ERNW
providing security.

Hands-On BloodHound

BruCon 2019

@SadProcessor

Agenda

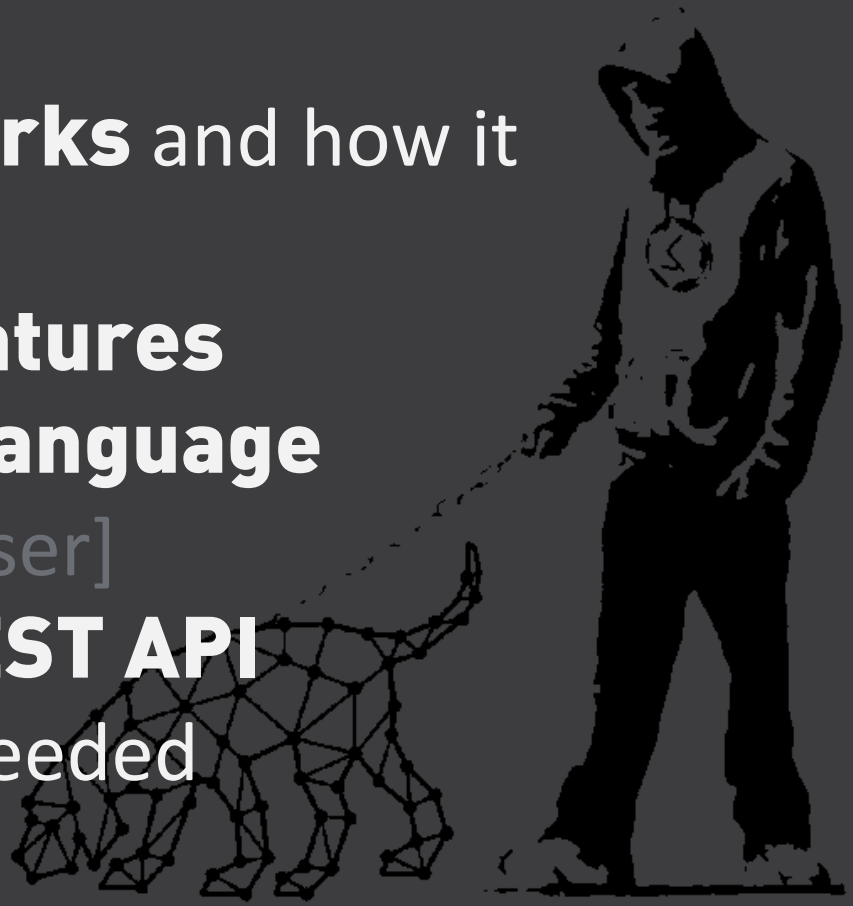
- BloodHound? Wut?
- Data & Collection
- Cypher - Basics
- Cypher - Advanced
- REST API & Automation

And memes…

Goal

At the end of this session, you should

- Understand **how BloodHound works** and how it could be useful for you [Red/Blue]
- Feel familiar with the **UI & tool features**
- Understand the basics of **Cypher language**
- **Create/Debug queries** [UI/Browser]
- Understand the workings of the **REST API**
- Know where to find **Info/Help** if needed



Prereq & Scope

If you want to follow along during the workshop

- Please **install BloodHound** BEFORE the session

[Due to time constrains this will NOT be done during session]

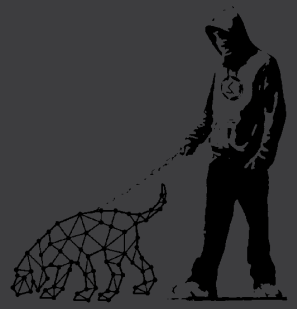
This training is about **Bloodhound & Cypher**, the following topics will not be covered in this workshop:

- Active Directory & Hardening in General
- Specific Attack Scenario

Whois

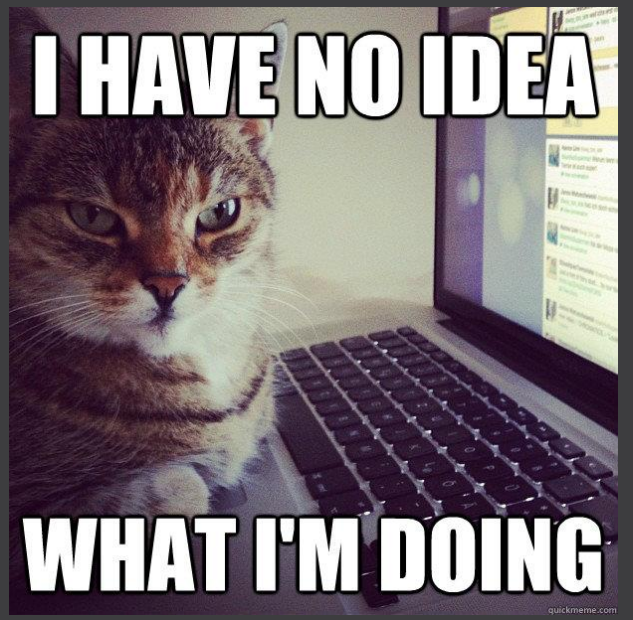
Walter Legowski - @SadProcessor

- Windows Security Consultant [ERNW]
- Born FR, Live NL, Work DE
- Like Buildings/Windows/Backdoors
- Like Cats/Trees/Backstreets
- Don't like Dogs
- Made exception for **BloodHound**

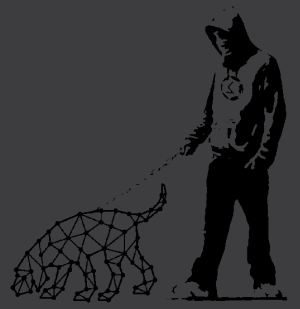


Disclaimer

- **I am not a Cypher expert** [I'm a Cat]
- Training only scratches the surface
- Excuses if any errors/typos in my slides

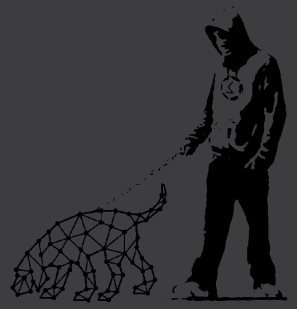


- **DO NOT SCAN A CORPORATE NETWORK WITHOUT PROPER AUTHORISATION**



Ready?

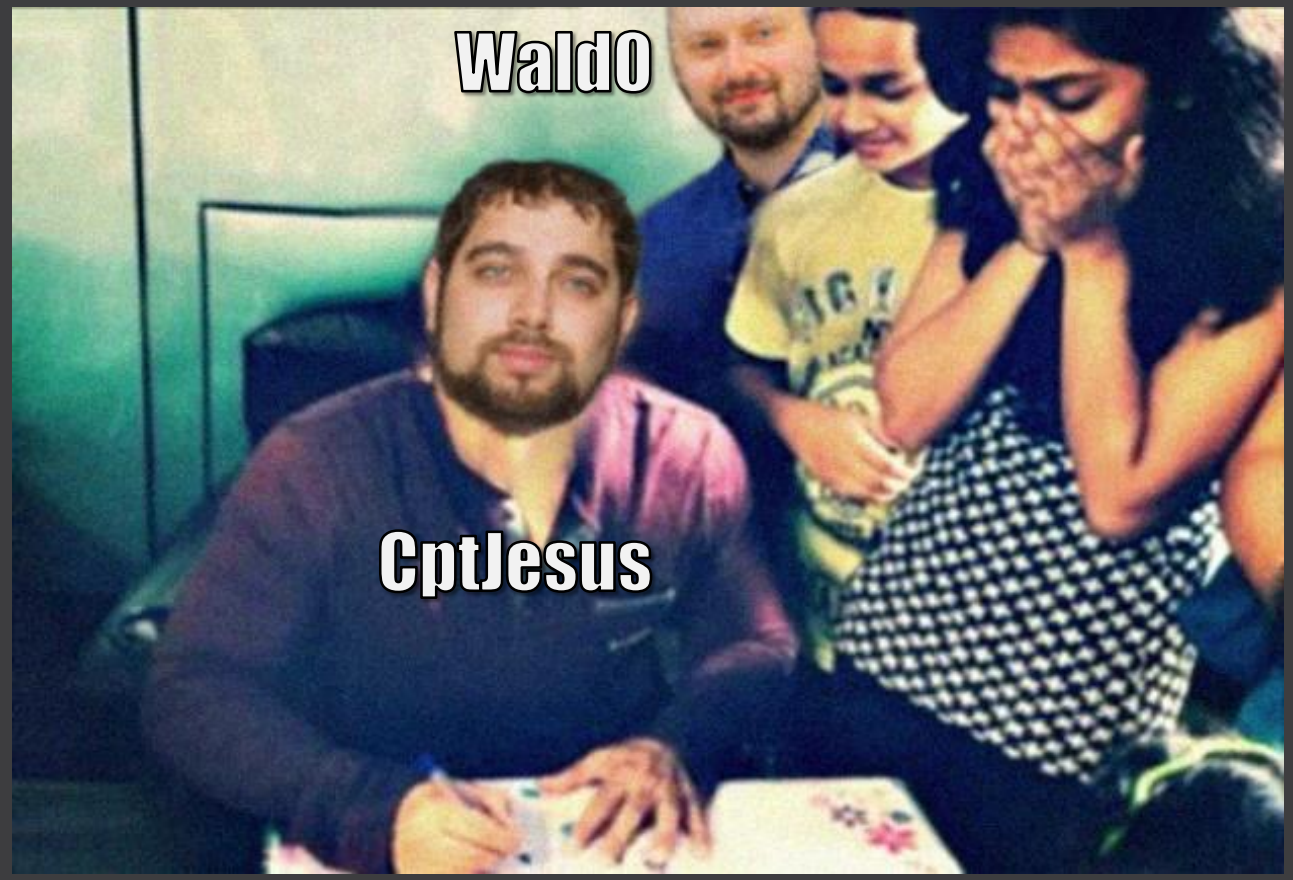
Let's Go...



1- BloodHound? Wut?

- Who?
- What?
- Why?
- Where?
- How?

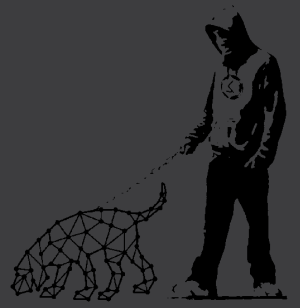
BloodHound - Who?



Created by

- @_WaldO
- @CptJesus
- @harmj0y

[click follow...]



BloodHound - What?

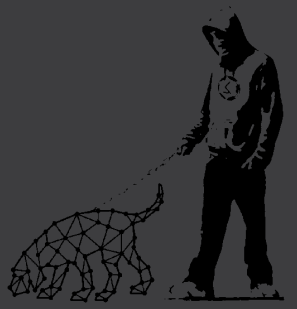
Defenders think in lists,
Attackers think in graphs,
As long as this is true,
Attackers win...
[John Lambert, MS Threat Intel]

Read: https://blogs.technet.microsoft.com/johnla/2015/04/26/defenders-think-in-lists-attackers-think-in-graphs-as-long-as-this-is-true-attackers-win/

BloodHound - What?

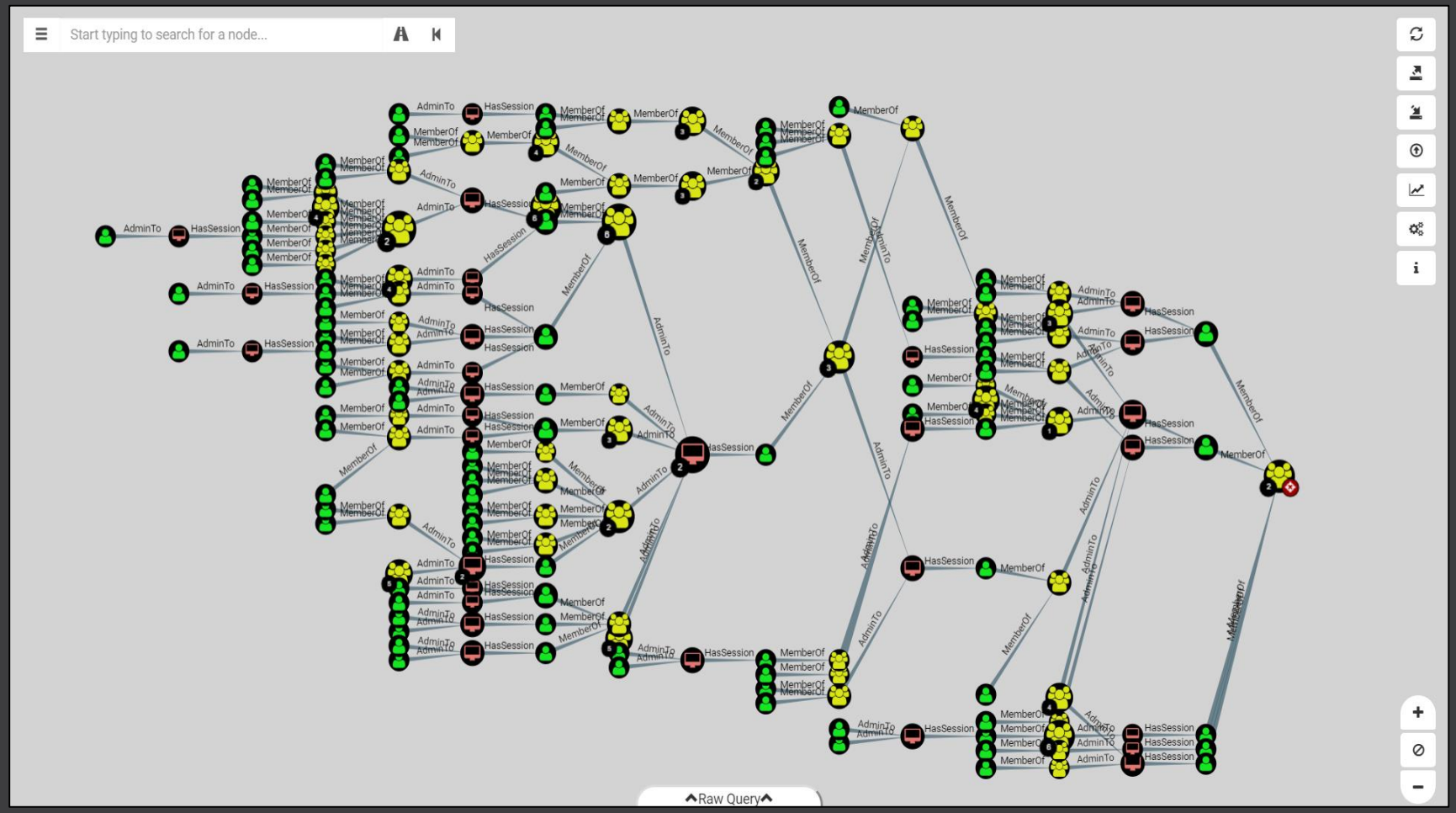
An **AD Attack Path Mapping Tool**, originally designed for **Post-Exploitation**, and useful for **AD Hardening** in general...

