Precious Gemstones: The New Generation of Kerberos Attacks

unit42.paloaltonetworks.com/next-gen-kerberos-attacks/

Oz Soprin, Shachar Roitman

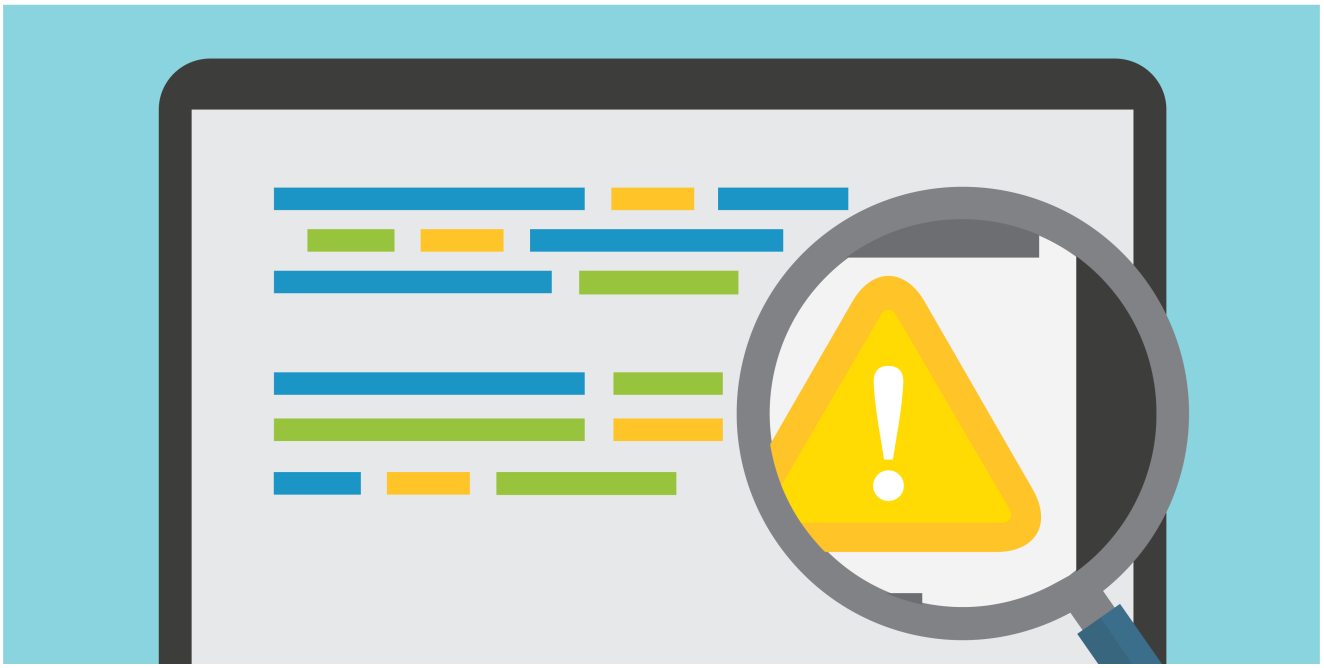
December 12, 2022

By Oz Soprin and Shachar Roitman

December 12, 2022 at 6:00 AM

Category: Threat Briefs and Assessments

Tags: Cortex XDR, Diamond Ticket, Golden Ticket, kerberos, privilege escalation, Sapphire Ticket



This post is also available in: 日本語 (Japanese)

Executive Summary

Unit 42 researchers show new detection methods that help improve detection of a new line of Kerberos attacks, which allow attackers to modify Kerberos tickets to maintain privileged access. The most well-known example of this is the Golden Ticket attack, which allows threat actors to forge a ticket to masquerade as a high-privileged user.

These two newer attacks extend the Golden Ticket attack in that the forged tickets are not created from scratch, but instead based on modifying an existing ticket to include high-privileged access. We'll discuss the difference between these three types of attacks, to explain why the newer ones are harder to detect.

The broad usage of Active Directory has made Kerberos attacks the bread and butter of many threat actors. Researchers have discovered the following new attack techniques that allow an adversary to gain unconstrained access to all services and resources within an Active Directory (AD) domain:

- Diamond Ticket
- Sapphire Ticket

Because of their similarity to the well-known Golden Ticket attack, threat actors might also use these attacks in future campaigns. People can better protect themselves by employing the new detection methods we'll discuss for these new Kerberos attacks.

Palo Alto Networks customers receive improved detection for the attacks discussed in this blog through Cortex XDR.

Related Unit 42 Topics privilege escalation, Golden Ticket, Kerberos

Table of Contents

Kerberos Refresher

To understand the ticket attacks and their implications, it helps to understand a few things about how Kerberos works. This includes some common terms for features used in these attacks, as well as the structure of how tickets are used.

Kerberos is a network authentication protocol that is primarily used in Active Directory (AD) environments. Thousands of companies across different industries use Active Directory technology for managing user accounts and other resources within an organization. Active Directory's first version was released in Windows Server 2000 and since then, it has become particularly common in businesses and other large organizations that have a significant number of users and resources to manage.

Kerberos provides strong authentication by issuing tickets to authenticate users and allow access to services. The tickets are distributed by the key distribution center (KDC). In most environments, the KDC is installed on the domain controller (DC).

During the initial authentication, a Ticket Granting Ticket (TGT) is a ticket assigned to a user. The TGT is later used to authenticate the user to the KDC and request a service ticket from the Ticket Granting Service (TGS). Service tickets are granted for authentication against services.

A Kerberos authentication would consist of the following steps:

1. The user requests (AS-REQ) a TGT from the KDC and the KDC verifies and validates the credentials and user information.
2. After authenticating the user, the KDC sends an encrypted TGT back to the requester (AS-REP).
3. The user presents the TGT to the DC and requests a TGS (TGS-REQ).
4. The TGS is encrypted and sent back to the requesting user (TGS-REP).
5. The user connects to the server hosting the service requested and presents the TGS (AP-REQ) in order to access the service.
6. The application server sends an (AP-REP) to the client.

