

SOC Workshop

David Crooks

UKRI STFC

EGI CSIRT/IRIS Security team

david.crooks@stfc.ac.uk



Backup

Zeek Exercises

1: Command line zeek

Run zeek from the command line

- On the zeek container, "zeekctl stop" (it runs as a daemon at startup)
- Gather a pcap file, "tcpdump -w /opt/pocketsoc-ng/data/somedata.pcap"
- Trigger "curl webservice" from client
- Tcpdump -r /opt/pocketsoc/data/somedata.pcap to test
- Use "zeek -r /opt/pocketsoc/data/somedata.pcap -C " to analyse pcap
- Check the logs in the current directory

OUTCOME

This shows that we can capture a set of traffic, and run zeek against it directly to obtain a set of logs. We will see later how we can achieve the same with zeek running as a daemon

2: zeek as a daemon

Run zeek as a daemon again

- Run “zeekctl start”
- cd /opt/zeek/logs/current/
- Trigger “curl webservers” from client
- Check the logs – these should contain similar results!

OUTCOME

We can compare the logs we see with zeek running as a daemon and those from running from the command line: note that the config we use may be different depending on what options are given to the command line

3: using tcpreplay to replay pcaps

Replay the captured pcap into the zeek daemon

- Run “tcpreplay -i eth0 /opt/pocketsoc-ng/data/somedata.pcap”
- cd /opt/zeek/logs/current/
- Check the logs – these should also contain similar results!

OUTCOME

We can use this method to replay prepared packet captures into a “normal” running zeek instance and perform the same analysis as if the traffic were live. This is particularly useful for validation purposes

4: main Zeek configuration

Zeek config

- Main config files are in “/opt/zeek/etc” and “/opt/zeek/share/zeek/site/”
- “networks.cfg node.cfg zeekctl.cfg” and “local.zeek”

Zeek intel config

- Observe the last config block in “local.zeek” following yesterday’s lecture

OUTCOME

We have looked at the key config files for Zeek

5: Zeek alerting: I

5. Check alerting configuration

- Going to use the CERN Mattermost for alerting in a private channel
- Webhook stored in ``/opt/pocketsoc/data/webhook``

5: Zeek alerting: II

- On zeek node, cd /opt/zeek/share/zeek/site/
- In local.zeek, check the following is present

```
@load ./mattermost.zeek

hook Notice::policy(n: Notice::Info)
{
    if ( n$note == Intel::Notice )
    {
        add n$actions[Notice::ACTION_MATTERMOST];
    }
}
```

5: Zeek alerting: III

- We use mattermost.zeek to call a helper script that actually does the webhook call
 - This is inefficient – there is a better way of doing this that will be implemented for the next time I use this
- We can test this now: on the zeek node, run
`/opt/pocketsoc-ng/bin/notifier.sh "Hi there!"`
- We (or at least I 😊) should see an update in the channel

6: Summary so far

- Now we have tested that we can:
 - Gather a packet capture file
 - Run zeek from the cli
 - Check the Zeek logs for recent activity
 - Use the helper script to raise a notification independently of Zeek
- Now let's do some alerting from a detection!
 - First: MISP

MISP Exercises

First steps: MISP

- Username: admin@admin.test
- Password: \$password



Login

Email

Password

Login

First steps: MISP

The screenshot displays the MISP web interface. At the top, a dark navigation bar contains links for Home, Event Actions, Dashboard, Galaxies, Input Filters, Global Actions, Sync Actions, Administration, Logs, and Alerts. Below this, a green notification bar states "Event deleted." with a close button. On the left, a sidebar menu lists various actions: List Events (highlighted), Add Event, Import from..., REST client, List Attributes, Search Attributes, View Proposals, Events with proposals, View delegation requests, Export, and Automation. The main content area is titled "Events" and features a search bar with a magnifying glass icon, buttons for "My Events" and "Org Events", and a dropdown menu. Below the search bar is a table header with columns: Published, Creator org, Owner org, ID, Clusters, Tags, #Attr., #Corr., #Sightings, Creator user, Date, Info, and I. The table currently shows 0 records. At the bottom, a pagination bar indicates "Page 1 of 1, showing 0 records out of 0 total, starting on record 0, ending on 0".

