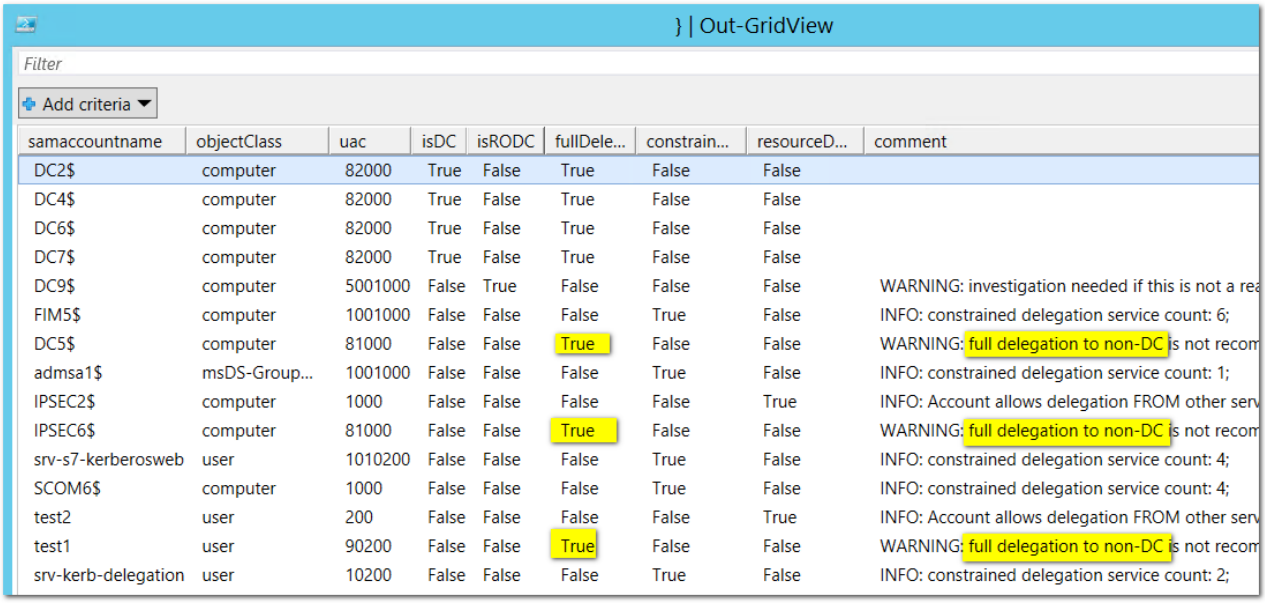
Get rid of accounts that use Kerberos Unconstrained Delegation

learn.microsoft.com/en-us/archive/blogs/389thoughts/get-rid-of-accounts-that-use-kerberos-unconstrained-delegation



samaccountname	objectClass	uac	isDC	isRODC	fullDele...	constrain...	resourceD...	comment
DC2$	computer	82000	True	False	True	False	False	
DC4$	computer	82000	True	False	True	False	False	
DC6$	computer	82000	True	False	True	False	False	
DC7$	computer	82000	True	False	True	False	False	
DC9$	computer	5001000	False	True	False	False	False	WARNING: investigation needed if this is not a rea
FIM5$	computer	1001000	False	False	False	True	False	INFO: constrained delegation service count: 6;
DC5$	computer	81000	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
admsa1$	msDS-Group...	1001000	False	False	False	True	False	INFO: constrained delegation service count: 1;
IPSEC2$	computer	1000	False	False	False	False	True	INFO: Account allows delegation FROM other serv
IPSEC6$	computer	81000	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
srv-s7-kerberosweb	user	1010200	False	False	False	True	False	INFO: constrained delegation service count: 4;
SCOM6$	computer	1000	False	False	False	True	False	INFO: constrained delegation service count: 4;
test2	user	200	False	False	False	False	True	INFO: Account allows delegation FROM other serv
test1	user	90200	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
srv-kerb-delegation	user	10200	False	False	False	True	False	INFO: constrained delegation service count: 2;

- Article
- 04/18/2017

Suppose you are managing an enterprise Active Directory. You will have people at your desk that need you to configure something in AD to support their applications: GPOs, service accounts, OUs and permissions, etc. Sometimes they will ask for Kerberos Delegation, a nebulous technology that is generally not well understood by admins or developers. There are multiple kinds of Kerberos delegation, but what you need to know is that that one form is particular dangerous from a security perspective: *unconstrained* delegation.