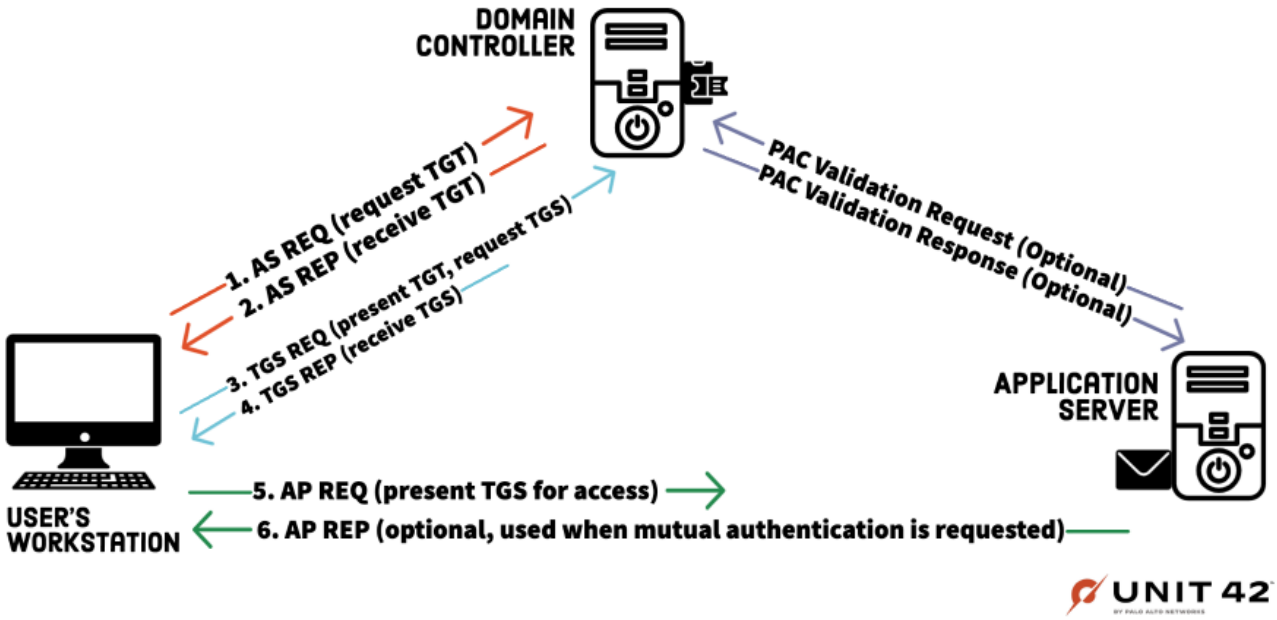


Figure 1. Kerberos authentication.

What Components Are in a Kerberos Ticket?

Each Kerberos ticket contains the following fields:

Kerberos Ticket	
Tkt-vno	Version number for the ticket format.
Realm	The realm that issued a ticket.
sname	Server name. All the components of the name are part of the server's identity.
Enc-part	Encrypted with the server's secret key.

Table 1. Kerberos ticket.

The enc-part contains different fields but we will focus on the following:

1. cname – The name part of the client's principal identifier.
2. authorization-data – used to pass authorization data from the principal on whose behalf a ticket was issued to the application service. This part includes the Privileged Attribute Certificate (PAC).

What Is a Kerberos Privilege Attribute Certificate (PAC)?

As Microsoft's documentation states, the PAC is a structure that conveys authorization-related information provided by DCs. The PAC is used by authentication protocols that verify identities to transport authorization information, which controls access to resources.

The DC includes authorization data such as security identifiers (SIDs) and relative identifiers (RIDs) in the PAC.

Kerberos Delegation

A common use case for Kerberos delegation is a web server fetching user data from a database server. The database server can grant access to the user data only for the user. In this scenario, the web server will need to impersonate the user. This impersonation is called Kerberos delegation.

We will focus on constrained delegation in this blog, but you can read about the different Kerberos delegation methods in Protecting Against the Bronze Bit Vulnerability with Cortex XDR.

Constrained Delegation: Microsoft has implemented Kerberos extensions in order to avoid keeping the user's TGT in memory. This extension is called S4U (Service for User), and it consists of S4U2Self and S4U2Proxy. S4U2Self allows a service to ask for a ticket to itself on behalf of any user. S4U2Proxy will allow a service to authenticate to a different service.

S4U2Self Extension

As described above, the S4U2self extension allows a service to receive the user's authorization data in the ticket (i.e., the PAC).

To use this protocol extension, the user that issues the KRB_TGS_REQ must have at least one Service Principal Name (SPN) to allow the DC to encrypt the generated service ticket with the service secret key.

S4U2U KRB_TGS exchange flow:

1. The service fills out the PA_FOR_USER data structure, which contains the information about the user on whose behalf the service requests the service ticket, and sends the KRB_TGS_REQ message to the TGS.
2. The service ticket will be sent back to the service via KRB_TGS_REQ message. The PAC returned in the service ticket contains the authorization data.

U2U Authentication

User-to-User authentication is described in the Kerberos RFC as a method that "allows the client to request that the ticket issued by the KDC be encrypted using a session key from a TGT issued to the party that will verify the authentication."

The KRB_TGS_REQ will have the following features:

- additional-tickets: will contain the TGT from which the secret-key is taken
- ENC-TKT-IN-SKEY: option indicates that the ticket for the end server is to be encrypted in the session key from the additional TGT provided.
- The service name (sname) can refer to a user and not necessarily a service with SPN.

U2U + S4U2Self

Combining the two methods together allows using the S4U2Self extension for users with no SPN. The service ticket received in this exchange is encrypted with the server's secret-key (in this specific case the server can be a user with no SPN), therefore, the PAC of the target user can be decrypted using the server's secret-key.

The KRB_TGS_REQ packet will have all the features of both methods.

How Do These Kerberos Attacks Work?

Both the Sapphire and Diamond Ticket attacks decrypt a legitimate TGT and change its PAC, and in order to do that, the adversary needs to have access to the KRBTGT account's key (the password hash). The KRBTGT account, as described by Microsoft TechNet (now Microsoft Docs), is a local default account that acts as a service account for the KDC service.

Sapphire Ticket

The Sapphire ticket attack (introduced by Charlie Bromberg), requires getting credentials of any user in the domain. Let's call that user Joe. Joe's credentials will be used to obtain a TGT (using a regular KRB_AS_REQ) and later to decrypt the PAC of a high-privileged user.

