

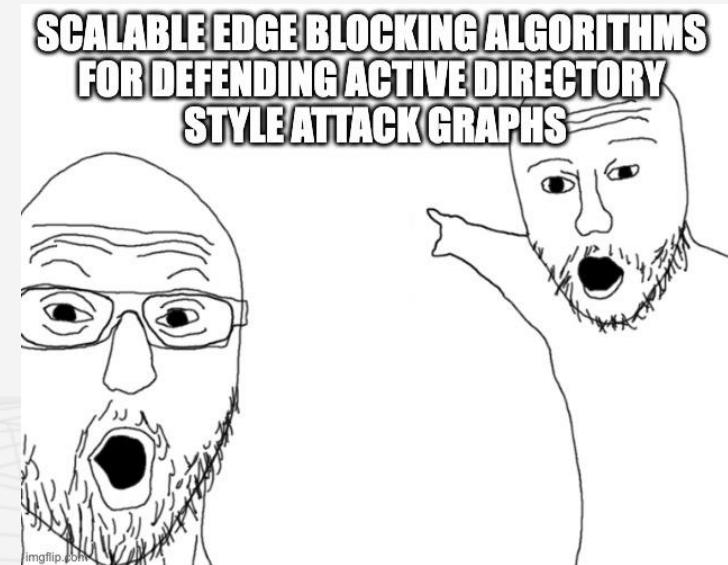


Woof, Woof! Meet Sniffer BloodHound

An Algorithm-Driven Framework for Attack Path Analysis

01 04, 2025

Big Big Thanks to
Mingyu Guo, Max Ward
Aneta Neumann, Frank Neumann, Hung Nguyen



ref.: <https://arxiv.org/abs/2212.04326>


JimmySu

- > Gamer / Researcher / Dog Lover (including Bloodhound)
- > Focus on identity-related attack technique




Maxine Shih

- > Mostly a homebody—unless you tempt me with a latte or a farmers' market
- > Focus on machine learning



**“Defenders think in lists.
Attackers think in graphs.
As long as this is true, attackers win.”**

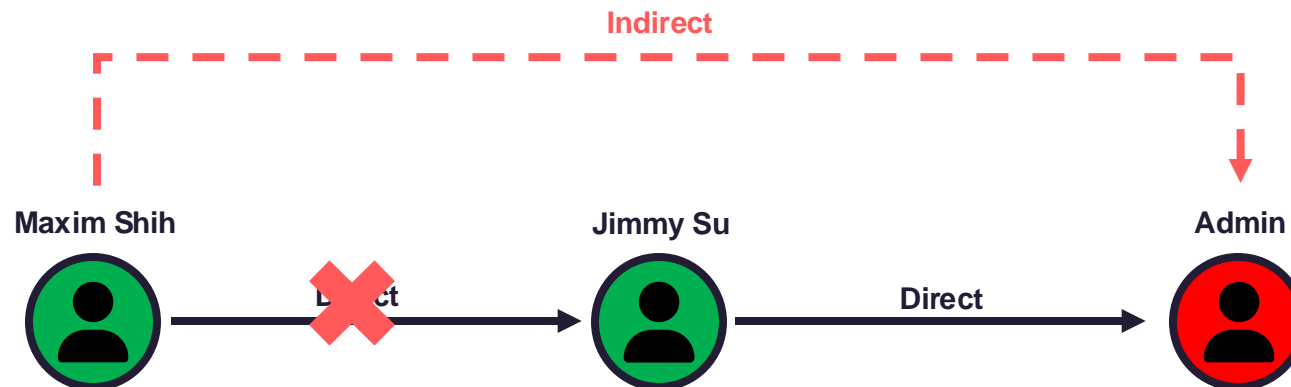


John Lambert



Six Degrees of Domain Admin

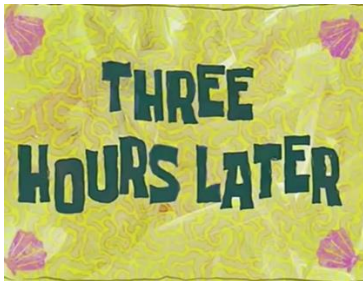
- > Lots of people have access to AD today, let's focus on AD.
- > Most of the time, we can turn usable permissions, misconfigurations into Graph.
- > With this, we can start to analysis what's the problem within this environment.
- > What is the problem? How to analysis this type AD graph?