In a nutshell, unconstrained Kerberos delegation gives a service to the ability impersonate *you* to any other service it likes. Suppose you have an IIS website, and its application pool account is configured with unconstrained delegation. The site also has Windows Authentication enabled, allowing native Kerberos authentication. It uses a SQL server backend for business data. You, with your Domain Admin account, browse to this website and authenticate to it. The website, using unconstrained delegation can now get a service ticket from a DC to the SQL service, and do so *in your name.*

The problem is, with unconstrained delegation you need to trust the application to do the right thing. But perhaps it won't. Remember that you logged on as Domain Admin? The site can create a ticket to whatever service it likes, as you, a Domain Admin. It could go to a DC, and change the Enterprise Admin group. It could get the hash of the krbtgt account. Or, it might download an interesting file from the HR department.

Perhaps you trust the intention of the application. But what if it gets compromised, and an attacker injects any code it wants? The sky is the limit. You get the point: unconstrained Kerberos delegation has a high security impact.

All of this is old information. It just seems to be underestimated somehow. Most customers I talk to do not consider this a serious issue or are not even aware of it. Before we go on, let me point you to some articles in case you need a background refresher:

- a general introduction on the ASKDS blog: Kerberos for the busy admin.
- the old but hardcore reference: How the Kerberos Version 5 Authentication Protocol Works.
- nice and explicit write-up (non-MSFT) on why unconstrained delegation is dangerous: Active Directory Security Risk #101: Kerberos Unconstrained Delegation (or How Compromise of a Single Server Can Compromise the Domain).

As mentioned, there are multiple types of Kerberos delegation. Besides the unconstrained kind, there is also *constrained delegation* introduced with Windows Server 2003, and *resource based constrained delegation* which was new with Windows Server 2012. These also have security consequences, but nowhere nearly as bad as the unconstrained variation.

For a customer engagement I needed an overview of delegated accounts, which include user accounts, computer accounts, and both kinds of managed service accounts. I could not find a single script to pull all of that together, so I rolled my own. The full script is on the TechNet gallery: Search-KerbDelegatedAccounts.ps1. This is the working but abbreviated version, with comments taken out:

```
[powershell]
[CmdletBinding()]
Param
(
# start the search at this DN. Default is to search all of the domain.
[string]$DN = (Get-ADDomain).DistinguishedName
)

$SERVER_TRUST_ACCOUNT = 0x2000
$TRUSTED_FOR_DELEGATION = 0x80000
$TRUSTED_TO_AUTH_FOR_DELEGATION= 0x1000000
$PARTIAL_SECRETS_ACCOUNT = 0x4000000
$bitmask = $TRUSTED_FOR_DELEGATION -bor
$TRUSTED_TO_AUTH_FOR_DELEGATION -bor $PARTIAL_SECRETS_ACCOUNT

# LDAP filter to find all accounts having some form of delegation.
# 1.2.840.113556.1.4.804 is an OR query.
$filter = @"
(&
(servicePrincipalname=*)
```

```
(|
(msDS-AllowedToActOnBehalfOfOtherIdentity=*)
(msDS-AllowedToDelegateTo=*)
(UserAccountControl:1.2.840.113556.1.4.804:=$bitmask)
)
(|
(objectcategory=computer)
(objectcategory=person)
(objectcategory=msDS-GroupManagedServiceAccount)
(objectcategory=msDS-ManagedServiceAccount)
)
)
"@ -replace "[\s\n]", "

$propertylist = @(
"servicePrincipalname",
"useraccountcontrol",
"samaccountname",
"msDS-AllowedToDelegateTo",
"msDS-AllowedToActOnBehalfOfOtherIdentity"
)
Get-ADObject -LDAPFilter $filter -SearchBase $DN -SearchScope Subtree -Properties
$propertylist -PipelineVariable account | ForEach-Object {
$isDC = ($account.useraccountcontrol -band $SERVER_TRUST_ACCOUNT) -ne 0
$fullDelegation = ($account.useraccountcontrol -band $TRUSTED_FOR_DELEGATION)
-ne 0
$constrainedDelegation = ($account.'msDS-AllowedToDelegateTo').count -gt 0
$isRODC = ($account.useraccountcontrol -band $PARTIAL_SECRETS_ACCOUNT) -ne
0
$resourceDelegation = $account.'msDS-AllowedToActOnBehalfOfOtherIdentity' -ne $null

$comment = ""
if ((-not $isDC) -and $fullDelegation) {
$comment += "WARNING: full delegation to non-DC is not recommended!; "
}
if ($isRODC) {
$comment += "WARNING: investigation needed if this is not a real RODC; "
}
if ($resourceDelegation) {
# to count it using PS, we need the object type to select the correct function... broken, but
there we are.
$comment += "INFO: Account allows delegation FROM other server(s); "
}
if ($constrainedDelegation) {
```