Once the TGT is received, the adversary will use the U2U + S4U2Self:

1. Decide on the user they want to impersonate (high-privileged user)
2. Generate a KRB_TGS_REQ with the following attributes:
    1. The PA_FOR_USER struct contains the impersonated user
    2. The service name (sname) is Joe's username
    3. Joe's TGT will be added to the additional-tickets field
    4. ENC-TKT-IN-SKEY flag in the KDC Option is set
3. Obtain a service ticket for the impersonated user

KRB_TGS_REQ -

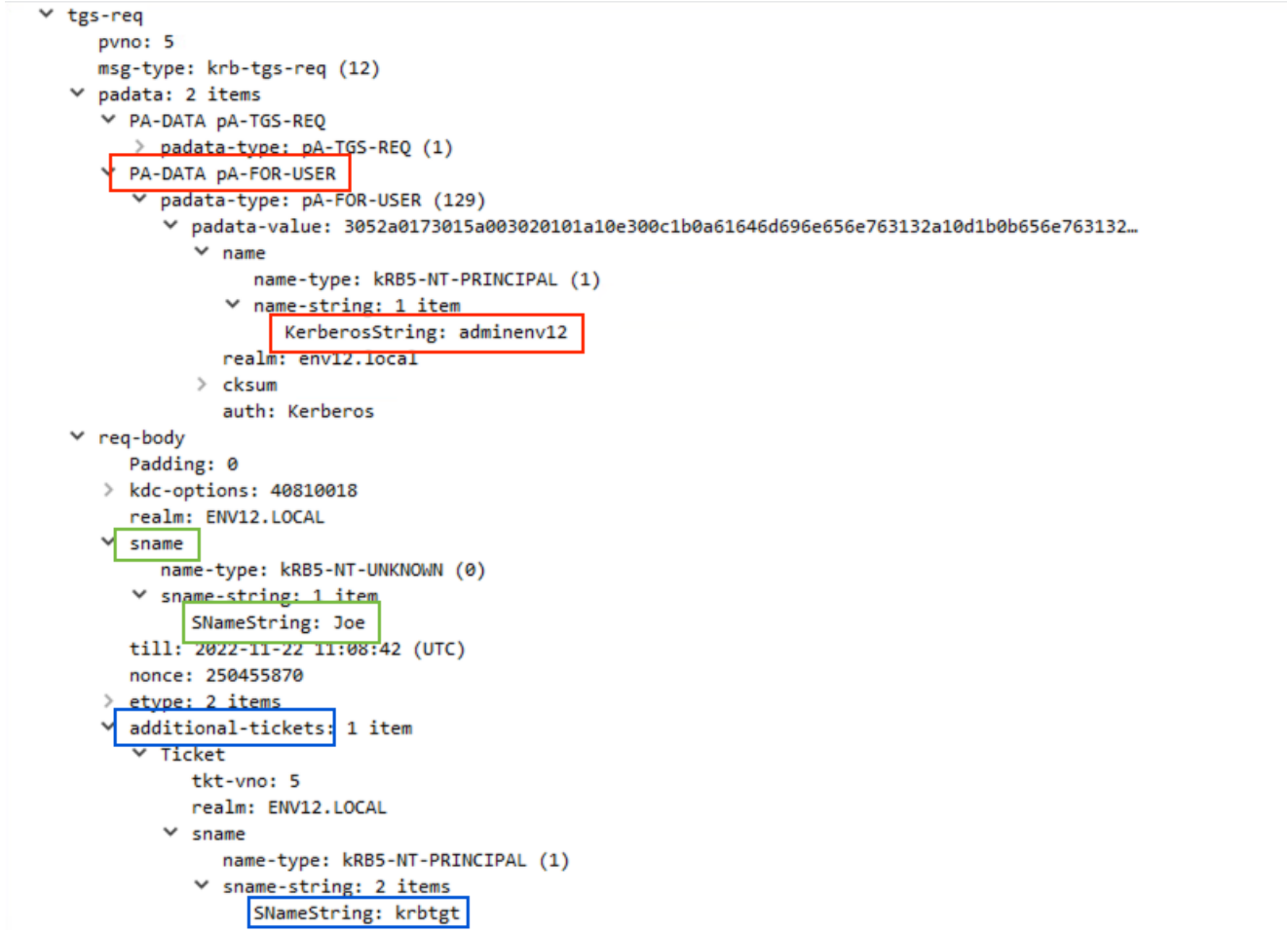


Figure 2. KRB_TGS_REQ created using U2U + S4U2Self methods.

```
kdc-options: 40810018
    0... .... = reserved: False
    .1.. .... = forwardable: True
    ..0. .... = forwarded: False
    ...0 .... = proxiable: False
    .... 0... = proxy: False
    .... .0.. = allow-postdate: False
    .... ..0. = postdated: False
    .... ...0 = unused7: False
    1... .... = renewable: True
    .0.. .... = unused9: False
    ..0. .... = unused10: False
    ...0 .... = opt-hardware-auth: False
    .... 0... = unused12: False
    .... .0.. = unused13: False
    .... ..0. = constrained-delegation: False
    .... ...1 = canonicalize: True
    0... .... = request-anonymous: False
    .0.. .... = unused17: False
    ..0. .... = unused18: False
    ...0 .... = unused19: False
    .... 0... = unused20: False
    .... .0.. = unused21: False
    .... ..0. = unused22: False
    .... ...0 = unused23: False
    0... .... = unused24: False
    .0.. .... = unused25: False
    ..0. .... = disable-transited-check: False
    ...1 .... = renewable-ok: True
    .... 1... = enc-tkt-in-skey: True
    .... .0.. = unused29: False
    .... ..0. = renew: False
    .... ...0 = validate: False
```

Figure 3. ENC-TKT-IN-SKEY flag is set in the KDC Option.

KRB_TGS_REP

```
tgs-rep
    pvno: 5
    msg-type: krb-tgs-rep (13)
    crealm: env12.local
    cname
        name-type: kRB5-NT-PRINCIPAL (1)
        cname-string: 1 item
            CNameString: adminenv12
    ticket
        tkt-vno: 5
        realm: ENV12.LOCAL
        sname
            name-type: kRB5-NT-UNKNOWN (0)
            sname-string: 1 item
                SNameString: Joe
        enc-part
    enc-part
```

Figure 4. U2U + S4U2Self TGS-REP.

As we know, a service ticket is encrypted with the service's long-term secret, which means that if the adversary has Joe's credentials, they will be able to decrypt the service ticket. The adversary also has the KRBTGT account's secret key and therefore they can decrypt and modify Joe's TGT, which is signed with said secret key.

To forge the Sapphire Ticket, the attacker will extract the PAC of the impersonated user, and modify Joe's TGT in 2 ways:

1. Replace Joe's original PAC with the high-privileged PAC.
2. Match the cname to be the impersonated user's name.

This gives the attacker a usable TGT of a high-privileged user.