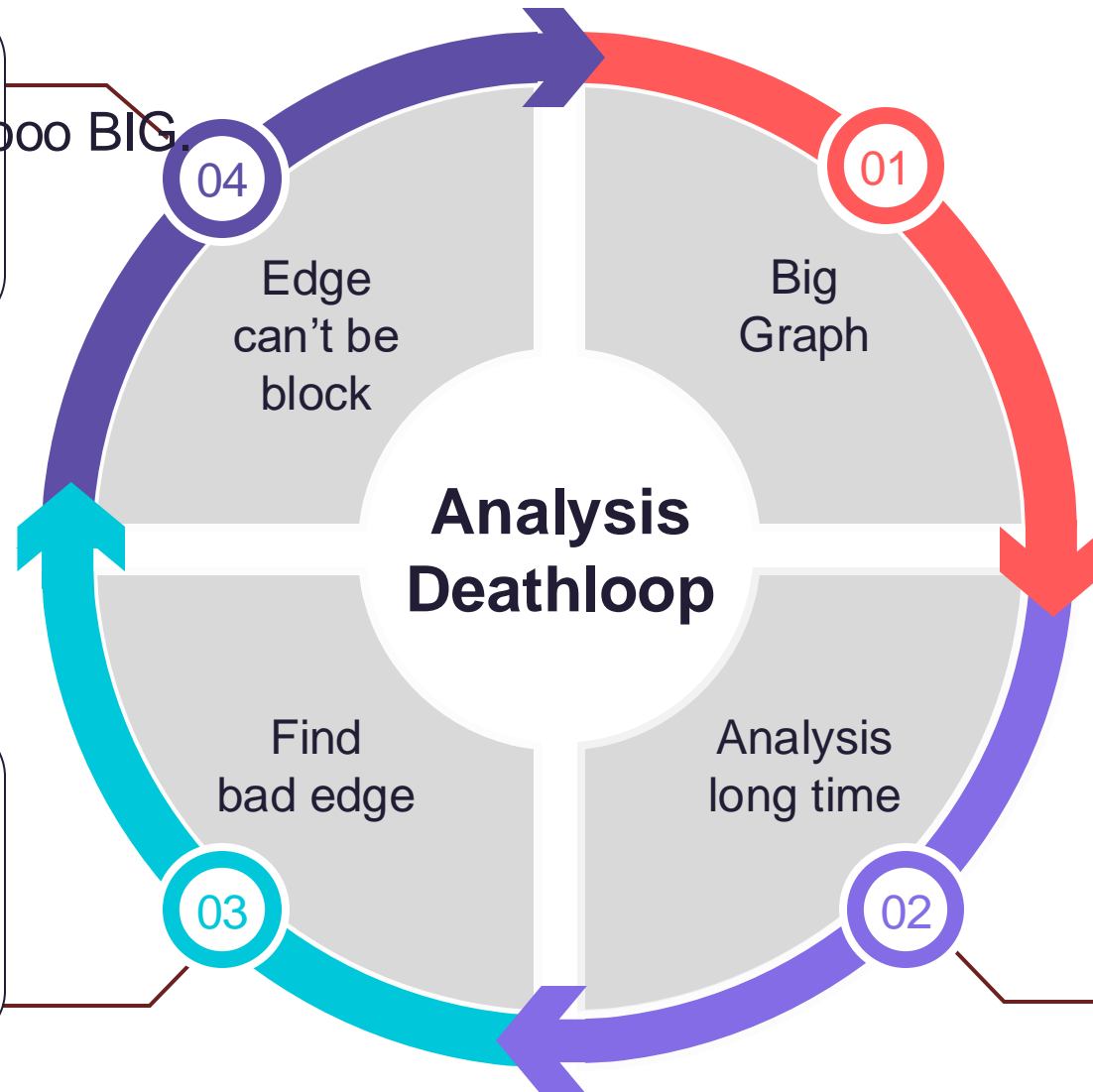
ref.: <https://gifer.com/en/TFdE>

Challenges We Faced...

- > Report to customer,
- > Customer said this edge is essential :(



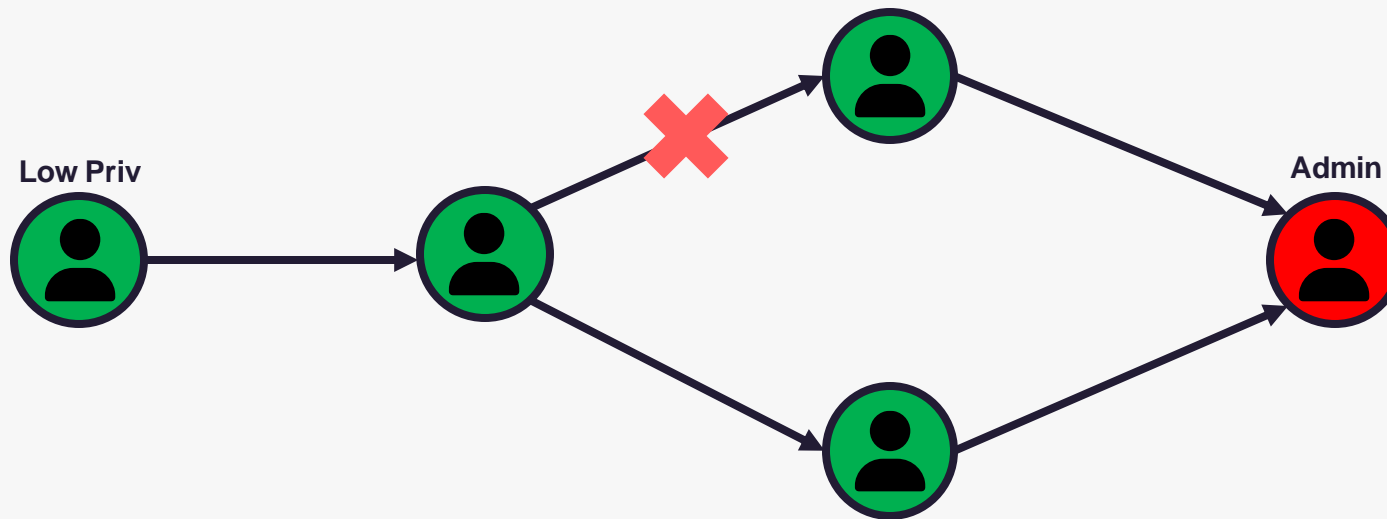
- > After long time, finally find a bad edge that can be block



- > Because Graph is too big, so it's hard to analysis in short time.

We also Faced this...

- > With limited human resources, we want to maximize their effectiveness.
- > Can we block **ONLY ONE** edge, then block **MORE THAN ONE** path?
- > Consider this situation:





Sniffer BloodHound Goals

- > Solving all challenges we mentioned earlier, including
 - > Analyzing big AD graph **efficiently**
 - > **Prioritizing** which attack path should analysis first by finding **critical edges**, helping reduce analysis time
- > Beside those challenges, we also want to
 - > Let user can use Sniffer BloodHound seamlessly on BloodHound
 - > Fit analysis methodology, e.g., Analysis in full path...

Main Questions to be solved

- > How to turn AD Graph into a **computable Graph**?
- > Even we can turn AD Graph into computable Graph, how can we find **critical edges** in this Graph?
- > Can we find critical edges efficiently? Can we provide a mathematical way that guarantee consistent results every time?
- > How to integrate into BloodHound?



