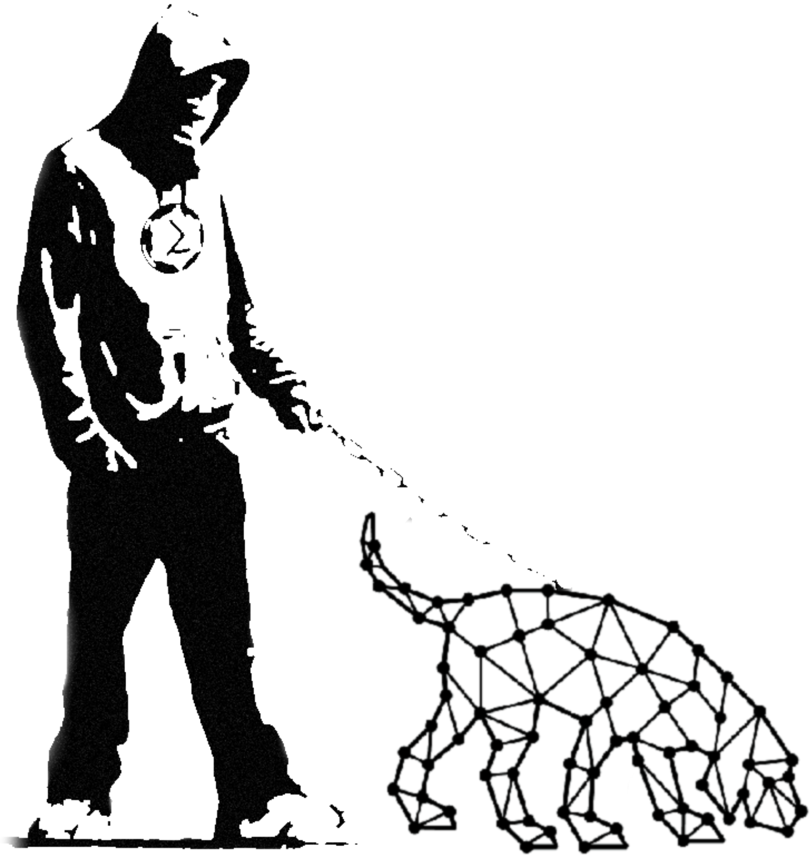
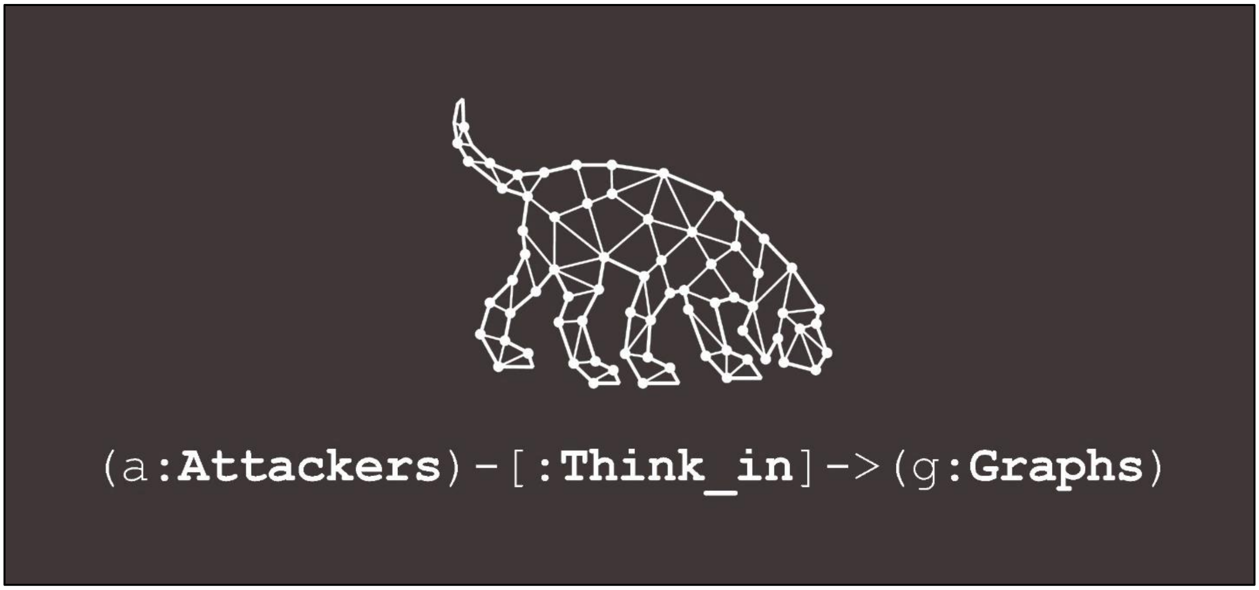


THE DOG WHISPERER'S HANDBOOK 4

A Hacker's Guide to the BloodHound Galaxy *– @SadProcessor*

Table of Contents

1. DOG WHISPERER

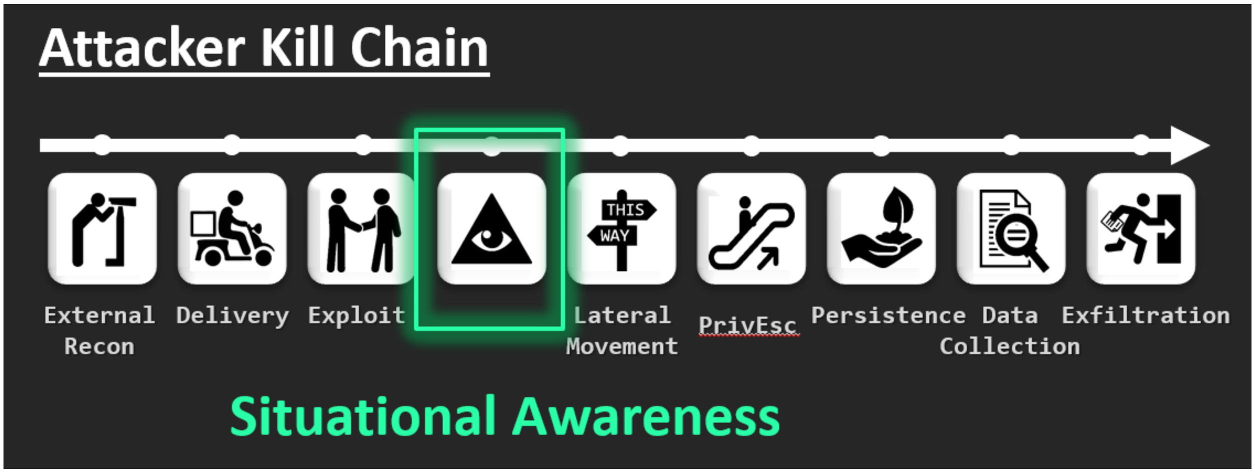


Heard the news? @CptJesus did it again… BloodHound 4 is out… and our favorite dog is going to the cloud.
This guide is an updated version with latest additions to BloodHound.

1.1. Attackers think in Graphs

First things first, I would like to give credits to @harmj0y, @Wald0 & @CptJesus, for creating and sharing BloodHound. Picking up on some early work by Jean-Baptiste Galet & Geraud de Drouas from the French ANSSI, they have brought a new way of looking at Active Directory from the attackers point of view, and this has great defensive value when trying to "Think In Graphs"…
What? You haven't read @JohnLaTwC's awesome post? A must read from Microsoft's Head of Threat Intel before diving into the bloodhound universe.

So, what is bloodhound?



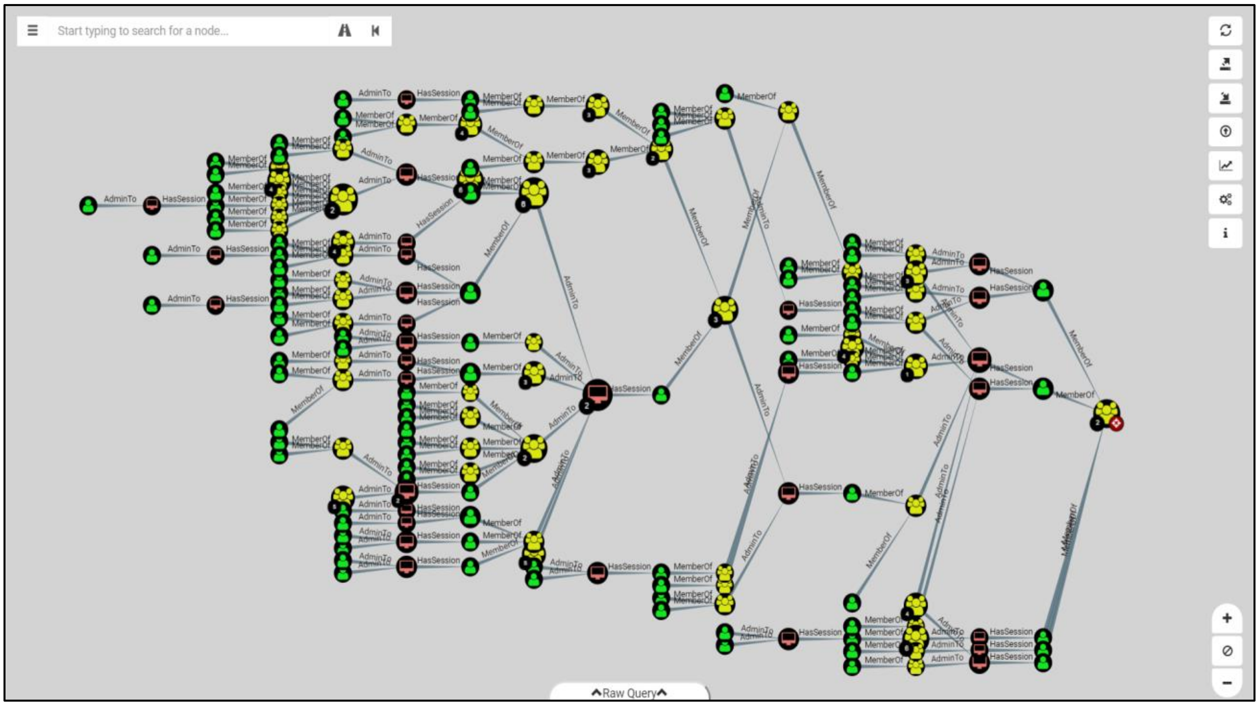
Attacker Kill Chain

External Recon · Delivery · Exploit · Lateral Movement · PrivEsc · Persistence · Data Collection · Exfiltration

Situational Awareness

Bloodhound is an open source reconnaissance tool designed to map Active Directory attack paths.

Originally build as a red team tool, it is also used by blue teams to identify possible abusable configuration in the environment, and better understand how to improve their Active Directory security posture in general.



Warning: Using bloodhound for the first time in your environment is often an eye opener… ;)

There is a very dynamic community around the tool, so if you haven't yet, I would highly recommend you invite yourself to the bloodhound slack and join the crew to talk AD security and more…

One last thing, these guys share all this for free and have a big heart…

If you like BloodHound, you can buy some cool BloodHound swag.

You will look awesome and support a good cause at the same time.

Do it.

1.2. Disclaimer

Content in this guide is mostly stuff I gathered from the internet while playing with bloodhound.

I am no kind of authority on anything and am just sharing what I got and what I understood so far…

Tools are shared as ideas and POCs. I'm not great with PRs…

¯_(ツ)_/¯ Sorry if any mistake has made its way into this document…

2. NEO4J

Before we dive into BloodHound, we need to talk a bit about neo4j, the graph database backing bloodhound.

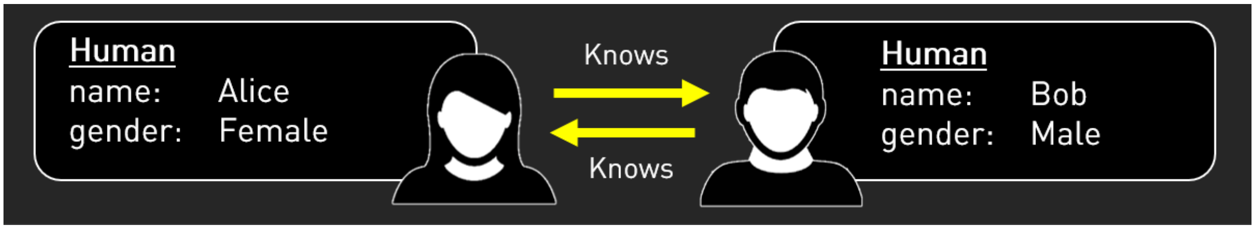
In this chapter, I'll introduce some basic concepts and terminology we will be using thru out this guide.

2.1. Concept & Terminology

Neo4j is a graph database. The power of these type of database is that we can query relationships between objects.

In the world of graph databases, an object is called a **node**. A relationship between two nodes is called an **edge**.

The following example illustrates this concept:



In this examples Alice and Bob are **nodes**, and the arrows between them represent **edges**.

It is important to note that a relationship is one way and has a direction.

In our example, Alice knows Bob and Bob knows Alice. These are two separate edges.

Using **Cypher**, the neo4j database query language, we could now ask several types of questions:

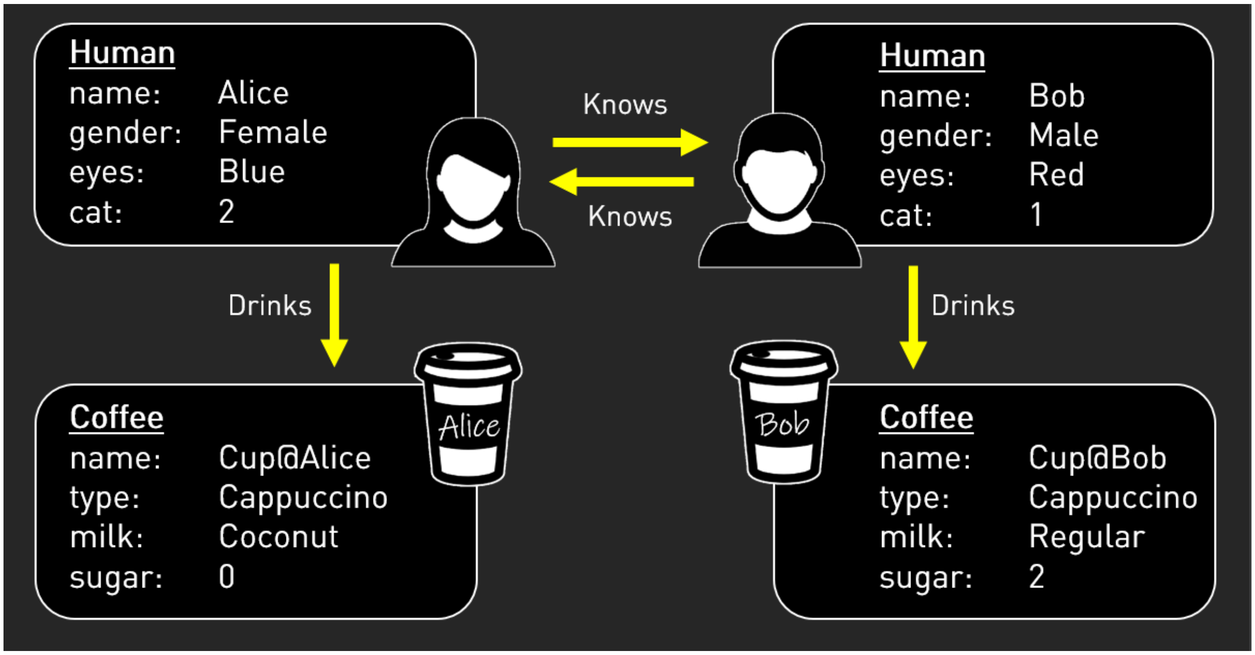
- Who is there?
- Who is Bob?
- Who Know Alice?
- Who does Bob Know?
- Who knows who?

The two first questions would return nodes.

The last three question would return nodes and relationships between them.

A series of nodes and relationships is called a **path**.

Now let's add more data to our dataset:



In the example above, we have more type of nodes: Human and Coffee.

In the graph database terminology, the type of a node is called a **label**.

Nodes have **properties**.

Edges can also have properties.

[And nodes can even have several labels… but let's not go too far for now]

With our above example, we could ask more questions:

- Who has more cats?
- Who drinks coffee?
- Who Drinks coffee with coconut milk?
- Who knows someone who drinks cappuccino with more sugar than him?
- Who has 2 cats and know someone with red eyes?
- Does anyone with cats know someone who drinks cappuccino with more sugar than her/him?

The more you add nodes and relationships types, the more complex it gets, but if you can ask the right question, you will get the right answer, and this is where **Cypher** becomes your friend… but let's not go too fast.

Now imagine what you would get if this concept was applied to Active directory objects and their possible abusable relationships…

3. BLOODHOUND

3.1. Install

In this guide, I will quickly go thru the install for Windows 10. Instruction for other OSs can be found here.

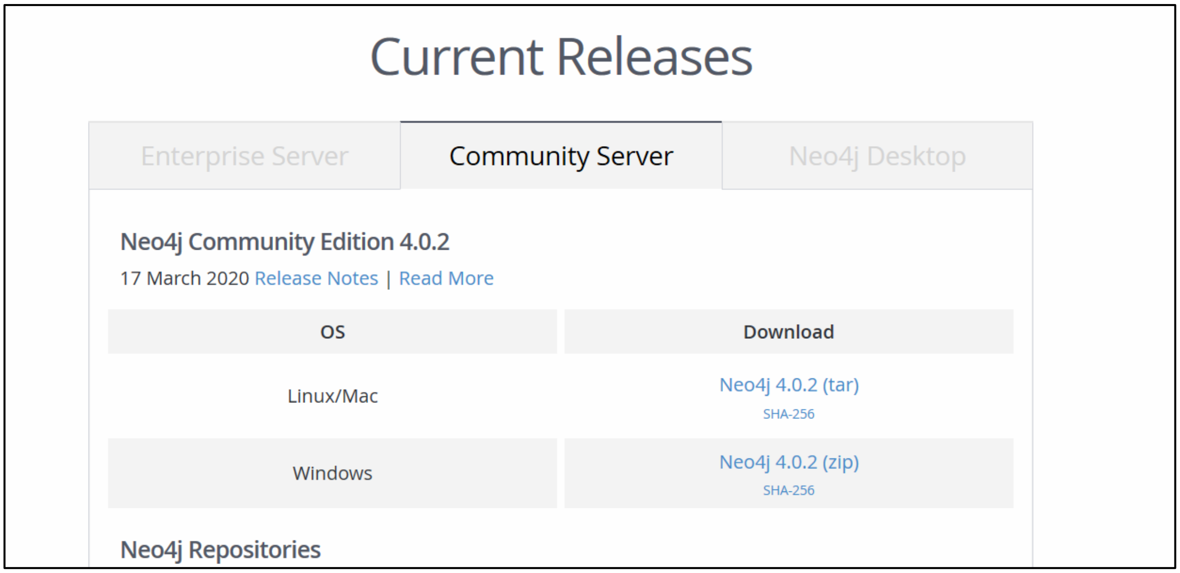
Note that Neo4j can get quite hungry when it comes to RAM. If you are playing around with a small dataset for testing, [a VM with] 4gb RAM should do. If you are dealing a consequent dataset in a real environment, a laptop with an i7 and 16gb RAM is often enough…

Here are the steps to follow for install:

Step1 – Download and install latest Java JDK

Step2 – Download and install neo4j Server – Community Edition

- Go to the following link
- Download latest **neo4j community edition**



Current Releases

Neo4j Community Edition 4.0.2

17 March 2020 Release Notes | Read More

OS	Download
Linux/Mac	Neo4j 4.0.2 (tar) SHA-256
Windows	Neo4j 4.0.2 (zip) SHA-256

Neo4j Repositories

- Extract in chosen location
- Open admin prompt and go to the /bin folder
- Type the following to install service:

```
> .\neo4j.bat install-service
```

- Type the following to start service

```
> .\neo4j.bat start
```

Note: Starting and stopping the service can also be done via the windows services console.

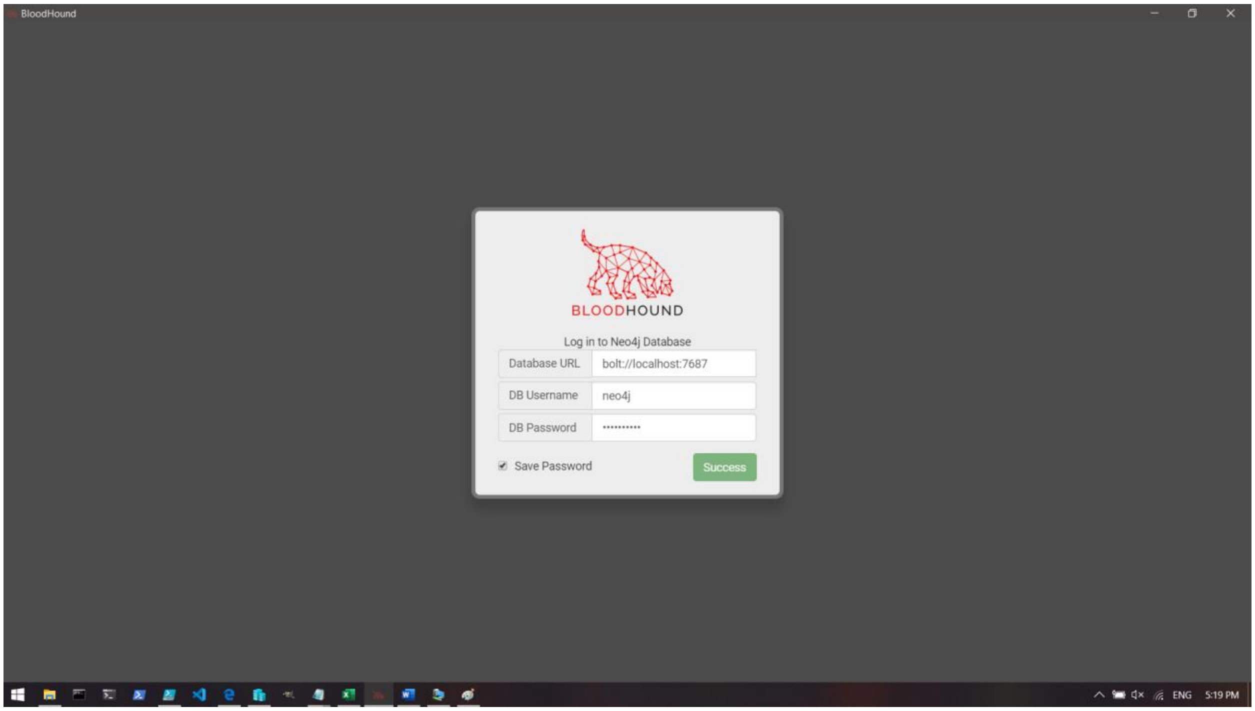
- Navigate to the neo4j browser at http://localhost:7474/Browser
- Enter username: **neo4j** & password: **neo4j**
- Enter new password
- Close browser for now

Step3 – Download BloodHound binaries

Latest binaries can be found here.