Diamond Ticket

The first part of the Diamond Ticket attack (introduced by Charlie Clark and Andrew Schwartz) is obtaining a TGT. This can be done by using one of the following techniques:

- Adversaries will use the low-privileged user's credentials to initiate a KRB_AS_REQ, which will result in a legitimate TGT.
- A tool called Rubeus has an option called tgtdeleg that abuses the Kerberos Generic Security Services Application Program Interface (GSS-API) to retrieve a usable TGT for the current user without needing elevation on the host.

Once receiving the TGT in the KRB_AS_REP, the adversary can now decrypt the TGT using the KRBTGT account's key and modify each part of the ticket.

After modifying the ticket, an attacker can escalate their privileges using one of the following approaches:

> Generating a TGT to impersonate a domain admin or any other user in the domain.

In this scenario the attacker will need to change the cname field in the ticket as well as the PAC. This method will result in a forged ticket under the name of the domain admin (or any other user in the domain), which is pretty similar to the Golden Ticket attack.

> Elevating a normal domain user's privileges to domain admin privileges by modifying the PAC.

This method will supply the attacker with a ticket under the name of the low-privileged user with domain admin permissions.

How Powerful Are the New Kerberos Attacks?

Diamond and Sapphire Tickets are forged TGTs created by modifying a legitimate TGT, which gives it additional privileges or a new identity. While many Golden Ticket detections are based on the absence of a TGT creation by a legitimate DC, the new attacks manipulate a legitimate TGT that was issued by the DC, which makes them harder to detect.

After the TGT is forged, the attacker uses it to request a TGS to any service or resource they desire. For example, they can request access to all computers, files and folders in the domain.

What's the Difference Between a Diamond Ticket and a Sapphire Ticket Attack?

In both attacks, the manipulation happens on the PAC of a legitimate TGT, but the main difference is in the way it is modified. With a Diamond Ticket, the modification is to the original PAC of the requested TGT, by adding additional privileges or modifying it completely. With a Sapphire Ticket, the attacker modifies the TGT by getting a legitimate PAC of a high-privileged user using Kerberos delegation, and replaces it with the original ticket's PAC.

Can Anyone Forge TGTs?

In order to perform such an attack, the basic requirement is that the adversary needs access to the KRBTGT account's password hash.

The KRBTGT account encrypts and signs all the Kerberos TGT tickets for the domain. All verification of Kerberos tickets, including encrypting and decrypting TGTs for the KDC service, is done by the KRBTGT. This means a forged TGT will be considered a valid ticket simply because it was encrypted with the KRBTGT account.

Moreover, the adversary has to encrypt the forged TGT with the KRBTGT password hash once the TGT is forged, to make sure the new TGT will pass the KDC's verification and issue a service ticket to the desired service.

With that being said, it's important to remember that getting the KRBTGT password hash is not an easy task. Doing so not only provides access to all the services and resources in the domain, but also enables network persistence.

Forged Ticket Attacks in the Wild

Forged ticket attacks have been sighted in the wild, such as in attacks by Playful Taurus, also known as APT15, Ke3chang and NICKEL. This group is attributed to actors operating out of China and has targeted oil, government, diplomatic, military and non-governmental organizations around the world since 2010.

An investigation done by NCC Group's IR team in May 2017 states that, to gain persistence in the victim's network in the event of remediation actions being undertaken, Playful Taurus used Mimikatz to dump credentials and generate Kerberos Golden Tickets.

In a different campaign around December 2021, MSTIC found that the same threat actor used malicious tools such as Mimikatz, WDigest, NTDSDump, and other password-dumping tools to gather credentials on a targeted system.

Mimikatz allows attackers to perform the following activities:

- Gathering credentials
- Performing DCSync attacks

These simulate the Active Directory replication process and retrieve the user's password hash. They can be used to collect the KRBTGT password hash.

Forge a Golden Ticket

Our assumption is that – because Sapphire and Diamond Ticket attacks resemble Golden Ticket attacks but are harder to discover – different threat actors (like Playful Taurus) are likely to use these attacks in the future.

Proposing Detection Methods for Forged Ticket Attacks

Due to the difficulty of finding the KRBTGT password hash, these attacks would probably cause a lot of noise even before the attack took place. In this case, the detection ideas that apply to Golden Ticket attacks are relevant here as well.

If you want to read more about such detections, check out our blog, Detecting and Preventing the Path to a Golden Ticket With Cortex XDR.

Sapphire Ticket

Since the Sapphire Ticket should appear valid (i.e., a ticket with a legitimate PAC), we would have a difficult time detecting its utilization solely by analyzing it. However, we can detect other suspicious activities on the host:

- Signs of KRBTGT password hash theft
- Suspicious tools usage
- DCSync attacks in the environment or suspicious connections from the host to a DC
- Irregular KRB_TGS_REQ with U2U + S4U2Self attributes
- KRB_TGS_REQ by a high-privileged user from that host

If the adversary uses the ticket right after performing the attack, we will observe a TGT request and TGS from the same IP for two different users. There would be no indication for the second user's (i.e., the one that appears in the TGS request) authentication to that machine. This can be monitored by Windows Event 4768 and 4769 as well as network traffic.