Inside the Algorithm: From Risk Modeling to Protection Strategy





**Now, everything is ready
Let's put things together**

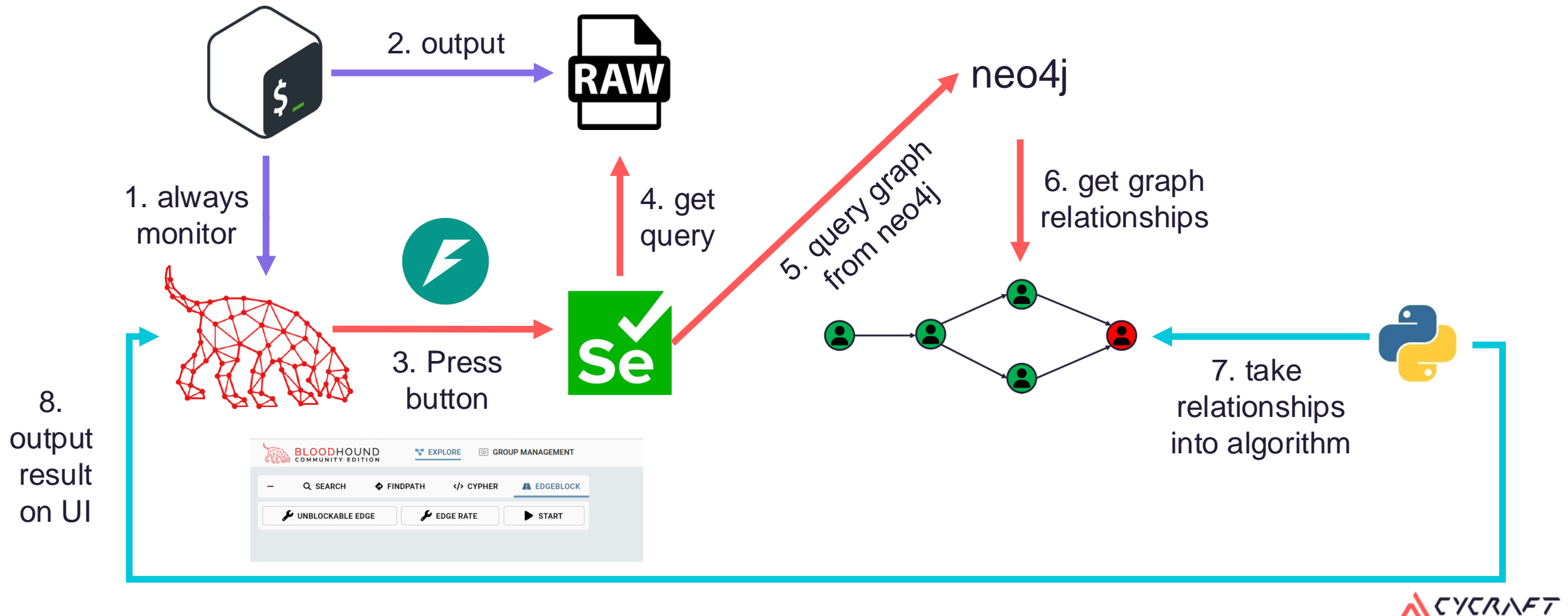


Sniffer BloodHound

- > An algo-driven framework
- > Design a flow that can call algorithm program inside BloodHound's UI
 - > We add few buttons on BloodHound UI
 - > Use FastAPI call other programs
 - > Use selenium to fetch neo4j data because algorithm need data from neo4j
 - > Auto-monitoring user's cypher query



Sniffer BloodHound Workflow



Other F

Blocking Plan Report

Please prioritize blocking these edges

> Blocked

> For defe

> For atta

```
1. Attack the best path
2. View paths
3. Select the attack path
4. Generate report

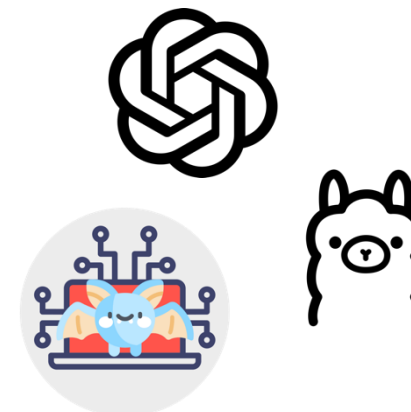
Choose options (1-4):
1
Select best attack paths timing:
1. Before block
2. After block
Choose options (1-2): 1
Start attack best attack paths before blocking.

C:\temp\bloodyAD\bloodyAD\bloodyAD\__init__.py
[+] Done, 2 edges have been found between STACY.GENEVIEVE@HYBRID.LOCAL and ESC13GROUP@HYBRID.LOCAL

Authenticated as None:

[+] [GenericAll given] on JIMMY@HYBIRD.LOCAL to STACY.GENEVIEVE@HYBRID.LOCAL
[+] [ESC1 attack] by JIMMY@HYBIRD.LOCAL using AdminCertLogin template become administrator
```