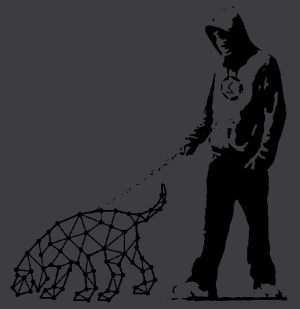
Collects and **graphs** relationships between **AD** Objects, and helps discover **security weaknesses**.



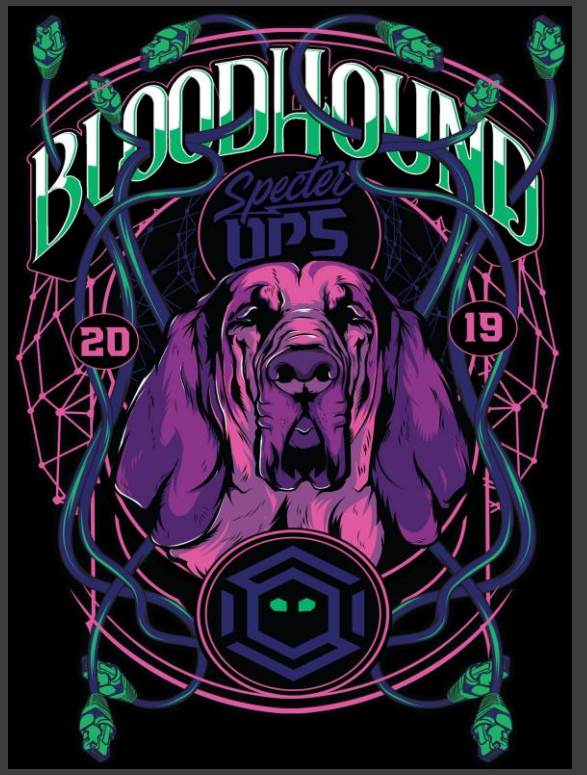
Code: https://github.com/BloodHoundAD/BloodHound

BloodHound - What?



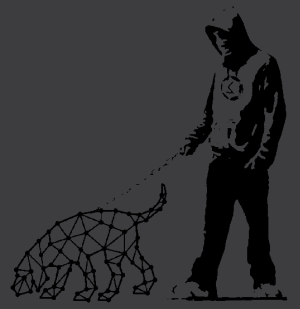


BloodHound - What?



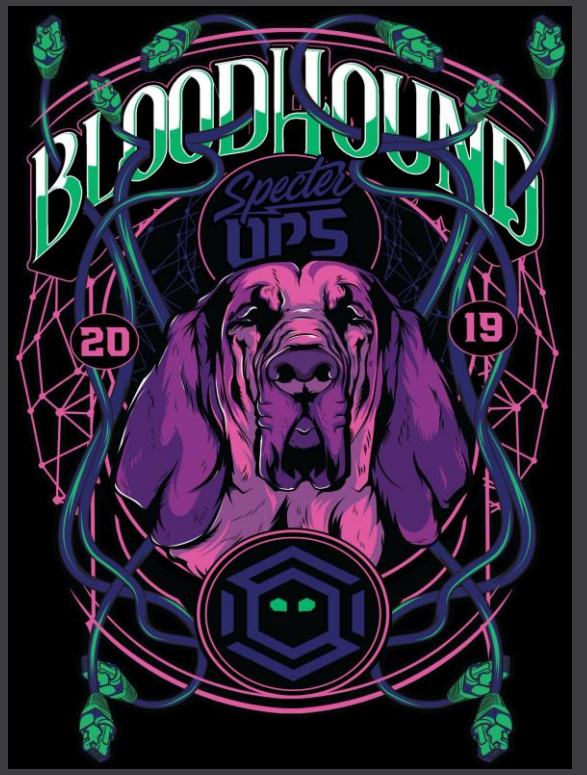
An **AD attack path** mapping tool...

- **Open Source** [all OS flavor]
- Based on **neo4j** graph DB
- Initial release: 2016
- Current version: 2.2
- Well maintained & documented
- User Community ++



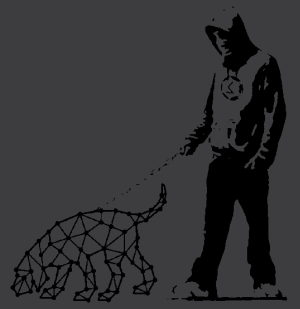
Wiki: https://github.com/BloodHoundAD/BloodHound/wiki

BloodHound - What?



An **AD Hardening** tool...

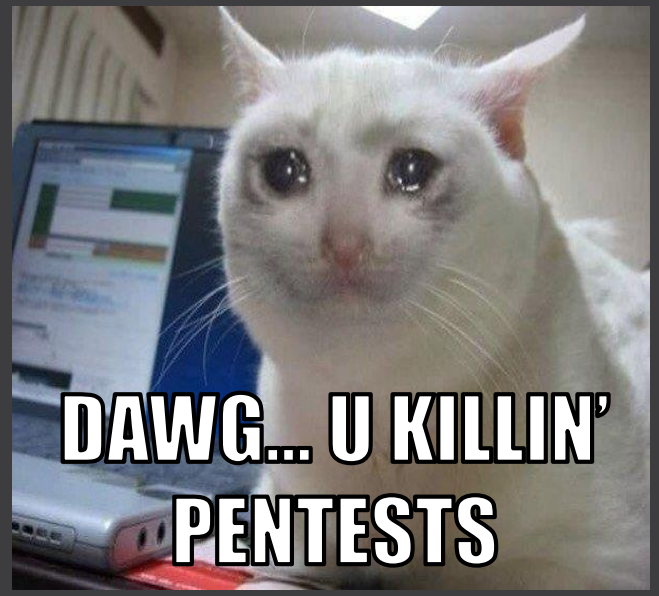
- Originally designed for **Red Team** reconnaissance [Post-Expoitation]
- Gaining popularity in **Blue Teams**
- Can be used for Security **Consulting** and **AD Auditing**
- **Expandable** & **Automatable**

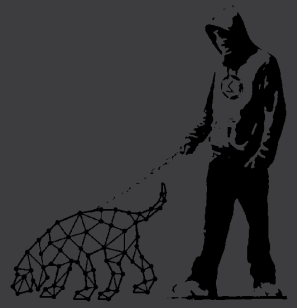


Wiki: https://github.com/BloodHoundAD/BloodHound/wiki

BloodHound - Why?

- **RED**: Find Path to Attack
  [Report > Blue > Consultant > Fix]
- **BLUE**: Find Paths to Defend
  [Report > Consultant > Fix]
- **CONSULTANT**: Find MisConfigs
  [Fix > Report]





BloodHound - Where?

Tool & Resources...

- Neo4j Community Edition

https://neo4j.com/download-center/#community

- BloodHound Source code

https://github.com/BloodHoundAD/BloodHound

- BloodHound Wiki

https://github.com/BloodHoundAD/BloodHound/wiki

- Neo4j Cypher Reference Card

https://neo4j.com/docs/cypher-refcard/current/

- Dog Whisperer Handbook

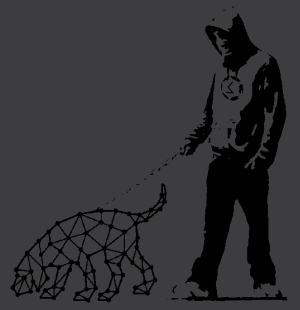
https://www.ernw.de/download/BloodHoundWorkshop/ERNW_DogWhispererHandbook.pdf

BloodHound - How?

Install... [Windows64]

- [Install Java64bit]
- Unzip Neo4j Community Edition
- Install & Start neo4j service
- Go to **http://localhost:7474**
- Set new password
- Unzip BloodHound Source
- Start bloodhound.exe & enter password

Wiki: https://github.com/BloodHoundAD/BloodHound/wiki/Getting-started



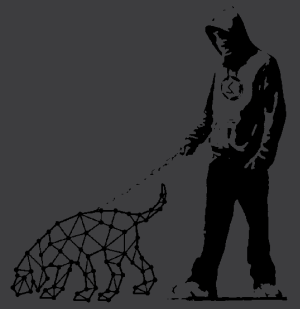
BloodHound - More

Get yourself on the BloodHound Slack

- Read tons of **interesting stuff**
- Meet tons of **interesting people** [4500+]
- Ask Wald0 about **#cypher_queries**
- Speak **#kerberos** with Harmj0y
- Ask Jesus **#random things**
- Hate @PrimaryTyler,CISSP as you like

And more…

Invite: https://bloodhoundgang.herokuapp.com/

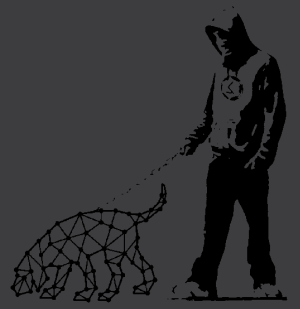


Hands-On: Slack

Join the Gang... [only if you like ofc]

- Invite yourself to the BloodHound Slack
- Check out some channels
- Join a few of your choosing

- Bonus:
  Ask @CptJesus for an autograph in #bloodhound-chat
  [or hate @PrimaryTyler,CISSP in his dedicated channel]
  and win a BloodHound sticker!!

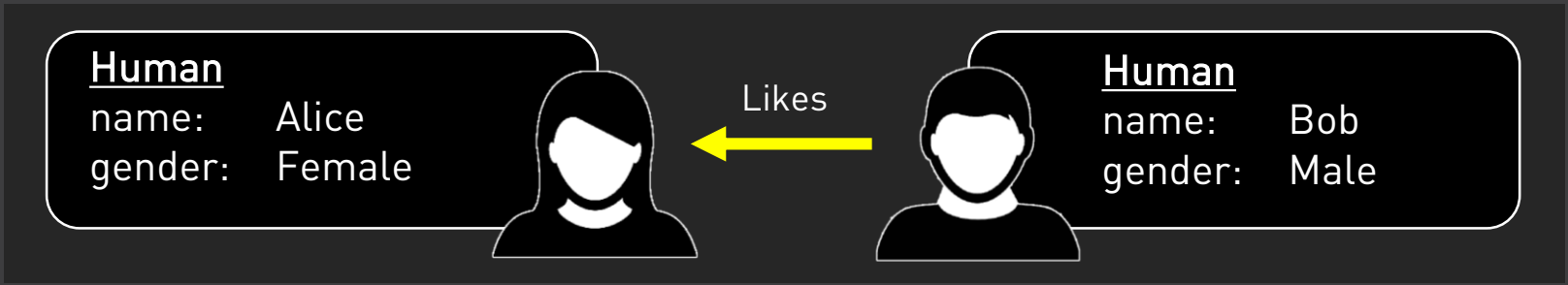


Invite: https://bloodhoundgang.herokuapp.com/

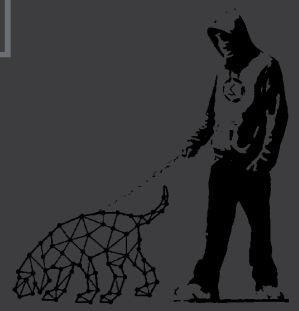
2- Data & Collection

- Alice & Bob
- BloodHound Data
- SharpHounds
- Collection Methods
- Data Import

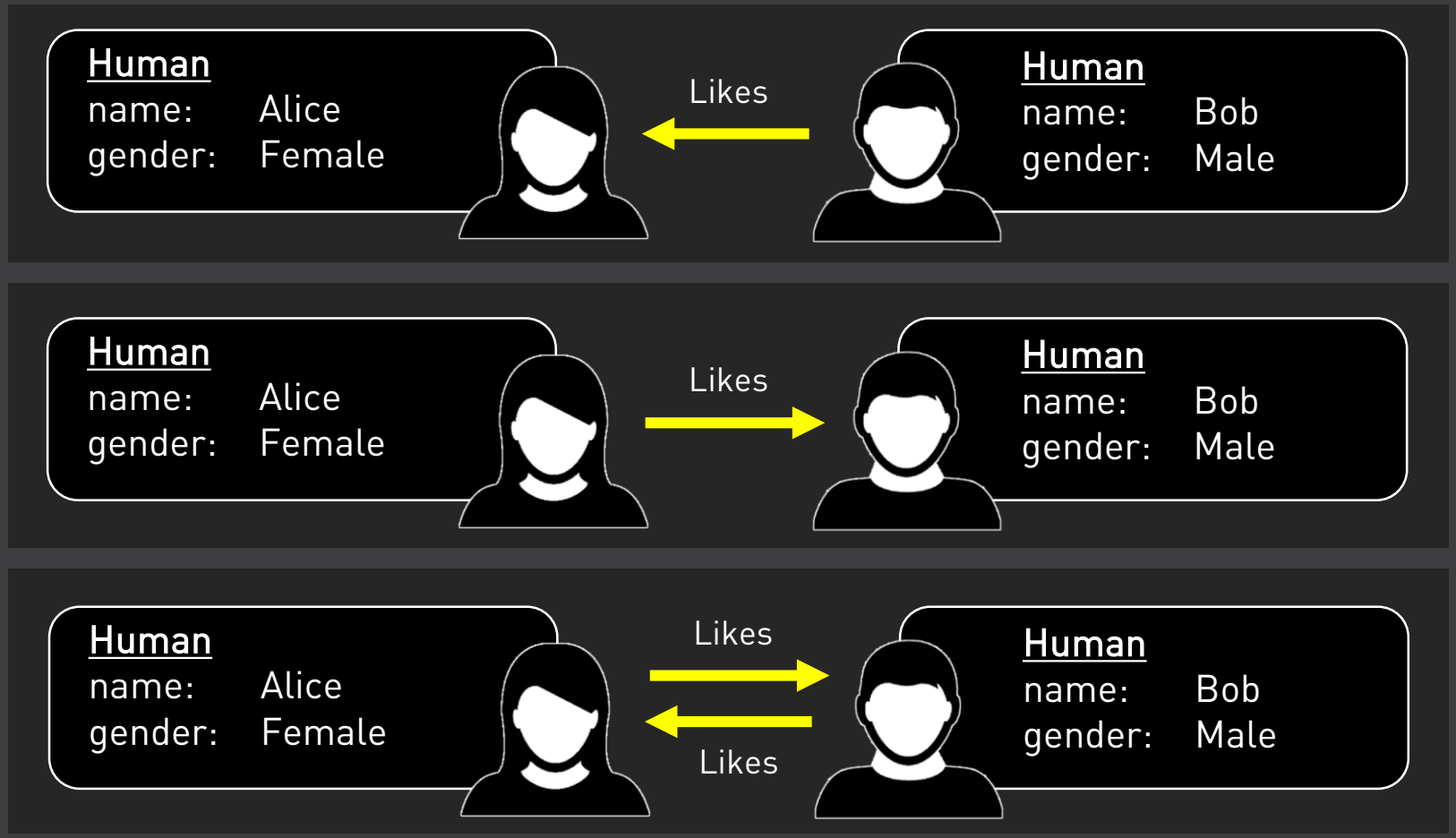
Alice & Bob - Terminology



- Objects are called **Nodes** [Humans Alice & Bob]
- Nodes have a type aka **Label** [Human]
- Nodes have **properties** [name/gender]
- Relationships are called an **Edges** [Likes]
- Edges can also have properties