Note: AV might flag on download. Use folder exclusion if needed.

Double-click on Bloodhound.exe to open application and enter chosen password.



BloodHound

BLOODHOUND

Log in to Neo4j Database

Database URL: bolt://localhost:7687

DB Username: neo4j

DB Password: ••••••••••

☑ Save Password

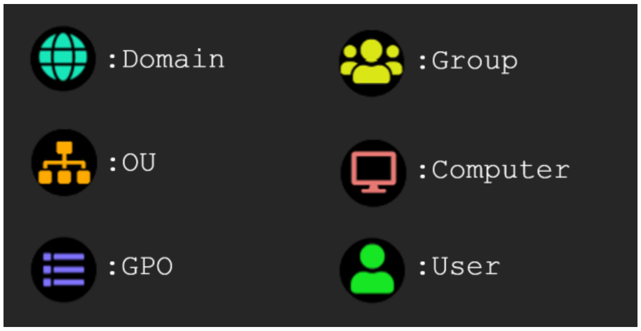
3.2. Data Types

We still need to explain a few things before we can start playing with BloodHound.

Let’s see how this neo4j graph database concept applies to Active Directory objects and relationship.

3.2.1. Bloodhound Nodes

In BloodHound, there are 6 types of nodes [labels]:



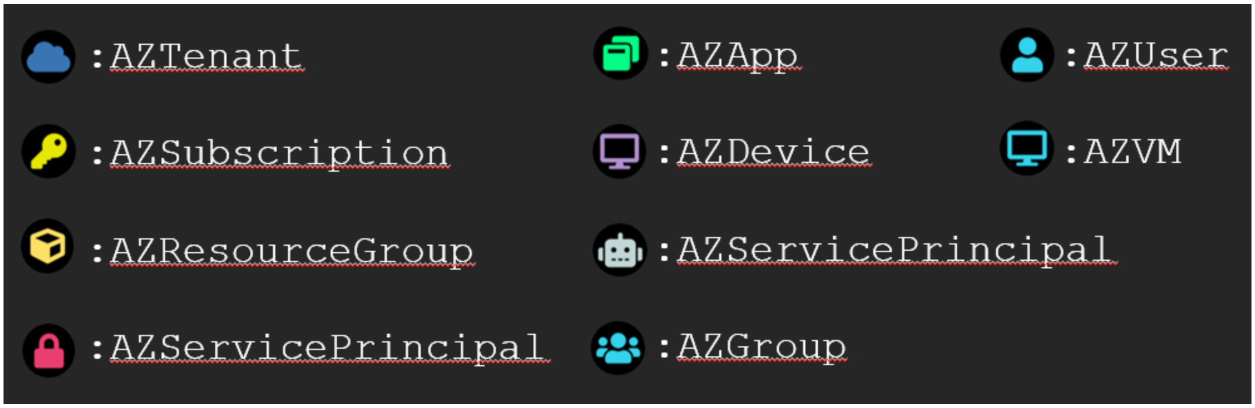
:Domain
:Group
:OU
:Computer
:GPO
:User

Each type of node has its own set of properties.

Note: Bloodhound nodes all have a unique *objectid* property.

BloodHound 4 includes Azure objects and relationships (collected separately with AzureHound).

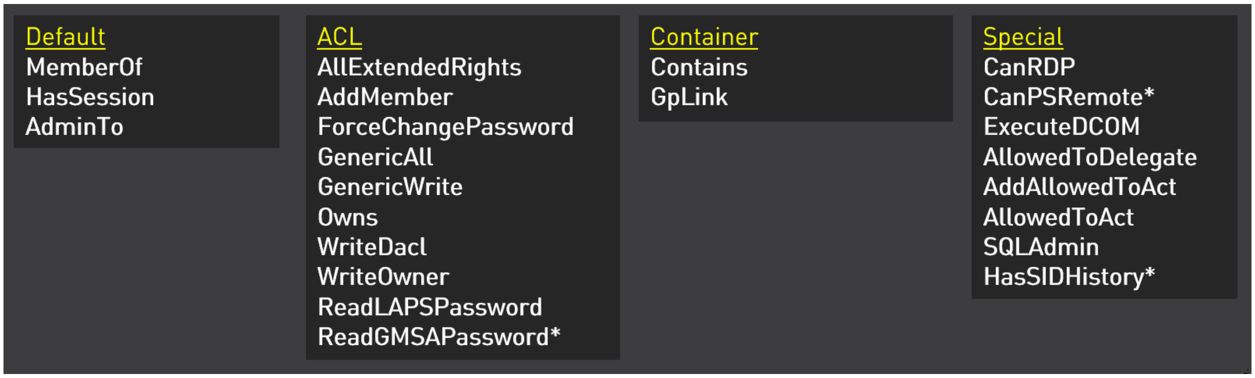
Following types of Azure objects are available:



:AZTenant
:AZApp
:AZUser
:AZSubscription
:AZDevice
:AZVM
:AZResourceGroup
:AZServicePrincipal
:AZServicePrincipal
:AZGroup

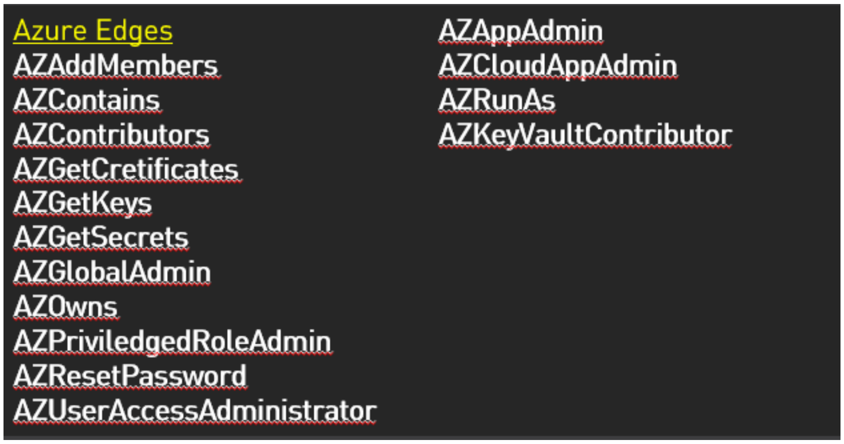
3.2.2. BloodHound Edges

In its original release, Bloodhound only had 3 types of relationships, but the little puppy grew quickly, and version 3 included 23 ‘on-prem’ edges:



Default	ACL	Container	Special
MemberOf	AllExtendedRights	Contains	CanRDP
HasSession	AddMember	GpLink	CanPSRemote*
AdminTo	ForceChangePassword		ExecuteDCOM
	GenericAll		AllowedToDelegate
	GenericWrite		AddAllowedToAct
	Owns		AllowedToAct
	WriteDacl		SQLAdmin
	WriteOwner		HasSIDHistory*
	ReadLAPSPassword		
	ReadGMSAPassword*		

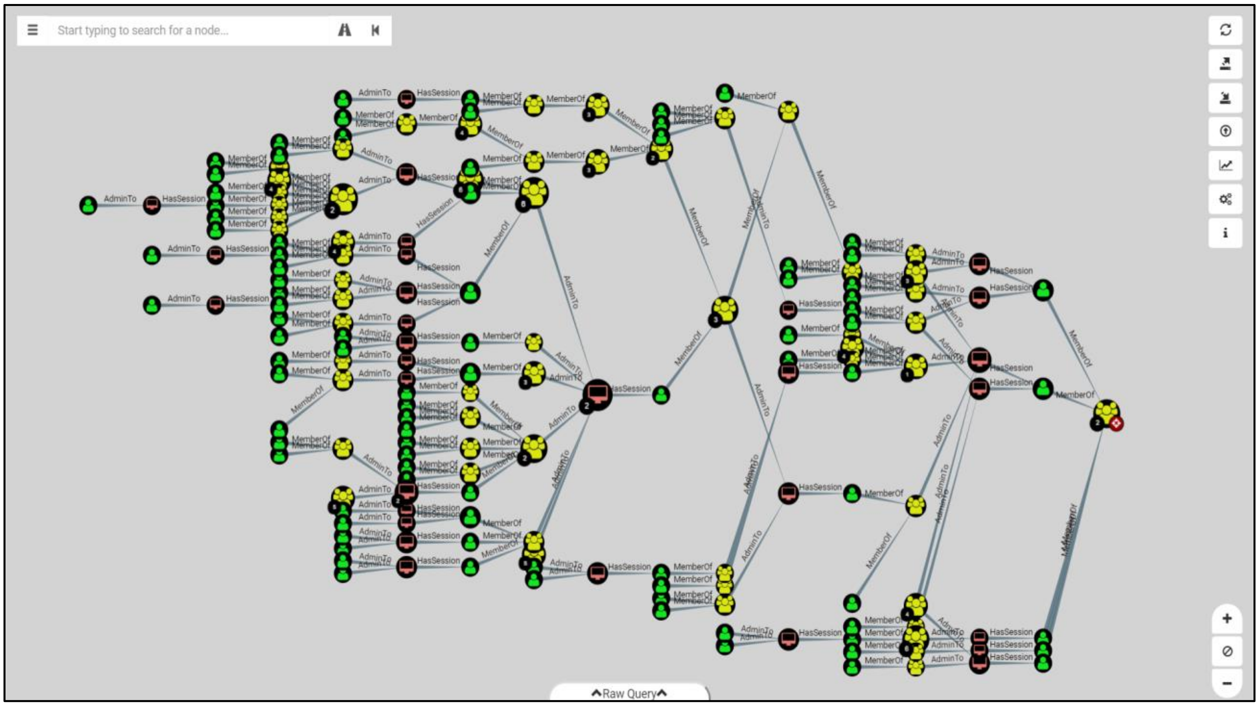
BloodHound 4 now adds the following Azure relationships between Azure nodes:



Azure Edges
AZAddMembers
AZContains
AZContributors
AZGetCretificates
AZGetKeys
AZGetSecrets
AZGlobalAdmin
AZOwns
AZPriviledgedRoleAdmin
AZResetPassword
AZUserAccessAdministrator
AZAppAdmin
AZCloudAppAdmin
AZRunAs
AZKeyVaultContributor

Each of these edges comes with an associated way of abusing it and moving to the next node on the graph.

When you put it all together, it looks something like this...



3.3. Edge Info

The guide is not about offensive tradecraft, so I won't go into detailing each edge and associated attacks. A lot of useful info can be found in Bloodhound itself by right-clicking on an Edge on the graph. This will bring up a contextual menu with more info on that edge including abuse info, OpSec Considerations and Links to further resources.
The following info is taken from Bloodhound.

3.3.1. Default

These edges are the base and have been here since the first version

3.3.1.1. MemberOf

X is a member of the group Y.

Groups in active directory grant their members any privileges the group itself has. If a group has rights to another principal, users/computers in the group, as well as other groups inside the group inherit those permissions.

https://adsecurity.org/?tag=ad-delegation

https://www.itprotoday.com/management-mobility/view-or-remove-active-directory-delegated-permissions

3.3.1.2. HasSession

The computer x has a session for user y.

When a user authenticates to a computer, they often leave credentials exposed on the system, which can be retrieved through LSASS injection, token manipulation/theft, or injecting into a user's process.

Any user that is an administrator to the system has the capability to retrieve the credential material from memory if it still exists.

Note: A session does not guarantee credential material is present, only possible.

Gathering Credentials

http://blog.gentilkiwi.com/mimikatz

https://github.com/gentilkiwi/mimikatz

https://adsecurity.org/?page_id=1821

https://attack.mitre.org/wiki/Credential_Access

Token Impersonation

https://labs.mwrinfosecurity.com/assets/BlogFiles/mwri-security-implications-of-windows-access-tokens-2008-04-14.pdf

https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Invoke-TokenManipulation.ps1

https://attack.mitre.org/wiki/Technique/T1134

3.3.1.3. AdminTo

X has admin rights to the computer y.

By default, administrators have several ways to perform remote code execution on Windows systems, including via RDP, WMI, WinRM, the Service Control Manager, and remote DCOM execution.

Further, administrators have several options for impersonating other users logged onto the system, including plaintext password extraction, token impersonation, and injecting into processes running as another user.

Finally, administrators can often disable host-based security controls that would otherwise prevent the aforementioned techniques.

Lateral movement

https://attack.mitre.org/wiki/Lateral_Movement

Gathering Credentials

http://blog.gentilkiwi.com/mimikatz

https://github.com/gentilkiwi/mimikatz

https://adsecurity.org/?page_id=1821

https://attack.mitre.org/wiki/Credential_Access

Token Impersonation

https://labs.mwrinfosecurity.com/assets/BlogFiles/mwri-security-implications-of-windows-access-tokens-2008-04-14.pdf

https://github.com/PowerShellMafia/PowerSploit/blob/master/Exfiltration/Invoke-TokenManipulation.ps1

https://attack.mitre.org/wiki/Technique/T1134

Disabling host-based security controls

https://blog.netspi.com/10-evil-user-tricks-for-bypassing-anti-virus/

https://www.blackhillsinfosec.com/bypass-anti-virus-run-mimikatz/

Opsec Considerations

https://blog.cobaltstrike.com/2017/06/23/opsec-considerations-for-beacon-commands/

3.3.2. ACL

These edges are all based on possible ACL abuse. They have been added since BloodHound 2 and greatly extend the attack possibilities.

3.3.2.1. AllExtendedRights

Extended rights are special rights granted on objects which allow reading of privileged attributes, as well as performing special actions.

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://www.youtube.com/watch?v=z8thoG7gPd0

3.3.2.2. AddMember

The user x has the ability to add arbitrary principals, including itself, to the group y. Because of security group delegation, the members of a security group have the same privileges as that group.

By adding itself to the group, x will gain the same privileges that group already has.

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://www.youtube.com/watch?v=z8thoG7gPd0

https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4728

3.3.2.3. ForceChangepassword

The user x has the capability to change the user y's password without knowing that user's current password.

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://www.youtube.com/watch?v=z8thoG7gPd0

https://www.sixdub.net/?p=579

https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4724

3.3.2.4. GenericAll

This is also known as full control. This privilege allows the trustee to manipulate the target object however they wish.

3.3.2.5. GenericWrite

X has generic write access to y.

Generic Write access grants you the ability to write to any non-protected attribute on the target object, including "members" for a group, and "serviceprincipalnames" for a user.

3.3.2.6. Owns

Object owners retain the ability to modify object security descriptors, regardless of permissions on the object's DACL

3.3.2.7. WriteDACL

With write access to the target object's DACL, you can grant yourself any privilege you want on the object.

3.3.2.8. WriteOwner

X has the ability to modify the owner of y.

Object owners retain the ability to modify object security descriptors, regardless of permissions on the object's DACL.

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://www.youtube.com/watch?v=z8thoG7gPd0

https://adsecurity.org/?p=1729

http://www.harmj0y.net/blog/activedirectory/targeted-kerberoasting/

https://posts.specterops.io/a-red-teamers-guide-to-gpos-and-ous-f0d03976a31e

https://eladshamir.com/2019/01/28/Wagging-the-Dog.html https://github.com/GhostPack/Rubeus#s4u

https://gist.github.com/HarmJ0y/224dbfef83febdaf885a8451e40d52ff

http://www.harmj0y.net/blog/redteaming/another-word-on-delegation/

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://github.com/Kevin-Robertson/Powermad#new-machineaccount

3.3.3. ReadLAPSPassword

X has the ability to read the password set by Local Administrator Password Solution (LAPS) on the computer y.

The local administrator password for a computer managed by LAPS is stored in the confidential LDAP attribute, "ms-mcs-AdmPwd".

https://www.specterops.io/assets/resources/an_ace_up_the_sleeve.pdf

https://adsecurity.org/?p=3164

3.3.3.1. ReadGMSAPassword

Y is a Group Managed Service Account. The user x can retrieve the password for the GMSA.

Group Managed Service Accounts are a special type of Active Directory object, where the password for that object is mananaged by and automatically changed by Domain Controllers on a set interval (check the MSDS-ManagedPasswordInterval attribute).

The intended use of a GMSA is to allow certain computer accounts to retrieve the password for the GMSA, then run local services as the GMSA. An attacker with control of an authorized principal may abuse that privilege to impersonate the GMSA.

https://www.dsinternals.com/en/retrieving-cleartext-gmsa-passwords-from-active-directory/

https://www.powershellgallery.com/packages/DSInternals/

https://github.com/markgamache/gMSA/tree/master/PSgMSAPwd

https://adsecurity.org/?p=36

https://adsecurity.org/?p=2535

https://www.ultimatewindowssecurity.com/securitylog/encyclopedia/event.aspx?eventID=4662

3.3.4. Container

Edges in this category have been added in BloodHound xx. Collecting this data gives an attacker the possibility to abuse GPO configuration to gain further control over the environment. This again adds complexity to the graph.

Note that due to the complexity of how GPOs are applied down the line, these attack path sometimes are false positives.

3.3.4.1. GpLink

A linked GPO applies its settings to objects in the linked container.

3.3.4.2. Contains

GPOs linked to a container apply to all objects that are contained by the container.

https://wald0.com/?p=179

https://blog.cptjesus.com/posts/bloodhound15

3.3.5. Special

In this category various 'exotic' abusable relationships.

3.3.5.1. CanRDP

X has the capability to create a Remote Desktop Connection with the computer y.

Remote Desktop access allows you to enter an interactive session with the target computer. If authenticating as a low privilege user, a privilege escalation may allow you to gain high privileges on the system.

Note: This edge does not guarantee privileged execution.

https://michael-eder.net/post/2018/native_rdp_pass_the_hash/

https://www.kali.org/penetration-testing/passing-hash-remote-desktop/

3.3.5.2. CanPSRemote

x has the capability to create a PSRemote Connection with the computer y.

PS Session access allows you to enter an interactive session with the target computer. If authenticating as a low privilege user, a privilege escalation may allow you to gain high privileges on the system.

Note: This edge does not guarantee privileged execution.

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/new-pssession?view=powershell-7/

https://docs.microsoft.com/en-us/powershell/module/microsoft.powershell.core/invoke-command?view=powershell-7

3.3.5.3. ExecuteDCOM

The user x has membership in the Distributed COM Users local group on the computer y.

This can allow code execution under certain conditions by instantiating a COM object on a remote machine and invoking its methods.

https://enigma0x3.net/2017/01/05/lateral-movement-using-the-mmc20-application-com-object/

https://enigma0x3.net/2017/01/23/lateral-movement-via-dcom-round-2/

https://enigma0x3.net/2017/09/11/lateral-movement-using-excel-application-and-dcom/

https://enigma0x3.net/2017/11/16/lateral-movement-using-outlooks-createobject-method-and-dotnettojscript/

https://www.cybereason.com/blog/leveraging-excel-dde-for-lateral-movement-via-dcom

https://www.cybereason.com/blog/dcom-lateral-movement-techniques

https://bohops.com/2018/04/28/abusing-dcom-for-yet-another-lateral-movement-technique/

https://attack.mitre.org/wiki/Technique/T1175

Invoke-DCOM

https://github.com/rvrsh3ll/Misc-Powershell-Scripts/blob/master/Invoke-DCOM.ps1

LethalHTA

https://codewhitesec.blogspot.com/2018/07/lethalhta.html

https://github.com/codewhitesec/LethalHTA/

3.3.5.4. AllowedToDelegate

The user x has the constrained delegation privilege to the computer y.

The constrained delegation primitive allows a principal to authenticate as any user to specific services (found in the msds-AllowedToDelegateTo LDAP property in the source node tab) on the target computer. That is, a node with this privilege can impersonate any domain principal (including Domain Admins) to the specific service on the target host. One caveat- impersonated users can not be in the "Protected Users" security group or otherwise have delegation privileges revoked.

An issue exists in the constrained delegation where the service name (sname) of the resulting ticket is not a part of the protected ticket information, meaning that an attacker can modify the target service name to any service of their choice. For example, if msds-AllowedToDelegateTo is “HTTP/host.domain.com”, tickets can be modified for LDAP/HOST/etc. service names, resulting in complete server compromise, regardless of the specific service listed.

https://github.com/GhostPack/Rubeus#s4u

https://labs.mwrinfosecurity.com/blog/trust-years-to-earn-seconds-to-break/

http://www.harmj0y.net/blog/activedirectory/s4u2pwnage/

https://twitter.com/gentilkiwi/status/806643377278173185

https://www.coresecurity.com/blog/kerberos-delegation-spns-and-more

http://www.harmj0y.net/blog/redteaming/from-kekeo-to-rubeus/

http://www.harmj0y.net/blog/redteaming/another-word-on-delegation/

3.3.5.5. AddAllowedToAct

The user x can modify the msds-AllowedToActOnBehalfOfOtherIdentity attribute on the computer y.

The ability to modify the msDS-AllowedToActOnBehalfOfOtherIdentity property allows an attacker to abuse resource-based constrained delegation to compromise the remote computer system. This property is a binary DACL that controls what security principals can pretend to be any domain user to the particular computer object.

If the msDS-AllowedToActOnBehalfOfOtherIdentity DACL is set to allow an attack-controller account, the attacker can use said account to execute a modified S4U2self/S4U2proxy abuse chain to impersonate any domain user to the target computer system and receive a valid service ticket "as" this user.

One caveat is that impersonated users can not be in the "Protected Users" security group or otherwise have delegation privileges revoked. Another caveat is that the principal added to the msDS-AllowedToActOnBehalfOfOtherIdentity DACL *must* have a service pricipal name (SPN) set in order to successfully abuse the S4U2self/S4U2proxy process. If an attacker does not currently control an account with a SPN set, an attacker can abuse the default domain MachineAccountQuota settings to add a computer account that the attacker controls via the Powermad project.

https://eladshamir.com/2019/01/28/Wagging-the-Dog.html

https://github.com/GhostPack/Rubeus#s4u

https://gist.github.com/HarmJ0y/224dbfef83febdaf885a8451e40d52ff

http://www.harmj0y.net/blog/redteaming/another-word-on-delegation/

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://github.com/Kevin-Robertson/Powermad#new-machineaccount

3.3.5.6. AllowedToAct

The user x has is added to the msds-AllowedToActOnBehalfOfOtherIdentity attribute on the computer y.

An attacker can use this account to execute a modified S4U2self/S4U2proxy abuse chain to impersonate any domain user to the target computer system and receive a valid service ticket "as" this user.