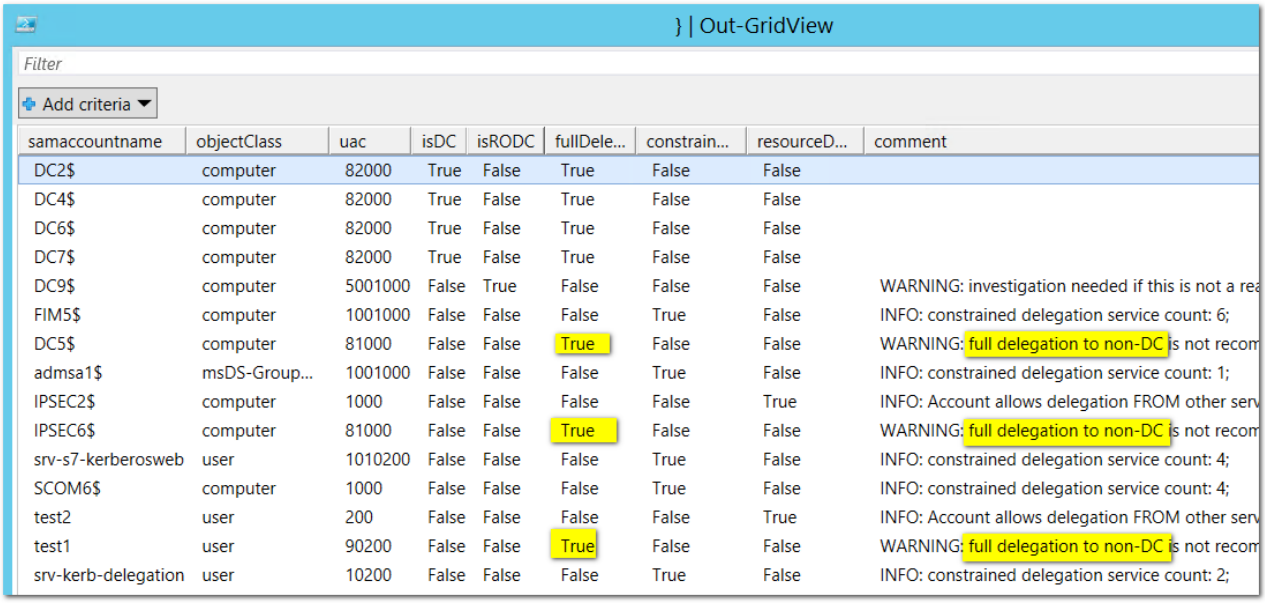
```
$comment += "INFO: constrained delegation service count: $(($account.'msDS-AllowedToDelegateTo').count); "
}

[PSCustomobject] @{
samaccountname = $account.samaccountname
objectClass = $account.objectclass
uac = ('{0:x}' -f $account.useraccountcontrol)
isDC = $isDC
isRODC = $isRODC
fullDelegation = $fullDelegation
constrainedDelegation = $constrainedDelegation
resourceDelegation = $resourceDelegation
comment = $comment
}
}
```
[/powershell]

By default, the script scans the entire current domain. If you like, you can point it to a specific OU using the "-DN" argument. The script uses a medium complicated LDAP query. The heart of the query is a binary OR condition on the useraccountControl attribute, selecting on the bits that represent some form of delegation: TRUSTED_FOR_DELEGATION (unconstrained), TRUSTED_TO_AUTH_FOR_DELEGATION (constrained with transition to any protocol), and PARTIAL_SECRETS_ACCOUNT (allowed to replicate password hashes, indicates RODC). Resource based constrained delegation is not represented in userAccountcontrol, because this particular form is configured on the target and not on the middle tier. That's the reason for the logical OR condition on a nonzero value of msDS-AllowedToActOnBehalfOfOtherIdentity.

Execution of the query generates a list of accounts with any form of delegation. The information is turned into Boolean flags for easier filtering later on. I added a comment field offering some explanation and warnings on the findings, specifically to point out unconstrained delegation and accounts that claim to be RODCs but perhaps are something else. Finally, all information is packed into an object and sent to the pipeline.