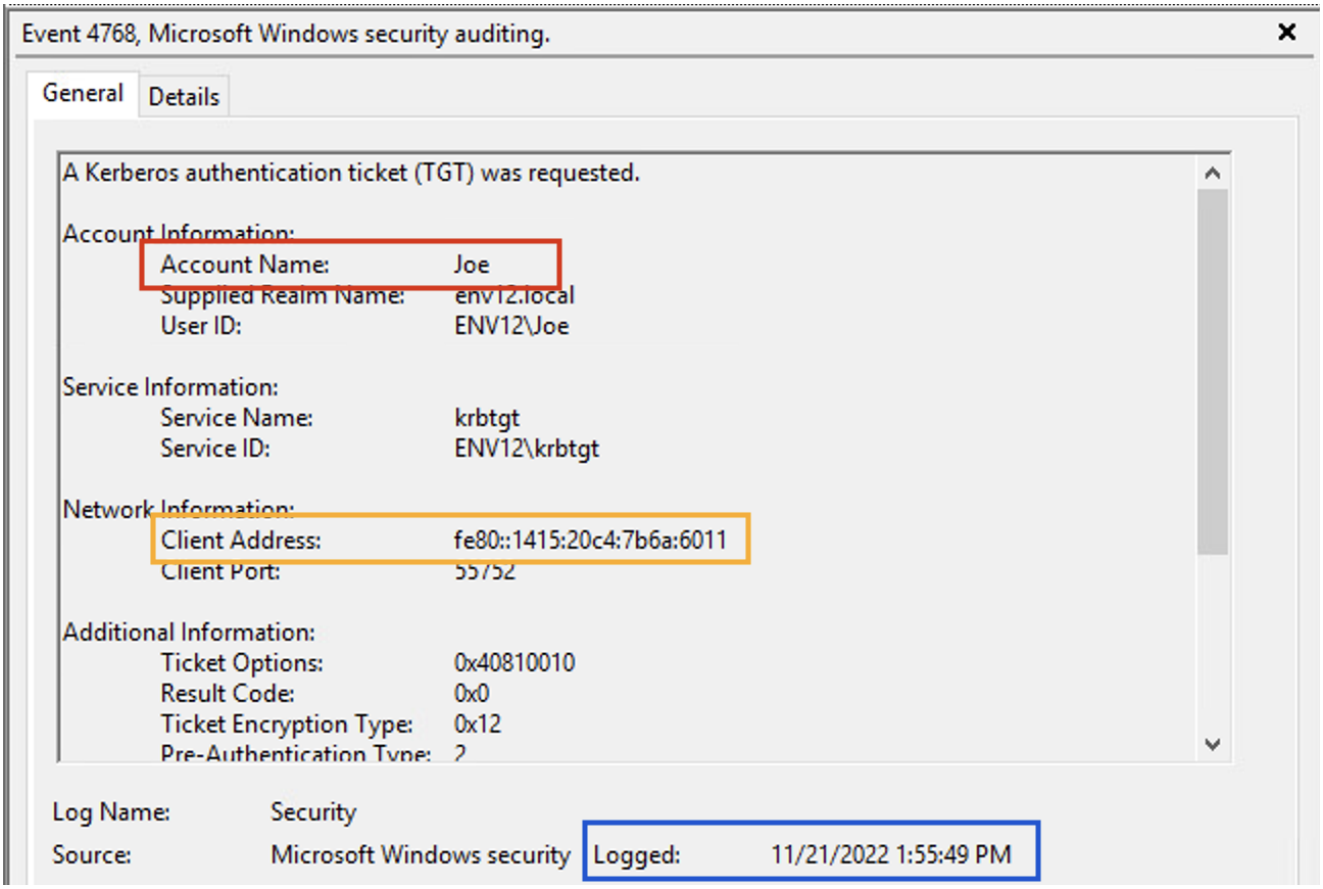


Figure 5. TGT request by low-privileged user.

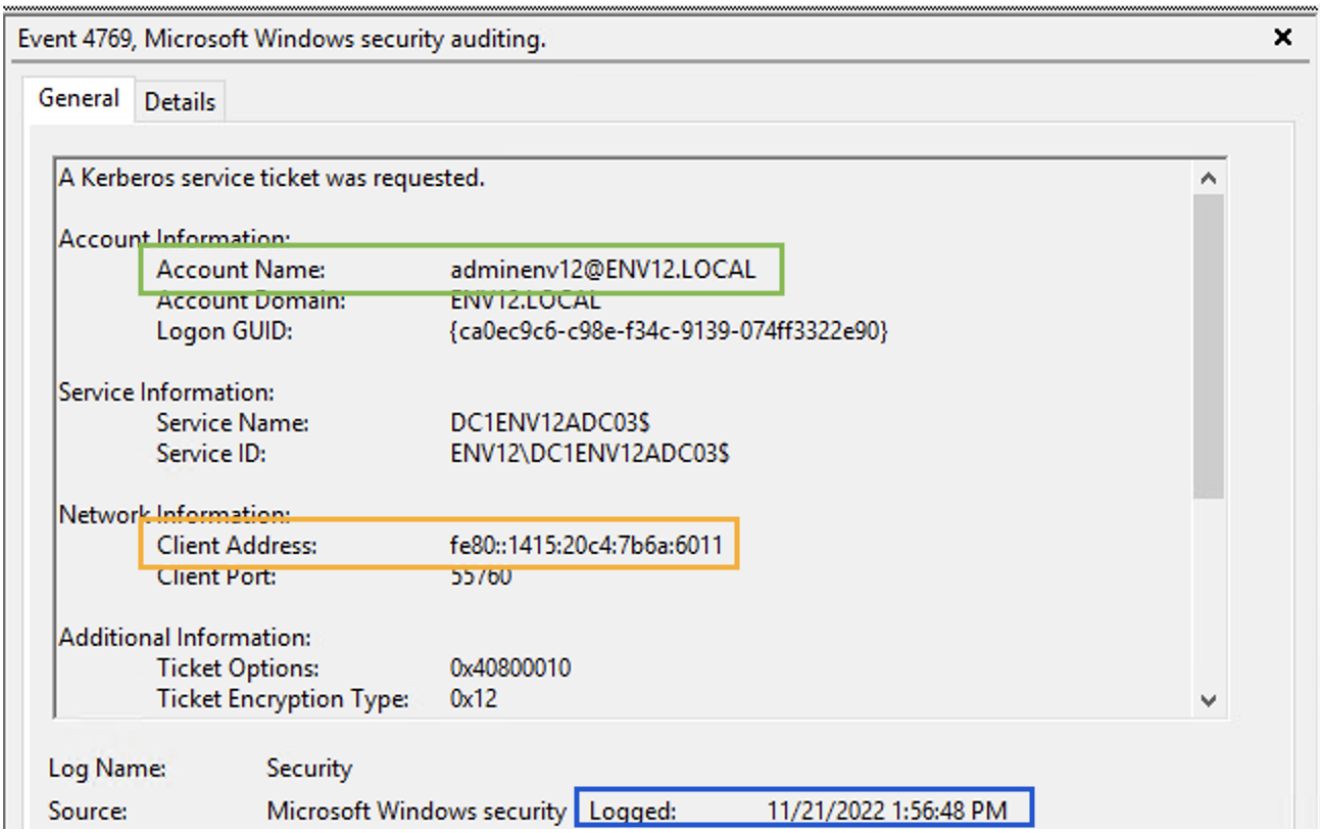


Figure 6. TGS request by high-privileged user.

Diamond Ticket

Generating a Ticket for Domain Admin Account

The detection here is pretty similar to any Golden Ticket detector or the suggested Sapphire Ticket detector.