One caveat is that impersonated users can not be in the "Protected Users" security group or otherwise have delegation privileges revoked. Another caveat is that the principal added to the msDS-

AllowedToActOnBehalfOfOtherIdentity DACL *must* have a service pricipal name (SPN) set in order to successfully abuse the S4U2self/S4U2proxy process. If an attacker does not currently control an account with a SPN set, an attacker can abuse the default domain MachineAccountQuota settings to add a computer account that the attacker controls via the Powermad project.

https://eladshamir.com/2019/01/28/Wagging-the-Dog.html

https://github.com/GhostPack/Rubeus#s4u

https://gist.github.com/HarmJ0y/224dbfef83febdaf885a8451e40d52ff

http://www.harmj0y.net/blog/redteaming/another-word-on-delegation/

https://github.com/PowerShellMafia/PowerSploit/blob/dev/Recon/PowerView.ps1

https://github.com/Kevin-Robertson/Powermad#new-machineaccount

3.3.5.7. SQLAdmin

X is a SQL admin on the computer y.

There is at least one MSSQL instance running on bob where the user alice is the account configured to run the SQL Server instance. The typical configuration for MSSQL is to have the local Windows account or Active Directory domain account that is configured to run the SQL Server service (the primary database engine for SQL Server) have sysadmin privileges in the SQL Server application. As a result, the SQL Server service account can be used to log into the SQL Server instance remotely, read all of the databases (including those protected with transparent encryption), and run operating systems command through SQL Server (as the service account) using a variety of techniques.

For Windows systems that have been joined to an Active Directory domain, the SQL Server instances and the associated service account can be identified by executing a LDAP query for a list of "MSSQLSvc" Service Principal Names (SPN) as a domain user. In short, when the Database Engine service starts, it attempts to register the SPN, and the SPN is then used to help facilitate Kerberos authentication.

https://github.com/NetSPI/PowerUpSQL/wiki

https://www.slideshare.net/nullbind/powerupsql-2018-blackhat-usa-arsenal-presentation

https://sqlwiki.netspi.com/attackQueries/executingOSCommands/#sqlserver

https://docs.microsoft.com/en-us/sql/database-engine/configure-windows/configure-windows-service-accounts-and-permissions?view=sql-server-2017

https://blog.netspi.com/finding-sensitive-data-domain-sql-servers-using-powerupsql/

3.3.5.8. HasSIDHistory

The user x has the SID for the user y in its SIDHistory attribute.

When a kerberos ticket is created for x, it will include the SID for y, and therefore grant x the same privileges and permissions as y.

http://www.harmj0y.net/blog/redteaming/the-trustpocalypse/

http://www.harmj0y.net/blog/redteaming/a-guide-to-attacking-domain-trusts/

https://adsecurity.org/?p=1772

https://adsecurity.org/?tag=sidhistory

https://attack.mitre.org/techniques/T1178/

https://dirkjanm.io/active-directory-forest-trusts-part-one-how-does-sid-filtering-work/

See online documentation for info on Azure Edges

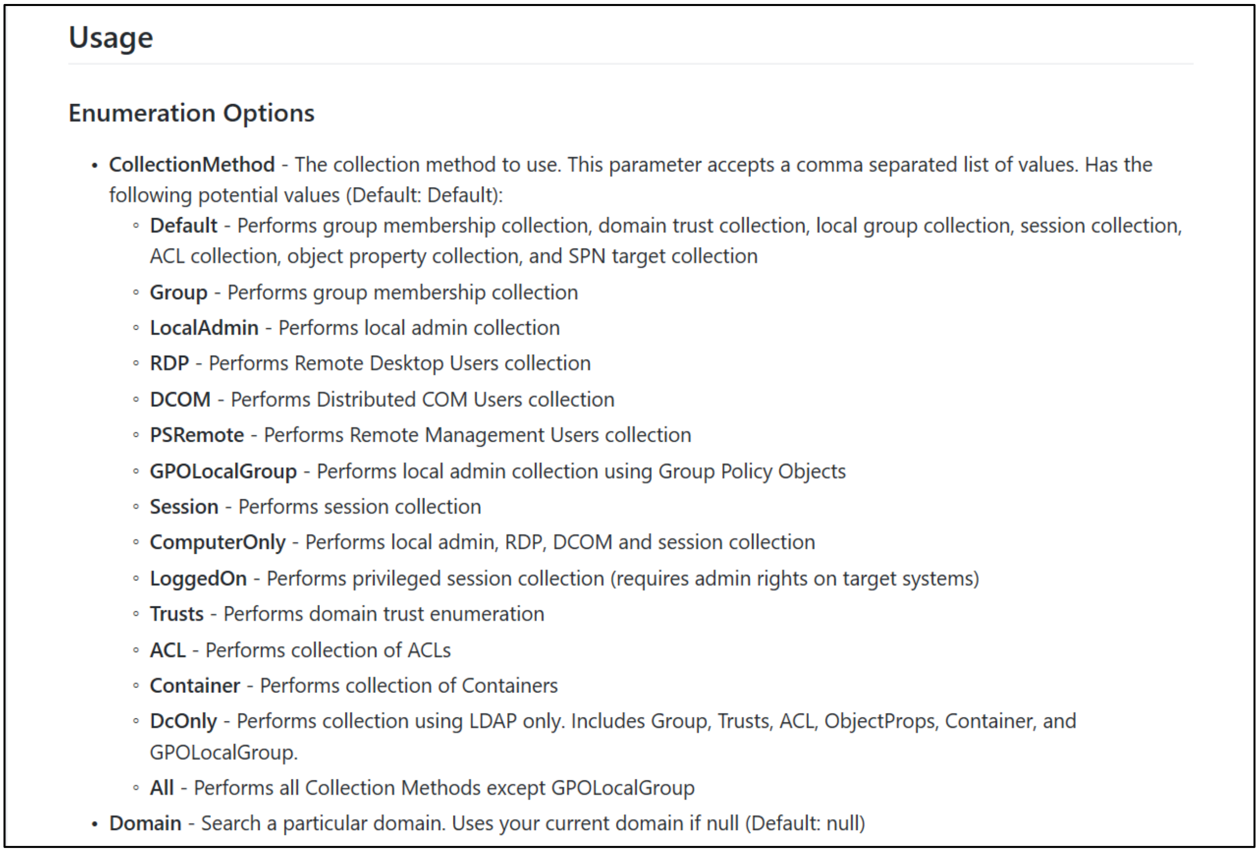
3.4. Data Collection

Ok, so last thing we need before being able to finally have a play with BloodHound is some data.
One of the important components of Bloodhound is SharpHound, the BloodHound data collector.

Sharphound comes in two flavors. An executable (.exe) and a PowerShell script (.ps1) and is what a red teamer would run in the targeted domain to collect all the information needed for mapping.
The ps1 is a wrapper holding same binary than executable as an encoded byte string.

If you are interested in how collection works, and I guess you should, you now are wondering how it's done.
CptJesus has got you covered in the following post and I guess you should read it carefully.

List of available switches for the executable can be found here.

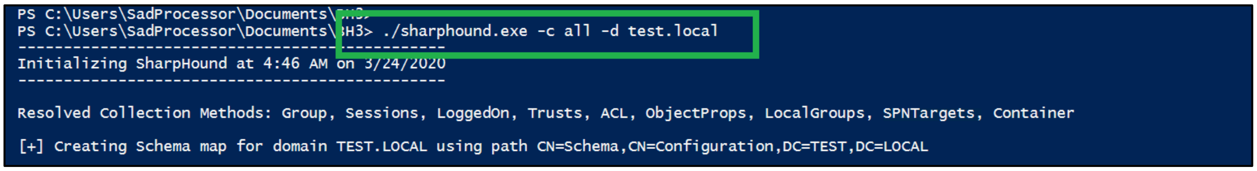


Usage

Enumeration Options

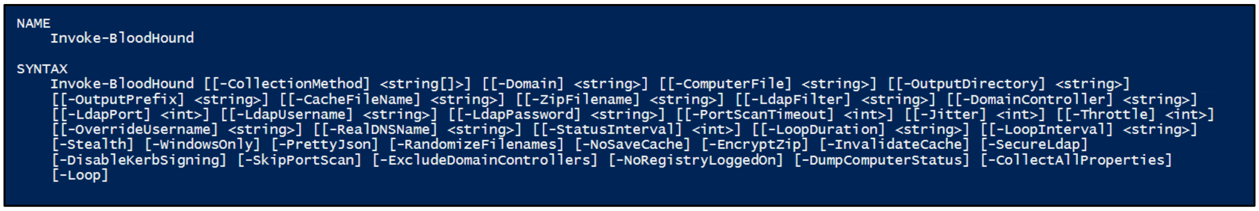
- **CollectionMethod** - The collection method to use. This parameter accepts a comma separated list of values. Has the following potential values (Default: Default):
  - **Default** - Performs group membership collection, domain trust collection, local group collection, session collection, ACL collection, object property collection, and SPN target collection
  - **Group** - Performs group membership collection
  - **LocalAdmin** - Performs local admin collection
  - **RDP** - Performs Remote Desktop Users collection
  - **DCOM** - Performs Distributed COM Users collection
  - **PSRemote** - Performs Remote Management Users collection
  - **GPOLocalGroup** - Performs local admin collection using Group Policy Objects
  - **Session** - Performs session collection
  - **ComputerOnly** - Performs local admin, RDP, DCOM and session collection
  - **LoggedOn** - Performs privileged session collection (requires admin rights on target systems)
  - **Trusts** - Performs domain trust enumeration
  - **ACL** - Performs collection of ACLs
  - **Container** - Performs collection of Containers
  - **DcOnly** - Performs collection using LDAP only. Includes Group, Trusts, ACL, ObjectProps, Container, and GPOLocalGroup.
  - **All** - Performs all Collection Methods except GPOLocalGroup
- **Domain** - Search a particular domain. Uses your current domain if null (Default: null)

Running the exe can be done with following command:



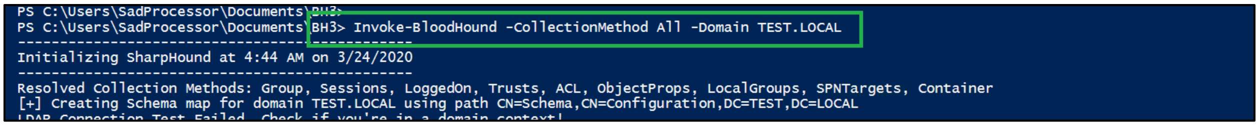
```
PS C:\Users\SadProcessor\Documents\BH3>
PS C:\Users\SadProcessor\Documents\BH3> ./sharphound.exe -c all -d test.local
-----------------------------------------------
Initializing SharpHound at 4:46 AM on 3/24/2020
-----------------------------------------------

Resolved Collection Methods: Group, Sessions, LoggedOn, Trusts, ACL, ObjectProps, LocalGroups, SPNTargets, Container

[+] Creating Schema map for domain TEST.LOCAL using path CN=Schema,CN=Configuration,DC=TEST,DC=LOCAL
```

If you are using the PowerShell script, these are the available switches:



```
NAME
    Invoke-BloodHound

SYNTAX
    Invoke-BloodHound [[-CollectionMethod] <string[]>] [[-Domain] <string>] [[-ComputerFile] <string>] [[-OutputDirectory] <string>]
    [[-OutputPrefix] <string>] [[-CacheFileName] <string>] [[-ZipFilename] <string>] [[-LdapFilter] <string>] [[-DomainController] <string>]
    [[-LdapPort] <int>] [[-LdapUsername] <string>] [[-LdapPassword] <string>] [[-PortScanTimeout] <int>] [[-Jitter] <int>] [[-Throttle] <int>]
    [[-OverrideUsername] <string>] [[-RealDNSName] <string>] [[-StatusInterval] <int>] [[-LoopDuration] <string>] [[-LoopInterval] <string>]
    [-Stealth] [-WindowsOnly] [-PrettyJson] [-RandomizeFilenames] [-NoSaveCache] [-EncryptZip] [-InvalidateCache] [-SecureLdap]
    [-DisableKerbSigning] [-SkipPortScan] [-ExcludeDomainControllers] [-NoRegistryLoggedOn] [-DumpComputerStatus] [-CollectAllProperties]
    [-Loop]
```

Running the PowerShell version can be done as follows:



```
PS C:\Users\SadProcessor\Documents\BH3>
PS C:\Users\SadProcessor\Documents\BH3> Invoke-BloodHound -CollectionMethod All -Domain TEST.LOCAL
-----------------------------------------------
Initializing SharpHound at 4:44 AM on 3/24/2020
-----------------------------------------------
Resolved Collection Methods: Group, Sessions, LoggedOn, Trusts, ACL, ObjectProps, LocalGroups, SPNTargets, Container
[+] Creating Schema map for domain TEST.LOCAL using path CN=Schema,CN=Configuration,DC=TEST,DC=LOCAL
LDAP Connection Test Failed. Check if you're in a domain context!
```

SharpHound outputs a zip file containing json objects.

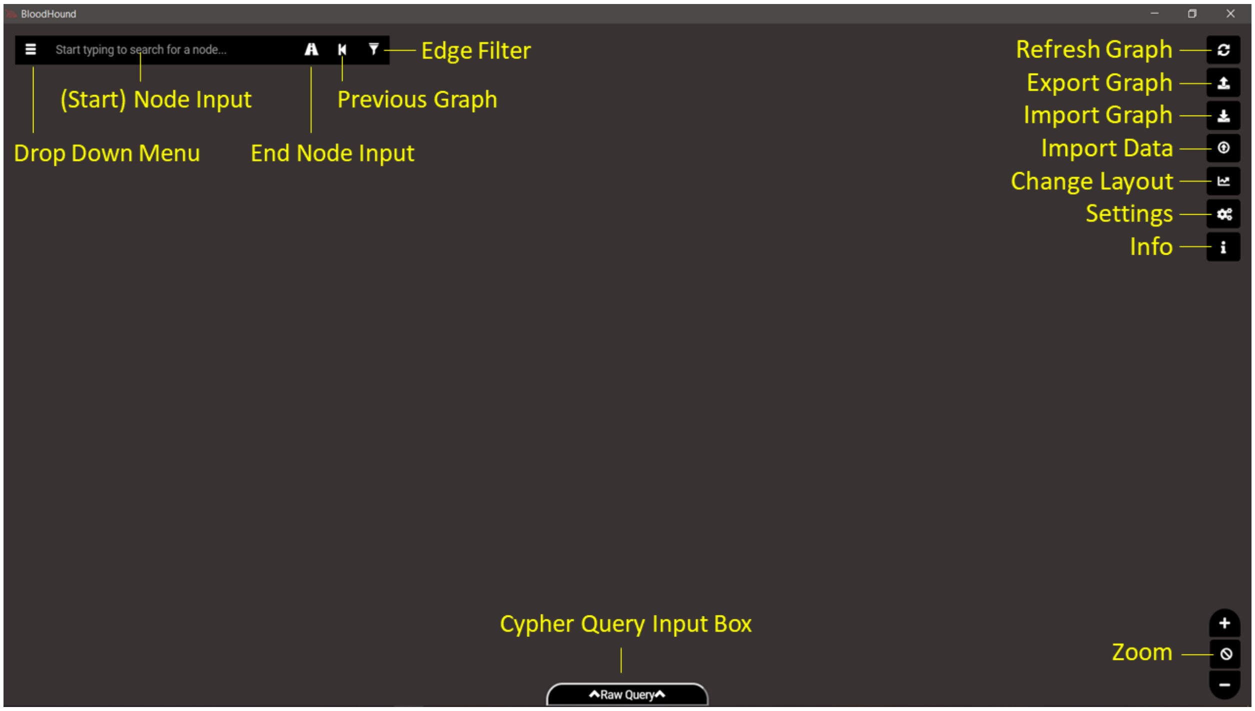
To ingest data on the bloodhound side, drag and drop the sharphound zip file into an empty part of the bloodhound graph. Once the progress bar reaches 100% the database is populated, and you are ready to start.

Collecting Azure Objects and relationships is done via AzureHound, a separate powershell script making various calls to the Azure APIs.

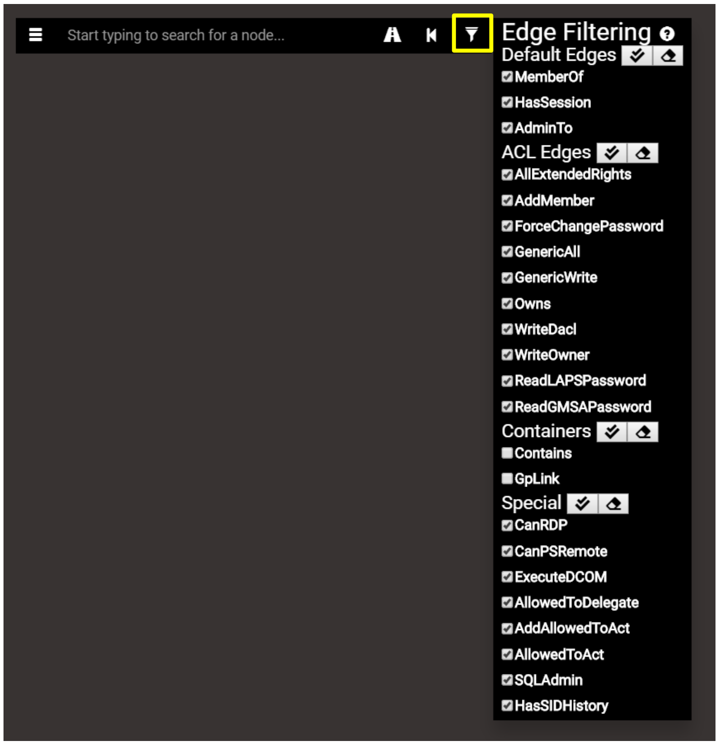
More info can be found here

3.5. User Interface

In this chapter, we will have a quick tour of the UI. Make yourself at home… Click everywhere to see what happens.



In the top right corner, the Node input allows to search for nodes. Clicking on the Highway icon will reveal a second input box for the end node, allowing to graph paths from node to node.
Clicking on the filter icon will open the edge filter drop down. Here you can select which edges are used in the path queries.



Edge Filtering

Default Edges

☑ MemberOf
☑ HasSession
☑ AdminTo

ACL Edges

☑ AllExtendedRights
☑ AddMember
☑ ForceChangePassword
☑ GenericAll
☑ GenericWrite
☑ Owns
☑ WriteDacl
☑ WriteOwner
☑ ReadLAPSPassword
☑ ReadGMSAPassword

Containers

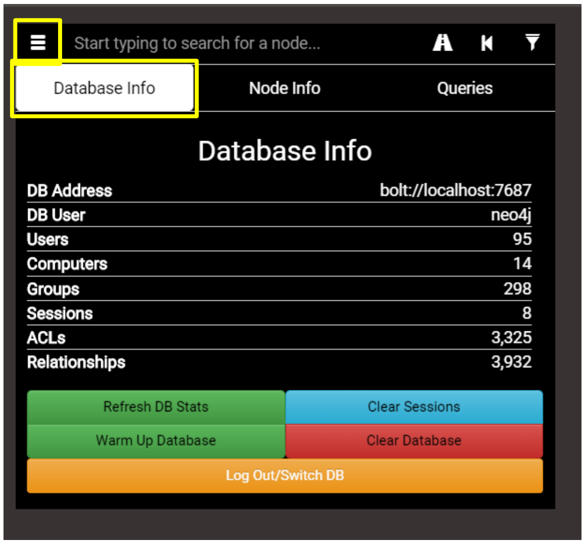
□ Contains
□ GpLink

Special

☑ CanRDP
☑ CanPSRemote
☑ ExecuteDCOM
☑ AllowedToDelegate
☑ AddAllowedToAct
☑ AllowedToAct
☑ SQLAdmin
☑ HasSIDHistory

Clicking on the back arrow will bring back previous graph.

Clicking on the menu icon will open a drop down with 3 tabs.



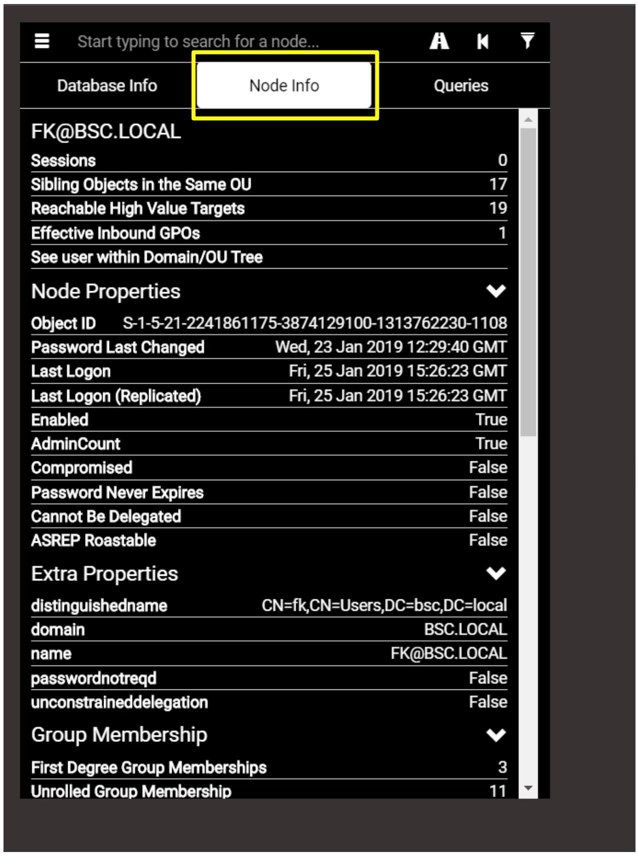
Database Info

DB Address	bolt://localhost:7687
DB User	neo4j
Users	95
Computers	14
Groups	298
Sessions	8
ACLs	3,325
Relationships	3,932

The **Database Tab** show database info and offers options to clear session data or full database.

The **Warm Up Database** button put the whole DB in RAM and speed up later queries.

If you click on a node on the graph, this will open the **Node Info Tab**



FK@BSC.LOCAL

Sessions	0
Sibling Objects in the Same OU	17
Reachable High Value Targets	19
Effective Inbound GPOs	1
See user within Domain/OU Tree	

Node Properties

Object ID	S-1-5-21-2241861175-3874129100-1313762230-1108
Password Last Changed	Wed, 23 Jan 2019 12:29:40 GMT
Last Logon	Fri, 25 Jan 2019 15:26:23 GMT
Last Logon (Replicated)	Fri, 25 Jan 2019 15:26:23 GMT
Enabled	True
AdminCount	True
Compromised	False
Password Never Expires	False
Cannot Be Delegated	False
ASREP Roastable	False

Extra Properties

distinguishedname	CN=fk,CN=Users,DC=bsc,DC=local
domain	BSC.LOCAL
name	FK@BSC.LOCAL
passwordnotreqd	False
unconstraineddelegation	False

Group Membership

First Degree Group Memberships	3
Unrolled Group Membership	11

Clicking on the numbers on the right side of the pane will display matching graph.