The following is an example from my lab. To get at this, save the script as "Search-KerbDelegatedAccounts.ps1", and execute it like this:

[powershell]
```
.\Search-KerbDelegatedAccounts.ps1 | Out-Gridview
```
[/powershell]



samaccountname	objectClass	uac	isDC	isRODC	fullDele...	constrain...	resourceD...	comment
DC2$	computer	82000	True	False	True	False	False	
DC4$	computer	82000	True	False	True	False	False	
DC6$	computer	82000	True	False	True	False	False	
DC7$	computer	82000	True	False	True	False	False	
DC9$	computer	5001000	False	True	False	False	False	WARNING: investigation needed if this is not a rea
FIM5$	computer	1001000	False	False	False	True	False	INFO: constrained delegation service count: 6;
DC5$	computer	81000	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
admsa1$	msDS-Group...	1001000	False	False	False	True	False	INFO: constrained delegation service count: 1;
IPSEC2$	computer	1000	False	False	False	False	True	INFO: Account allows delegation FROM other serv
IPSEC6$	computer	81000	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
srv-s7-kerberosweb	user	1010200	False	False	False	True	False	INFO: constrained delegation service count: 4;
SCOM6$	computer	1000	False	False	False	True	False	INFO: constrained delegation service count: 4;
test2	user	200	False	False	False	False	True	INFO: Account allows delegation FROM other serv
test1	user	90200	False	False	True	False	False	WARNING: full delegation to non-DC is not recom
srv-kerb-delegation	user	10200	False	False	False	True	False	INFO: constrained delegation service count: 2;

This domain has a small but varied collection of delegations, including some cases for unconstrained delegation that should be hunted down. It also detects the presence of an RODC account, which is real in this case.

The next obvious question is: how do I get rid of unconstrained delegation? The high-level answer is easy: convert to constrained or resource-based delegation. How to do this is a bit out of scope for a single blog entry, but have a look here for starters: Understanding Kerberos Double Hop.

In practice this conversion may not be so easy. The application may simply not work with anything but unconstrained delegation; especially non-Windows machines may have this problem. Or maybe it does work, but the vendor will not support it. If for whatever reason conversion to constrained delegation is not feasible you still have some options:

- Accept the risk, and move on. Make sure to involve a security officer for approval, if applicable to your company.
- Recognize that the affected accounts operate (almost) on DC security level, and put the account and servers under control of the Domain Administrators (real people, not the group).
- Enable Kerberos auditing using an advanced audit policy on all DCs, and start monitoring for tickets from delegated accounts to unusual services. Not easy, but software exists to help with this.
- Set outbound (!) firewall restrictions on the servers using the unconstrained delegated account.

Looking at it from another angle, it is also possible to specifically protect your important admin accounts while leaving the application alone. These accounts would then not be able to use Kerberos delegation of any kind. Three possibilities are, in order of increasing impact:

1. For each account that you want to protect, open its properties in ADU&C, and set the checkbox "Account is sensitive and cannot be delegated". This simply blocks all delegation scenarios.
2. Add the accounts that you want to protect to the group "Protected Users". This groups blocks its members from using Kerberos delegation, blocks NTLM, forces AES, disables cached logon, and more. It requires a domain functional level of 2012 or higher. Before you do this, read: How to Configure Protected Accounts.
3. Use authentication policy silos, which builds on Protected Users and also limits the machines that its members can authenticate to. This is a powerful but complex approach, usually done as part of a larger security project. The link under the previous point contains some information. More in-depth information is here: Authentication Policies and Authentication Policy Silos.

Pick the first one if you are not sure what to do. You should tick that checkbox in any case for your Domain Admins.

Main takeaway: chase all services with unconstrained delegation. If these are *not* DC accounts, reconfigure them with constrained delegation, OR claim them als DCs from a security perspective. Meaning, the AD team manages the service and the servers it runs on. In any case, make sure that at least the Domain Admin accounts cannot be used for delegation.

Comments

- daemonR00t May 24, 2017
  Great post Willem! Thanks

- Jeroen de Bonte

  May 25, 2017
  Good stuff! :)

- JDsst84 January 28, 2019
  this is amazing, I only ask one thing is there a way to run this on other domains in my forest?

  > Willem Kasdorp January 29, 2019
  > The easiest way would be to use an account of that domain to run the script. OTOH, it would not be too much work to change the script to get all domains in the forest and to loop over that. Exercise for the reader ;)
  >
  > > JDsst84 January 29, 2019
  > > Thanks much I will play with it

- cappel2007 March 12, 2019
  Hi there! Awesome post and script! Is there a reason why DC's need full delegation? I have that as well when I ran the script and believe it is a default setting for Windows 2008 R2. Because it is default, I am leery to change. Online searches aren't turning up much information about it. Thanks!

  > Willem Kasdorp March 12, 2019
  > DC's are just about the only machines that need full delegation because they need to be able to issue any ticket. And you're right, information about that is hard to find. Probably most of it was lost with the Windows 2003 documentation purge.
  >
  > > cappel2007 March 13, 2019
  > > Thank you for the clarification!