Elevating Normal Domain User's Privileges

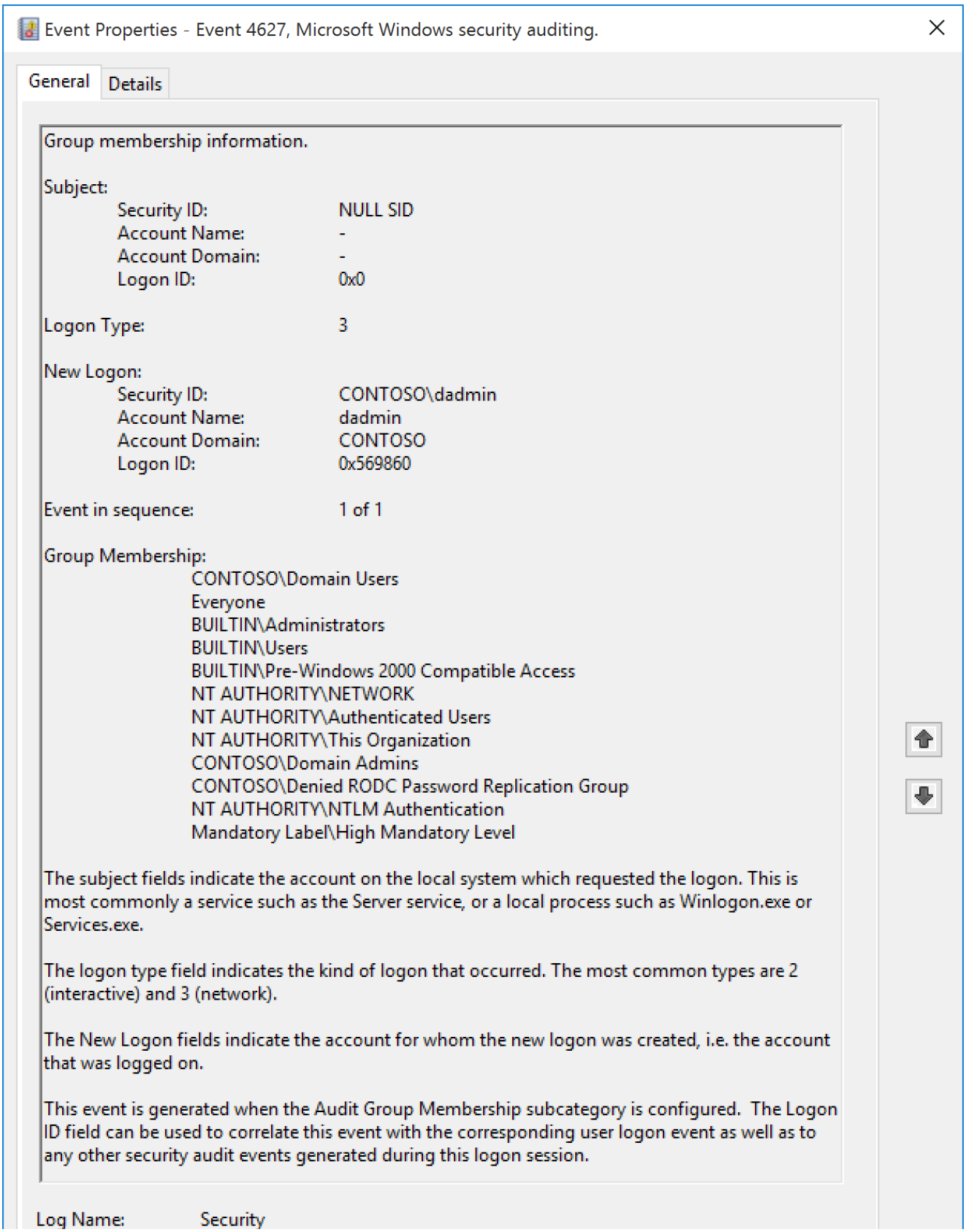
If the adversary uses this technique, it will look like legitimate authentication. The TGT and TGS requests will be for the same user, and the suggested detectors won't be relevant.

This activity can be monitored by looking for anomalies in both the number of resources being accessed and what resources the user accessed. Attackers will likely be accessing high-privileged resources, so this activity will be new. Anomaly detection can be tricky in big environments and can cause distraction rather than finding an attack.

It's important to note that the main difference between Diamond Ticket and Sapphire Ticket is that a Sapphire Ticket has an actual PAC of a real high-privileged user. In Diamond Ticket and Golden Ticket attacks, the PAC is modified by the attacker, and might therefore be inaccurate. This information can be used to detect these attacks.

Group Membership Detection

To make this detection method more approachable, let me introduce you to Windows event 4627(S): Group membership information. This event is generated with Windows event "4624(S): An account was successfully logged on," and it shows the list of groups that the logged-on account belongs to.



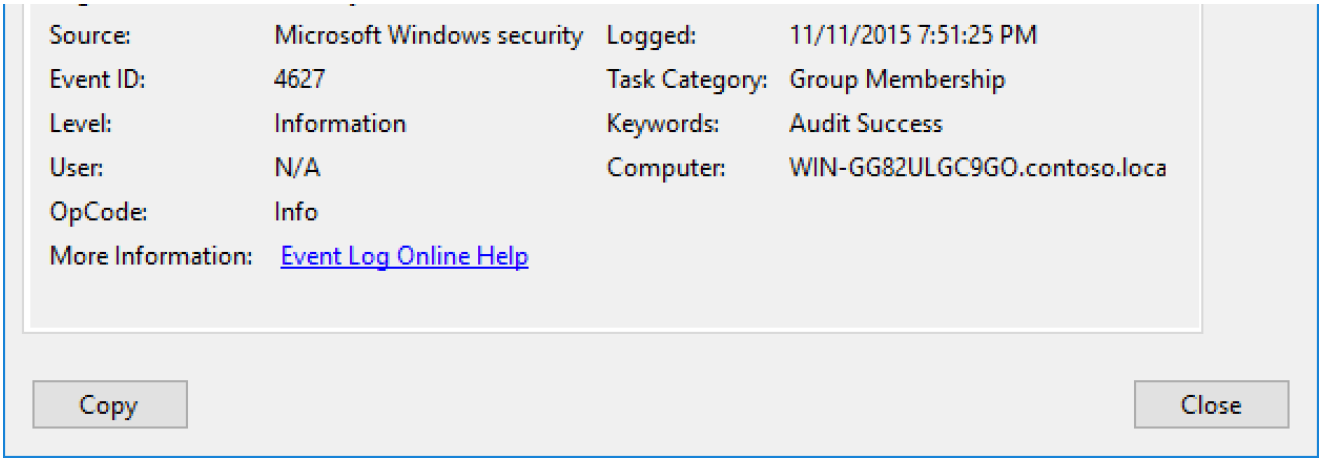


Figure 7. Windows Event 4627.
Source - Microsoft

This requires OS Version Windows Server 2016, Windows 10 and above. To enable it, you must enable the Success Audit for the Audit Logon subcategory.