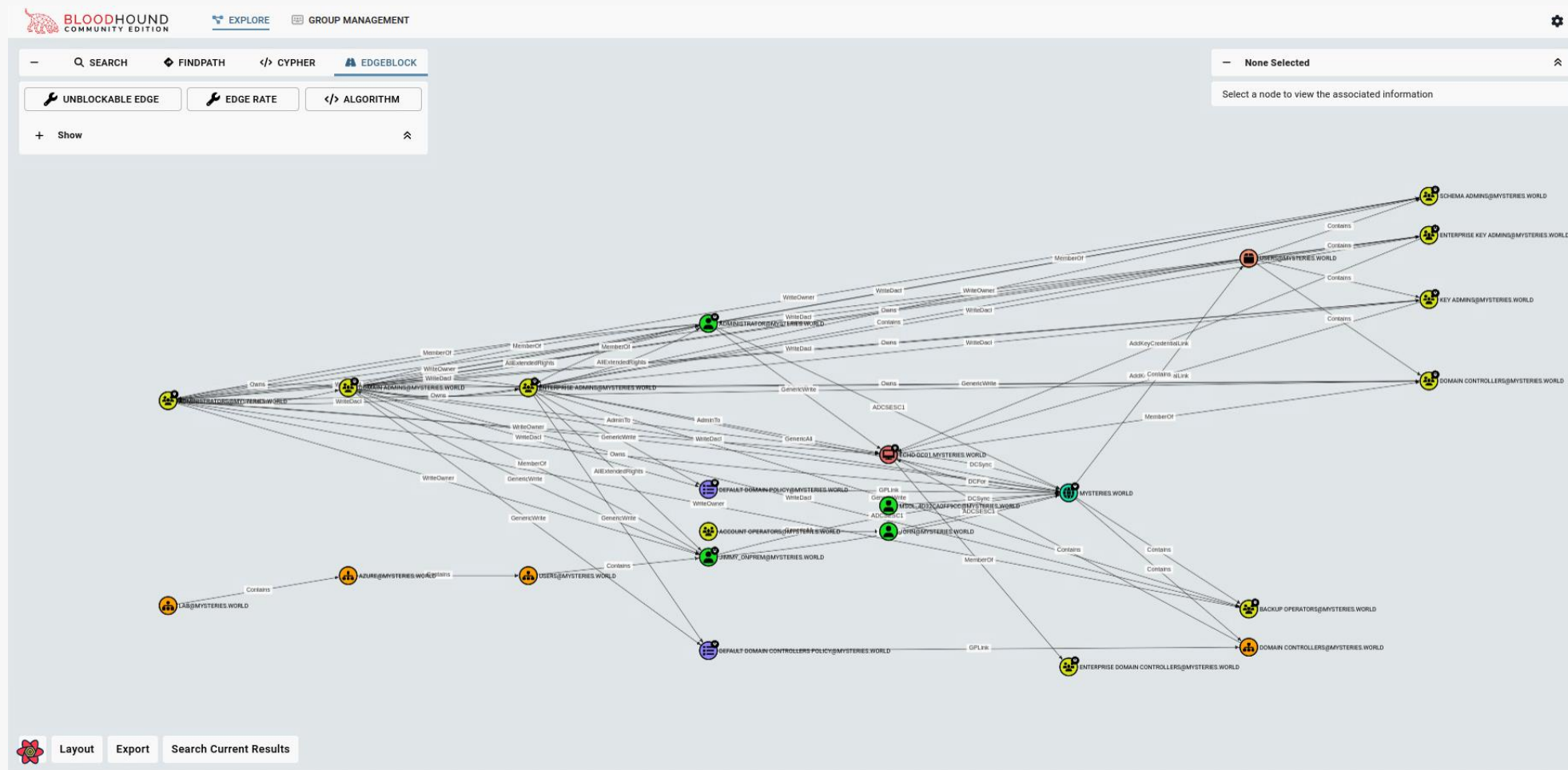
If the recommended edges are not addressed, potential attack paths remain within the environment. Pay special attention to the activities of objects mentioned in the "Unexploitable Attack Paths after Block" section to prevent misuse.