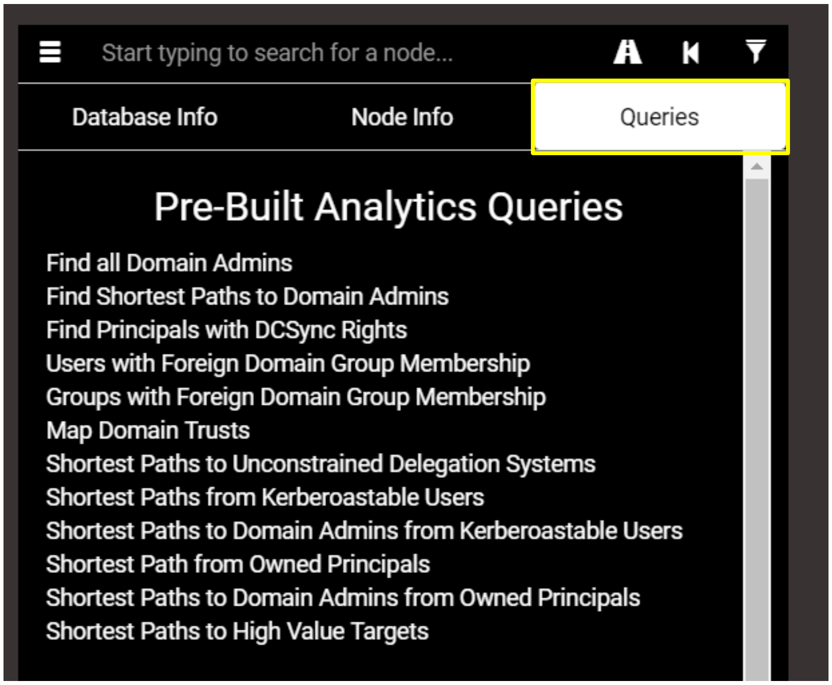
This is how you navigate the BloodHound data.

Third tab is the **Query Tab**.

Here you will find a lot of interesting pre-build analytic queries to visualize your data.

This should keep you busy for a while...

And later you can add your own custom queries if you like.

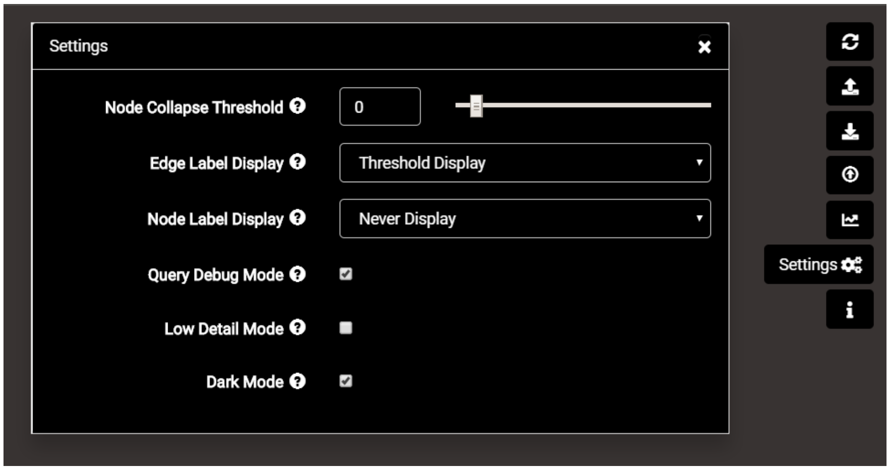


Pre-Built Analytics Queries

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

On the top right side of the UI are a few more buttons. Mostly self-explanatory.

Clicking on the Setting icon will open a menu with a few handy options.



Settings

Node Collapse Threshold	0
Edge Label Display	Threshold Display
Node Label Display	Never Display
Query Debug Mode	☑
Low Detail Mode	□
Dark Mode	☑

Here you can control how Nodes are collapsed, and how Node names and Edge labels are shown.

Query Debug Mode will display matching cypher in Query Box each time you click on something.

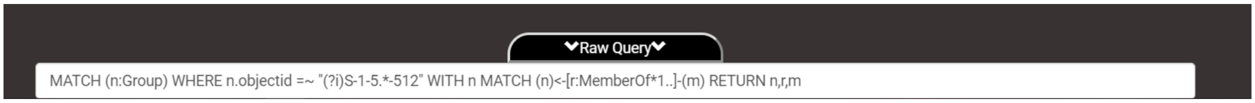
I highly recommend you turn it on if you want to learn cypher.

Dark Mode is also a must. But that's only my opinion.

On the bottom right corner is the **Zoom** function (in/out/reset)

Finally, and most importantly, hidden in the botton is that little raw query tab.

Clicking on it will reveal the **Cypher Query Input Box**...



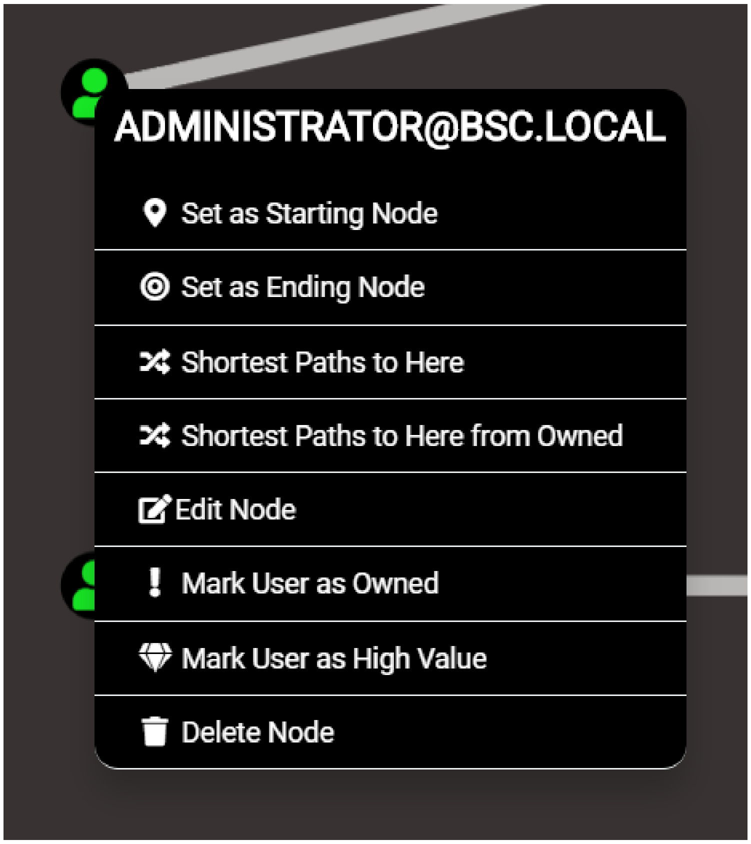
```
MATCH (n:Group) WHERE n.objectid =~ "(?i)S-1-5.*-512" WITH n MATCH (n)<-[r:MemberOf*1..]-(m) RETURN n,r,m
```

This is where the magic happens... But we have to learn some Cypher first.

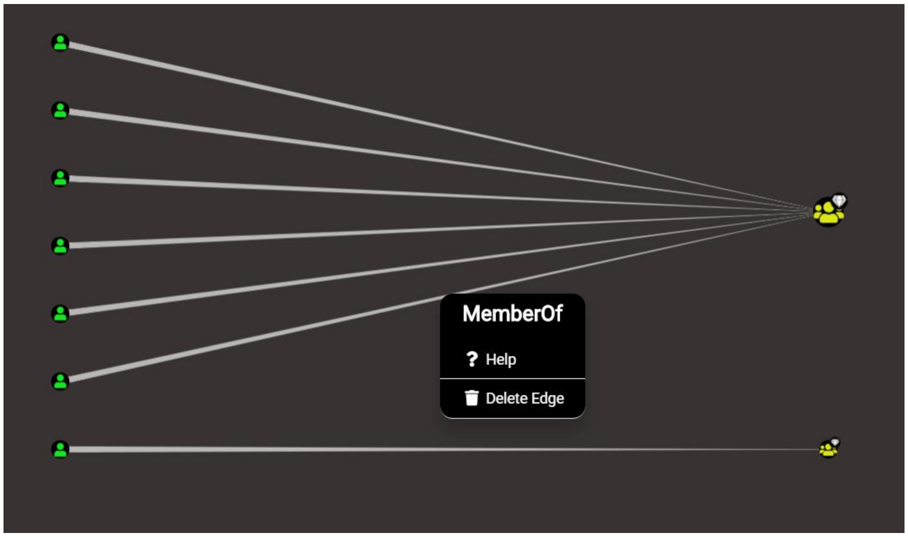
[Tip: Use **{}** to use edgefilter edges in raw query]

There are a few more menus.

If you right click on a node:

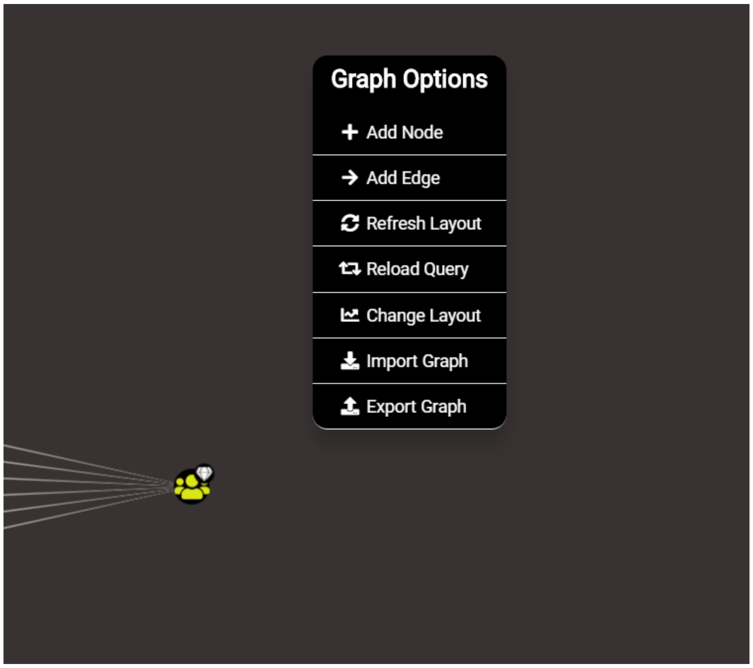


If you right-click on an edge:



Clicking on **Help** will bring up info on how to abuse this edge (including commands!)

If you right-click on empty graph space:



All of them are quite self-explanatory, so I'll let you explore rather than try to explain everything…

Last but not least, a few keyboard shortcuts:

Key	Action
[SPACE]	Node Search

[CTRL]	Node Names ON/OFF
[CTRL+SHIFT+I]	Dev Console / Debug
[CTRL+R]	Restart Bloodhound

4. CYPHER

A lot can be done via the UI, but you can do even more if you use Cypher, the neo4j database query language.
In this chapter, I'll share just a few examples that should be enough to get you started. Once you understood how it works, a lot of info can be found online, and the Cypher Reference Card is your best friend.

4.1. Cypher – Basics

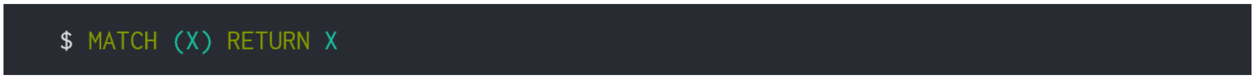
Cypher is a very "visual" language. It was designed with ASCII art in mind. Who doesn't love ASCII art...
A simplified cypher path query could look like this:

(This)-[IsConnectedTo]->(That)

This and **That** are Nodes. **IsConnectedTo** is the Edge between them, but this is not what I want to highlight here.

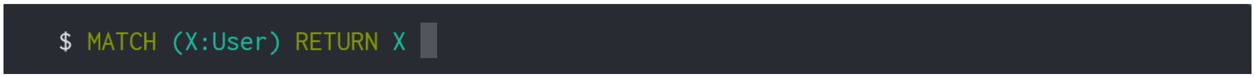
The important parts are the **brackets and arrow** (the ASCII art). This is your basic Path query construct.
It's a bit confusing at first, but you will get used to it very quickly. Now let's look at some real query syntax...

In cypher, your two basic instructions will be **MATCH** and **RETURN**.
MATCH will instruct neo4j what to look for.
RETURN indicates what results you want to see.



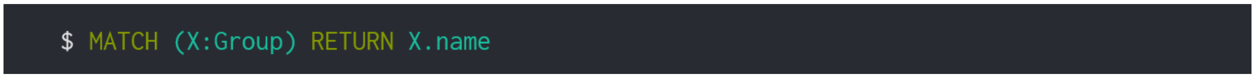
```
$ MATCH (X) RETURN X
```

returns all Nodes in the database



```
$ MATCH (X:User) RETURN X
```

returns all Users in the database

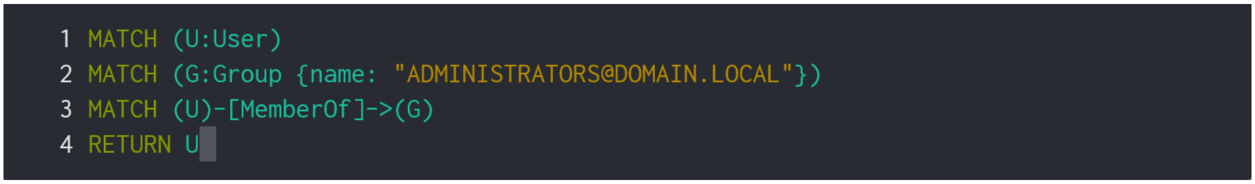


```
$ MATCH (X:Group) RETURN X.name
```

returns only the name property of all Groups

Note: The type of node is called a "**Label**" in the official neo4j vocabulary.

Now we can make things a bit more interesting. Let's return all users member of a specific group:

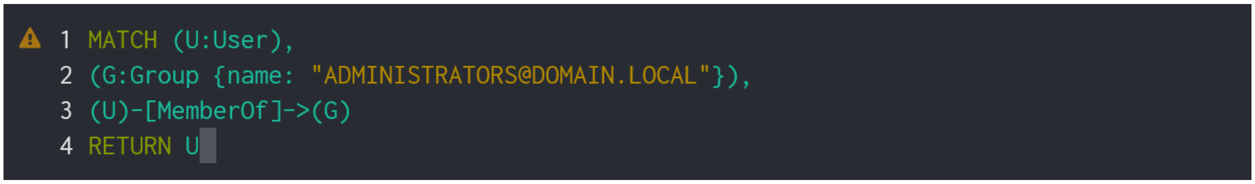


```
MATCH (U:User)
MATCH (G:Group {name: "ADMINISTRATORS@DOMAIN.LOCAL"})
MATCH (U)-[MemberOf]->(G)
RETURN U
```

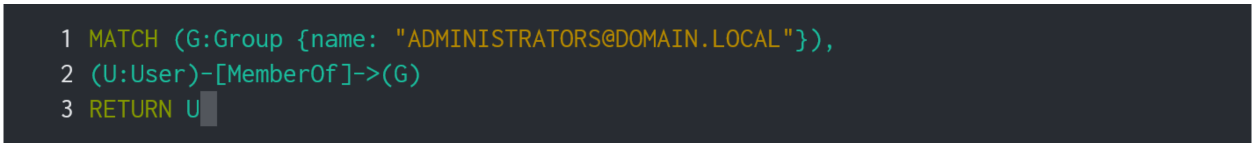
Here we first ask for all users and store it in a variable U, then for a group called "ADMINISTRATORS@DOMAIN.LOCAL" and we store it in a variable G.

From that list of users U, we filter who is member of the specified group G, and finally return these User nodes.

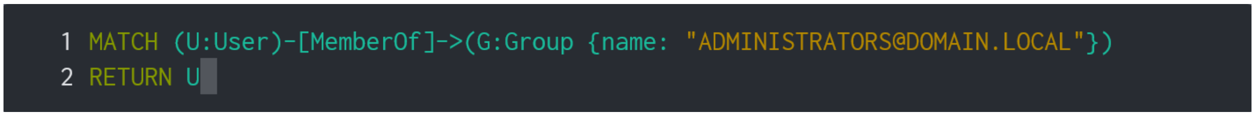
This query can also be written with the following **equivalent syntaxes**:



```
MATCH (U:User),
(G:Group {name: "ADMINISTRATORS@DOMAIN.LOCAL"}),
(U)-[MemberOf]->(G)
RETURN U
```



```
MATCH (G:Group {name: "ADMINISTRATORS@DOMAIN.LOCAL"}),
(U:User)-[MemberOf]->(G)
RETURN U
```



```
MATCH (U:User)-[MemberOf]->(G:Group {name: "ADMINISTRATORS@DOMAIN.LOCAL"})
RETURN U
```

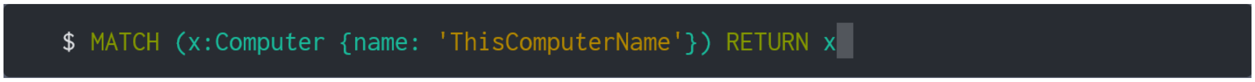
Note: Cypher language is case-sensitive, Proper casing of Nodes properties, Labels and other syntax elements is the first thing to check when debugging hanging queries...

(Do not worry about the warning icons for now, more on this later...)

Things will get more complicated as we dig deeper, but for now, if you understood the above syntaxes, you are good to go.

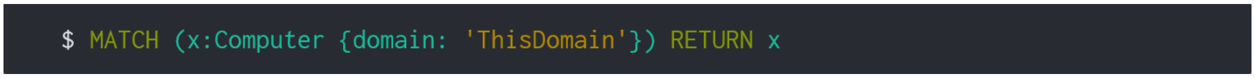
Note: Cypher queries in this guide cannot be copy-pasted. This was done on purpose. The idea is that you type them.

4.1.1. Querying Nodes



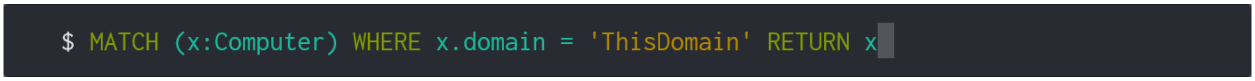
```
$ MATCH (x:Computer {name: 'ThisComputerName'}) RETURN x
```

Returns Computer nodes with name 'ThisComputerName'



```
$ MATCH (x:Computer {domain: 'ThisDomain'}) RETURN x
```

Returns Computer nodes where the domain property is equal to 'ThisDomain'

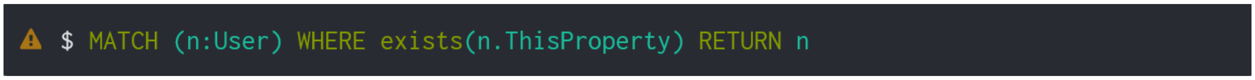


```
$ MATCH (x:Computer) WHERE x.domain = 'ThisDomain' RETURN x
```

Same as previous using the WHERE clause

The **WHERE** clause is used to filter Nodes per property. It is used in combination with Comparison Operators. In case of “Is Equal To” comparison, the shorter construct (“Map”) is preferred.