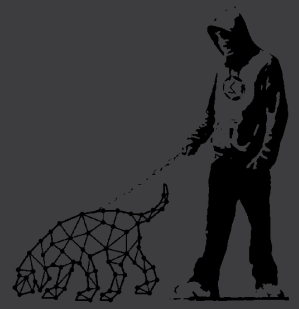


Alice & Bob - Terminology

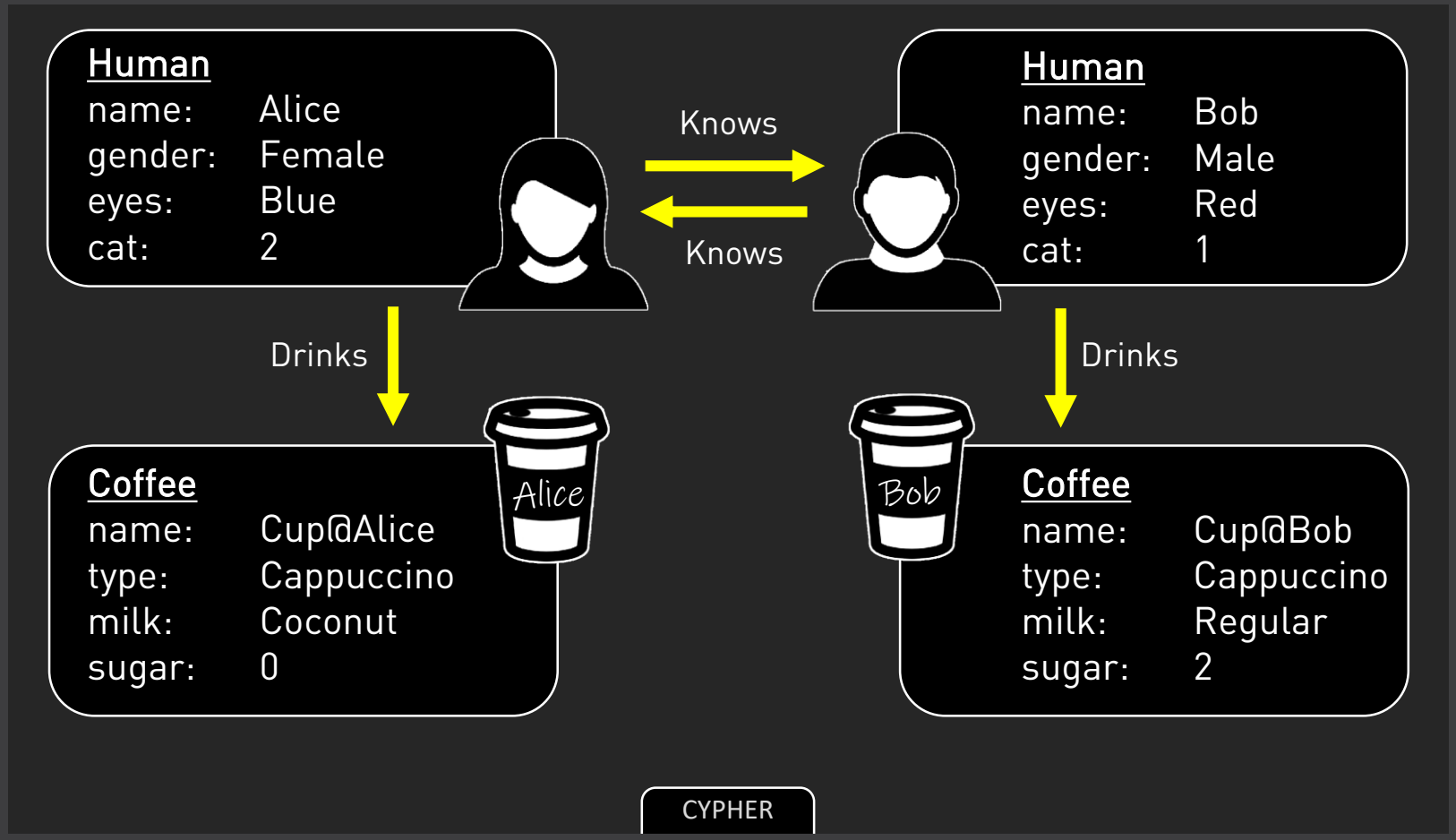


Important:
Edges are directional

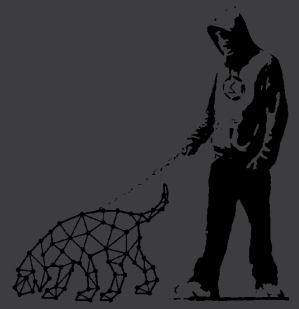
[Need two Edges for relationship to go both ways...]



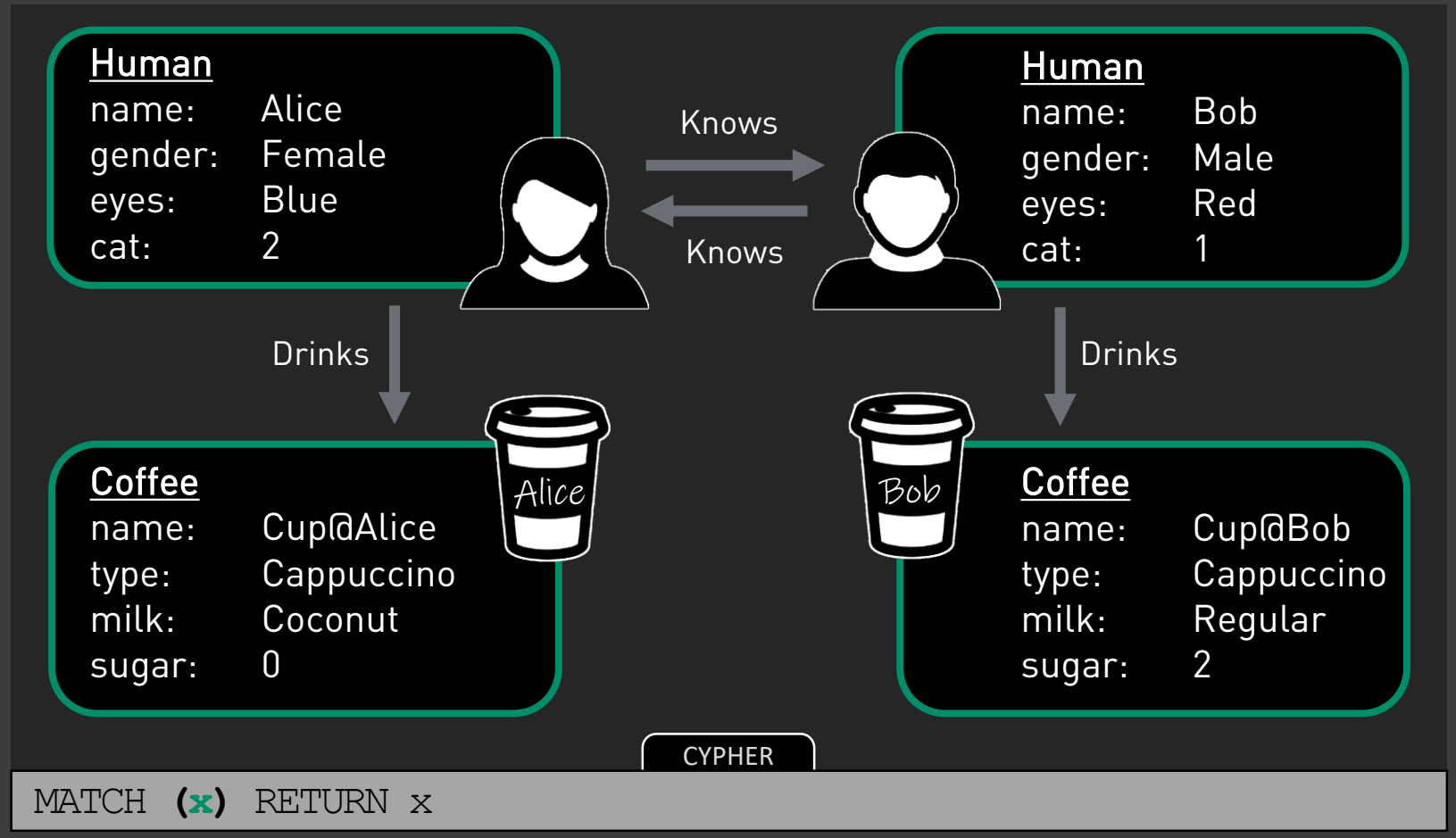
Alice & Bob - Dataset



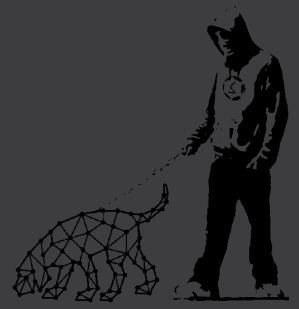
Now let's imagine this is our data. **Let's see what we can ask neo4j...**



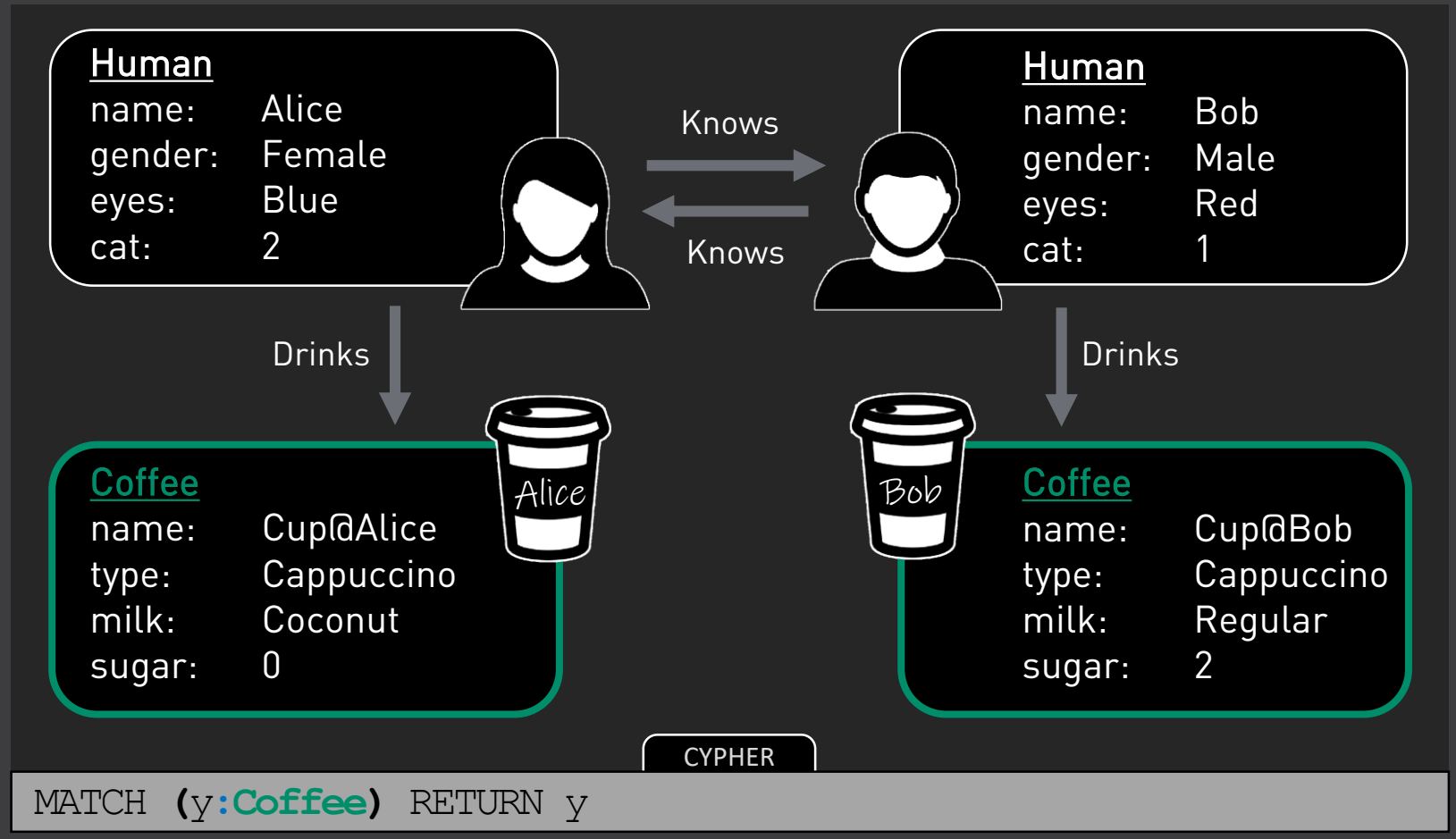
Alice & Bob - Queries



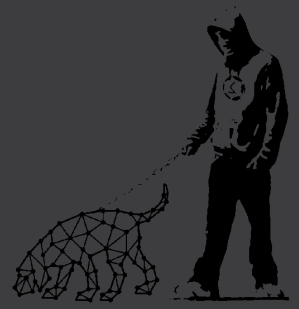
Return all Objects... [Nodes]