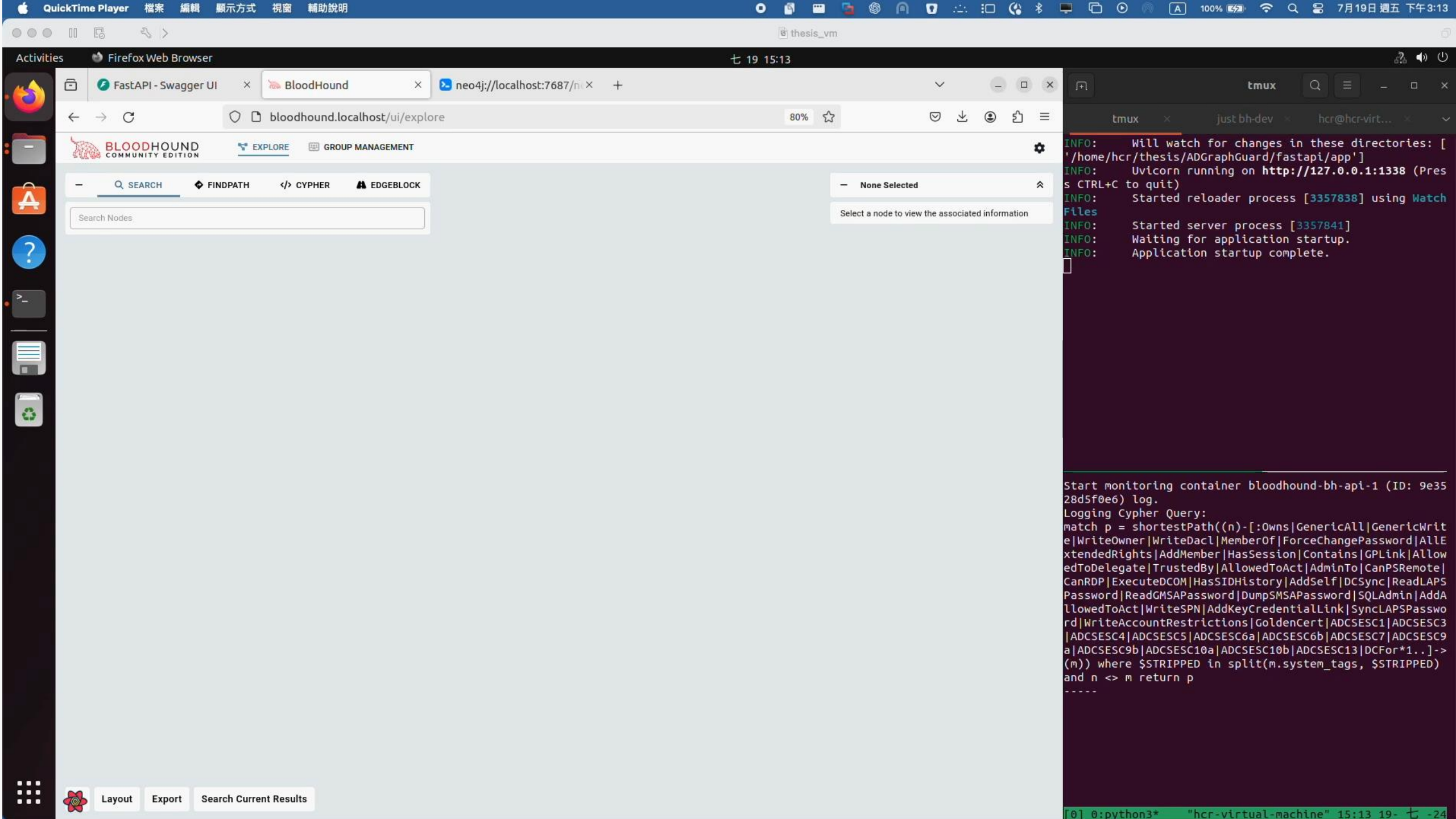
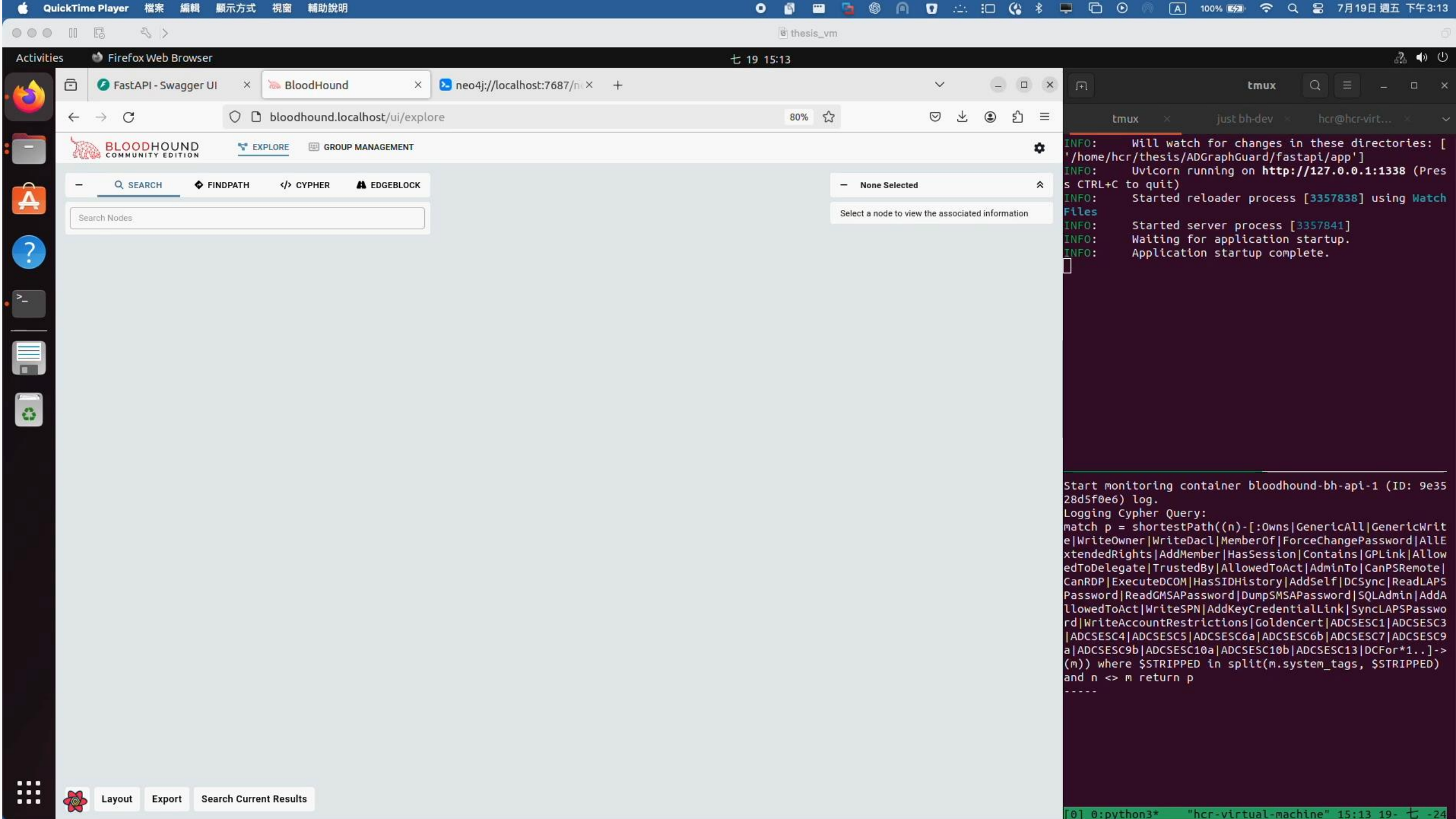


Cast Study



Cast Study 1





Recommend Block Edge

- > 1. 2. 3. 4. 5. 6. The output paths can be observed to decide whether to block them.
- > 7. is the AD CS attacks, ESC1.

```
1.MSOL_4D32CA0FF9CC@MYSTERIES.WORLD = [DCSync] => MYSTERIES.WORLD

2.DEFAULT DOMAIN POLICY@MYSTERIES.WORLD = [GPLink] => MYSTERIES.WORLD

3.DEFAULT DOMAIN CONTROLLERS POLICY@MYSTERIES.WORLD = [GPLink] => DOMAIN CONTROLLERS@MYSTERIES.WORLD

4.AZURE@MYSTERIES.WORLD = [Contains] => USERS@MYSTERIES.WORLD

5.ECHO-DC01.MYSTERIES.WORLD = [MemberOf] => DOMAIN CONTROLLERS@MYSTERIES.WORLD

6.ECHO-DC01.MYSTERIES.WORLD = [DCSync] => MYSTERIES.WORLD

7.JOHN@MYSTERIES.WORLD = [ADCS1] => MYSTERIES.WORLD
```

Before Blocking, Attack Path on AD

- > Upon reviewing the complete path, we determined that certain edges should not be blocked due to potential operational issues within the environment.
- > Consequently, these edges were added to the unblockable path list.

```
data > best_attack_path > before_attack_paths.csv
1 0,0.8,DEFAULT DOMAIN CONTROLLERS POLICY@MYSTERIES.WORLD,GPLink,DOMAIN CONTROLLERS@MYSTERIES.WORLD
2 1,0.8,ECHO-DC01.MYSTERIES.WORLD,MemberOf,DOMAIN CONTROLLERS@MYSTERIES.WORLD
3
```