Monitoring this event will allow you to discover irregularities in the user's group memberships. When an attacker wants to assign domain admin privileges to a forged ticket, they add the domain admin rid (512) to the PAC. Thus, during the login, it appears the user is part of the domain admins.

An example of this would be using Diamond Ticket attacks to elevate a low-privileged user (we'll continue to use "Joe") to domain admin privileges, as shown in Figures 8 and 9.

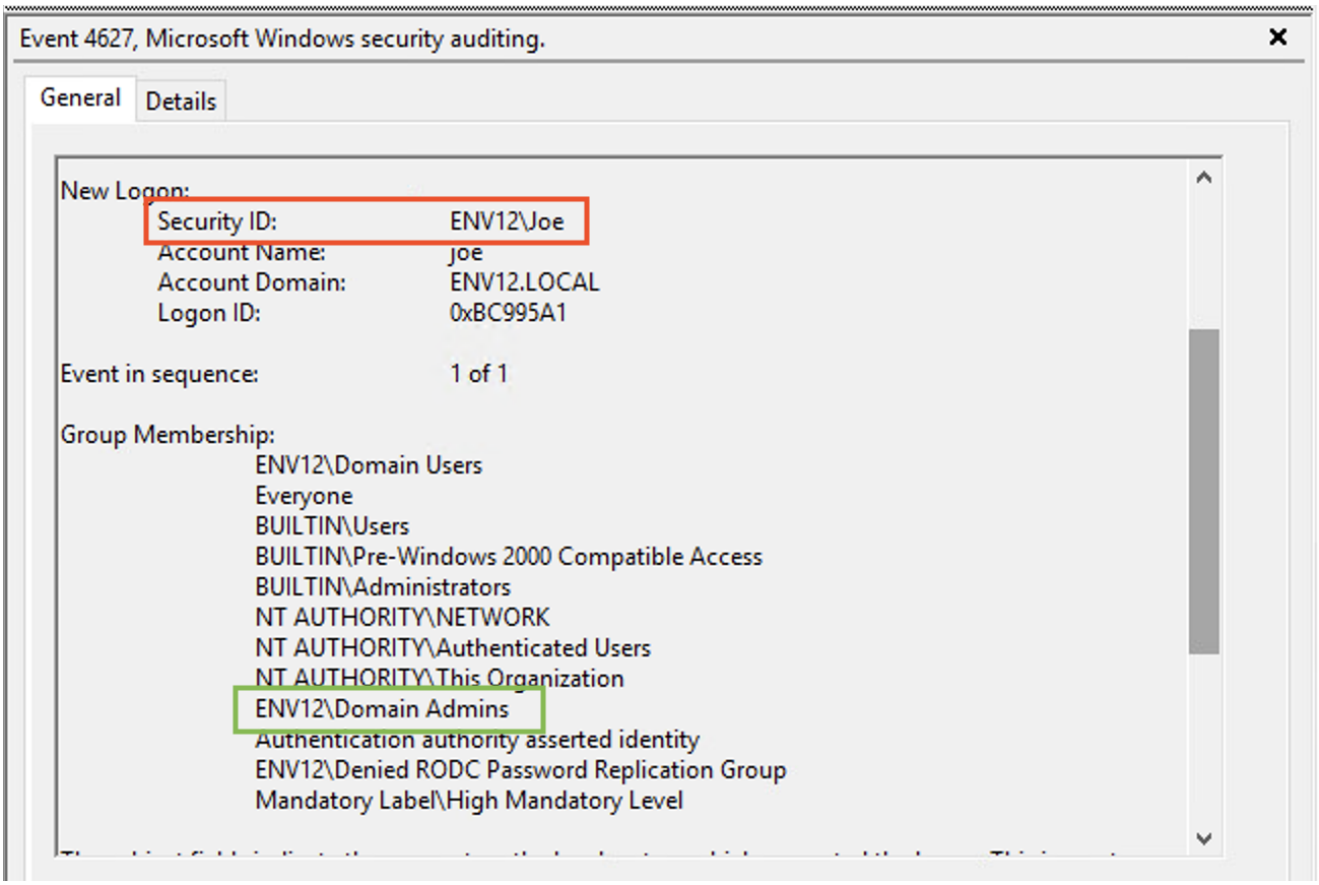


Figure 8. Low-privileged user appears in domain admin.