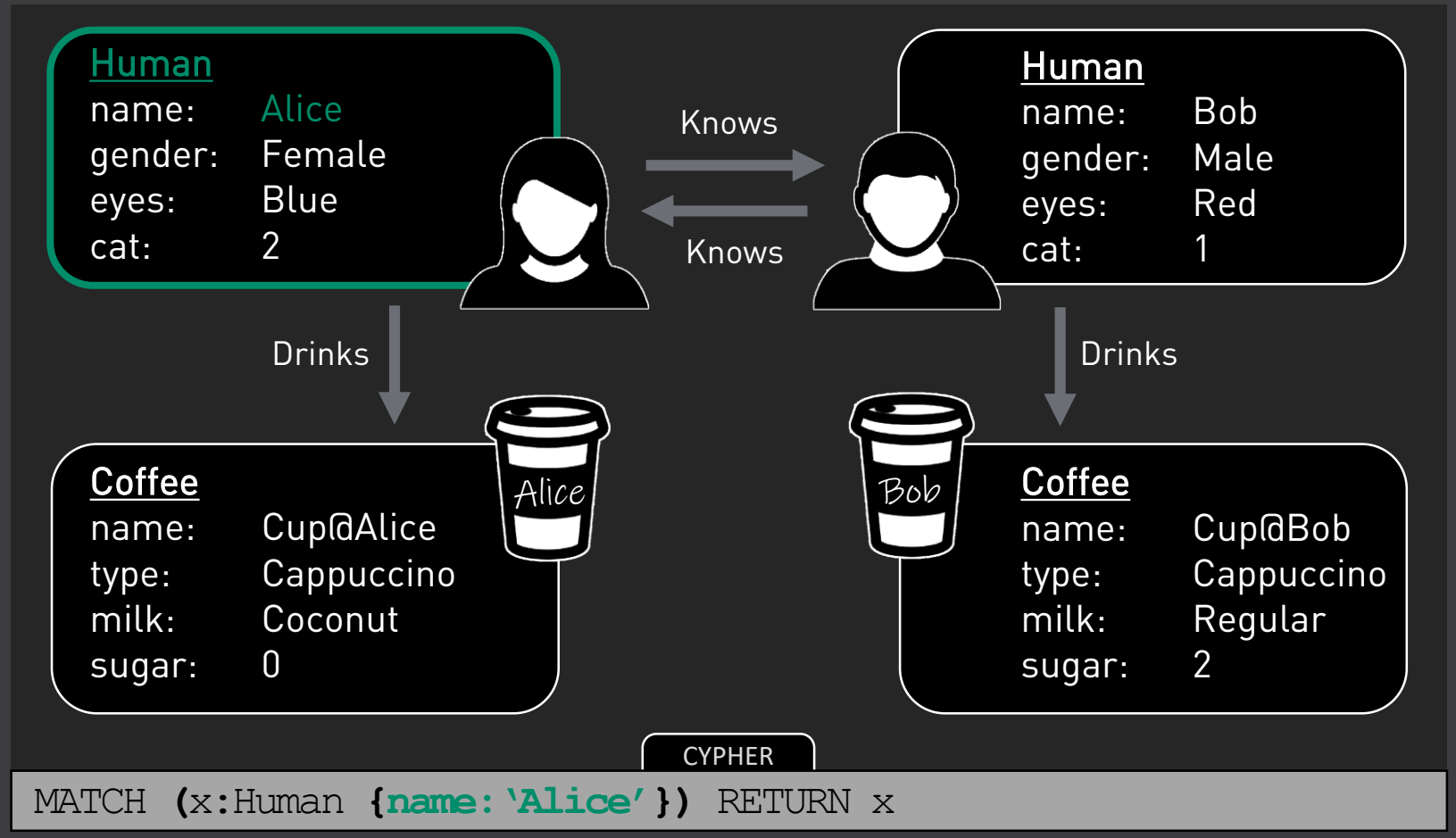
Alice & Bob - Queries



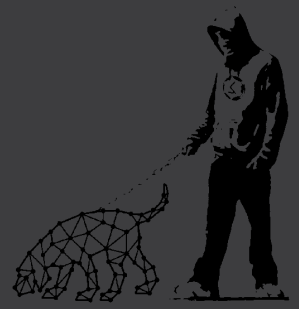
Return all Nodes of type Coffee [Label]



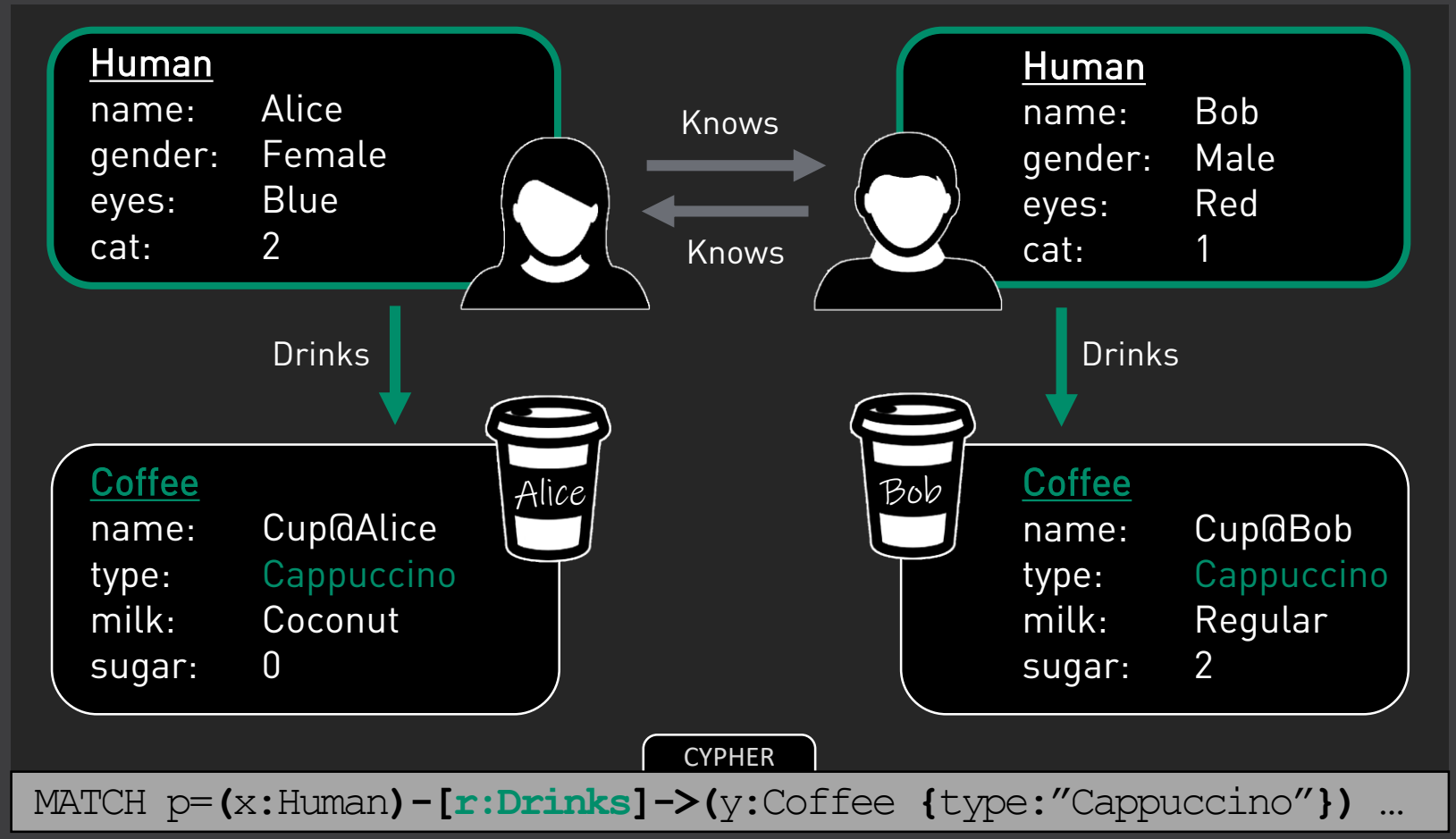
Alice & Bob - Queries



Is there a Human with name Alice?
[Property]

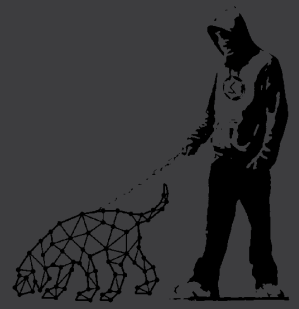


Alice & Bob - Queries

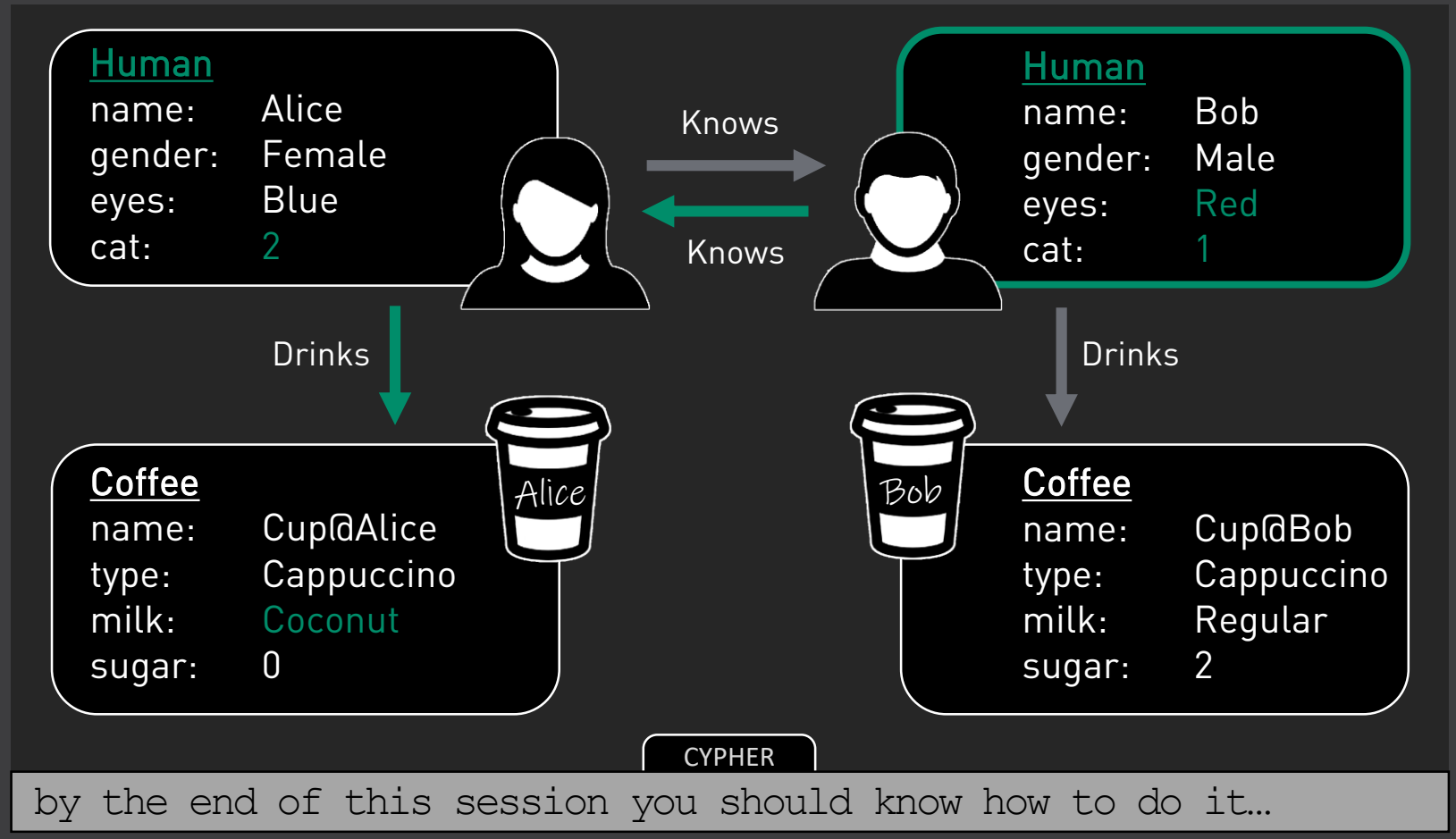


Who Drinks Cappuccino?

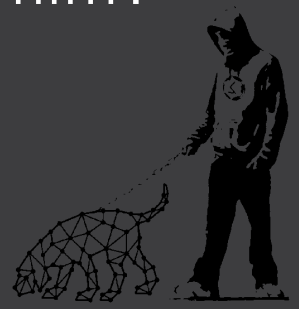
[Edge]



Alice & Bob - Queries



Does anyone with red eyes know somebody that drinks cappuccino with coconut milk and that has more cats than him?



