton Key. Certain antivirus or endpoint solutions may access the LSASS process; therefore, it is important to determine what security solutions are present and expected on the host.
4673	Domain Controllers	This event is generated when a privileged service is called. This event triggers when the 'SeDebugPrivilege' privilege is enabled, which is required to successfully execute a Skeleton Key. This event also triggers when the 'SeTCBPrivilege' privilege is used. The 'SeTCBPrivilege' privilege allows for the impersonation of the system account and is often requested by Mimikatz.
4697	Domain Controllers	This event is generated when a service has been installed on the system. If this is an unknown kernel mode driver it may indicate a malicious or vulnerable driver being leveraged for exploitation, such as to bypass LSA protection. A service type field of '0x1' or '0x2' can indicate kernel driver services. Services are also installed with the use of some remoting tools, such as PSExec.
4703	Domain Controllers	This event is generated when a user right is adjusted. The addition of the 'SeDebugPrivilege' privilege, or other sensitive privileges such as 'SeTCBPrivilege', for an account may indicate attempts to deploy a Skeleton Key.

Detecting Active Directory compromises with canaries

Detecting Active Directory compromises can be difficult, time consuming and resource intensive, even for organisations with mature security information and event management (SIEM) and security operations centre (SOC) capabilities. This is because many Active Directory compromises exploit legitimate functionality and generate the same events that are generated by normal activity. Distinguishing malicious activity from normal activity often requires correlating different events, sometimes from different sources, and analysing these events for discrepancies. For some Active Directory compromises, the detection relies on the presence of one event and the absence of another. The complexity of detecting Active Directory compromises is one of the leading causes of their success and their prevalence against organisations.

The use canary objects in Active Directory is an effective technique to detect Active Directory compromises. The benefit of this technique is that it does not rely on correlating event logs, providing a strong indication a compromise has happened. Notably, this technique does not rely on detecting the tooling used by malicious actors (like some other detection techniques do), but instead detects the compromise itself. As such, it is more likely to accurately detect compromises against Active Directory. There are both open source and commercially available tools, such as the canaries developed by Airbus, that use this and similar techniques to detect Active Directory compromises.

Canary objects can be configured to deny read access to all user objects via the Everyone security group. This, combined with configuring the Directory Service Access audit policy to both 'Success' and 'Failure', means that when anyone attempts to read the properties of one of the canary objects, an audit failure event (event 4662) is generated. This event can be ingested into a SIEM configured with the Globally Unique Identifier (GUID) of the canary objects to alert on any corresponding events containing this GUID. This provides a high value alert that an attempt to enumerate one of these canary objects has occurred, which is indicative of a compromise against Active Directory.

Any compromise against Active Directory that enumerates objects in the domain can be detected using this technique. This is important as most compromises against Active Directory start with enumerating all objects in a domain using a tool, such as SharpHound, which collects information from Active Directory. Malicious actors employ this tactic to identify misconfigurations, weaknesses and vulnerabilities that can be exploited to escalate privileges and move laterally. This type of enumeration is detected by this technique and can provide an early warning to organisations that an Active Directory compromise is underway.

The following Active Directory compromises can be detected using this technique.

- Kerberoasting
- AS-REP Roasting
- DCSync.

A limitation of this technique is that malicious actors may choose to only target a single or a small number of user objects. If so, they are unlikely to try and read the canary objects. As a result, this technique would not generate the desired audit failure event and, subsequently, would not be detected by the SIEM.

Appendix A – Active Directory security controls

Table 17 shows a checklist representing the mitigations for each of the Active Directory compromises detailed in this guidance.

Table 17. Active Directory security controls checklist

Mitigating Kerberoasting	
☐	Minimise the number of user objects configured with SPNs.
☐	Create user objects with SPNs as gMSAs. However, if creating user objects with SPNs as gMSAs is not feasible, set a minimum 30-character password that is unique, unpredictable and managed.
☐	Assign user objects with SPNs to the minimum privileges necessary to perform their functions and make sure they are not members of highly privileged security groups, such as the Domain Admins security group.
Mitigating AS-REP Roasting	
☐	Ensure user objects require Kerberos pre-authentication. However, if user objects must be configured to bypass Kerberos pre-authentication, then these user objects should be granted the minimum set of privileges required for them to perform their functions. They should not be members of highly privileged security groups, such as Domain Admins. Additionally, set a minimum 30-character password that is unique, unpredictable and managed.
Mitigating password spraying	
☐	Create passwords for local administrator accounts, service accounts, and break glass accounts that are long (30-character minimum), unique, unpredictable and managed.
☐	Create passwords used for single-factor authentication that consist of at least four random words with a total minimum length of 15-characters.
☐	Lock out user objects, except for break glass accounts, after a maximum of five failed logon attempts.
☐	Ensure passwords created for user objects are randomly generated, such as when a user object is created, or a user requests a password reset.
☐	Configure the built-in ‘Administrator’ domain account as sensitive to ensure it cannot be delegated.
☐	Scan networks at least monthly to identify any credentials that are being stored in the clear.
☐	Disable the NTLM protocol.
Mitigating a MachineAccountQuota compromise	