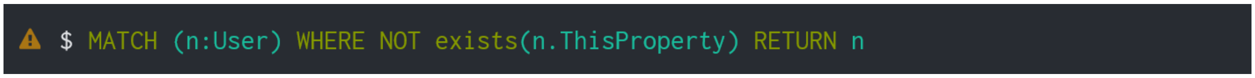
4.1.1.1. Node by Property - Property Exists



```
⚠ $ MATCH (n:User) WHERE exists(n.ThisProperty) RETURN n
```

Returns all Nodes that have a property 'ThisProperty' (value or not)

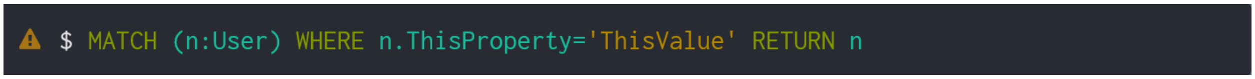
4.1.1.2. Node by Property - Does Not Exists



```
⚠ $ MATCH (n:User) WHERE NOT exists(n.ThisProperty) RETURN n
```

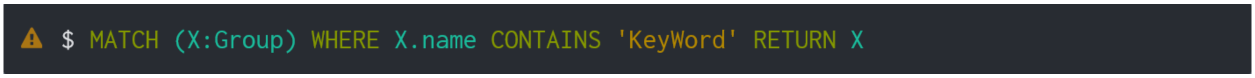
Returns all Users that don’t have a property called 'ThisProperty'

4.1.1.3. Node by Property - Property Value



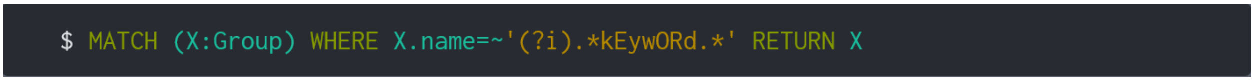
```
⚠ $ MATCH (n:User) WHERE n.ThisProperty='ThisValue' RETURN n
```

Returns all Users that have a property 'ThisProperty' with value 'ThisValue'



```
⚠ $ MATCH (X:Group) WHERE X.name CONTAINS 'KeyWord' RETURN X
```

Returns All Groups with 'KeyWord' in name property (case sensitive)



```
$ MATCH (X:Group) WHERE X.name=~'(?i).*kEywORd.*' RETURN X
```

Same as previous example but using RegEx [(?i) = case insensitive]

4.1.1.4. Comparison Operators

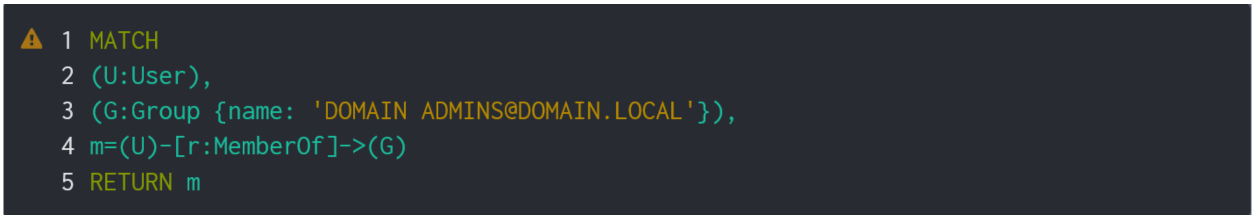
List of operators that can be used with the WHERE clause:

OPERATOR	SYNTAX
Is Equal To	=
Is Not Equal To	<>
Is Less Than	<
Is Greater Than	>
Is Less or Equal	<=
Is Greater or Equal	>=
Is Null	IS NULL
Is Not Null	IS NOT NULL
Prefix Search*	STARTS WITH
Suffix Search*	ENDS WITH
Inclusion Search*	CONTAINS
RegEx*	=~

* String specific

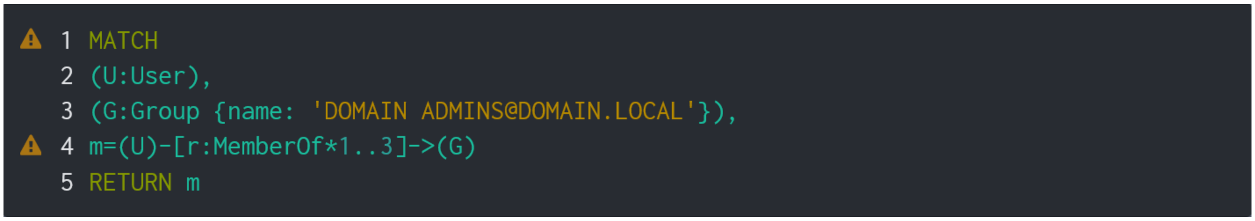
4.1.2. Querying Edges

4.1.2.1. Group Membership – Direct



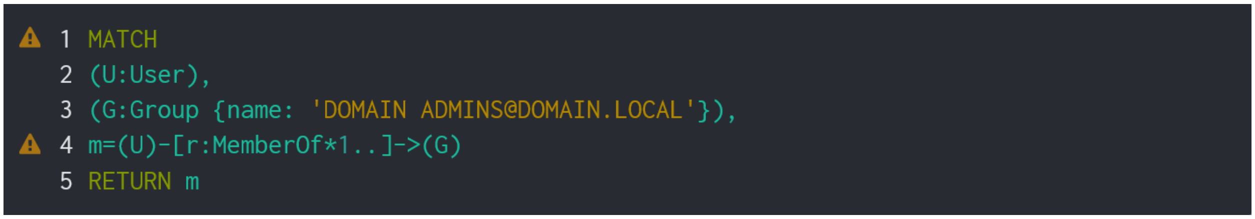
```
MATCH
(U:User),
(G:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
m=(U)-[r:MemberOf]->(G)
RETURN m
```

4.1.2.2. Group Membership – Max Degree 3



```
MATCH
(U:User),
(G:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
m=(U)-[r:MemberOf*1..3]->(G)
RETURN m
```

4.1.2.3. Group Membership – Any Degree

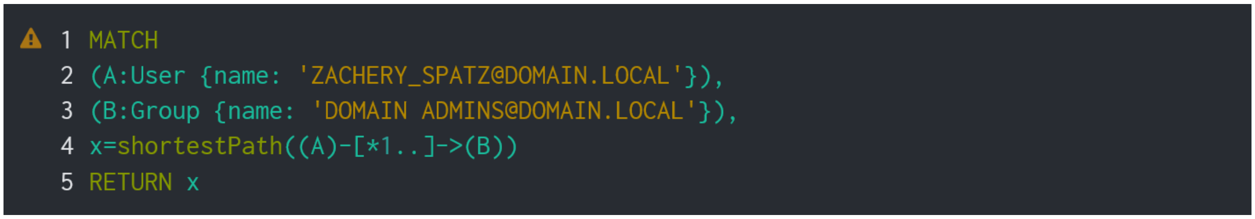


```
MATCH
(U:User),
(G:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
m=(U)-[r:MemberOf*1..]->(G)
RETURN m
```

Note: Here we return paths to visualize nested groups in BloodHound. If you want to return just the User Nodes you can replace m by U in the RETURN clause of the queries.

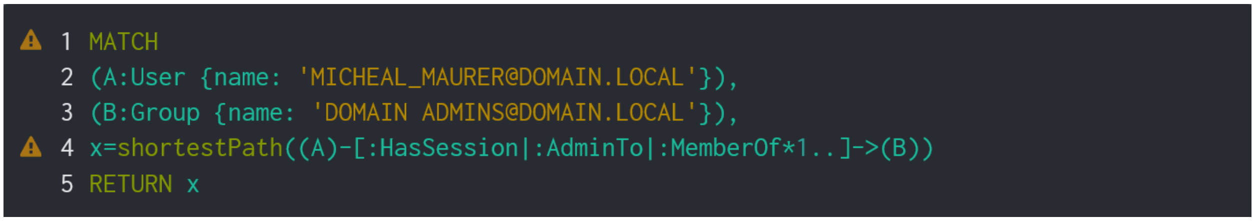
4.1.3. Querying Paths

4.1.3.1. Shortest Path from A to B - any Edge type / One or more hops



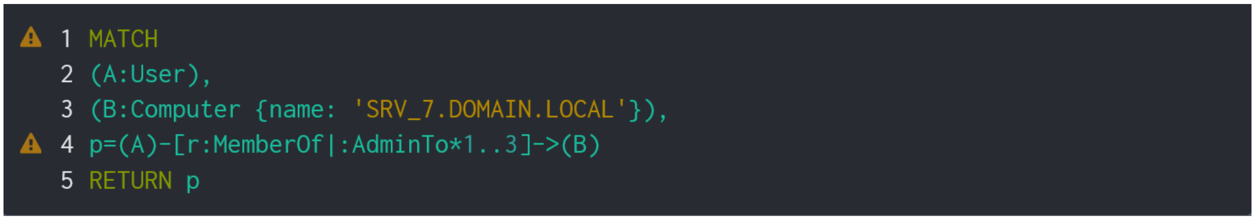
```
MATCH
(A:User {name: 'ZACHERY_SPATZ@DOMAIN.LOCAL'}),
(B:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
x=shortestPath((A)-[*1..]->(B))
RETURN x
```

4.1.3.2. Shortest Path from A to B - specific Edge types / One or more hops



```
MATCH
(A:User {name: 'MICHEAL_MAURER@DOMAIN.LOCAL'}),
(B:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
x=shortestPath((A)-[:HasSession|:AdminTo|:MemberOf*1..]->(B))
RETURN x
```

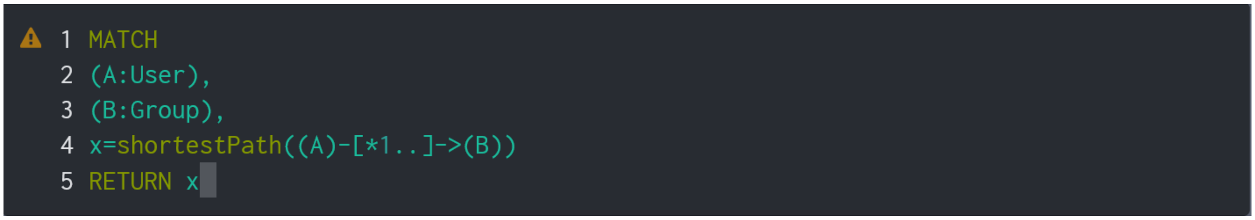
4.1.3.3. Shortest Path Any to One – Specific Edge type / Max hop count



```
MATCH
(A:User),
(B:Computer {name: 'SRV_7.DOMAIN.LOCAL'}),
p=(A)-[r:MemberOf|:AdminTo*1..3]->(B)
RETURN p
```

All user, max 3 degrees away by group membership, admin to specified target computer

4.1.3.4. Shortest Path Any to Any

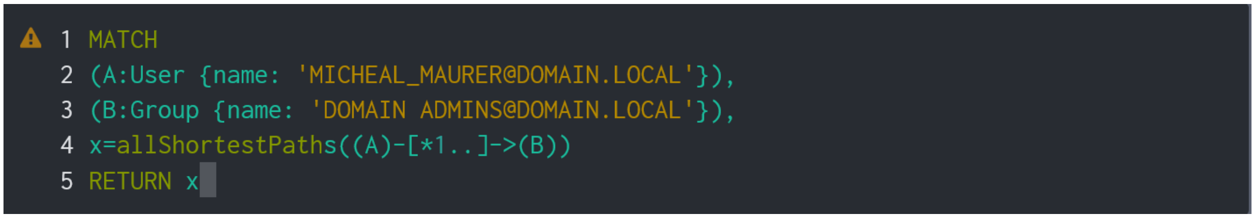


```
MATCH
(A:User),
(B:Group),
x=shortestPath((A)-[*1..]->(B))
RETURN x
```

Shortest paths from any user to any group

/!\ Any-to-Any are heavy queries and might hang with large datasets

4.1.3.5. All Shortest Paths



```
MATCH
(A:User {name: 'MICHEAL_MAURER@DOMAIN.LOCAL'}),
(B:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
x=allShortestPaths((A)-[*1..]->(B))
RETURN x
```

The **allShortestPaths()** function works the same way as shortestPath() but returns all possible shortest paths [= more ways to get to target with same amount of hops]

/!\ Might need to restrict Edge type/max hops for heavy queries

4.1.3.6. All Paths

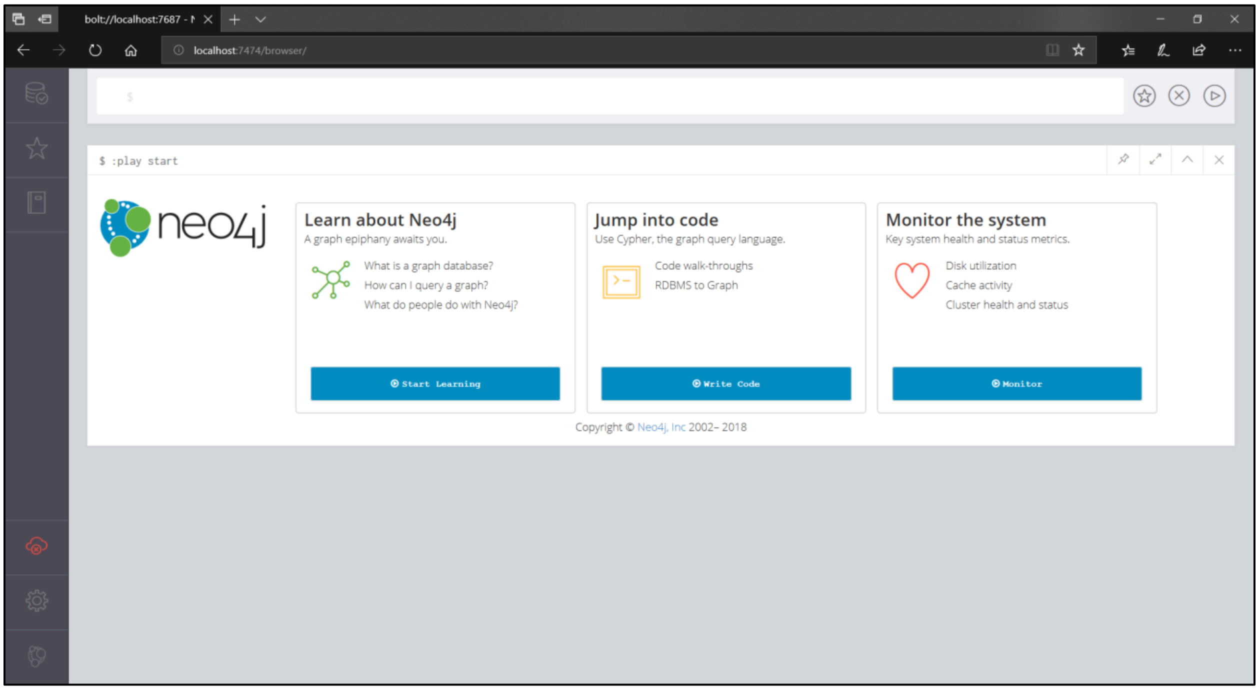
It is possible to request all available paths, even the longer ones, if you remove the **shortestPath()** or **allShortestPaths()** from your queries. This is however risky, and your query might hang. Make sure you specify at least one node name when using it. Do not try this on an Any-to-Any query.

There is way more to the Cypher language, but with the above syntaxes, you should be able to graph most of what you need from the bloodhound database.

4.2. Neo4j Browser

BloodHound is awesome for visualizing complex Active Directory object relationships in a graphical way. However, the graph output is not always what you are looking for.
Let's say you want to count how many users are members of a specific group and have a specific property with cypher… Or you want the list of names… The UI is not really designed for this type of queries.

For these queries, you can use the **Neo4j browser** located at http://localhost:7474/Browser
this will take you to the following page:



```
$ :play start
```

Learn about Neo4j
A graph epiphany awaits you.
What is a graph database?
How can I query a graph?
What do people do with Neo4j?

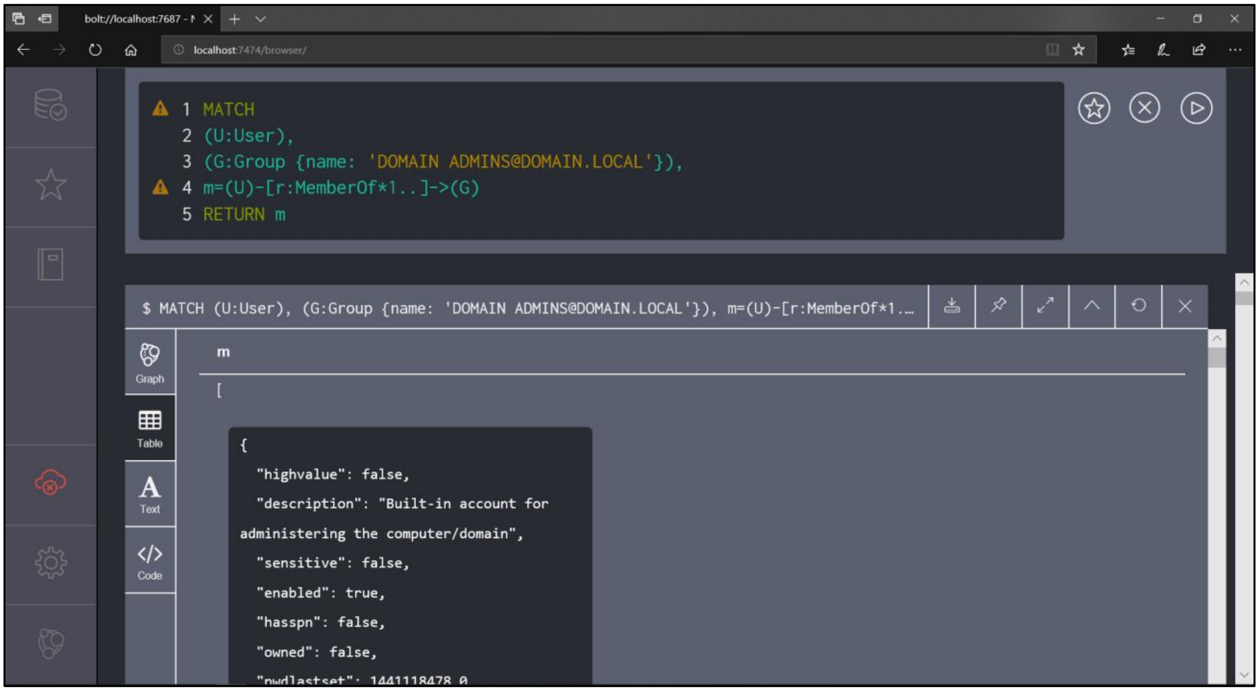
Jump into code
Use Cypher, the graph query language.
Code walk-throughs
RDBMS to Graph

Monitor the system
Key system health and status metrics.
Disk utilization
Cache activity
Cluster health and status

Copyright © Neo4j, Inc 2002– 2018

The neo4j browser is an awesome place to practice writing your queries. You now have multiline, syntax highlighting, error messages, and you can even set the font size (ctrl +/-).

The interface is quite simple and intuitive, I'll let you have a look around and click everywhere…



```
MATCH
(U:User),
(G:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}),
m=(U)-[r:MemberOf*1..]->(G)
RETURN m
```

```
$ MATCH (U:User), (G:Group {name: 'DOMAIN ADMINS@DOMAIN.LOCAL'}), m=(U)-[r:MemberOf*1...
```

```
m
[
  {
    "highvalue": false,
    "description": "Built-in account for
administering the computer/domain",
    "sensitive": false,
    "enabled": true,
    "hasspn": false,
    "owned": false,
    "pwdlastset": 1441118478.0
```

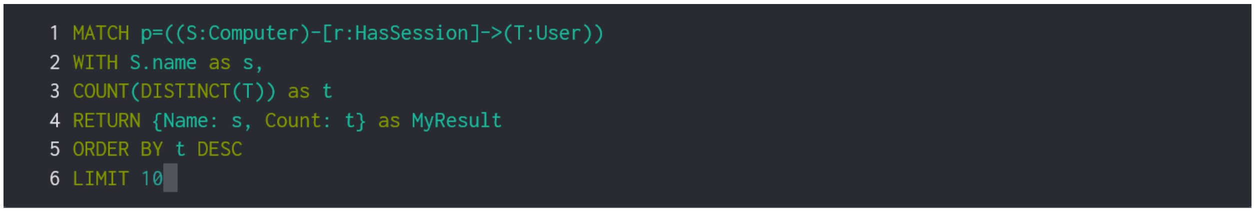
TIP: When in query input box, hit the Up arrow for previous queries (or Ctrl+Up if multiline). Hit escape to toggle editor mode. If you really like that query you just wrote, save for future usage by hitting on the star next to the cross and the play buttons. Hit the star on the left sidebar to view all your saves queries.

4.3. Cypher – Advanced

4.3.1. Calculating Metrics

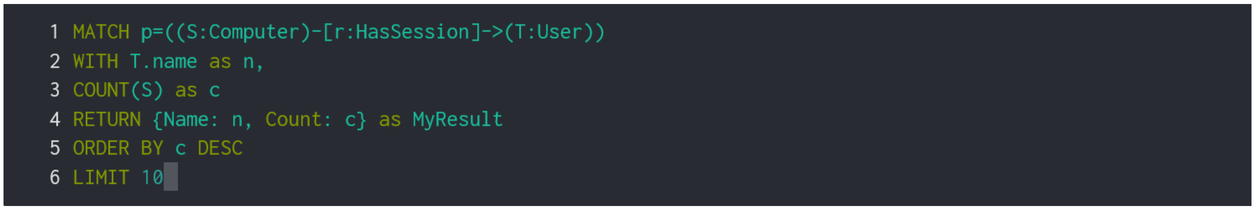
For this purpose, we will now introduce new cypher building blocks: the **count()** function, and the **ORDER BY** and **LIMIT** clauses

4.3.1.1. Top 10 Computers with Most Sessions



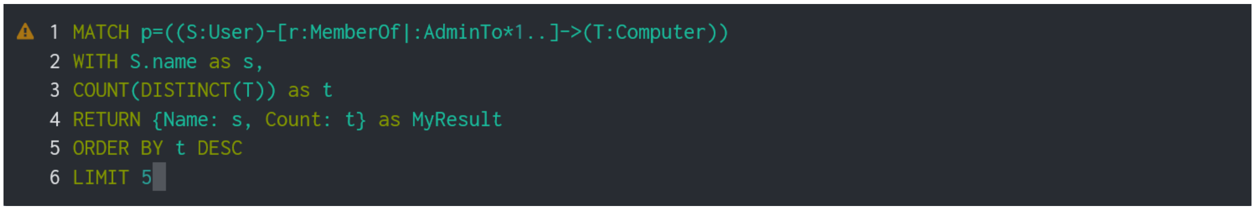
```
MATCH p=((S:Computer)-[r:HasSession]->(T:User))
WITH S.name as s,
COUNT(DISTINCT(T)) as t
RETURN {Name: s, Count: t} as MyResult
ORDER BY t DESC
LIMIT 10
```

4.3.1.2. Top 10 Users with Most Sessions



```
MATCH p=((S:Computer)-[r:HasSession]->(T:User))
WITH T.name as n,
COUNT(S) as c
RETURN {Name: n, Count: c} as MyResult
ORDER BY c DESC
LIMIT 10
```

4.3.1.3. Top 5 Users with Most Admin Rights



```
MATCH p=((S:User)-[r:MemberOf|:AdminTo*1..]->(T:Computer))
WITH S.name as s,
COUNT(DISTINCT(T)) as t
RETURN {Name: s, Count: t} as MyResult
ORDER BY t DESC
LIMIT 5
```

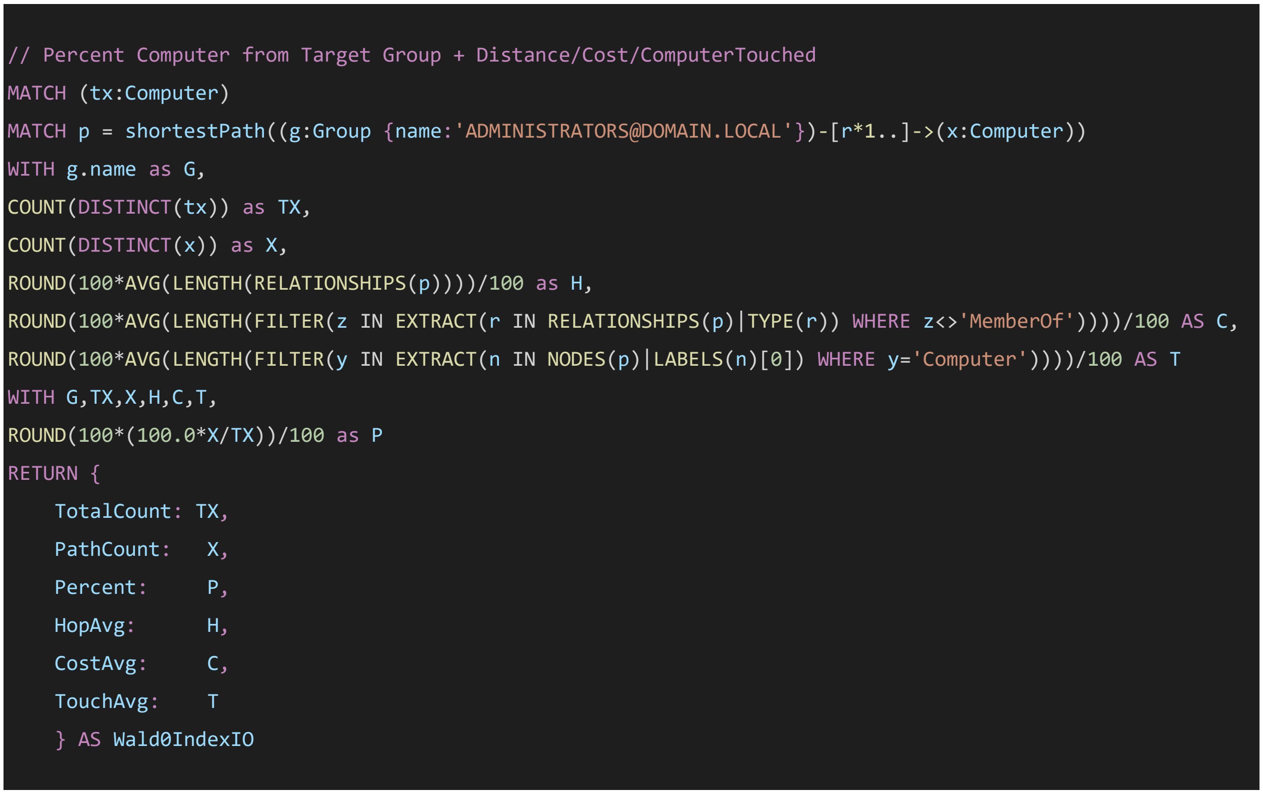
Notice the cool "Map" construct on the RETURN clause. This is a cool tip to build nice objects on the fly and return only the data you want. You are going to love this trick if you work with the REST API.