Sample DB - Install

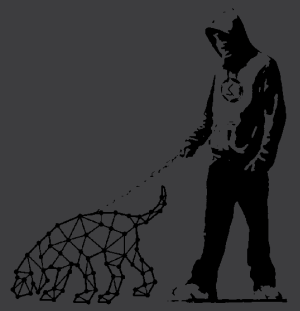
For this workshop, we will use fake AD data

Install as follows:

- Download & Unzip folder

https://github.com/SadProcessor/HandsOnBloodHound/blob/master/SampleData/graph.db.practice.zip

- Place in neo4j \data\databases folder
- **Stop neo4j service**
- Rename graph.db to graph.db.old
- **Rename graph.db.practice to graph.db**
- **Start service** & Restart BloodHound



BloodHound Data

BloodHound Data - Nodes

BloodHound uses **6 Node types** [aka Node Labels]


:Domain


:Group

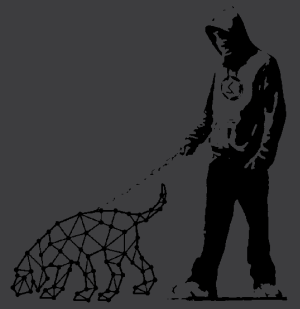

:OU


:Computer


:GPO

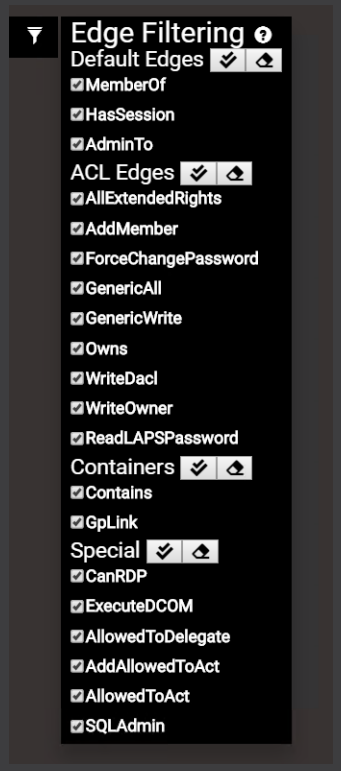

:User

Each Node type has matching set of **properties**



BloodHound Data - Edges

BloodHound uses **19 Edge types**

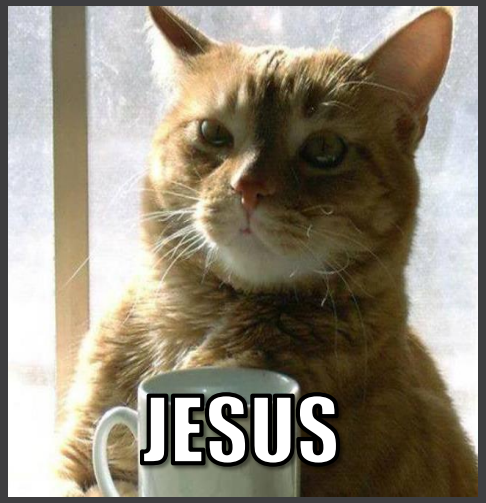


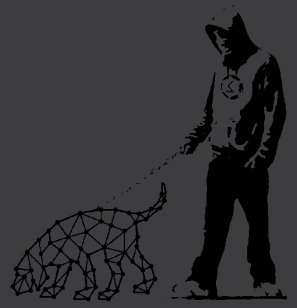
Default
MemberOf
HasSession
AdminTo

Special
ExecuteDCOM
AllowedToDelegate
AddAllowedToAct
AllowedToAct
SQLAdmin

ACL
AllExtendedRights
AddMember
ForceChangePassword
GenericAll
GenericWrite
Owns
WriteDacl
WriteOwner
ReadLAPSPassword

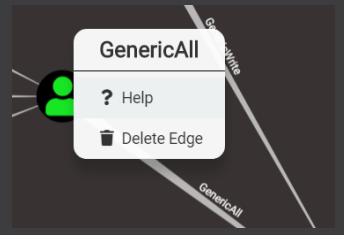
Container
Contains
GpLink



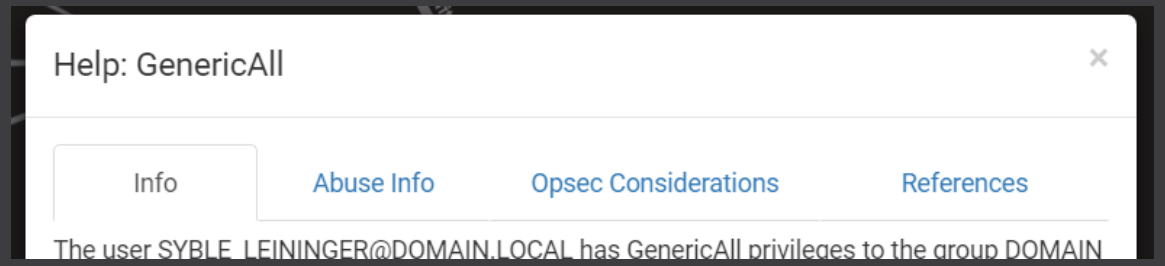


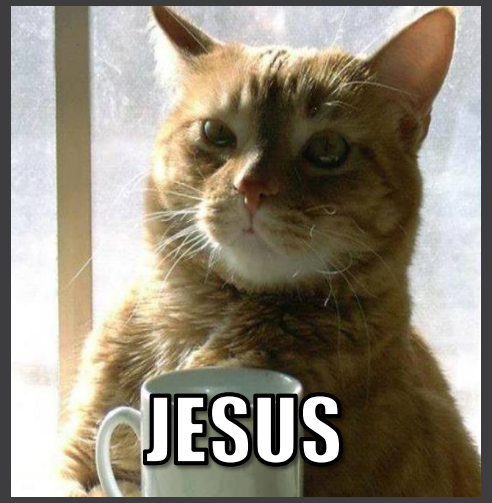
BloodHound Data - Edges

Each Edge represents a [one]way to move



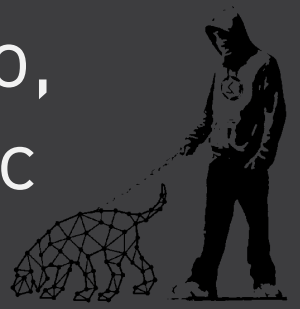
Right-Click Edge for Edge Help





This will open a contextual menu with General Info, Attack Info, OpSec Info, and Extra Refs on the topic

[Attacks are out-of-scope for today, but check it out during next exercise]



Hands-On: UI Tour

In the **BloodHound UI**, find how to:

- Check BD Properties
- Toggle Dark Mode On/Off [Keep your fav]
- Set Debug Mode On [forever]
- View a Node and it's properties
- View a Path One-to-One
- Request shortest Paths Any-to-One
- Run Build-In Queries

[And more by [right-]clicking around…]

SharpHounds - Info

SharpHound is the BloodHound Data Collector

- [Re]Written in **C#** for better performances
- Uses **LDAP & win32 API Calls** to gather info
- Most data can be collected at user level
- Comes in two flavors [.exe/.ps1]
- Various **Collection Methods** [switches]

Read: https://blog.cptjesus.com/posts/sharphoundtargeting

Collection Methods - Overview

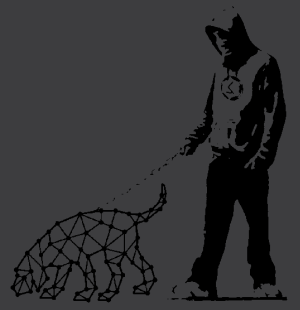
Data collected depends on Collection Method chosen

- To collect everything [no admin needed]

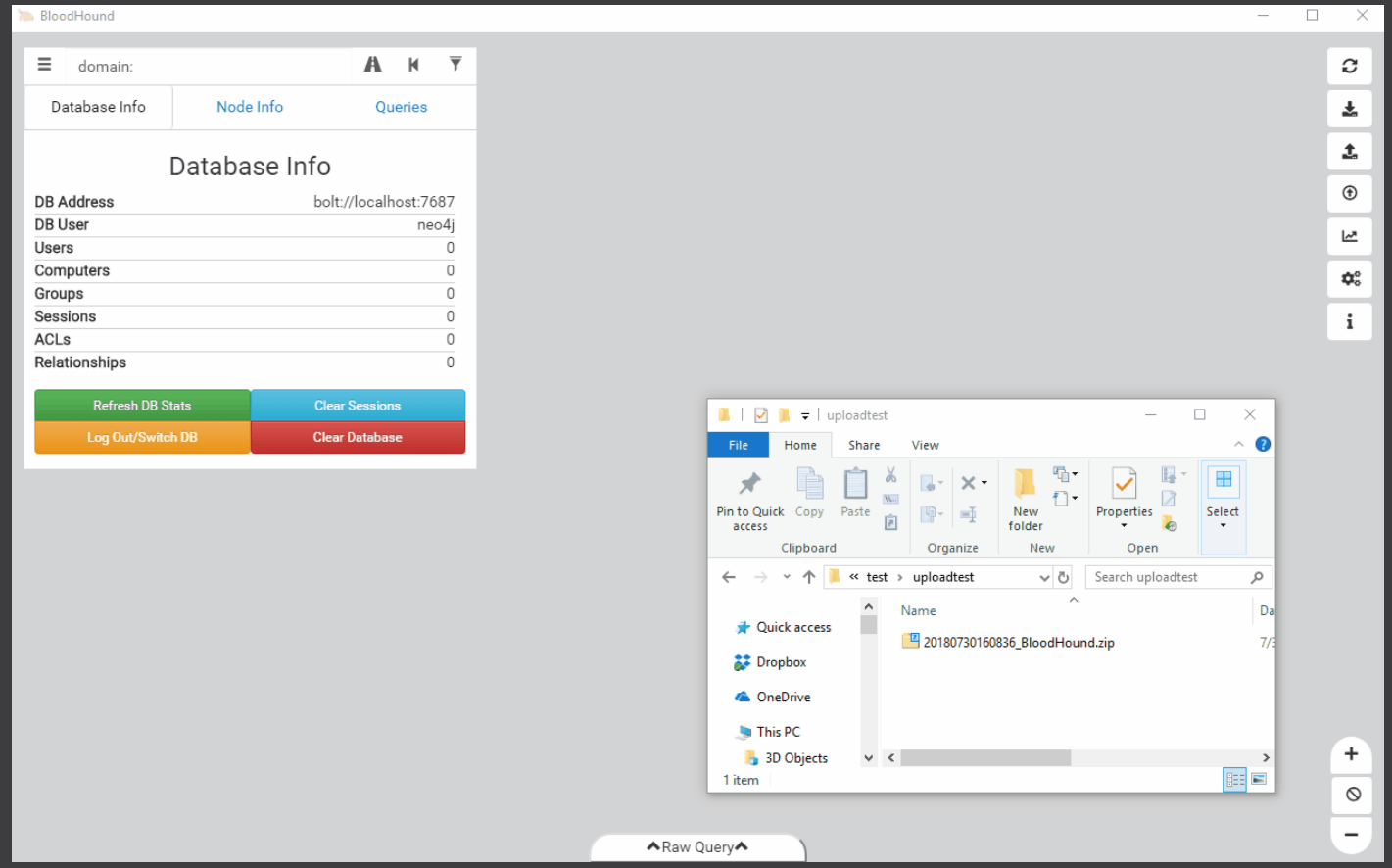
Invoke-BloodHound –CollectionMethod **All** -SearchForest

- Make sure to read Wiki & CptJesus's post on the topic
- Check .ps1 code & Help pages

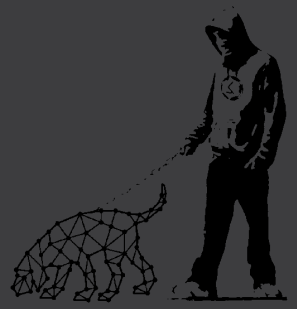
Read: https://github.com/BloodHoundAD/BloodHound/wiki/Data-Collector



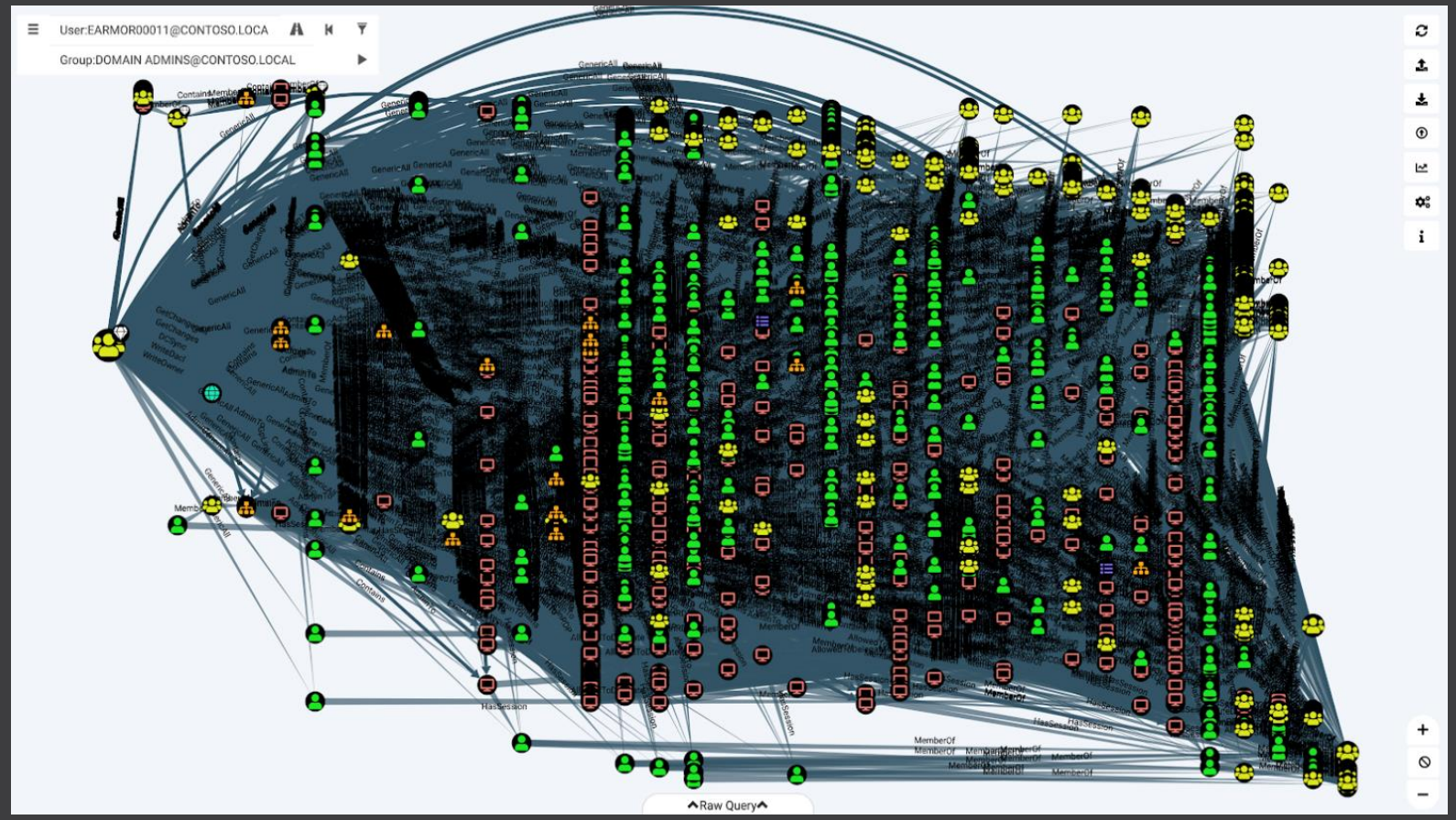
Data Import - HowTo



To Import [more] collected data, simply **drag** [extra] **zip files into the UI**



Data - A lot of data...