☐	Configure unprivileged user objects so they cannot add computer objects to the domain.
☐	Ensure the Domain Computers security group is not a member of privileged security groups.
☐	Ensure the Domain Computers security group does not have write privileges to any objects in Active Directory.
☐	Enable LDAP signing for Domain Controllers.
Mitigating an unconstrained delegation compromise	
☐	Ensure computer objects are not configured for unconstrained delegation.
☐	Ensure privileged user objects are configured as ‘sensitive and cannot be delegated’.
☐	Ensure privileged user objects are members of the Protected Users security group.
☐	Disable the Print Spooler service on Domain Controllers.
Mitigating a password in GPP compromise	
☐	Remove all GPP passwords.
☐	Apply Microsoft’s security patch 2962486 to remove the functionality to create cpasswords.
Mitigating an AD CS compromise	
☐	Remove the ‘Enrollee Supplies Subject’ flag.
☐	Restrict standard user object permissions on certificate templates.
☐	Remove vulnerable AD CS CA configurations.
☐	Require CA Certificate Manager approval for certificate templates that allow the SAN to be supplied.
☐	Remove EKUs that enable user authentication.
☐	Limit access to AD CS CA servers to only privileged users that require access.
☐	Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.

☐	Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.
☐	Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.
☐	Centrally log and analyse AD CS CA server logs in a timely manner to identify malicious activity.
Mitigating a Golden Certificate	
☐	Use MFA to authenticate privileged users of systems.
☐	Implement application control on AD CS CAs.
☐	Use a HSM to protect key material for AD CS CAs.
☐	Limit access to AD CS CAs to only privileged users that require access.
☐	Restrict privileged access pathways to AD CS CA servers to jump servers and secure admin workstations using only the ports and services that are required for administration.
☐	Only use AD CS CA servers for AD CS and do not install any non-security-related services or applications.
☐	Encrypt and securely store backups of AD CS CA servers and limit access to only Backup Administrators.
☐	Centrally log and analyse AD CS CA logs in a timely manner to identify malicious activity.
Mitigating DCSync	
☐	Minimise the number of user objects with DCSync permissions.
☐	Ensure user objects that are configured with a SPN do not have DCSync permissions.
☐	Ensure user objects with DCSync permissions cannot log on to unprivileged operating environments.
☐	Review user objects with DCSync permissions every 12 months to determine if these permissions are still required.
☐	Disable the NTLMv1 protocol.
☐	Ensure LM password hashes are not used.
Mitigating dumping ntds.dit	

☐	Limit access to Domain Controllers to only privileged users that require access.
☐	Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.
☐	Encrypt and securely store backups of Domain Controllers and limit access to only Backup Administrators.
☐	Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.
☐	Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity
☐	Disable the Print Spooler service on Domain Controllers.
☐	Disable the SMB version 1 protocol on Domain Controllers.
Mitigating a Golden Ticket	
☐	Change the KRBTGT password every 12 months, or when the domain has been compromised or suspected to have been compromised.
Mitigating a Silver Ticket	
☐	Create User objects with SPNs as group Managed Service Accounts (gMSAs).
☐	Change all computer object (including Domain Controller) passwords every 30 days.
☐	Ensure computer objects are not members of privileged security groups, such as the Domain Admins security group.
☐	Ensure the Domain Computers security group does not have write or modify permissions to any objects in Active Directory.
Mitigating a Golden SAML	
☐	Ensure the AD FS service account is a gMSA.
☐	Ensure the AD FS service account is used only for AD FS and no other purpose.
☐	Ensure passwords for AD FS server local administrator accounts are long (30-character minimum), unique, unpredictable and managed.
☐	Limit access to AD FS servers to only privileged users that require access.

