SharpHound Enumeration Options

--CollectionMethod
Default – Group membership, domain trust, local admin, sessions
Group – Group membership
LocalAdmin – Local admin rights
RDP – Remote Desktop Users
ComputerOnly – Local admin, RDP, DCOM and, sessions
Trusts – Domain trust collection
DcOnly – Only uses LDAP to collect groups, trusts, ACL, ObjectProps, Container and GPOLocalGroup
All – All of the above except GPOLocalGroup and LoggedOn

--d *<domainname>*
Provide domain name to enumerate
--Stealth
Lowers noise, runs single-threaded
--ExcludeDomainControllers
Exclude domain controllers, stealthier
--ComputerFile *<file>*
Provide a list of computer names or IPs
--LDAPFilter <filter>
Filter on specific AD attributes

SharpHound Connection & Performance Options

--DomainController *<domain controller>*
Specify which Domain Controller to use
--Stealth
Lowers noise, runs single-threaded
--Throttle *<value>*
Delay between requests in milliseconds, default=0
--Jitter *<value>*
Jitter to the Throttle value, in %

SharpHound Output Options

--OutputDirectory *<directory>*
To store the JSON output files, default = .
--OutputPrefix *<prefix>*
Prefix to the JSON output files
--PrettyJson
Add indentation to JSON for readability
Increases file size.
--NoZip
Do not zip JSON files
--ZipFileName *<name>*
Specify the filename for the zip
--EncryptZip
Add password to zip (randomly generated)

SharpHound Loop Options

--Loop
Loop computer collections
--LoopDuration hh:mm:ss
Duration of the loop
--LoopInterval hh:mm:ss
Wait time between loops

Download – Install - Use

SharpHound download
```
C#: https://github.com/BloodHoundAD/SharpHound3
ps1: https://github.com/BloodHound/Collectors
Python: https://github.com/fox-it/BloodHound.py
```

BloodHound Installation:
`https://bloodhound.readthedocs.io`

Using BloodHound:
```
1. Start Neo4j
2. Drag-and-Drop the .zip file from SharpHound
   onto the BloodHound interface
3. The hamburger on the top left shows the menu
4. <Analysis> contains pre-build queries
5. The highway-button opens the path-finding mode
6. Click a node to show details about the node
7. <Unrolled> items will show parent items
```

OFFENSIVE OPERATIONS

BloodHound Cheat Sheet v1.0

By Michiel Lemmens @mchllmmns

SANS

sans.org/offensive-operations

This cheat sheet will help you in your Active Directory attacks.
Related course – SANS SEC560: Network Penettration Testing and Ethical Hacking

How To Use This Sheet

This cheat sheet will help you in Active Directory data collection, analysis and visualization using BloodHound.

Bloodhound uses Neo4j as database, with Cypher as the query language.

SharpHound is a popular tool for collecting the raw AD data through a domain-connected system.

This sheet is split into these sections:
- Page 1:
  - SharpHound options, usage
- Page 2:
  - Handy DB queries, DB query buildup

The proof of the pudding is in the tasting...

Handy DB queries

Computers

Count the LAPS-status of all computers
```
MATCH (c:Computer) RETURN c.haslaps, COUNT(*)
```

Get a list of all OS versions with a count
```
MATCH (c:Computer) RETURN DISTINCT
c.operatingsystem, COUNT(c.operatingsystem)
```

Get a list of all OS versions containing 'Server'
```
MATCH (c:Computer) WHERE c.operatingsystem
CONTAINS 'Server' RETURN DISTINCT
c.operatingsystem
```

Get all Windows 2008 computers and sort by last logon timestamp descending and human readable
```
MATCH (c:Computer) WHERE c.operatingsystem
CONTAINS '2008' RETURN c.name,
c.operatingsystem, datetime({ epochSeconds:
toInteger(c.lastlogontimestamp) }) AS rdate
ORDER BY rdate DESC
```