How to navigate all this?? **LEARN CYPHER**...

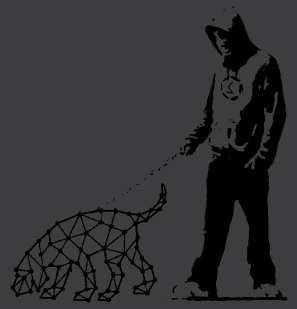
3- Cypher Basics

- Node & Path Queries
- Filtering & Comparing
- Adding & Deleting Stuff
- Debugging Queries
- Neo4j Browser
- Counting Nodes

Node Queries - Basic

Example of basic Node Queries

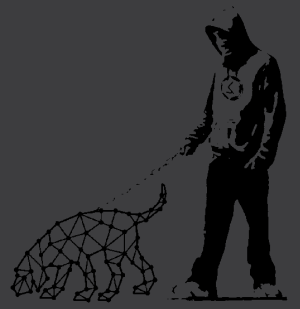
```
// All Nodes
MATCH (x) RETURN x

// All User Nodes
MATCH (x:User) RETURN x

// Node by Property
MATCH (x:User {name: 'BOB@DEMO.LAB'}) RETURN x
```



Path Queries - Basic

Example of Basic Path Query

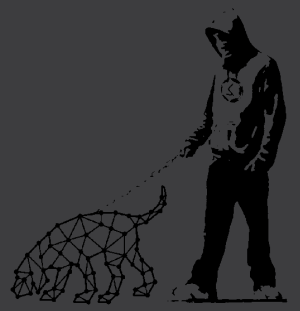
```
// Path User Bob to DA
MATCH (u:User {name:'BOB@DEMO.LAB'})
MATCH (c:Group {name:'DOMAIN ADMINS@DEMO.LAB'})
MATCH p=shortestPath((u)-[r*1..]->(c))
RETURN p
```



Path Queries - Basic

Example Path – Owned to HighValue

```
// Path Own to High Value – All Shortest
MATCH (u:User {highvalue:true})
MATCH (c:Computer {owned:true})
MATCH p=allShortestPaths((c)-[r*1..]->(u))
RETURN p
```

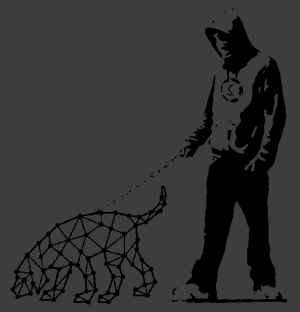


Filtering Stuff - WHERE

The WHERE clause can be used to filter:

```
MATCH (x:User {name: 'BOB@DEMO.LAB'}) RETURN x
// same as
MATCH (x:User)
WHERE x.name='BOB@DEMO.LAB'
RETURN x
```

[WHERE can be used with other operators than equal]

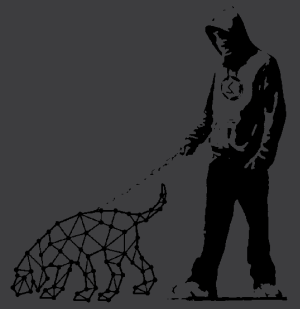


Comparing Stuff - Operators

List of Comparaison Operators:

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*	=~

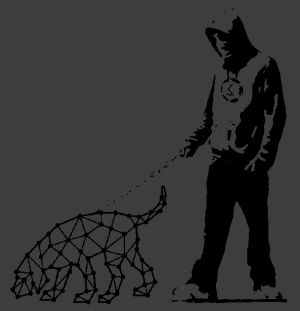
* String specific



Comparing Stuff - Operators

Example using RegEx

```
// Group SID with regex
MATCH (g:Group) WHERE g.objectsid =~ '^S-1-5-32-548$'
RETURN g
```



Hands-On: Nodes&Paths

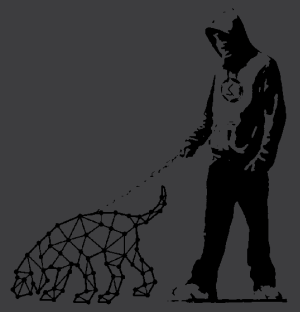
In the **BloodHound UI**:

- Search for User TRUDY_NEELD@DOMAIN.LOCAL
- Check Node properties
- Ask for shortest path from Trudy to Group DOMAIN_ADMINS@DOMAIN.LOCAL
- Ask for same path but without ACL Edges

Adding Stuff - Node

The following syntax is used to create a Node & Add props

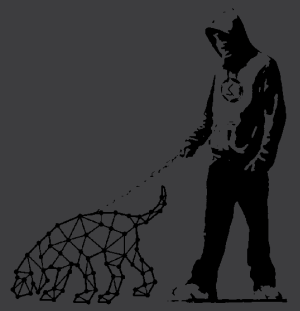
```
// Create Node
MERGE (u:User {name: 'BOB'})

// Add Props
MATCH (u:User {name: 'BOB'})
SET u.age=23, u.hair='Black'
```



Adding Stuff - Edge

The following syntax is used to create an Edge:

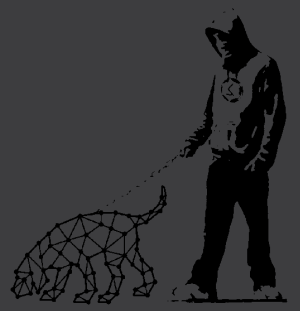
```
// Create Edge Between Nodes
MATCH (b:Human {name: 'BOB'})
MATCH (a:Human {name: 'ALICE'})
CREATE (b)-[r:Likes]->(a)
```



Deleting Stuff - Edge

The following syntax is used to delete an Edge:

```
// Delete Relationship
MATCH (b:Human {name: 'BOB'})-[r:Likes]->(a:Human {name: 'ALICE'})
DELETE r
```

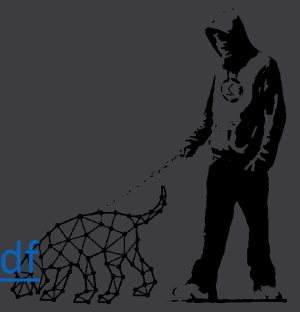


Deleting Stuff - Node

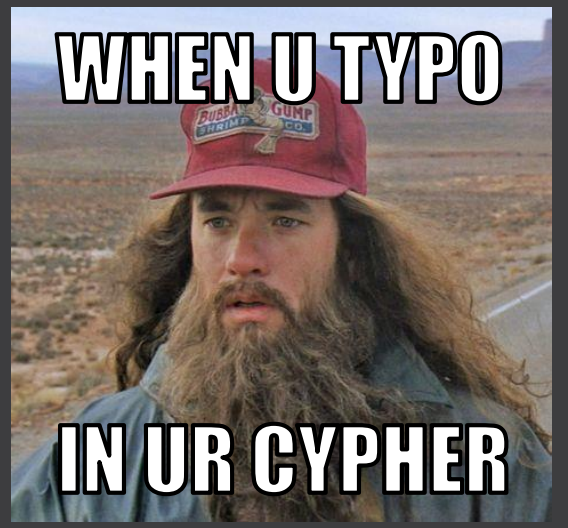
The following syntax is used to delete a Node:

```
// Delete Node
MATCH (u:User {name: 'BOB'})
DETACH DELETE u
```

More: https://www.ernw.de/download/BloodHoundWorkshop/ERNW_DogWhispererHandbook.pdf



Making Errors - Typos



It happens...

- UI will not show any error message
- Your dog will run forever
- Prefer Browser to Build/Debug queries

ERROR

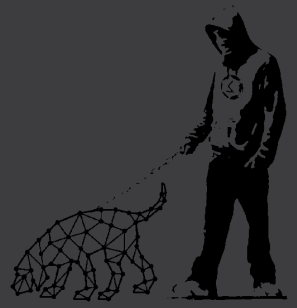
Neo.ClientError.Statement.SyntaxError

```
Neo.ClientError.Statement.SyntaxError: Invalid input ';': expected an identifier character, node la
pattern (line 1, column 9 (offset: 8))
"MATCH (u;User) RETURN u"
         ^
```

Hands-On: GodMode

Perform the following:

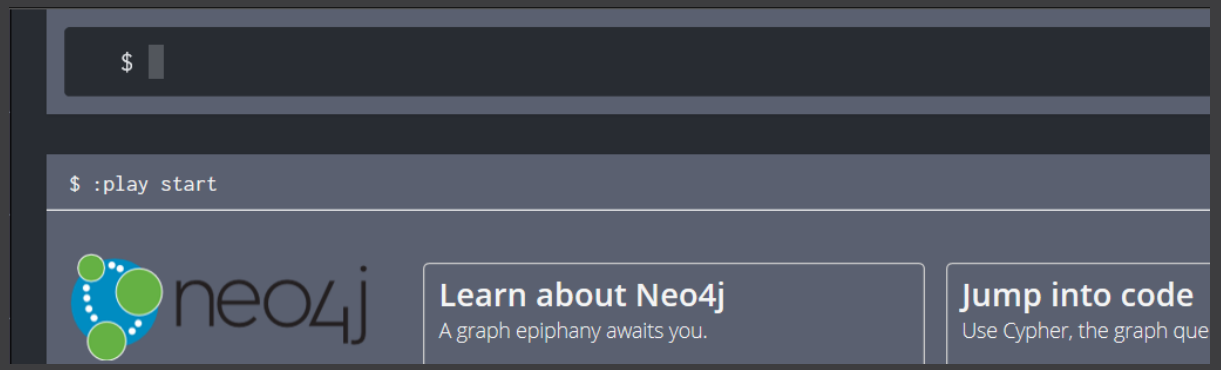
- **Create Humans** named Alice & Bob
- **Add an age property** to each
- **Create relationships** between them
- **Request path** from Alice to Bob
- **Delete an Edge**
- **Delete Both Nodes**



Neo4j Browser - 2nd Home

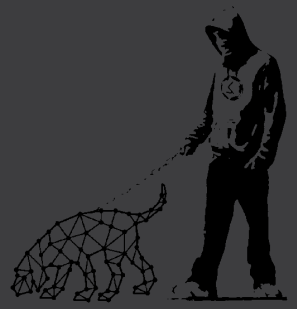
The Neo4j Browser is the best place to work on queries:

- Bigger font
- Syntax coloring
- Error messages
- **Return numbers**



And more...

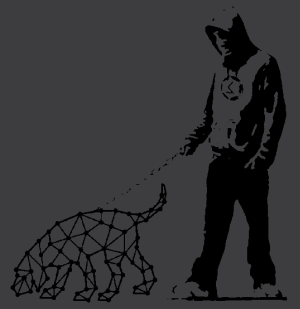
It is located at http://localhost:7474



Counting Stuff - COUNT()

The following syntax can be used to count Nodes

```
MATCH
(g:Group {name: 'DOMAIN ADMINS@SUB.DOMAIN.LOCAL'}),
p=shortestPath((x:User)-[r*1..]->(g))
RETURN COUNT(DISTINCT(x))
```



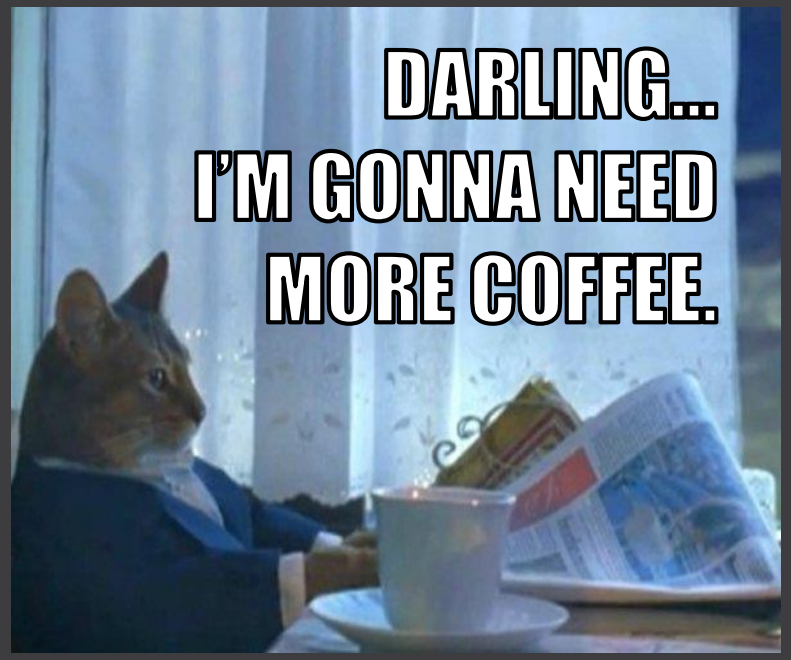
Hands-On: Browser

Using the **neo4j browser**:

- **Click around** & make yourself at home
- Try **Nodes&Paths** exercise again
- Try **GodMode** exercise again
- Try **counting** some Nodes

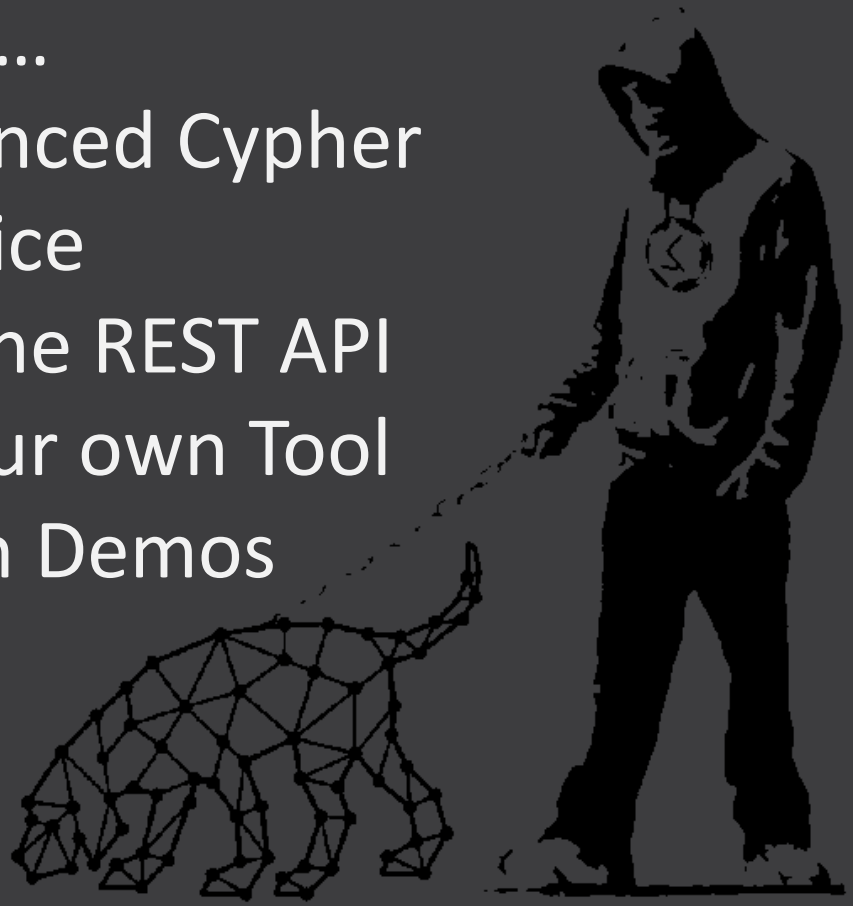
[Make Errors...]