☐	Restrict privileged access pathways to AD FS servers to jump servers and secure admin workstations using only the ports and services that are required.
☐	Only use AD FS servers for AD FS and ensure no other non-security-related services or applications are installed.
☐	Centrally log and analyse AD FS server logs in a timely manner to identify malicious activity.
☐	Encrypt and securely store backups of AD FS servers and limit access to only Backup Administrators.
☐	Rotate AD FS token-signing and encryption certificates every 12 months, or sooner if an AD FS server has been compromised or suspected to have been compromised.
Mitigating a Microsoft Entra Connect compromise	
☐	Disable hard match takeover.
☐	Disable soft matching.
☐	Do not synchronise privileged user objects from AD DS to Microsoft Entra ID. Use separate privileged accounts for AD DS and Microsoft Entra ID.
☐	Enable MFA for all privileged users in Microsoft Entra ID.
☐	Limit access to Microsoft Entra Connect servers to only privileged users that require access.
☐	Restrict privileged access pathways to Microsoft Entra Connect servers to jump servers and secure admin workstations using only the ports and services that are required for administration.
☐	Ensure passwords for Microsoft Entra Connect server local administrator accounts are long (30-character minimum), unique, unpredictable and managed.
☐	Only use Microsoft Entra Connect servers for Microsoft Entra Connect and ensure no other non-security-related services or applications are installed.
☐	Encrypt and securely store backups of Microsoft Entra Connect and limit access to only Backup Administrators.
☐	Centrally log and analyse Microsoft Entra Connect server logs in a timely manner to identify malicious activity.
Mitigating a one-way domain trust bypass	
☐	Limit access to Domain Controllers to only privileged users that require access.

☐	Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.
☐	Encrypt and securely store backups of Domain Controllers and limit access to only Backup Administrators.
☐	Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.
☐	Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.
☐	Disable the Print Spooler service on Domain Controllers.
Mitigating a SID History compromise	
☐	Ensure the 'sIDHistory' attribute is not used.
☐	Ensure the 'sIDHistory' attribute is checked weekly.
☐	Enable SID Filtering for domain and forest trusts.
Mitigating Skeleton Key	
☐	Limit access to Domain Controllers to only privileged users that require access.
☐	Restrict privileged access pathways to Domain Controllers to jump servers and secure admin workstations using only the ports and services that are required for administration.
☐	Run the LSASS process in protected mode.
☐	Implement Microsoft's vulnerable driver blocklist.
☐	Restrict driver execution to an approved set.
☐	Only use Domain Controllers for AD DS and do not install any non-security-related services or applications.
☐	Centrally log and analyse Domain Controller logs in a timely manner to identify malicious activity.
☐	Disable the Print Spooler service on Domain Controllers.

Appendix B – Active Directory events

Table 18 through **Table 23** contain the recommended event IDs to log and monitor to detect the Active Directory compromises detailed in this guidance.

The below tables contain the recommended events to log and monitor to detect the Active Directory compromises detailed in this guidance. Some of the events in this guidance are not logged as part of the default Windows Audit Policy and additional configuration is required to log these events. For more information, see ASD's Windows Event Logging and Forwarding guidance.

Domain Controller events

The events in **Table 18** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving Domain Controllers.

Table 18. Events that detect compromises involving Domain Controllers

Event ID	Compromise	Description
39	AD CS	The KDC encountered a user certificate that was valid but could not be mapped to a user in a secure way (such as via explicit mapping, key trust mapping or a SID).
40	AD CS	A certificate is issued before the user existed in Active Directory, and no explicit mapping could be found. This event is only logged when the KDC is in Compatibility mode.
41	AD CS	A certificate contains the new SID extension, but it does not match the SID of the corresponding user account.
1102	Dumping ntds.dit, One-way Trust Bypass, SID History, Skeleton Key	The 'Security' audit log is cleared.
2889	Password Spray	A computer object tries to make an unsigned LDAP bind.
3033	Skeleton Key	A driver fails to load because it does not meet Microsoft's signing requirements.
3063	Skeleton Key	A driver fails to load because it does not meet the security requirements for shared sections.
4103	Dumping ntds.dit, One-way Trust Bypass, SID History, Skeleton Key	PowerShell executes and logs pipeline execution details.
4104	Dumping ntds.dit, One-way Trust	PowerShell executes code to capture scripts and commands.