1.MSOL_4D32CA0FF9CC@MYSTERIES.WORLD = [DCSync] => MYSTERIES.WORLD

2.DEFAULT DOMAIN POLICY@MYSTERIES.WORLD = [GPLink] => MYSTERIES.WORLD

3.DEFAULT DOMAIN CONTROLLERS POLICY@MYSTERIES.WORLD = [GPLink] => DOMAIN CONTROLLERS@MYSTERIES.WORLD

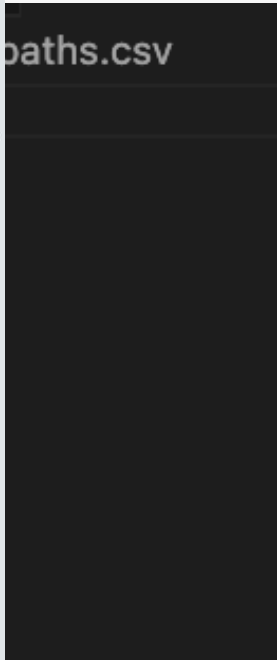
4.AZURE@MYSTERIES.WORLD = [Contains] => USERS@MYSTERIES.WORLD

5.ECHO-DC01.MYSTERIES.WORLD = [MemberOf] => DOMAIN CONTROLLERS@MYSTERIES.WORLD

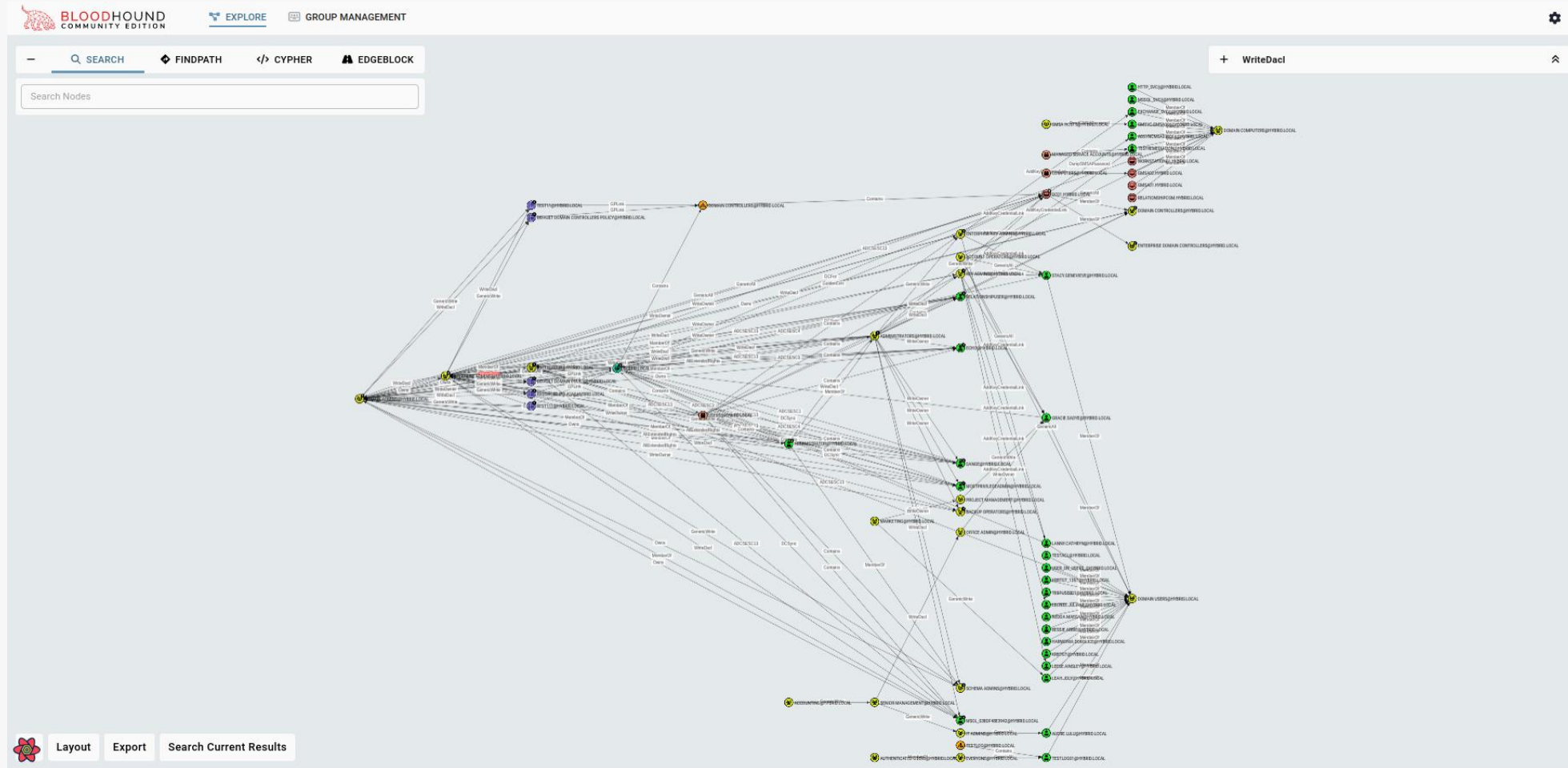
6.ECHO-DC01.MYSTERIES.WORLD = [DCSync] => MYSTERIES.WORLD