Users

Get all Domain Admins
```
MATCH (g:Group) WHERE g.name =~ "(?i).*DOMAIN
ADMINS.*" WITH g MATCH (g)<-[r:MemberOf*1..]-
(a) RETURN a.name
```

Get active sessions of Domain Admins
```
MATCH (u:User)-[:MemberOf*1..]->(g:Group)
WHERE g.objectid ENDS WITH '-512' MATCH p =
(c:Computer)-[:HasSession]->(u) return
c.name, u.name
```

Find all Kerberoastable users
```
MATCH (u:User) WHERE u.hasspn=true
RETURN u.name
```

Find all AS-REP-roastable users
```
MATCH (u:User {dontreqpreauth: true}) RETURN
u.name
```

Handy DB queries

Users (cont.)

Get the local admins to all computers
```
MATCH p=(u:User)-[r:AdminTo]->(c:Computer)
RETURN u.name, c.name ORDER BY u.name
```

Find all Kerberoastable users with path to DA
```
MATCH (u:User {hasspn:true}) MATCH (g:Group)
WHERE g.name CONTAINS 'DOMAIN ADMINS' MATCH p
= shortestPath( (u)-[*1..]->(g) ) RETURN p
```

Find all computers domain users can RDP to
```
MATCH p=(g:Group)-[:CanRDP]->(c:Computer)
WHERE g.objectid ENDS WITH '-513' RETURN p
```

Find users that haven't logged in for 90 days
```
MATCH (u:User) WHERE u.lastlogon >
(datetime().epochseconds - (90 * 86400)) AND
NOT u.lastlogon IN [-1.0, 0.0] RETURN u.name
```

Find users with passwords older than 90 days
```
MATCH (u:User) WHERE u.pwdlastset >
(datetime().epochseconds - (90 * 86400)) AND
NOT u.pwdlastset IN [-1.0, 0.0] RETURN u.name
```

Find all sessions a user has in a domain
```
MATCH p=(c:Computer)-[r:HasSession]->(u:User
{domain: "LAB.LOCAL"}) RETURN c.name, u.name
```

Find all users member of high-value groups
```
MATCH p=(u:User)-[r:MemberOf*1..]->(g:Group
{highvalue:true}) RETURN u.name
```

Find dangerous rights for Domain Users group
```
MATCH p=(g:Group)-
[:Owns|WriteDacl|GenericAll|WriteOwner|Execut
eDCOM|GenericWrite|AllowedToDelegate|ForceCha
ngePassword]->(c:Computer) WHERE g.objectid
ENDS WITH "-513" RETURN p
```

GPOs

Find all GPOs that contain keyword
```
Match (g:GPO) WHERE toUpper(g.name) CONTAINS
"SERVER" RETURN g
```

DB query buildup

MATCH p=(c:Computer)-[r:HasSession]->(u:User {domain:"lab"})

MATCH indicates a search
p,c,r,u are parameters (you choose)
Computer, User are the AD object type
HasSession is a relationship

WHERE c.operatingsystem CONTAINS 'Server'

WHERE will allow for filtering (optional). Can be followed by NOT. Case sensitive.
Combinations possible:
WHERE NOT (c.name = "DC" AND c.domain = "lab")
CONTAINS does substring search. Others: STARTS WITH, ENDS WITH, =, <, >, ...

RETURN c.name, COUNT(*)

RETURN indicates the return value
c.name is the name property for c
, COUNT(*) Count per c.name the number of occurrences (optional)
Alternative:
RETURN COUNT(*): count the total of items (can contain multiple times the same item)
RETURN DISTINCT COUNT(*): Count unique items

Sources

https://bloodhound.readthedocs.io/en/latest/data-collection/sharphound-all-flags.html

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

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

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

https://github.com/SadProcessor/Cheats/blob/master/DogWhispererV2.md

https://blog.compass-security.com/2020/07/make-the-most-out-of-bloodhound/

https://phackt.com/pentesting-bloodhound-cypher-queries

https://github.com/awsmhacks/awsmBloodhoundCustomQueries