	Bypass, SID History, Skeleton Key	
4624	Password Spray, MachineAccountQuota, Unconstrained Delegation	An account is successfully logged on.
4625	AS-REP Roasting, Password Spray	An account fails to log on.
4656	Dumping ntds.dit	A handle to an object is requested.
4662	DCSync, Golden SAML	An operation is performed on an object.
4663	Dumping ntds.dit, Skeleton Key	An attempt is made to access an object.
4673	Skeleton Key	A privileged service is called.
4674	AD CS	An operation is attempted on a privileged object.
4675	SID History (Domain hopping with Golden Tickets and SID History)	SIDs were filtered.
4688	Dumping ntds.dit	A new process is created.
4697	Skeleton Key	A service is installed in the system.
4703	Skeleton Key	A user right is adjusted.
4724	MachineAccountQuota	An attempt is made to reset an account's password.
4738	Kerberoasting, AS-REP Roasting, SID History	A user account is changed.
4740	Password Spray	A user account is locked out.
4741	MachineAccountQuota	A computer account was created in Active Directory.
4768	AS-REP Roasting, AD CS, Golden	A Kerberos TGT is requested.

	Ticket, One-way Trust Bypass	
4769	Kerberoasting, Golden Ticket	A TGS is requested.
4770	Unconstrained Delegation	A Kerberos TGT is renewed.
4771	Password Spray	Kerberos pre-authentication fails.
5136	Kerberoasting, AS-REP Roasting	A directory service object was modified.
8222	Dumping ntds.dit	A shadow copy is created.

Active Directory Certificate Services Certificate Authority (AD CS CA) events

The events in **Table 19** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving AD CS CA servers.

Table 19. Events that detect compromises involving AD CS CA servers

Event ID	Compromise	Description
1102	AD CS, Golden Certificate	The 'Security' audit log was cleared.
4103	Golden Certificate	PowerShell module logging.
4104	Golden Certificate	PowerShell script block logging.
4876	Golden Certificate	Certificate Services backup was started.
4886	AD CS	Certificate Services received a certificate request.
4887	AD CS	Certificate Services approved a certificate request and issued a certificate.
4899	AD CS	A Certificate Services template was updated.
4900	AD CS	Certificate Services template security was updated.

Active Directory Federation Services (AD FS) events

The events in **Table 20** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving AD FS servers.

Table 20. Events that detect compromises involving AD FS servers

Event ID	Compromise	Description
70	Golden SAML	A Certificate Private Key was acquired.
307	Golden SAML	The Federation Service configuration was changed.
510	Golden SAML	Additional information about events such as federation service configuration changes was requested.
1007	Golden SAML	A certificate was exported.
1102	Golden SAML	The 'Security' audit log was cleared.
1200	Golden SAML	The Federation Service issued a valid token.
1202	Golden SAML	The Federation Service validated a new credential.

Microsoft Entra Connect server events

The events in **Table 21** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving Microsoft Entra Connect servers.

Table 21. Events that detect compromises involving Microsoft Entra Connect servers

Event ID	Compromise	Description
611	Microsoft Entra Connect	PHS failed for the domain.
650	Microsoft Entra Connect	Password synchronisation starts retrieving updated passwords from the on-premises AD DS.
651	Microsoft Entra Connect	Password synchronisation finishes retrieving updated passwords from the on-premises AD DS.
656	Microsoft Entra Connect	Password synchronisation indicates that a password change was detected and there was an attempt to sync it to Microsoft Entra ID.
657	Microsoft Entra Connect	A password was successfully synced for a user object.
1102	Microsoft Entra Connect	The security audit log was cleared.
4103	Microsoft Entra Connect	PowerShell module logging.

4104	Microsoft Entra Connect	PowerShell script block logging.

Computer objects configured for unconstrained delegation events

The events in **Table 22** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving computer objects configured for unconstrained delegation.

Table 22. Events that detect compromises involving computer objects configured for unconstrained delegation

Event ID	Compromise	Description
4103	Unconstrained delegation	PowerShell executes and logs pipeline execution details.
4104	Unconstrained delegation	PowerShell executes code to capture scripts and commands.
4624	Unconstrained delegation	An account is successfully logged on.
4688	Unconstrained delegation	A new process is created.

Computer objects compromised by a Silver Ticket

The events in **Table 23** should be centrally logged and analysed in a timely manner to identify Active Directory compromises involving Silver Tickets.

Table 23. Events that detect Silver Ticket compromises

Event ID	Compromise	Description
4624	Silver Ticket	This event is generated when an account is logged into a computer. It can be correlated and analysed with event 4627 for signs of a potential Silver Ticket.
4627	Silver Ticket	This event is generated alongside event 4624 and provides additional information regarding the group membership of the account that logged in. This event can be analysed for discrepancies, such as mismatching SID and group membership information for the user object that logged on. Note that a Silver Ticket forges the TGS, which can contain false information, such as a different SID to the user object logging on and different group memberships. Malicious actors falsify this information to escalate their privileges on the target computer object.