Break [10m]



And after that...

- Some Advanced Cypher
- More Practice
- A bit over the REST API
- Building your own Tool
- Automation Demos

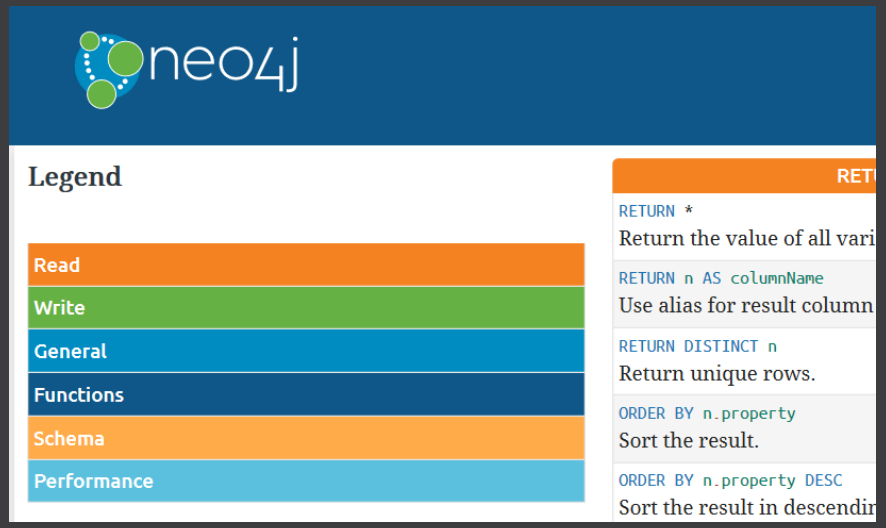


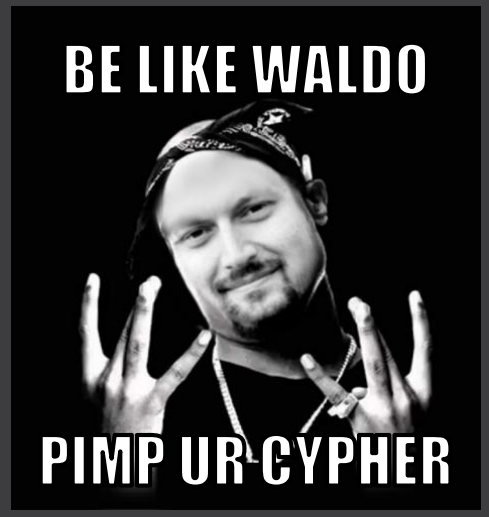
4- Cypher Advanced

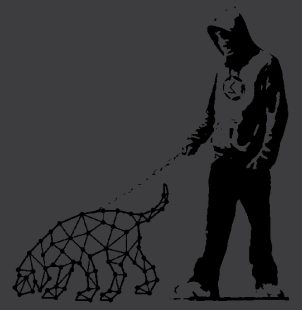
- More Functions()
- Cypher Manual
- Cypher Gallery
- Query Tuning

More Functions() - Ref Card

There are many Functions available...
Check out the **cypher Reference Card** for a quick overview...





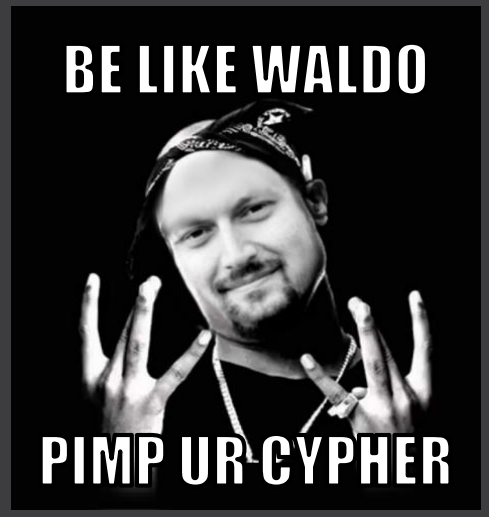


Ref: https://neo4j.com/docs/cypher-refcard/current/

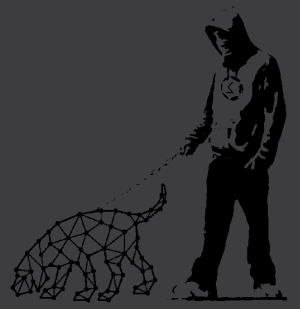
More Functions() - Example

See if you can understand what happens in this bit of cypher [use the ref card]

```
...
 WHERE ALL(x in RELS(p) WHERE (TYPE(x)='MemberOf' OR x.isacl=true))
...
```

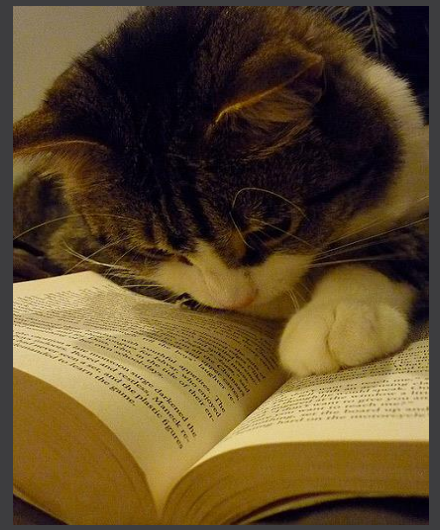


Ref: https://neo4j.com/docs/cypher-manual/3.5/



Cypher Manual - The Bible

We have only scratched the surface...
There is a full Cypher online reference waiting for you



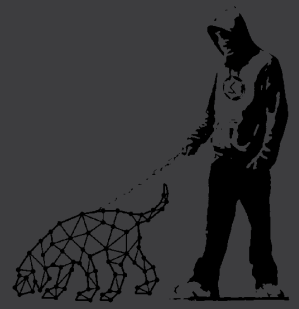
The Neo4j Cypher Manual v3.5

License: Creative Commons 4.0

This is the Cypher manual for Neo4j version 3.5, authored by the Neo4j Team.

This manual covers the following areas:

- Chapter 1, *Introduction* — Introducing the Cypher query language.
- Chapter 2, *Syntax* — Learn Cypher query syntax.
- Chapter 3, *Clauses* — Reference of Cypher query clauses.
- Chapter 4, *Functions* — Reference of Cypher query functions.
- Chapter 5, *Schema* — Working with indexes and constraints in Cypher.
- Chapter 6, *Query tuning* — Learn to analyze queries and tune them for performance.
- Chapter 7, *Execution plans* — Cypher execution plans and operators.
- Chapter 8, *Deprecations, additions and compatibility* — An overview of language developments across



Manual: https://neo4j.com/docs/cypher-manual/3.5/

Cypher Gallery - Community

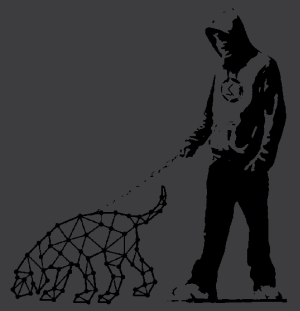
List of Cypher cheats by Community Members

https://gist.github.com/jeffmcjunkin/7b4a67bb7dd0cfbfbd83768f3aa6eb12

https://hausec.com/2019/09/09/bloodhound-cypher-cheatsheet/

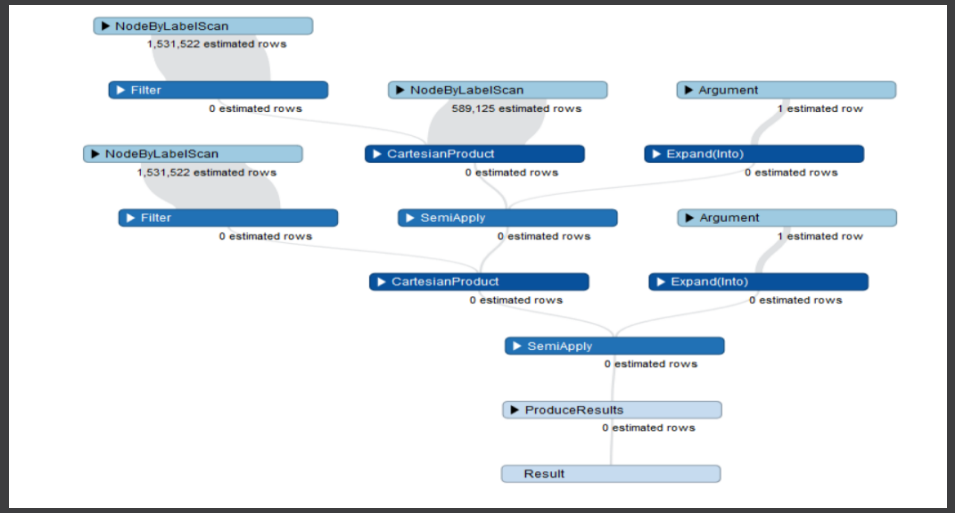
https://github.com/BloodHoundAD/BloodHound/wiki/Cypher-Query-Gallery

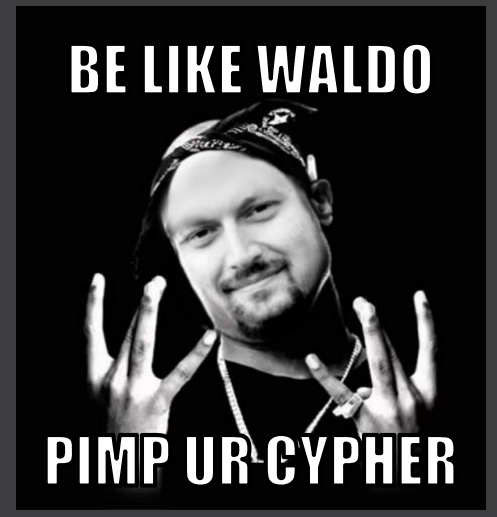
Share your cool queries on BloodHound slack...



Query Tuning - Performance

Tip: Add **EXPLAIN** or **PROFILE** in front of your Cypher Query to understand how it performs under the hood... [Browser Only]





Manual: https://neo4j.com/docs/cypher-manual/3.5/query-tuning/

Hands-On: Moar Cypher

Check out the **Cypher Gallery** links and:

- Find some cool **Queries** you like
- **Tweak** them to your pleasure
- Run them in **UI** & **Browser**
- Add **EXPLAIN** & **PROFILE** in front

[Make moar errors…]

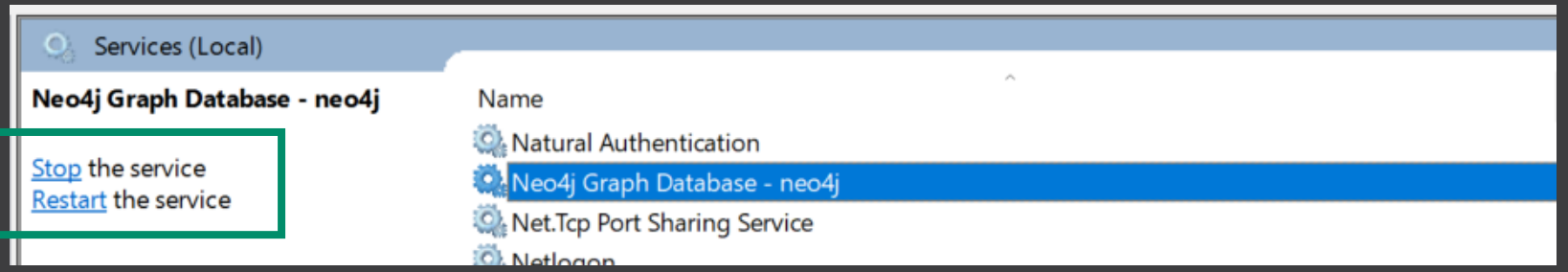
5- REST API & Automation

- REST API Basics
- Invoke-Cypher
- CypherDog Demo [Full Client]
- WatchDog Demo [Metrics & Report]

REST API - Setup

Enable **Unauthenticated** API requests [/!\ LocalHost Only /!\]

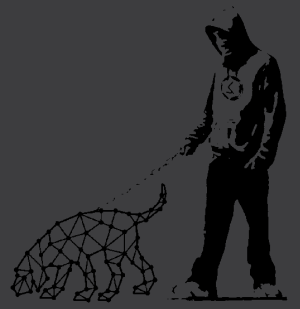
- Stop neo4j service



- Uncomment in neo4j\conf\neo4j.conf

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

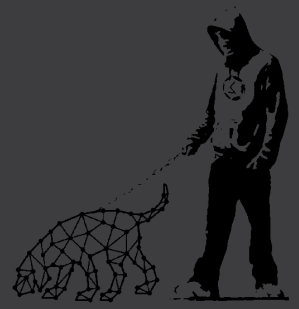
- Start neo4j service



REST API - Basic Call

```powershell
# 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