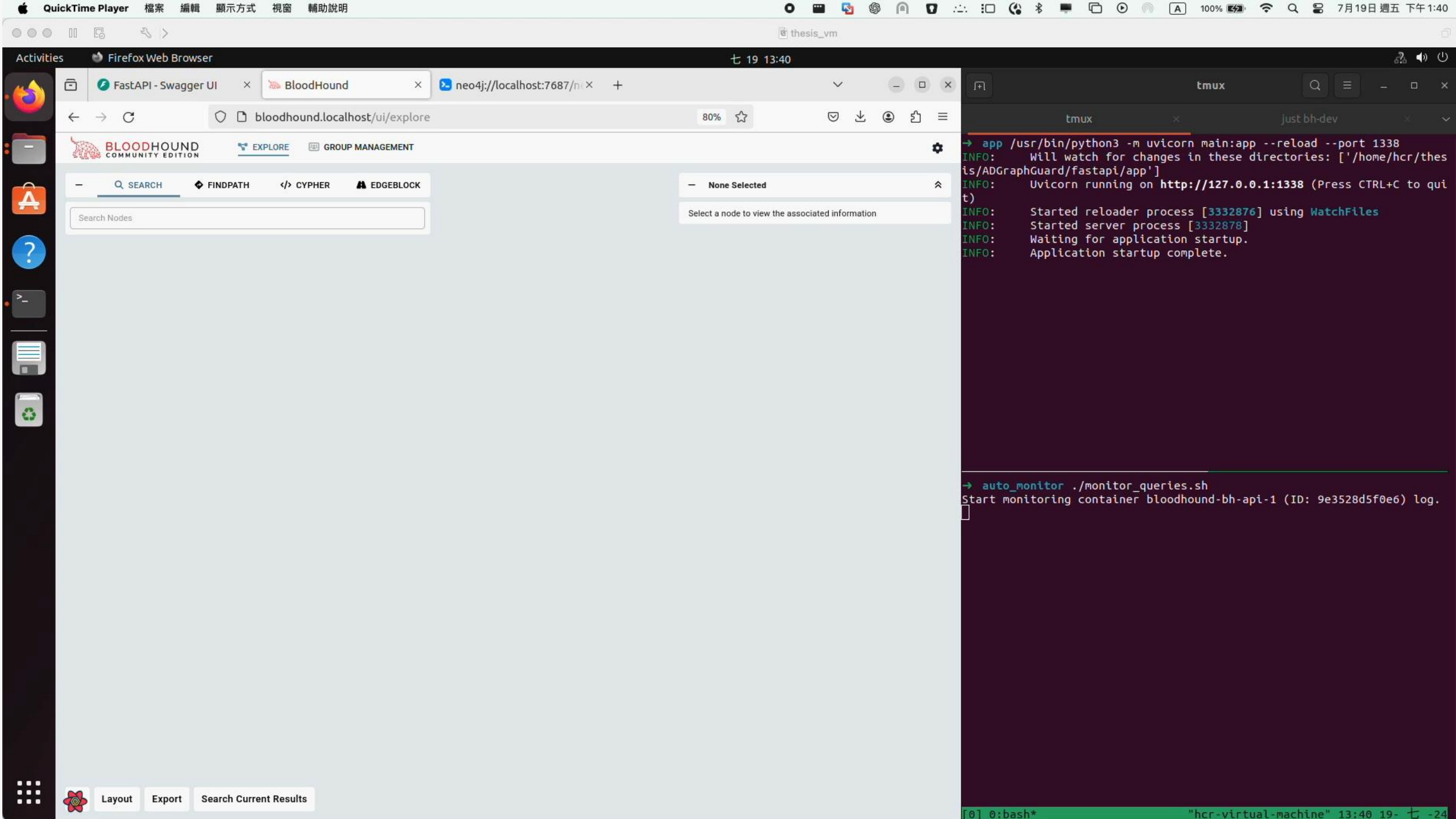
7.JOHN@MYSTERIES.WORLD = [ADCSesC1] => MYSTERIES.WORLD

> After the



Cast Study 2





Recommend Block Edge

- > 1. 2. These are the AD CS attack, ESC13.
- > 3. 4. 5. 6. The output paths can be observed to decide whether to block them.
- > 7. 8. 9. These need to be reviewed as they may indicate unexpected user logins to the DC.

1.DOMAIN USERS@HYBRID.LOCAL = [ADCSESC13] =>
ESC13GROUP@HYBRID.LOCAL

2.DOMAIN COMPUTERS@HYBRID.LOCAL = [ADCSESC13] =>
ESC13GROUP@HYBRID.LOCAL

3.HYBRID.LOCAL = [Contains] => ADMINISTRATORS@HYBRID.LOCAL

4.HYBRID.LOCAL = [Contains] => USERS@HYBRID.LOCAL

5.HYBRID.LOCAL = [Contains] => BACKUP OPERATORS@HYBRID.LOCAL

6.DC01.HYBRID.LOCAL = [MemberOf] => DOMAIN
CONTROLLERS@HYBRID.LOCAL

7.DC01.HYBRID.LOCAL = [HasSession] => ECHO@HYBRID.LOCAL

8.DC01.HYBRID.LOCAL = [HasSession] => DANGE@HYBRID.LOCAL

9.DC01.HYBRID.LOCAL = [HasSession] =>
ADMINISTRATOR@HYBRID.LOCAL

Before Blocking, Attack Path on AD

- > Upon reviewing the complete path, we determined that certain edges should not be blocked due to potential operational issues within the environment.
- > Consequently, these edges were added to the unblockable path list.

```
data > best_attack_path > before_attack_paths.csv
1 0,0.64,PASSWORD POLICY@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,ADMINISTRATORS@HYBRID.LOCAL
2 1,0.64,PASSWORD POLICY@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,BACKUP OPERATORS@HYBRID.LOCAL
3 2,0.64,DEFAULT DOMAIN POLICY@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,ADMINISTRATORS@HYBRID.LOCAL
4 3,0.64,DEFAULT DOMAIN POLICY@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,BACKUP OPERATORS@HYBRID.LOCAL
5 4,0.64,TEST111@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,ADMINISTRATORS@HYBRID.LOCAL
6 5,0.64,TEST111@HYBRID.LOCAL,GPLink,HYBRID.LOCAL,Contains,BACKUP OPERATORS@HYBRID.LOCAL
7 |
```