Using constructs similar to the above queries, you can count about anything you like in BloodHound.

4.3.1.4. Extreme Example

There is a lot of things that can be done in cypher, and this guide can only scratch the surface.

Below an example of a advanced query to calculate percentage of computers that can be reached from a specific group, with average distance, attack “cost” (MemberOf = No cost) and number of computer touched on the way…



```
// Percent Computer from Target Group + Distance/Cost/ComputerTouched
MATCH (tx:Computer)
MATCH p = shortestPath((g:Group {name:'ADMINISTRATORS@DOMAIN.LOCAL'})-[r*1..]->(x:Computer))
WITH g.name as G,
COUNT(DISTINCT(tx)) as TX,
COUNT(DISTINCT(x)) as X,
ROUND(100*AVG(LENGTH(RELATIONSHIPS(p))))/100 as H,
ROUND(100*AVG(LENGTH(FILTER(z IN EXTRACT(r IN RELATIONSHIPS(p)|TYPE(r)) WHERE z<>'MemberOf'))))/100 AS C,
ROUND(100*AVG(LENGTH(FILTER(y IN EXTRACT(n IN NODES(p)|LABELS(n)[0]) WHERE y='Computer'))))/100 AS T
WITH G,TX,X,H,C,T,
ROUND(100*(100.0*X/TX))/100 as P
RETURN {
    TotalCount: TX,
    PathCount:   X,
    Percent:     P,
    HopAvg:      H,
    CostAvg:     C,
    TouchAvg:    T
    } AS Wald0IndexIO
```

This is as extreme as it gets in this guide, and just here to give an idea of what it would look like.

An easier option is to work via the REST API to return objects and do the math on the client side using your favorite scripting language (see chapter 5)

4.3.2. Manipulating Data

Until now we have been querying data from the database. But what if I told you can also manipulate that database with Cypher…

For this will use the **SET**, **REMOVE** & **DETACH DELETE** instructions.

You might be asking:"why would I want to manipulate the data?".

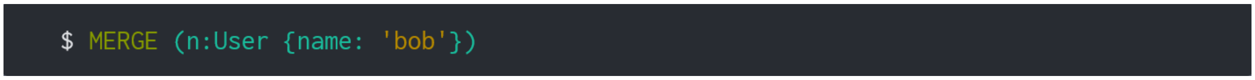
That's a good question. Here are a few scenarios where you might need it:

- Marking nodes as Owned/HighValue at scale…
- Deleting all sessions in one go for a specific computer or more…
- Removing "out-of-scope" nodes from the graph…
- Adding non-windows systems to the graph… (linux jumphosts etc…)
- Adding your own custom properties to nodes during an engagement…
- Simulating changes to the AD infrastructure…
- Testing environment hardening hypothesis…
- Adding your attack infrastructure to the BloodHound graph… Why not?

Anything you can think of really, BloodHound is just a database, you can do what you want with it.

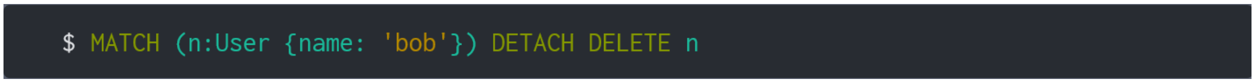
Be creative, share your ideas on slack… Hack the planet!

4.3.2.1. Creating/Deleting Nodes



```
$ MERGE (n:User {name: 'bob'})
```

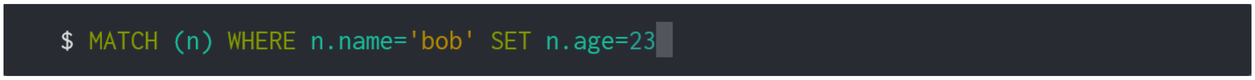
Creates a User named "bob" if it doesn't already exist



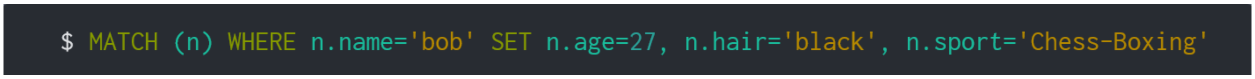
```
$ MATCH (n:User {name: 'bob'}) DETACH DELETE n
```

Delete that User (and connected Edges of course…)

4.3.2.2. Adding/Updating/Removing Node property

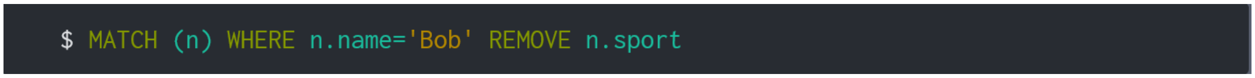


```
$ MATCH (n) WHERE n.name='bob' SET n.age=23
```

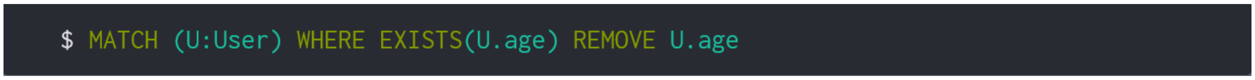


```
$ MATCH (n) WHERE n.name='bob' SET n.age=27, n.hair='black', n.sport='Chess-Boxing'
```

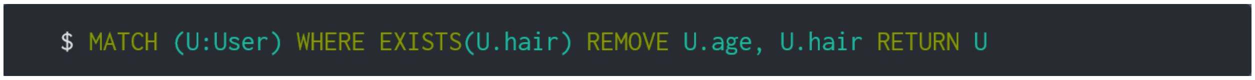
Both create missing properties and overwrite existing property values



```
$ MATCH (n) WHERE n.name='Bob' REMOVE n.sport
```



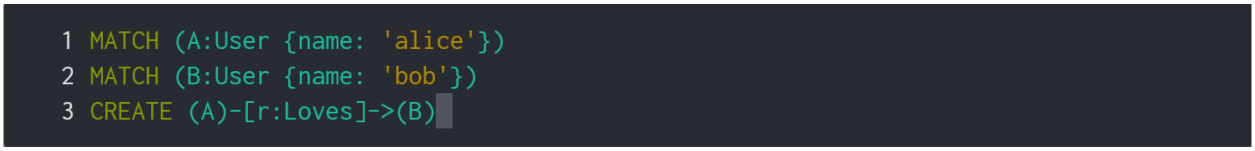
```
$ MATCH (U:User) WHERE EXISTS(U.age) REMOVE U.age
```



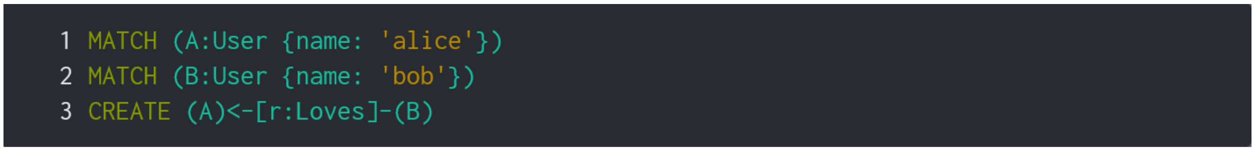
```
$ MATCH (U:User) WHERE EXISTS(U.hair) REMOVE U.age, U.hair RETURN U
```

Remove property from node (Single Node / multiple Nodes / multiple props)

4.3.2.3. Creating/Removing Edges



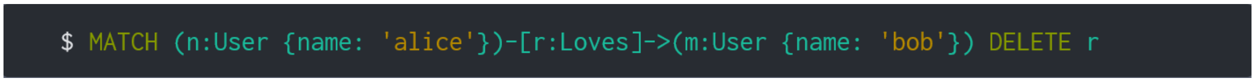
```
MATCH (A:User {name: 'alice'})
MATCH (B:User {name: 'bob'})
CREATE (A)-[r:Loves]->(B)
```



```
MATCH (A:User {name: 'alice'})
MATCH (B:User {name: 'bob'})
CREATE (A)<-[r:Loves]-(B)
```

Create Edges between Nodes.

/!\ Reminder: Edges are directional.

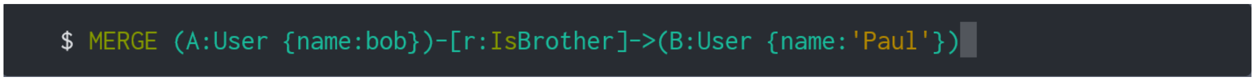


```
$ MATCH (n:User {name: 'alice'})-[r:Loves]->(m:User {name: 'bob'}) DELETE r
```

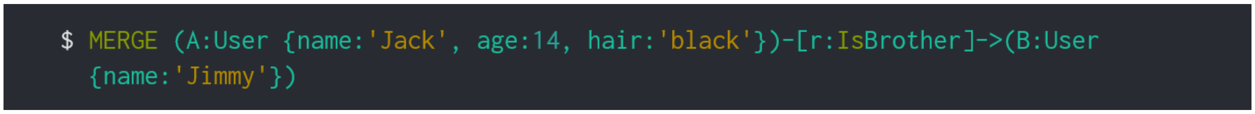
Remove Edge between Nodes

/!\ not specifying any Edge type will remove all Edges between specified Nodes

4.3.2.4. Creating Nodes with Properties & Edges



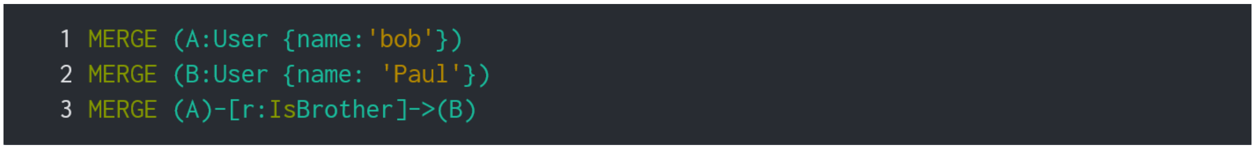
```
$ MERGE (A:User {name:bob})-[r:IsBrother]->(B:User {name:'Paul'})
```



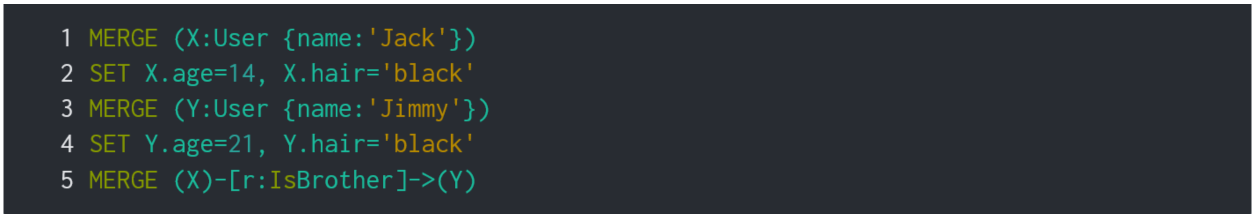
```
$ MERGE (A:User {name:'Jack', age:14, hair:'black'})-[r:IsBrother]->(B:User
  {name:'Jimmy'})
```

/!\ Use these syntaxes only if Nodes don't already exist. Otherwise use MERGE or MERGE/SET for each block separately as shown below

Recommended syntax:

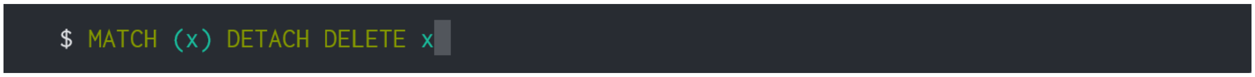


```
MERGE (A:User {name:'bob'})
MERGE (B:User {name: 'Paul'})
MERGE (A)-[r:IsBrother]->(B)
```



```
MERGE (X:User {name:'Jack'})
SET X.age=14, X.hair='black'
MERGE (Y:User {name:'Jimmy'})
SET Y.age=21, Y.hair='black'
MERGE (X)-[r:IsBrother]->(Y)
```

4.3.2.5. Nuke the DB

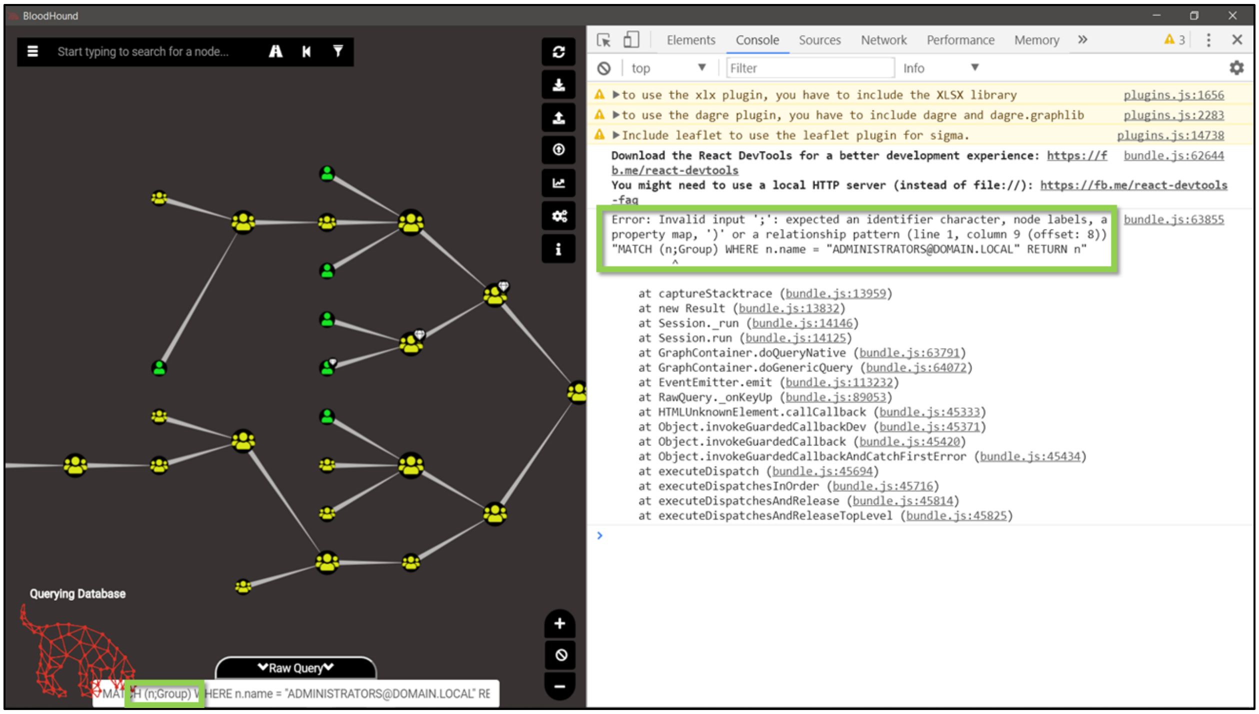


```
$ MATCH (x) DETACH DELETE x
```

4.4. Debugging Queries

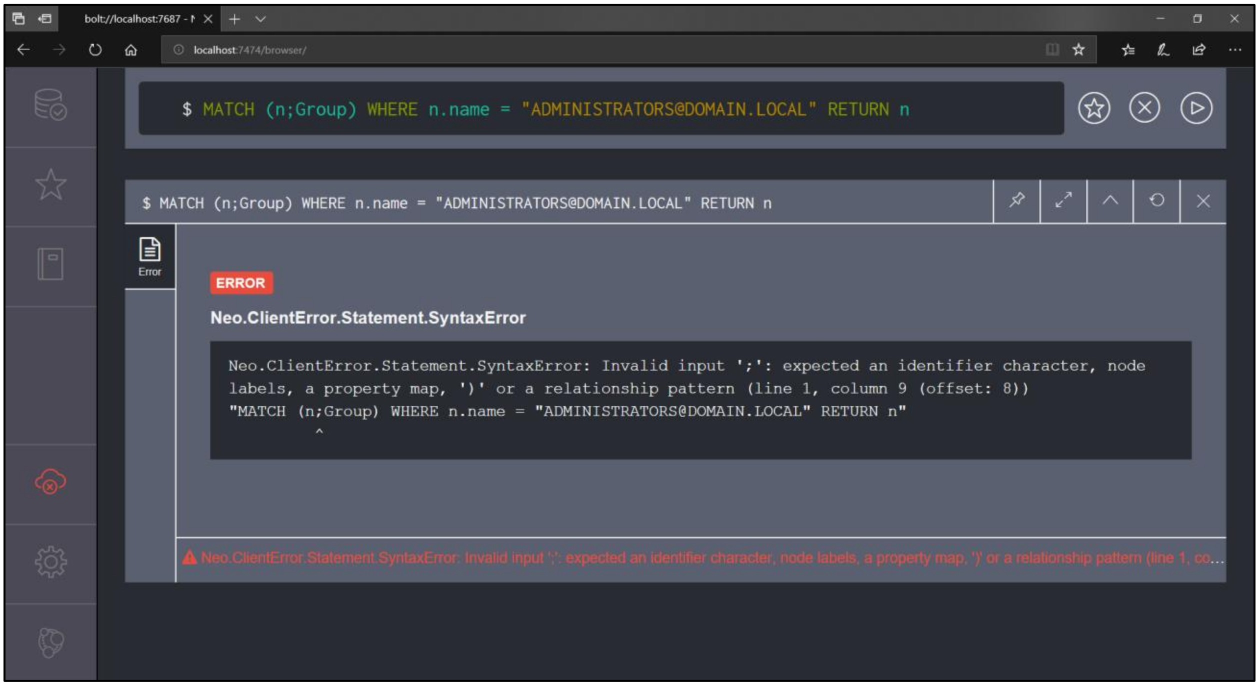
Sometimes, you just fat-finger your Cypher… it happens. In that case, the BloodHound logo animation will keep walking forever. This is the moment you have to check your Cypher syntax.

For a quick fix, you can tweak your query directly into the Raw Query input box. If you can't catch your typo there, you can hit **[CTRL+SHIFT+I]**. This will open the web dev tools, and you can find your error messages in the Console tab.


BloodHound
Start typing to search for a node...
Querying Database
Raw Query
MATCH (n;Group) WHERE n.name = "ADMINISTRATORS@DOMAIN.LOCAL" RE
Elements
Console
Sources
Network
Performance
Memory
top
Filter
Info

```
▶to use the xlx plugin, you have to include the XLSX library                plugins.js:1656
▶to use the dagre plugin, you have to include dagre and dagre.graphlib      plugins.js:2283
▶Include leaflet to use the leaflet plugin for sigma.                       plugins.js:14738
Download the React DevTools for a better development experience: https://f  bundle.js:62644
b.me/react-devtools
You might need to use a local HTTP server (instead of file://): https://fb.me/react-devtools
-faq
Error: Invalid input ';': expected an identifier character, node labels, a  bundle.js:63855
property map, ')' or a relationship pattern (line 1, column 9 (offset: 8))
"MATCH (n;Group) WHERE n.name = "ADMINISTRATORS@DOMAIN.LOCAL" RETURN n"
         ^

    at captureStacktrace (bundle.js:13959)
    at new Result (bundle.js:13832)
    at Session._run (bundle.js:14146)
    at Session.run (bundle.js:14125)
    at GraphContainer.doQueryNative (bundle.js:63791)
    at GraphContainer.doGenericQuery (bundle.js:64072)
    at EventEmitter.emit (bundle.js:113232)
    at RawQuery._onKeyUp (bundle.js:89053)
    at HTMLUnknownElement.callCallback (bundle.js:45333)
    at Object.invokeGuardedCallbackDev (bundle.js:45371)
    at Object.invokeGuardedCallback (bundle.js:45420)
    at Object.invokeGuardedCallbackAndCatchFirstError (bundle.js:45434)
    at executeDispatch (bundle.js:45694)
    at executeDispatchesInOrder (bundle.js:45716)
    at executeDispatchesAndRelease (bundle.js:45814)
    at executeDispatchesAndReleaseTopLevel (bundle.js:45825)
```

A good option when working on complex queries, is to debug them in the neo4j browser. There, you will get tips while writing them (warning icons), a bigger font size with syntax highlighting, and you will get error messages when running them. It's way easier than in Bloodhound itself.



```
$ MATCH (n;Group) WHERE n.name = "ADMINISTRATORS@DOMAIN.LOCAL" RETURN n
```

```
$ MATCH (n;Group) WHERE n.name = "ADMINISTRATORS@DOMAIN.LOCAL" RETURN n
```

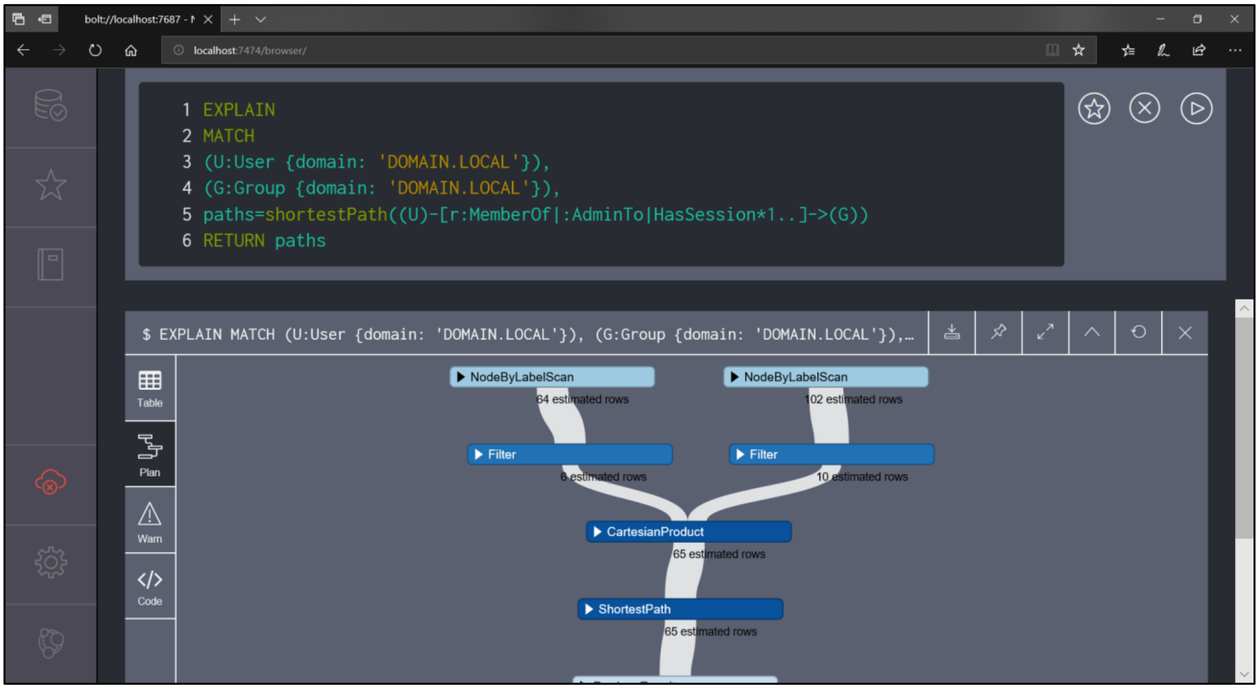
ERROR

Neo.ClientError.Statement.SyntaxError

```
Neo.ClientError.Statement.SyntaxError: Invalid input ';': expected an identifier character, node
labels, a property map, ')' or a relationship pattern (line 1, column 9 (offset: 8))
"MATCH (n;Group) WHERE n.name = "ADMINISTRATORS@DOMAIN.LOCAL" RETURN n"
         ^
```

Neo.ClientError.Statement.SyntaxError: Invalid input ';': expected an identifier character, node labels, a property map, ')' or a relationship pattern (line 1, co...