```

PowerShell example of a basic call to API

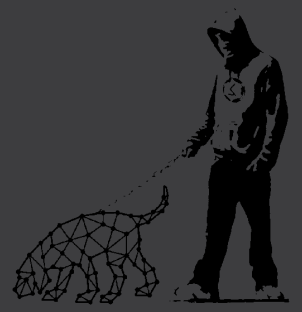


Bash: https://github.com/BloodHoundAD/BloodHound/wiki/neo4j-REST-API

REST API - Nodes

This is what a returned Node looks like [PoSh]

```
> cypher "MATCH (n:User {name: 'JACOB_NEWELL@SUB.DOMAIN.LOCAL'}) RETURN n" -Expand data


metadata                       : @{id=410; labels=System.Object[]}
data                           : @{highvalue=False; sensitive=True; enabled=True; hasspn=False
                                 domain=SUB.DOMAIN.LOCAL; name=JACOB_NEWELL@SUB.DOMAIN.LOCAL;
                                 objectsid=S-1-5-21-2505991005-2303352498-2358670217-2111; adm
paged_traverse                 : http://localhost:7474/db/data/node/410/paged/traverse/{returr
outgoing_relationships         : http://localhost:7474/db/data/node/410/relationships/out
outgoing_typed_relationships   : http://localhost:7474/db/data/node/410/relationships/out/{-li
labels                         : http://localhost:7474/db/data/node/410/labels
create_relationship            : http://localhost:7474/db/data/node/410/relationships
traverse                       : http://localhost:7474/db/data/node/410/traverse/{returnType}
extensions                     :
all_relationships              : http://localhost:7474/db/data/node/410/relationships/all
all_typed_relationships        : http://localhost:7474/db/data/node/410/relationships/all/{-li
property                       : http://localhost:7474/db/data/node/410/properties/{key}
self                           : http://localhost:7474/db/data/node/410
incoming_relationships         : http://localhost:7474/db/data/node/410/relationships/in
properties                     : http://localhost:7474/db/data/node/410/properties
incoming_typed_relationships   : http://localhost:7474/db/data/node/410/relationships/in/{-lis
```

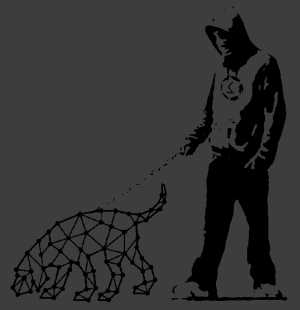


REST API - Paths

This is what a returned Path looks like [PoSh]

```
relationships : {http://localhost:7474/db/data/relationship/50808, http://localhost:7474/db/data/relationship/46395,
                http://localhost:7474/db/data/relationship/50312, http://localhost:7474/db/data/relationship/50313...}
nodes         : {http://localhost:7474/db/data/node/373, http://localhost:7474/db/data/node/477, http://localhost:7474/db/data/node/48395,
                http://localhost:7474/db/data/node/472...}
directions    : {->, ->, ->, ->...}
length        : 5
start         : http://localhost:7474/db/data/node/373
end           : http://localhost:7474/db/data/node/47074
```

More calls will need to be made to retrieve Node & Edge info [curl/irm]



Invoke-Cypher - Cmdlet

Invoke-Cypher is a simple Cmdlet to send Cypher queries to the BloodHound REST API.

```
> help Invoke-Cypher

NAME
    Invoke-Cypher

SYNOPSIS
    Invoke Cypher


SYNTAX
    Invoke-Cypher [-Query] <String> [[-Params] <Hashtable>] [[-Expand] <String[]>] [<CommonParameters>]


DESCRIPTION
    Post Cypher Query to BloodHound REST API
```

Code: https://github.com/SadProcessor/HandsOnBloodHound/blob/master/Workshop/Invoke-Cypher.ps1

Hands-On: API Calls

Read the **Invoke-Cypher**.ps1 Cmdlet code and help page. Using the Cmdlet:

- Try **Nodes&Paths e**xercise again
- Try **GodMode** exercise again
- Try other queries of your creation

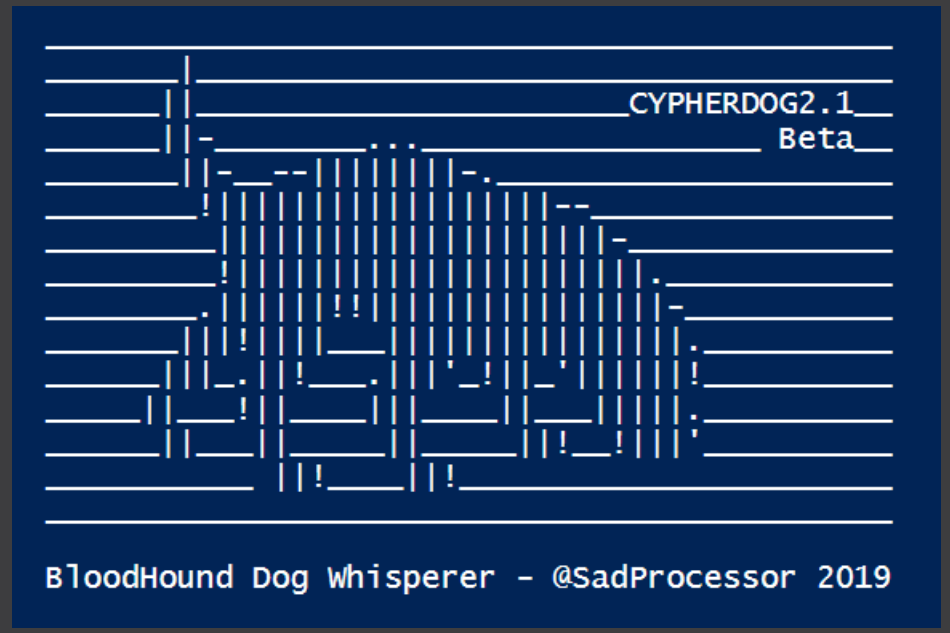
[**Get-Help** Invoke-Cypher -Full]

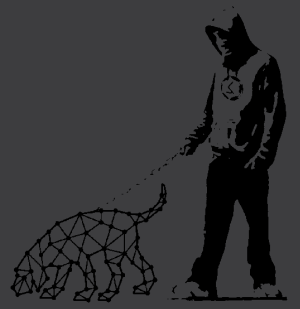
Tool Demos [Sit back, relax...]

- CypherDog
- WatchDog

CypherDog - Demo

CypherDog is a PowerShell BloodHound Client allowing Data Manipulation & Automation

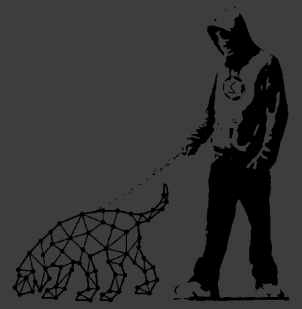




WatchDog - Demo

WatchDog is an BloodHound Data Scanner [POC/WiP]

```
Top20 Overall - TotalImpact [ 19 : 637 : 100 ]
--------------------------

Type     Name                                    Hit Weight Impact
----     ----                                    --- ------ ------
Group    ENTERPRISE ADMINS@DOMAIN.LOCAL           19    158   24.8
User     SYBLE_LEININGER@DOMAIN.LOCAL             19    143   22.4
Group    ACCOUNT OPERATORS@DOMAIN.LOCAL           18    141   22.1
User     HAZEL_DUNFEE@SUB.DOMAIN.LOCAL            19    104   16.3
Computer WS_4.DOMAIN.LOCAL                        19     95   14.9
Group    REMOTE MANAGEMENT USERS@DOMAIN.LOCAL     19     95   14.9
User     EUGENIE_HITES@DOMAIN.LOCAL               19     86   13.5
Computer WS_17.SUB.DOMAIN.LOCAL                   18     85   13.3
Computer WS_12.SUB.DOMAIN.LOCAL                   19     79   12.4
Group    DOMAIN GUESTS@SUB.DOMAIN.LOCAL           19     79   12.4
Group    DOMAIN ADMINS@DOMAIN.LOCAL               18     77   12.1
Group    DISTRIBUTED COM USERS@SUB.DOMAIN.LOCAL   16     66   10.4
User     SHERWOOD_ENDRES@DOMAIN.LOCAL             19     65   10.2
User     MICHEAL_MAURER@DOMAIN.LOCAL              19     55    8.6
Computer SRV_9.DOMAIN.LOCAL                       19     55    8.6
User     PENNI_ROGAN@DOMAIN.LOCAL                 19     54    8.5
Group    RAS AND IAS SERVERS@SUB.DOMAIN.LOCAL     19     54    8.5
User     SOLEDAD_UHRIG@DOMAIN.LOCAL               19     47    7.4
User     THI_RODKEY@DOMAIN.LOCAL                  19     47    7.4
User     LOREAN_EUGENE@DOMAIN.LOCAL               19     45    7.1
```

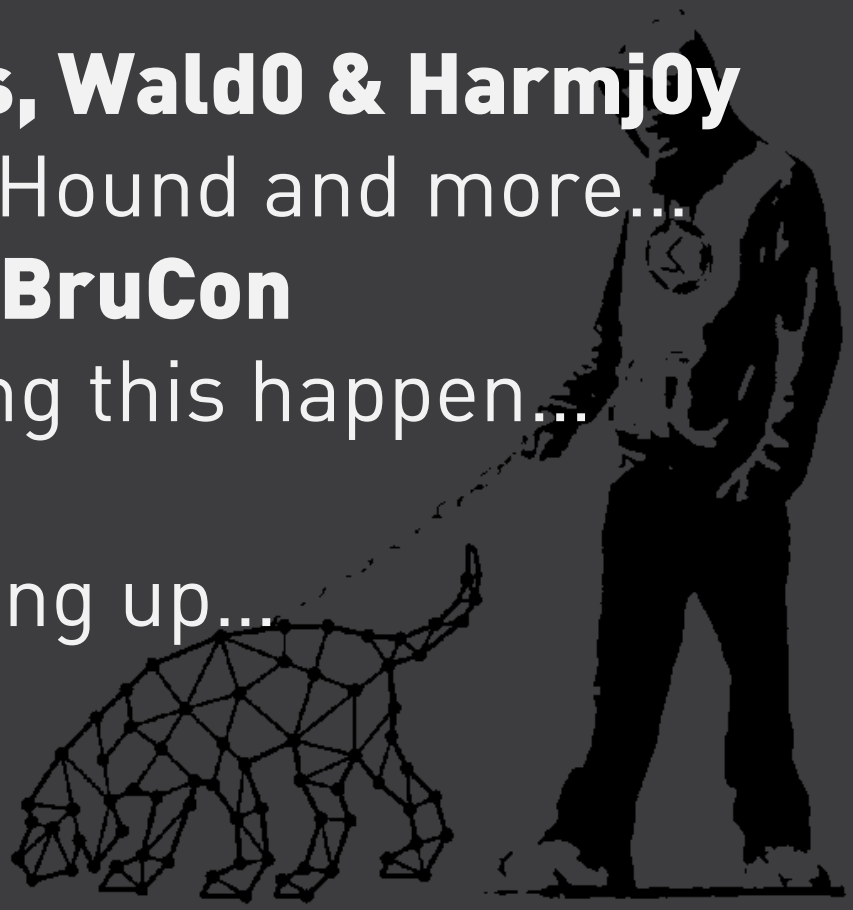


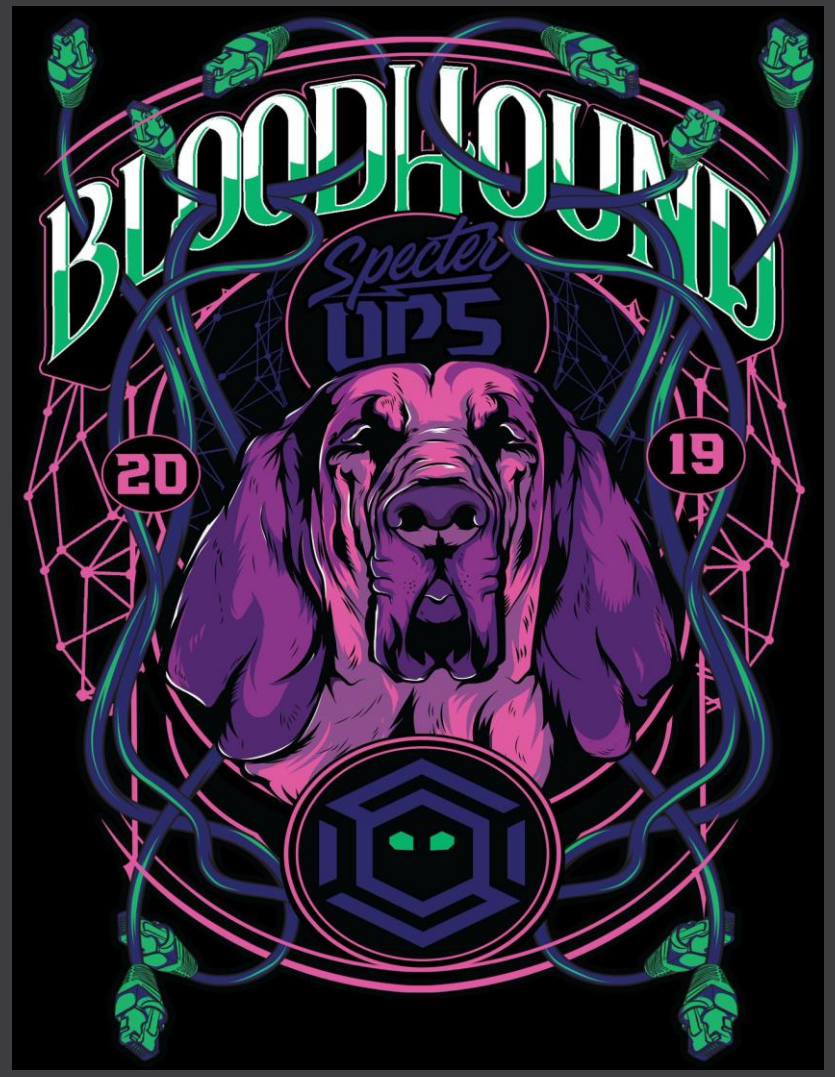
Questions...

If we have time.
If not, see you on the
BloodHound Slack...

Thank You...

- **CptJesus, Wald0 & Harmj0y**
  for BloodHound and more...
- **ERNW & BruCon**
  for making this happen...
- **You**
  for showing up...





Support...

BloodHound is a great tool.
And it's **free**.

If you use it on a regular basis, why not support a **good cause** and treat yourself with some **BloodHound Swag**...

All benefits go to **Charity**