1.DOMAIN USERS@HYBRID.LOCAL = [ADCSERC13] =>
ESC13GROUP@HYBRID.LOCAL

2.DOMAIN COMPUTERS@HYBRID.LOCAL = [ADCSERC13] =>
ESC13GROUP@HYBRID.LOCAL

3.HYBRID.LOCAL = [Contains] => ADMINISTRATORS@HYBRID.LOCAL

4.HYBRID.LOCAL = [Contains] => USERS@HYBRID.LOCAL

5.HYBRID.LOCAL = [Contains] => BACKUP OPERATORS@HYBRID.LOCAL

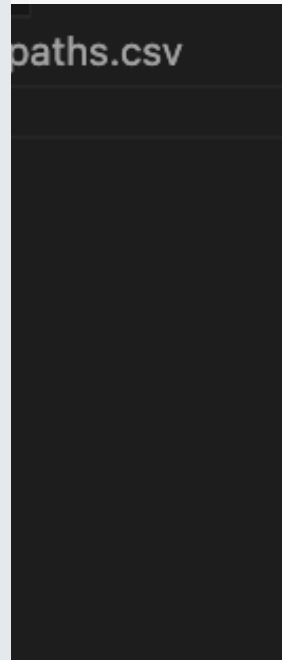
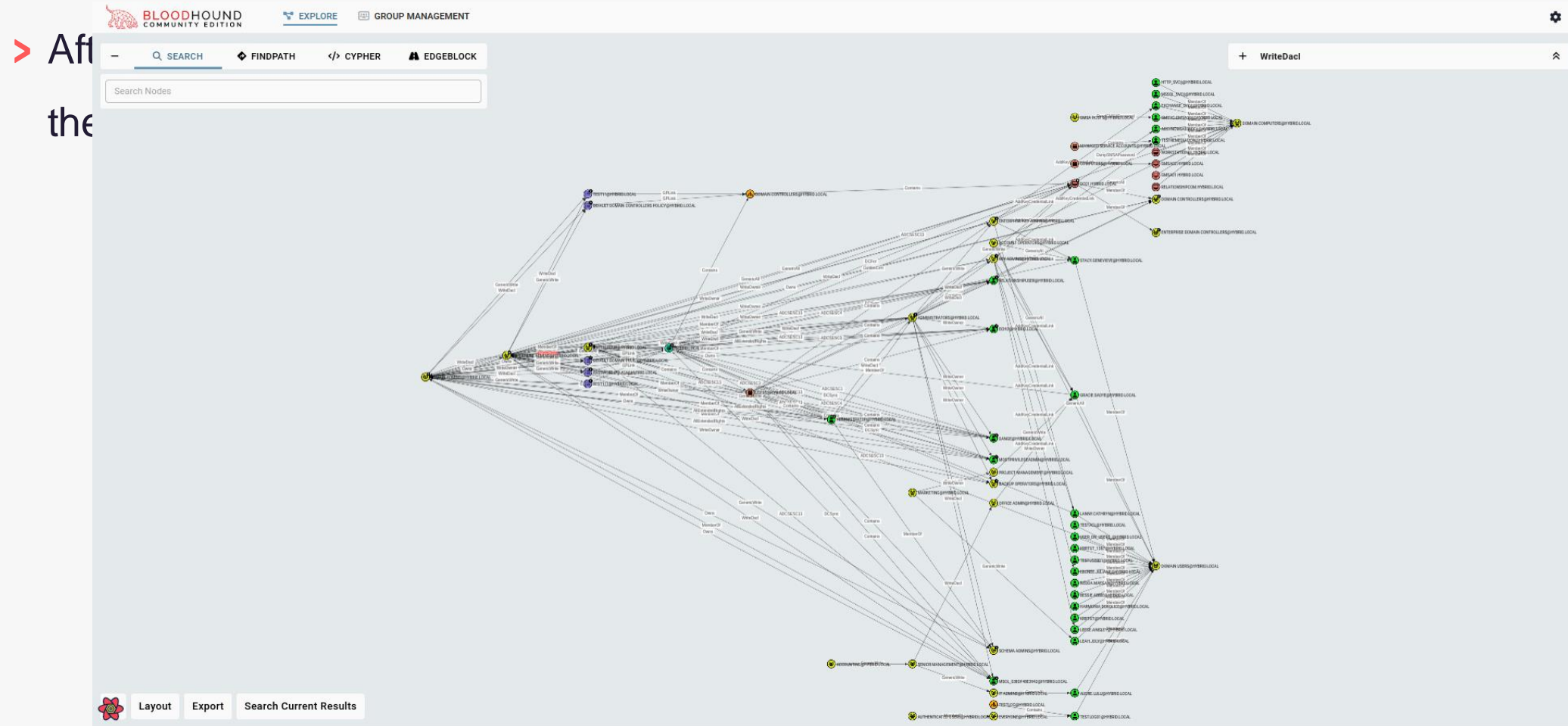
6.DC01.HYBRID.LOCAL = [MemberOf] => DOMAIN
CONTROLLERS@HYBRID.LOCAL

7.DC01.HYBRID.LOCAL = [HasSession] => ECHO@HYBRID.LOCAL

8.DC01.HYBRID.LOCAL = [HasSession] => DANGE@HYBRID.LOCAL

9.DC01.HYBRID.LOCAL = [HasSession] =>
ADMINISTRATOR@HYBRID.LOCAL

After Blocking, Attack Path on AD



Limitations / Drawbacks

- > ~~It's UGLY~~
- > Efficiency will be low under these situations:

- > Large Graph
- > Tree unlike Graph
- > Higher Budget

budget=5	R2000	R4000	ADS10
GREEDY	0.521 (0.04s)	0.376 (0.34s)	0.448 (4.57s)
TDCYCLE	0.480 (0.10s)	0.373 (15366s)	-
IP	0.480 (0.01s)	0.373 (0.06s)	0.409 (0.09s)

- > Need to self-defined edge's **Risk Score**
- > Need to tune back-and-forth few times, because we don't know which edge can be blocked
 - > self-defined **Unblockable Edge**

Future Work

- > Better algorithm
 - > Analysis bigger environment
 - > Let algo can deal with tree unlike AD graph
- > Containerize Sniffer BloodHound, simplify workflow
- > More beautiful UI, don't let algo output so ugly
- > Thinks like attacker and defender, provide more functionality to make everyone's life easier

Thanks! / Questions

Jiun-Ming, Su (Jimmy) | @jimmysured

Shih-Nai, Shin (Maxine)