Another cool thing in the neo4j browser, is to add **EXPLAIN** or **PROFILE** in front of your query. You will get loads of info on what is happening under the neo4j hood.



```
EXPLAIN
MATCH
(U:User {domain: 'DOMAIN.LOCAL'}),
(G:Group {domain: 'DOMAIN.LOCAL'}),
paths=shortestPath((U)-[r:MemberOf|:AdminTo|HasSession*1..]->(G))
RETURN paths
```

```
$ EXPLAIN MATCH (U:User {domain: 'DOMAIN.LOCAL'}), (G:Group {domain: 'DOMAIN.LOCAL'}),…
```

4.5. More Resources

For a quick syntax check you can use the Neo4j Cypher Reference Card.

For a deeper dive, use the Neo4j Cypher Manual.

There are tons of non-bloodhound-specific Cypher resources available online. Google if needed.

For BloodHound specific queries click here, and check this cool cheat sheet by @haus3c.

And of course, the Bloodhound Slack #Cypher channel is the best place to ask if you have any questions…

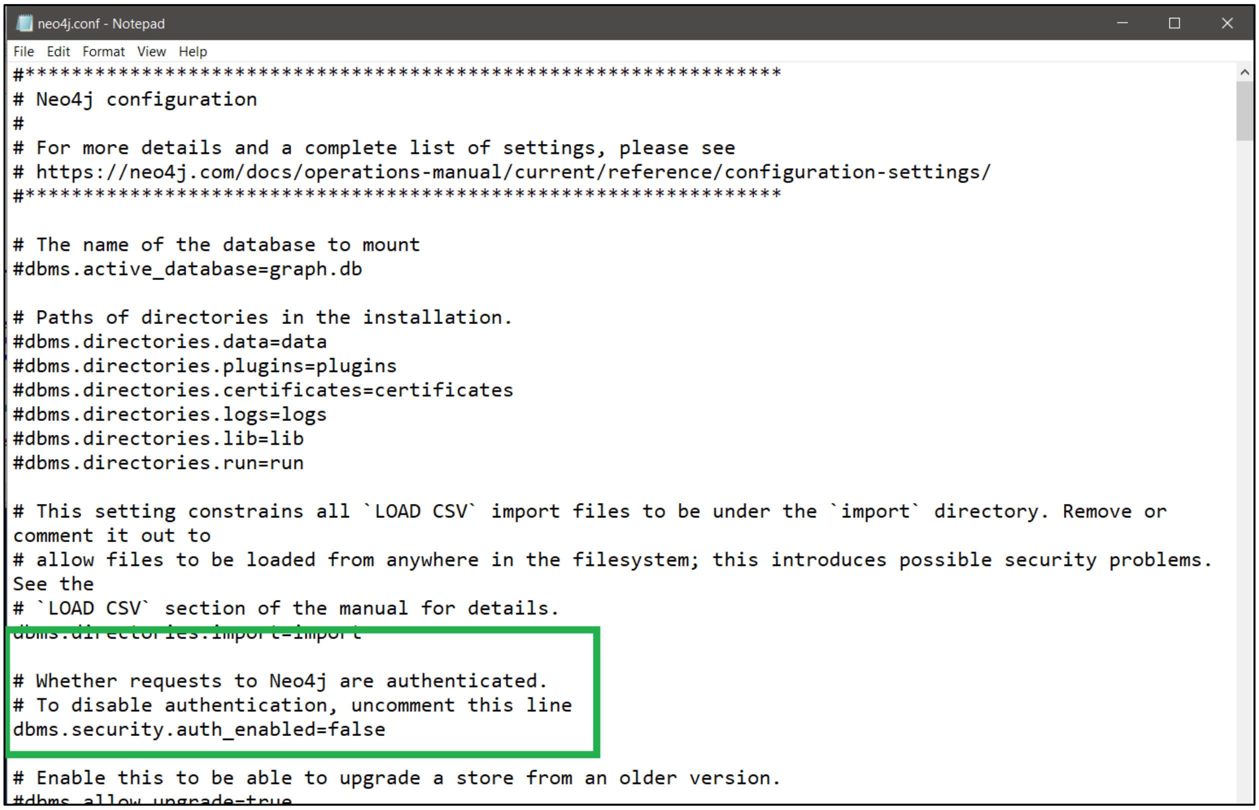
5. REST API

Ok, so now you're cool with Bloodhound and Cypher. You are ready for BloodHound's best kept secret:

Neo4j has a REST API, and with a little bit of scripting, you can build whatever tooling your like around it.

5.1. Setup

To allow unauthenticated requests to the rest API, uncomment the /conf/neo4j.conf file as follow.

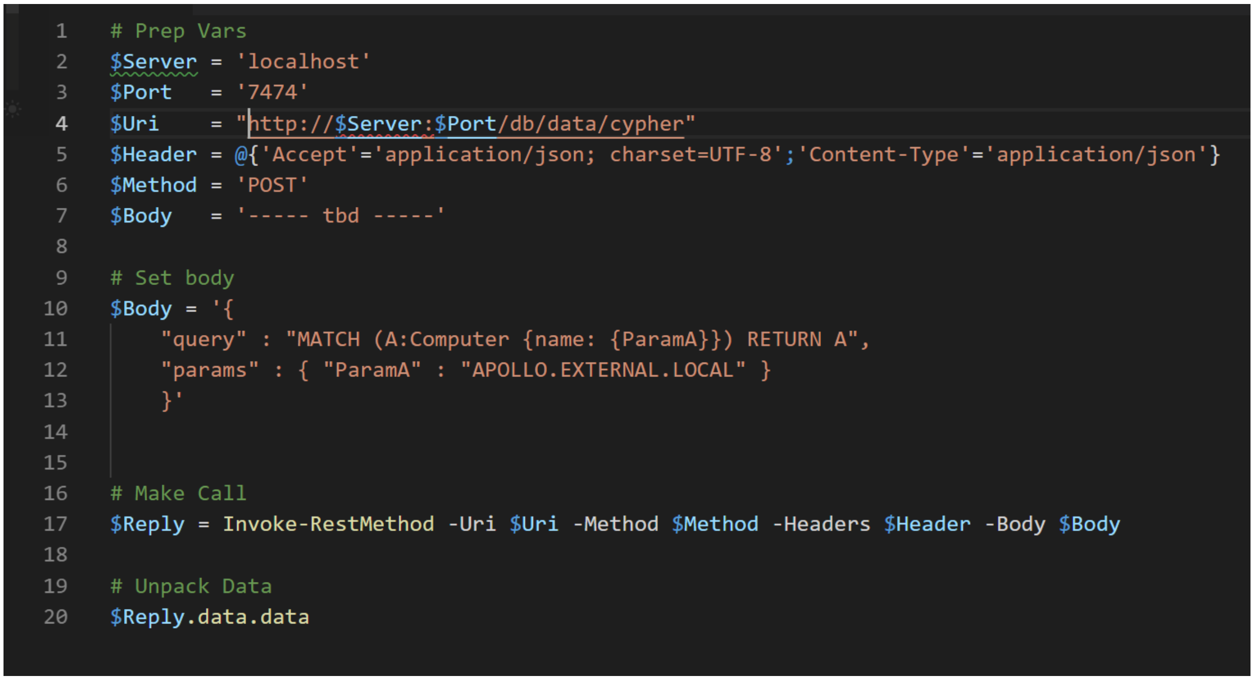


```
#*****************************************************************
# Neo4j configuration
#
# For more details and a complete list of settings, please see
# https://neo4j.com/docs/operations-manual/current/reference/configuration-settings/
#*****************************************************************

# The name of the database to mount
#dbms.active_database=graph.db

# Paths of directories in the installation.
#dbms.directories.data=data
#dbms.directories.plugins=plugins
#dbms.directories.certificates=certificates
#dbms.directories.logs=logs
#dbms.directories.lib=lib
#dbms.directories.run=run

# This setting constrains all `LOAD CSV` import files to be under the `import` directory. Remove or
comment it out to
# allow files to be loaded from anywhere in the filesystem; this introduces possible security problems.
See the
# `LOAD CSV` section of the manual for details.
dbms.directories.import=import

# Whether requests to Neo4j are authenticated.
# To disable authentication, uncomment this line
dbms.security.auth_enabled=false

# Enable this to be able to upgrade a store from an older version.
#dbms.allow_upgrade=true
```

If you want to allow remote access to the REST API, you have to uncomment another line further down in the config, but in that case make sure to implement some kind of authentication. I'm guessing having an open bloodhound DB on the network is not best practices…

5.2. Basic call

Below an example in PowerShell. Example in Bash can be found here.

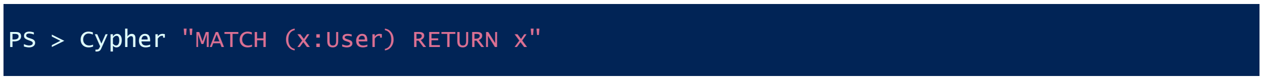


```
# Prep Vars
$Server = 'localhost'
$Port   = '7474'
$Uri    = "http://$Server:$Port/db/data/cypher"
$Header = @{'Accept'='application/json; charset=UTF-8';'Content-Type'='application/json'}
$Method = 'POST'
$Body   = '----- tbd -----'

# Set body
$Body = '{
    "query" : "MATCH (A:Computer {name: {ParamA}}) RETURN A",
    "params" : { "ParamA" : "APOLLO.EXTERNAL.LOCAL" }
    }'


# Make Call
$Reply = Invoke-RestMethod -Uri $Uri -Method $Method -Headers $Header -Body $Body

# Unpack Data
$Reply.data.data
```

5.3. Invoke-Cypher

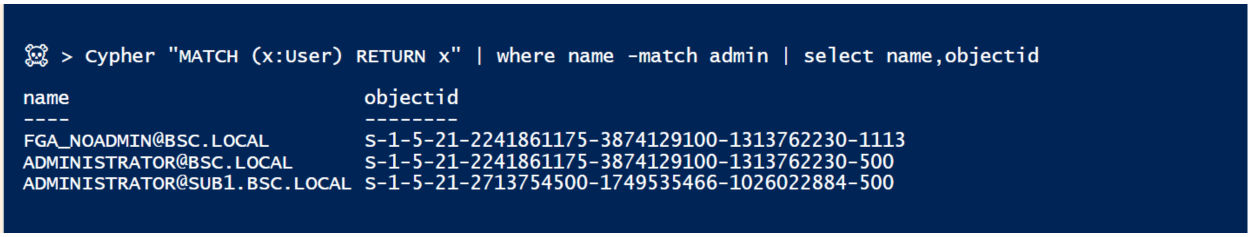
Invoke-Cypher is a basic function to send Cypher queries to the REST API. Can be used as a stand-alone or as a base for further tooling.

Example Syntax is as follows:



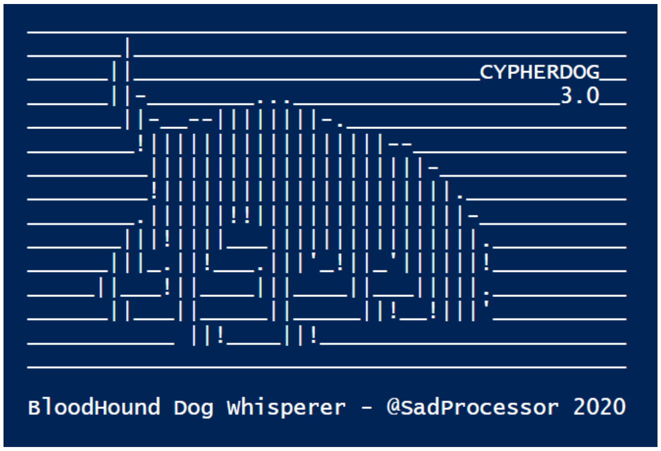
```
PS > Cypher "MATCH (x:User) RETURN x"
```

You can send any type of Cypher query via the rest API, just like you would in the neo4j browser, except now you are returning objects, and you can further manipulate them…



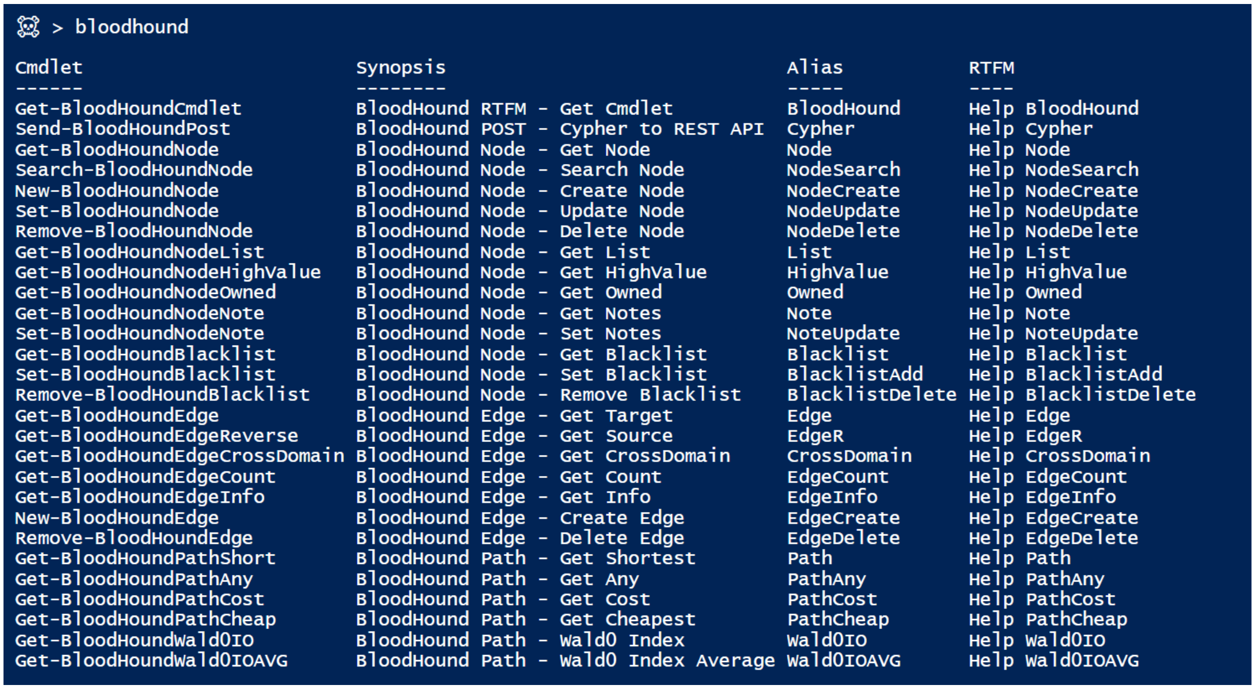
```
☠ > Cypher "MATCH (x:User) RETURN x" | where name -match admin | select name,objectid

name                          objectid
----                          --------
FGA_NOADMIN@BSC.LOCAL         S-1-5-21-2241861175-3874129100-1313762230-1113
ADMINISTRATOR@BSC.LOCAL       S-1-5-21-2241861175-3874129100-1313762230-500
ADMINISTRATOR@SUB1.BSC.LOCAL  S-1-5-21-2713754500-1749535466-1026022884-500
```

5.4. CypherDog



```
BloodHound Dog Whisperer - @SadProcessor 2020
```

[CypherDog](#) is a full set of PowerShell Cmdlets build around the same idea…



```
> bloodhound

Cmdlet                        Synopsis                              Alias           RTFM
------                        --------                              -----           ----
Get-BloodHoundCmdlet          BloodHound RTFM - Get Cmdlet          BloodHound      Help BloodHound
Send-BloodHoundPost           BloodHound POST - Cypher to REST API  Cypher          Help Cypher
Get-BloodHoundNode            BloodHound Node - Get Node            Node            Help Node
Search-BloodHoundNode         BloodHound Node - Search Node         NodeSearch      Help NodeSearch
New-BloodHoundNode            BloodHound Node - Create Node         NodeCreate      Help NodeCreate
Set-BloodHoundNode            BloodHound Node - Update Node         NodeUpdate      Help NodeUpdate
Remove-BloodHoundNode         BloodHound Node - Delete Node         NodeDelete      Help NodeDelete
Get-BloodHoundNodeList        BloodHound Node - Get List            List            Help List
Get-BloodHoundNodeHighValue   BloodHound Node - Get HighValue       HighValue       Help HighValue
Get-BloodHoundNodeOwned       BloodHound Node - Get Owned           Owned           Help Owned
Get-BloodHoundNodeNote        BloodHound Node - Get Notes           Note            Help Note
Set-BloodHoundNodeNote        BloodHound Node - Set Notes           NoteUpdate      Help NoteUpdate
Get-BloodHoundBlacklist       BloodHound Node - Get Blacklist       Blacklist       Help Blacklist
Set-BloodHoundBlacklist       BloodHound Node - Set Blacklist       BlacklistAdd    Help BlacklistAdd
Remove-BloodHoundBlacklist    BloodHound Node - Remove Blacklist    BlacklistDelete Help BlacklistDelete
Get-BloodHoundEdge            BloodHound Edge - Get Target          Edge            Help Edge
Get-BloodHoundEdgeReverse     BloodHound Edge - Get Source          EdgeR           Help EdgeR
Get-BloodHoundEdgeCrossDomain BloodHound Edge - Get CrossDomain     CrossDomain     Help CrossDomain
Get-BloodHoundEdgeCount       BloodHound Edge - Get Count           EdgeCount       Help EdgeCount
Get-BloodHoundEdgeInfo        BloodHound Edge - Get Info            EdgeInfo        Help EdgeInfo
New-BloodHoundEdge            BloodHound Edge - Create Edge         EdgeCreate      Help EdgeCreate
Remove-BloodHoundEdge         BloodHound Edge - Delete Edge         EdgeDelete      Help EdgeDelete
Get-BloodHoundPathShort       BloodHound Path - Get Shortest        Path            Help Path
Get-BloodHoundPathAny         BloodHound Path - Get Any             PathAny         Help PathAny
Get-BloodHoundPathCost        BloodHound Path - Get Cost            PathCost        Help PathCost
Get-BloodHoundPathCheap       BloodHound Path - Get Cheapest        PathCheap       Help PathCheap
Get-BloodHoundWald0IO         BloodHound Path - Wald0 Index         Wald0IO         Help Wald0IO
Get-BloodHoundWald0IOAVG      BloodHound Path - Wald0 Index Average Wald0IOAVG      Help Wald0IOAVG
```

Too many cmdlets to explain everything here so I'll shamelessly plug my [Troopers talk](#) from last year.

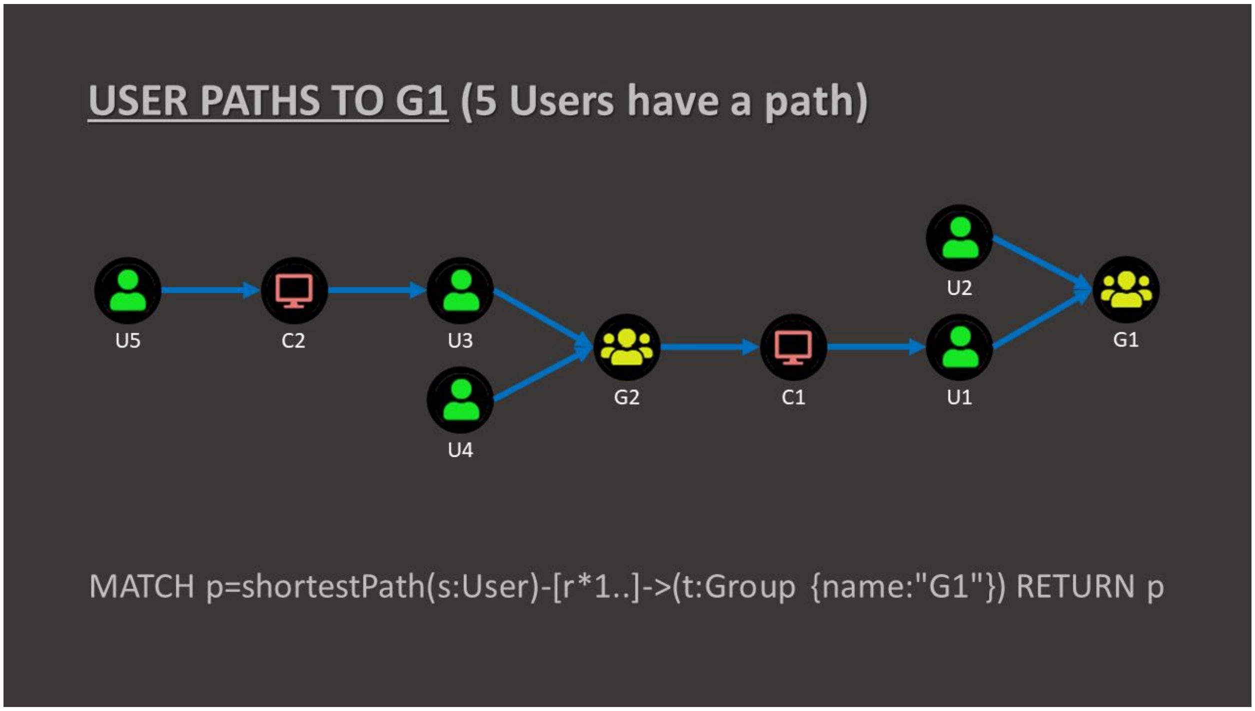
Ping me on the [Bloodhound Slack](#) if you have questions.