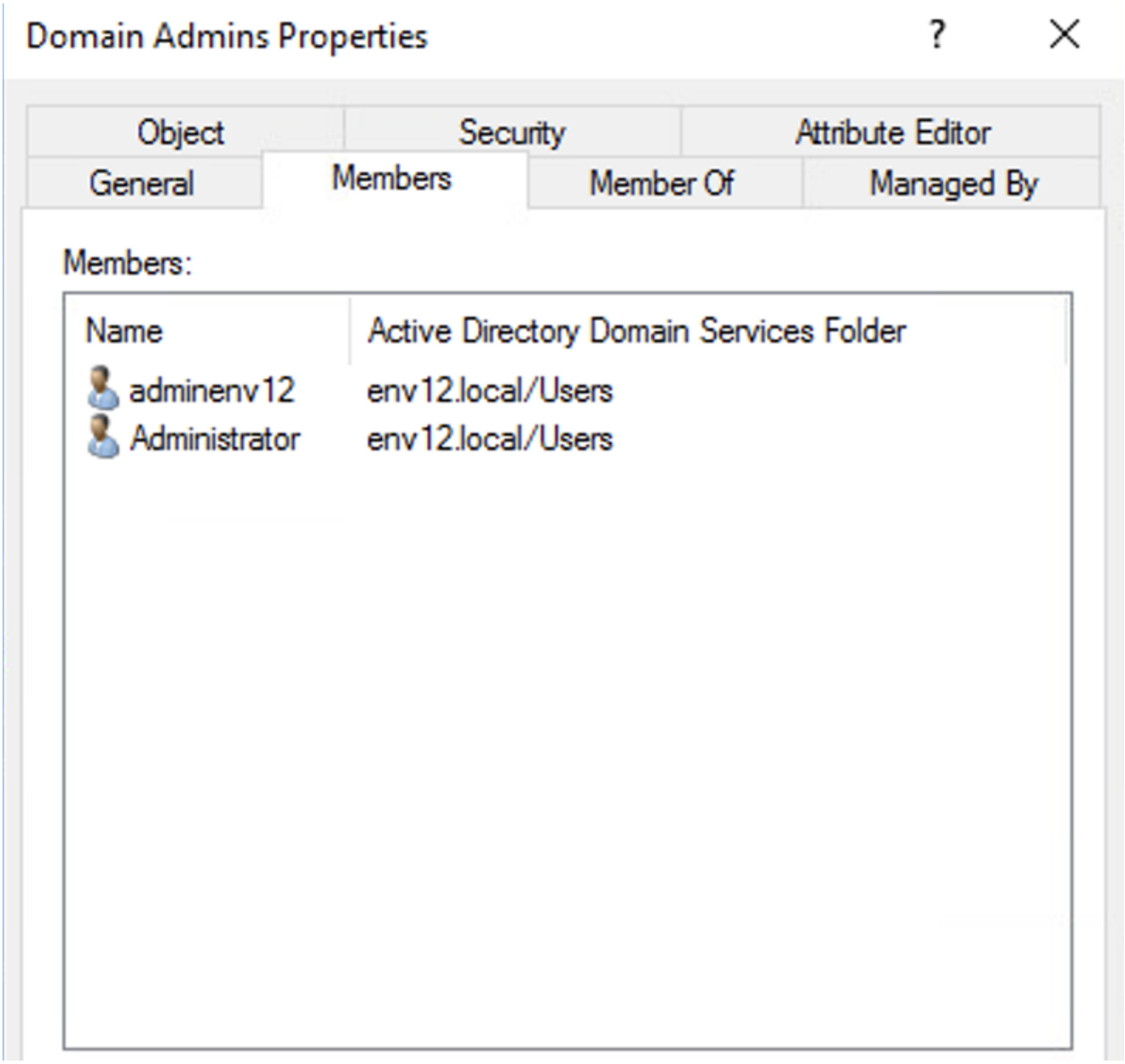


Figure 9. Joe is not a member of domain admins.

It is worth noting that, if an adversary impersonates a high-privileged user using a forged TGT, they might assign the wrong group membership. Therefore, it is worth paying attention to those as well.

To avoid false positives (when a user is intentionally added to a high-privileged group), event 4627 can be correlated with Windows events 4732 and 4728, which indicate when a user is added to a group.

Conclusion

In this blog, after a brief primer on relevant Kerberos terms and the attacks themselves, we discussed the privileges required to perform such attacks and the importance of monitoring different forged ticket attacks. Additionally, we examined possible detection ideas that might

help cover Golden Ticket attacks as well as new attack methods.

Forged ticket attacks might be hard to detect with a cursory glance, since they can initially appear to be legitimate. However, if enough information is collected about suspicious network activity, malicious tool usage, or Windows events, we might be able to detect some of the most effective Kerberos attacks.

In addition, deploying a security platform such as Cortex XDR can provide an additional layer of protection and visibility to the various stages of the attack.

Additional Resources

Detecting and Preventing the Path to a Golden Ticket With Cortex XDR - Palo Alto Networks
Protecting Against the Bronze Bit Vulnerability with Cortex XDR - Palo Alto Networks
The Essential Guide to XDR - Palo Alto Networks
Understanding Microsoft Kerberos PAC Validation - Microsoft
The Kerberos Network Authentication Service (V5) - MIT
A Diamond (Ticket) in the Ruff - Semperis

Appendix: Glossary

Constrained delegation – Provides a safer form of delegation that could be used by services. When it is configured, constrained delegation restricts the services to which the specified server can act on the behalf of a user.

DCSync Attack – Abusing a Windows Domain Controller's application programming interface (API) to simulate the replication process from a remote domain controller.

Domain controller (DC) – A server computer that responds to security authentication requests within a computer network domain. It is a network server that is responsible for allowing host access to domain resources. Serves as the KDC in AD environments.

ENC-TKT-IN-SKEY – The option indicates that the ticket for the end server is to be encrypted in the session key from the additional TGT provided.

Golden Ticket attack – Adversaries who have the KRBTGT account password hash may forge Kerberos ticket-granting tickets (TGT) called “Golden ticket” which enable adversaries to generate authentication material for any account in Active Directory.

Kerberos – Kerberos is a network authentication protocol that is primarily used in Active Directory environments.

Kerberos delegation – Kerberos delegation is a delegation setting that allows applications to request end-user access credentials to access resources on behalf of the originating user.

Key Distribution Center (KDC) – A key distribution center (KDC) in cryptography is a system that is responsible for providing keys to the users in a network that shares sensitive or private data.

KRBTGT account – A local default account that acts as a service account for the KDC service.

PA-FOR-USER – A struct used to identify the user, on whose behalf the service requests the service ticket, using the user name and user realm.

Privileged Attribute Certificate (PAC) – The PAC is a structure that conveys authorization-related information provided by domain controllers (DCs). The PAC is used by authentication protocols that verify identities to transport authorization information, which controls access to resources.

S4U – Kerberos extensions in order to avoid keeping the user's TGT in memory. It consists of S4U2Self and S4U2Proxy.

S4U2Proxy – Will allow a service to authenticate to a different service.

S4U2Self – Allows a service to ask for a ticket to itself on behalf of any user.

Service Principal Names (SPN) – A unique identifier of a service instance. SPNs are used by Kerberos authentication to associate a service instance with a service logon account.

Ticket Granting Server (TGS) – A TGS validates the use of a ticket for a specified purpose, such as network service access.

Ticket Granting Ticket (TGT) – A user authentication token issued by the KDC that is used to request access tokens from the Ticket Granting Service (TGS) for specific resources/systems joined to the domain.

U2U authentication – A method to perform authentication when the verifier does not have access to a long-term service key. To address this problem, the Kerberos protocol allows the client to request that the ticket issued by the KDC be encrypted using a session key from a TGT issued to the party that will verify the authentication.

Updated December 13, 2022, at 8:25 a.m. PT.

By submitting this form, you agree to our Terms of Use and acknowledge our Privacy Statement.