5.5. WatchDog

[WatchDog](#) is an attempt at extracting metrics out of Bloodhound data.

Not a silver bullet, but I believe it does at good job at identifying nodes with high impact on paths to sensitive groups.

The basics idea of NodeWeight calculation is as follow:



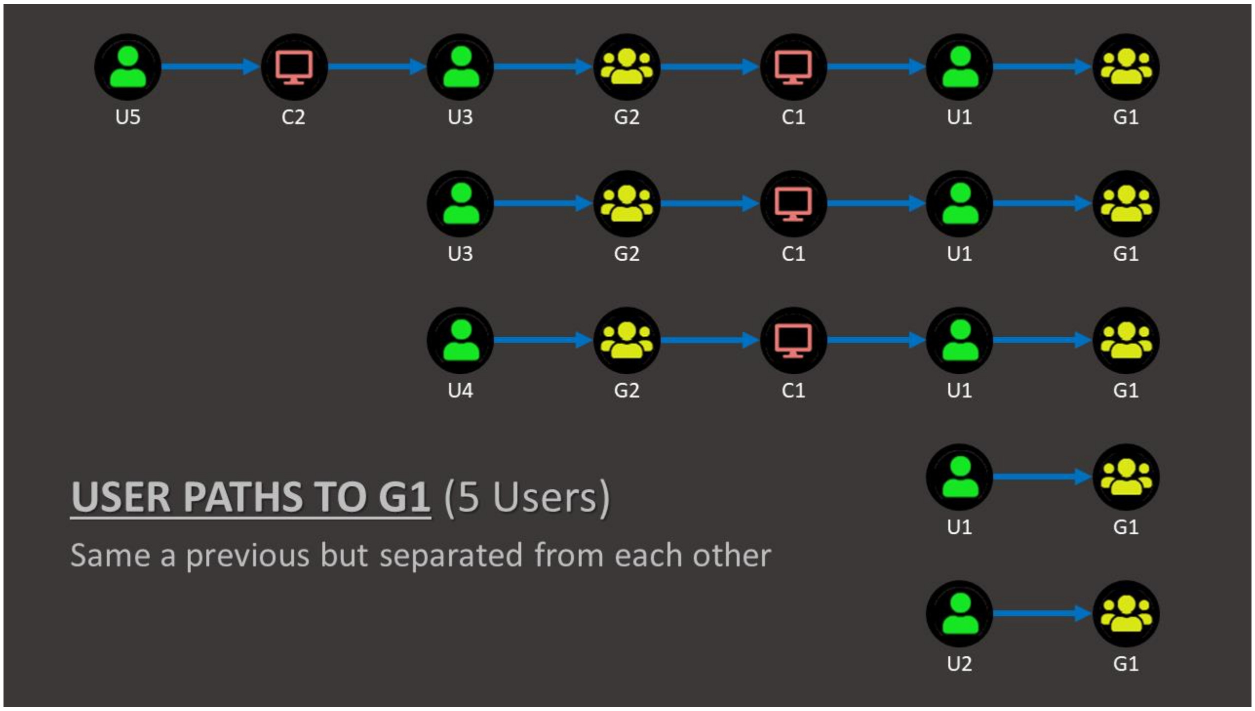
USER PATHS TO G1 (5 Users have a path)

```
MATCH p=shortestPath(s:User)-[r*1..]->(t:Group {name:"G1"}) RETURN p
```

In the example above, we ask Bloodhound to return Users with a path to a group G1.

5 users have a path to this group.

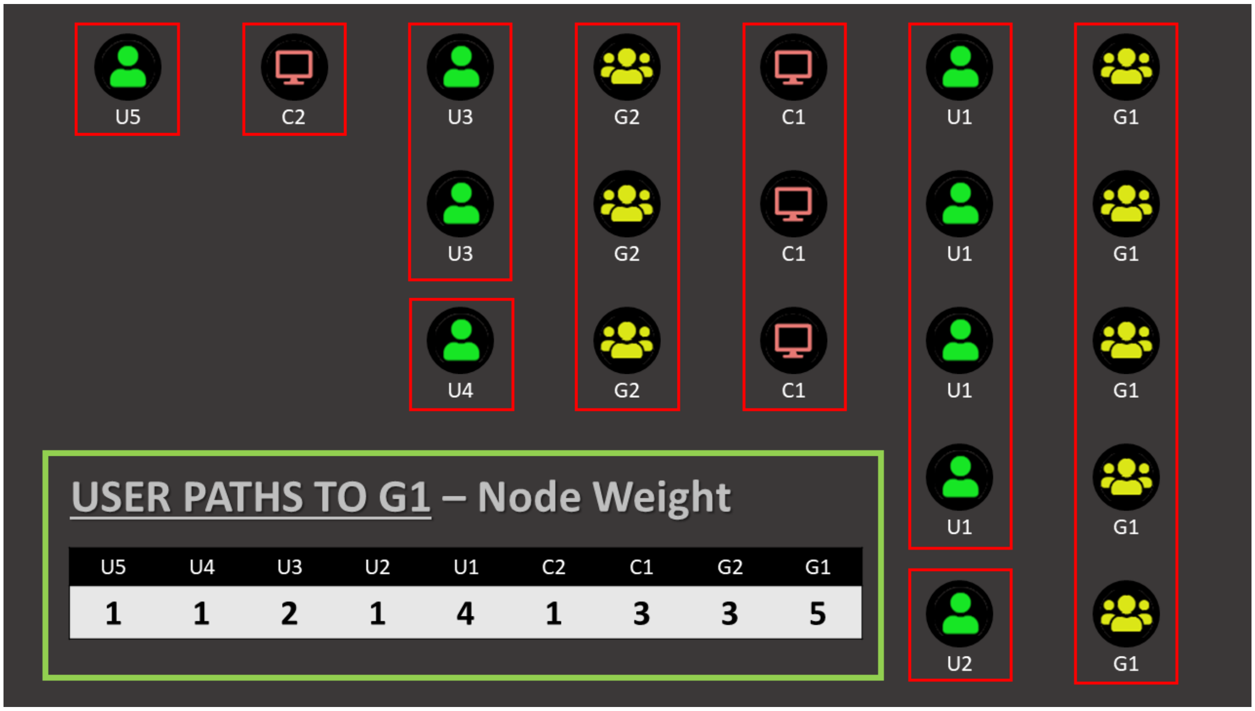
The graph above is the same thing as below:



USER PATHS TO G1 (5 Users)

Same a previous but separated from each other

Once we look at it this way, counting the incidence of nodes give us the NodeWeight:

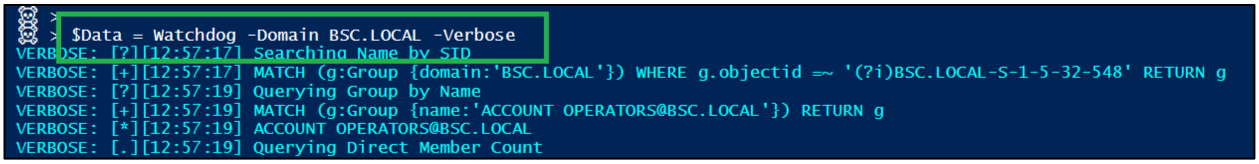


USER PATHS TO G1 – Node Weight

U5	U4	U3	U2	U1	C2	C1	G2	G1
1	1	2	1	4	1	3	3	5

WatchDog uses this technique to calculate NodeWeight for each node on user paths to a set of active directory sensitive groups at the domain level. (extra groups can be added to the scan)

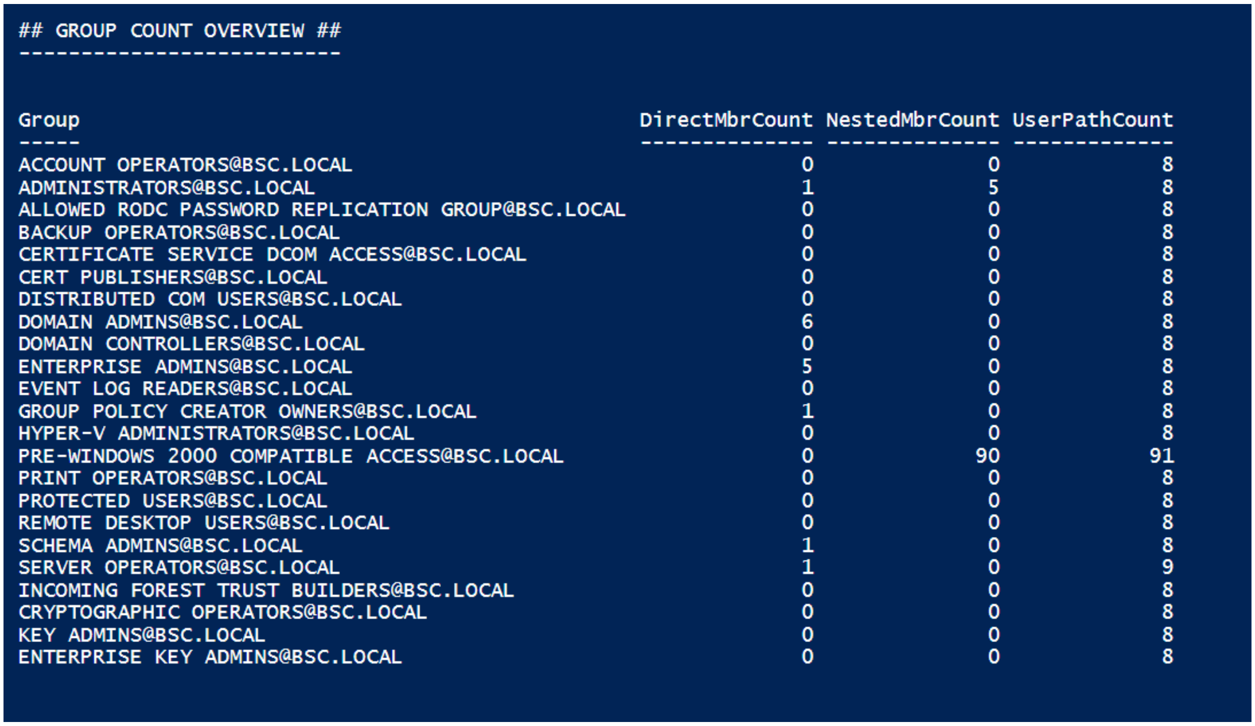
Basic command is as follows:



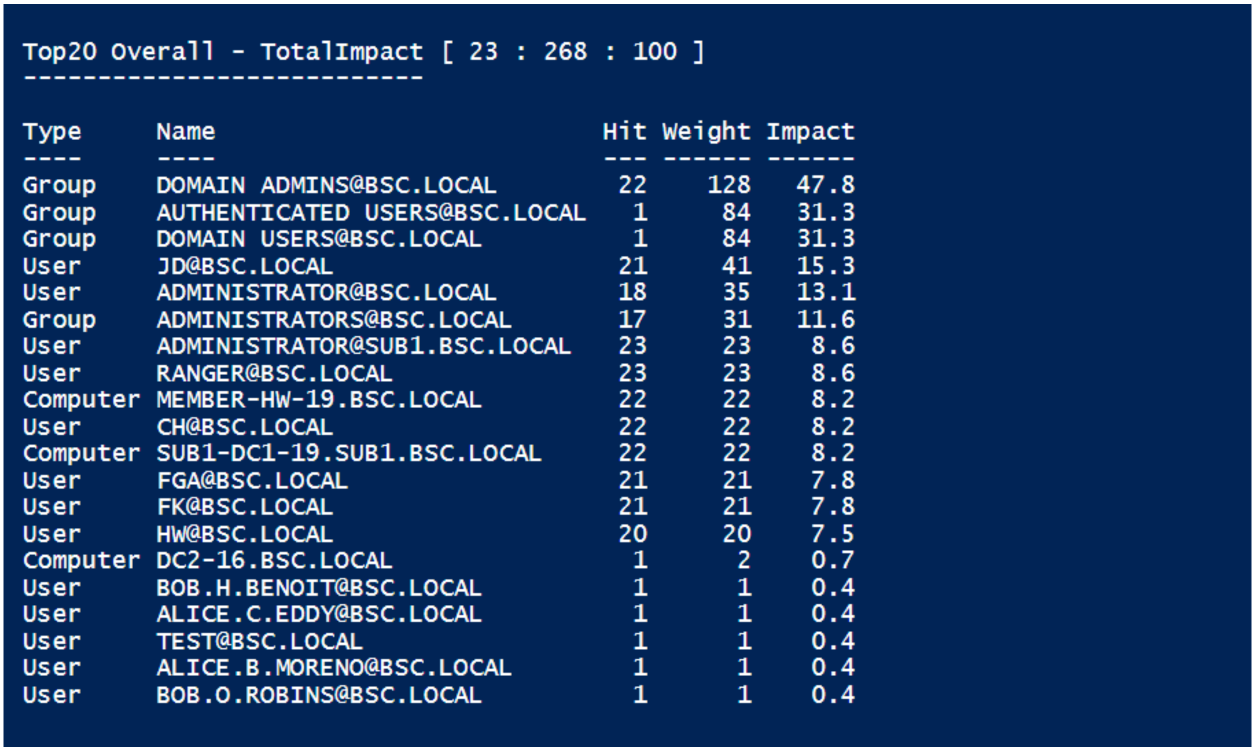
```
 >
 > $Data = Watchdog -Domain BSC.LOCAL -Verbose
VERBOSE: [?][12:57:17] Searching Name by SID
VERBOSE: [+][12:57:17] MATCH (g:Group {domain:'BSC.LOCAL'}) WHERE g.objectid =~ '(?i)BSC.LOCAL-S-1-5-32-548' RETURN g
VERBOSE: [?][12:57:19] Querying Group by Name
VERBOSE: [+][12:57:19] MATCH (g:Group {name:'ACCOUNT OPERATORS@BSC.LOCAL'}) RETURN g
VERBOSE: [*][12:57:19] ACCOUNT OPERATORS@BSC.LOCAL
VERBOSE: [.][12:57:19] Querying Direct Member Count
```

This will output a watchdog object per targeted group, and store that collection in the $Data variable.

From there you can do all sorts of manipulation for further analysis. Generate a report, maybe build a dashboard…

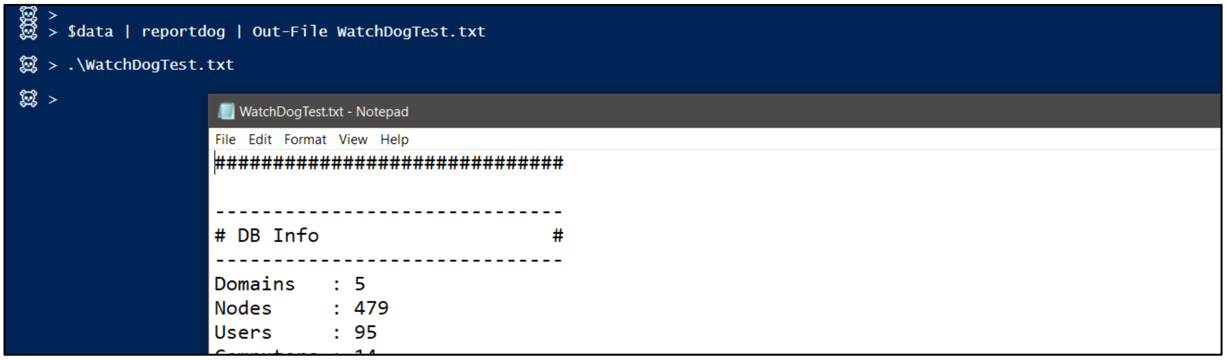


```
## GROUP COUNT OVERVIEW ##
--------------------------


Group                                                DirectMbrCount NestedMbrCount UserPathCount
-----                                                -------------- -------------- -------------
ACCOUNT OPERATORS@BSC.LOCAL                                       0              0             8
ADMINISTRATORS@BSC.LOCAL                                          1              5             8
ALLOWED RODC PASSWORD REPLICATION GROUP@BSC.LOCAL                 0              0             8
BACKUP OPERATORS@BSC.LOCAL                                        0              0             8
CERTIFICATE SERVICE DCOM ACCESS@BSC.LOCAL                         0              0             8
CERT PUBLISHERS@BSC.LOCAL                                         0              0             8
DISTRIBUTED COM USERS@BSC.LOCAL                                   0              0             8
DOMAIN ADMINS@BSC.LOCAL                                           6              0             8
DOMAIN CONTROLLERS@BSC.LOCAL                                      0              0             8
ENTERPRISE ADMINS@BSC.LOCAL                                       5              0             8
EVENT LOG READERS@BSC.LOCAL                                       0              0             8
GROUP POLICY CREATOR OWNERS@BSC.LOCAL                             1              0             8
HYPER-V ADMINISTRATORS@BSC.LOCAL                                  0              0             8
PRE-WINDOWS 2000 COMPATIBLE ACCESS@BSC.LOCAL                      0             90            91
PRINT OPERATORS@BSC.LOCAL                                         0              0             8
PROTECTED USERS@BSC.LOCAL                                         0              0             8
REMOTE DESKTOP USERS@BSC.LOCAL                                    0              0             8
SCHEMA ADMINS@BSC.LOCAL                                           1              0             8
SERVER OPERATORS@BSC.LOCAL                                        1              0             9
INCOMING FOREST TRUST BUILDERS@BSC.LOCAL                          0              0             8
CRYPTOGRAPHIC OPERATORS@BSC.LOCAL                                 0              0             8
KEY ADMINS@BSC.LOCAL                                              0              0             8
ENTERPRISE KEY ADMINS@BSC.LOCAL                                   0              0             8
```



```
Top20 Overall - TotalImpact [ 23 : 268 : 100 ]
---------------------------

Type     Name                         Hit Weight Impact
----     ----                         --- ------ ------
Group    DOMAIN ADMINS@BSC.LOCAL       22    128   47.8
Group    AUTHENTICATED USERS@BSC.LOCAL  1     84   31.3
Group    DOMAIN USERS@BSC.LOCAL         1     84   31.3
User     JD@BSC.LOCAL                  21     41   15.3
User     ADMINISTRATOR@BSC.LOCAL       18     35   13.1
Group    ADMINISTRATORS@BSC.LOCAL      17     31   11.6
User     ADMINISTRATOR@SUB1.BSC.LOCAL  23     23    8.6
User     RANGER@BSC.LOCAL              23     23    8.6
Computer MEMBER-HW-19.BSC.LOCAL        22     22    8.2
User     CH@BSC.LOCAL                  22     22    8.2
Computer SUB1-DC1-19.SUB1.BSC.LOCAL    22     22    8.2
User     FGA@BSC.LOCAL                 21     21    7.8
User     FK@BSC.LOCAL                  21     21    7.8
User     HW@BSC.LOCAL                  20     20    7.5
Computer DC2-16.BSC.LOCAL               1      2    0.7
User     BOB.H.BENOIT@BSC.LOCAL         1      1    0.4
User     ALICE.C.EDDY@BSC.LOCAL         1      1    0.4
User     TEST@BSC.LOCAL                 1      1    0.4
User     ALICE.B.MORENO@BSC.LOCAL       1      1    0.4
User     BOB.O.ROBINS@BSC.LOCAL         1      1    0.4
```

The following command outputs a text report summarizing the data in a human readable format, and saves it to a file.



```
☠ >
☠ > $data | reportdog | Out-File WatchDogTest.txt

☠ > .\WatchDogTest.txt

☠ >
```

```
###############################

-------------------------------
# DB Info                     #
-------------------------------
Domains    : 5
Nodes      : 479
Users      : 95
Computers  : 14
```

You can then review report to spot interesting nodes…

(And yes… Exchange is a High Impact node and has control over your DCs… ;))

I'll try to present that tool in a conference somewhere this year, in the meanwhile you can read more about it here.

Make sure to check it out if you haven't, and again ping me on slack if needed…

… And that's about it for now… Hope you liked it.

Happy hunting with your dog…

@SadProcessor

6. Appendix

BloodHound links for further digging…

6.1. BloodHound Crew

Twitter:	@harmj0y / @_Wald0 / @CptJesus
Slack:	https://bloodhoundhq.slack.com/
Slack Invite:	https://bloodhoundgang.herokuapp.com

6.2. BloodHound Posts

Automated Derivative Admin Search	by @_Wald0
Introducing BloodHound	by @_Wald0
Intro to Cypher	by @CptJesus
The ACL Attack Path Update (v1.3)	by @_Wald0
Evolution of the BloodHound Ingestor	by @CptJesus
The Object Properties Update (v1.4)	by @CptJesus
SharpHound: Technical Details	by @CptJesus
SharpHound: Target Selection and API Usage	by @CptJesus
The Container Update (v1.5)	by @CptJesus
A Red Teamer's Guide to GPOs & OUs	by @_Wald0
BloodHound 2.0 (v2.0)	by @CptJesus
Broken Stuff Update (v2.1)	by @CptJesus
Introducing BloodHound 3 (v3.0)	by @Wald0
Introducing BloodHound 4 (v4.0)	by @Wald0

6.1. Bloodhound Code

GitHub:	https://github.com/BloodHoundAD/BloodHound
Docs:	https://bloodhound.readthedocs.io/en/latest/index.html

6.2. BloodHound Videos

Six Degrees of Domain Admin	BSides LV 2016
Here Be Dragons...	DerbyCon 2017
An ACE Up the Sleeves	DEFCON 2017
Bloodhound: He Attac, But He Also Protec...	SecDSM 2018
How to Download and Install Neo4j	SpecterOps 2018
How does Session Collection Work	SpecterOps 2018
BloodHound 2.1 Computer Account Takeover	SpecterOps 2019
BloodHound 3 Release – Webinar	SpecterOps 2020
SharpHound Detection – Webinar	SpecterOps 2020

6.3. Neo4j Cypher

Cypher Reference Card:	http://neo4j.com/docs/cypher-refcard/current/
Cypher Syntax Online Doc:	https://neo4j.com/docs/developer-manual/current/cypher/syntax/
Common Cypher Confusions:	https://neo4j.com/blog/common-confusions-cypher/

BLOODHOUND

*

***