Home Event Actions Dashboard Galaxies Input Filters Global Actions Sync Actions Administration Logs Alerts

Event deleted. x

List Events
Add Event
Import from...
REST client

List Attributes
Search Attributes

View Proposals
Events with proposals
View delegation requests

Export
Automation

Events

« previous next »

🔍 My Events Org Events 📄

Enter value to search Event info Filter

<input type="checkbox"/>	Published	Creator org	Owner org	ID	Clusters	Tags	#Attr.	#Corr.	#Sightings	Creator user	Date	Info	I
--------------------------	-----------	-------------	-----------	----	----------	------	--------	--------	------------	--------------	------	------	---

Page 1 of 1, showing 0 records out of 0 total, starting on record 0, ending on 0

« previous next »

MISP exercises

- Log into your MISP instance
- [https://scsc-2022-\[01-39\].cern.ch](https://scsc-2022-[01-39].cern.ch)
 - admin@admin.test + \$password
- We want to create an event with the webserver as `ip_dst`
 - And a filehash too if we want
- Start with an event

MISP exercises

- Click “add an event” and we’ll work through the steps
- We want to add a “network” object
- Ip_dst= the webserver IP (should be 172.18.0.2)
 - On the client container, you should be able to `dig webserver` to confirm
- Make sure that To IDS is clicked
- Publish (no email)

MISP exercises

- On the events page, check that you have one event!
- Next, we want to download this to Zeek
- In MISP, go to Global Actions -> My Profile and copy your authkey
- In Zeek, ``export authkey=$AUTHKEY`` and ``/opt/pocketsoc-ng/bin/pull_misp.sh``
 - Should see a list of the intel in `/opt/zeek/feeds/intel.txt`

MISP exercises

- Now, trigger the “bad” activity! Either:
 - On the client node, curl the webserver one more time OR
 - On the zeek node, we can replay the pcap file into zeek again
- `tcpreplay -i eth0 $pcapfile`
- Either of these should
 - Create a new entry in `/opt/zeek/logs/current/intel.log`
 - Raise an alert in mattermost

Building a MISP network

- Use `scsc-2022-00.cern.ch` as our central instance
- I have prepopulated it with “sync users” that will let you sync your instance to mine
- User: [scsc@scsc-2022-\[01-39\].cern.ch](mailto:scsc@scsc-2022-[01-39].cern.ch)
- Password: the same password
- You should now see the `scsc-2022-00.cern.ch` events

Building a MISP network

- **In the -00 instance**, again go to Global actions -> My profile and copy the **different** authkey
- **On your instance** go to “Sync actions -> List Servers” and click on “New Servers”

Building a MISP network

Base URL: <https://scsc-2022-00.cern.ch>

Instance Name: Central

Organisation Type: Local

Local organization type: PocketSOC

Authkey: the key you copied from the **-00** instance

Enabled synchronisation methods: Pull

Allow self signed certificates (unsecure): check

(This shouldn't be needed, this is on my snaglist)

-> **Submit**

Building a MISP network

- Check the server list (or click list servers)
- **RUN** Connection test
- If this fails, we can look at it
- On the far right side of that row, click the down arrow (hover text: pull all to pull all events)
- That's it!
- You can also set up regular synching **which will only pull deltas**

First steps: OpenSearch Dashboards

- Username: admin
- Password: \$password



Please login to OpenSearch Dashboards

If you have forgotten your username or password, please ask your system administrator



Username



Password

Log In

First steps: OpenSearch Dashboards

Welcome to OpenSearch
Dashboards



Start by adding your data

Add data to your cluster from any source, then analyze and visualize it in real time. Use our solutions to add search anywhere, observe your ecosystem, and protect against security threats.

Add data

Explore on my own

First steps: OpenSearch Dashboards

✕

Select your tenant

Tenants are useful for safely sharing your work with other OpenSearch Dashboards users. You can switch your tenant anytime by clicking the user avatar on top right.

Global
The global tenant is shared between every OpenSearch Dashboards user.

Private
The private tenant is exclusive to each user and can't be shared. You might use the private tenant for exploratory work.

Choose from custom

Cancel

Confirm

First steps: OpenSearch Dashboards

You have data in
OpenSearch.

Now, create an index
pattern.

OpenSearch Dashboards requires an index pattern to identify which indices you want to explore. An index pattern can point to a specific index, for example, your log data from yesterday, or all indices that contain your log data.



[+ Create index pattern](#)

First steps: OpenSearch Dashboards

Create index pattern

An index pattern can match a single source, for example,

`filebeat-4-3-22` , or **multiple** data sources, `filebeat-*` .

[Read documentation](#) 

Step 1 of 2: Define an index pattern

Index pattern name

Next step >

Use an asterisk (*) to match multiple indices. Spaces and the characters `\,/, ?, ", <, >, |` are not allowed.

 Include system and hidden indices

First steps: OpenSearch Dashboards

Create index pattern

An index pattern can match a single source, for example, `filebeat-4-3-22` , or **multiple** data sources, `filebeat-*` .

[Read documentation](#) 

Step 2 of 2: Configure settings

Specify settings for your **opensearch-logstash-zeek*** index pattern.

Select a primary time field for use with the global time filter.

Time field

Refresh

@timestamp

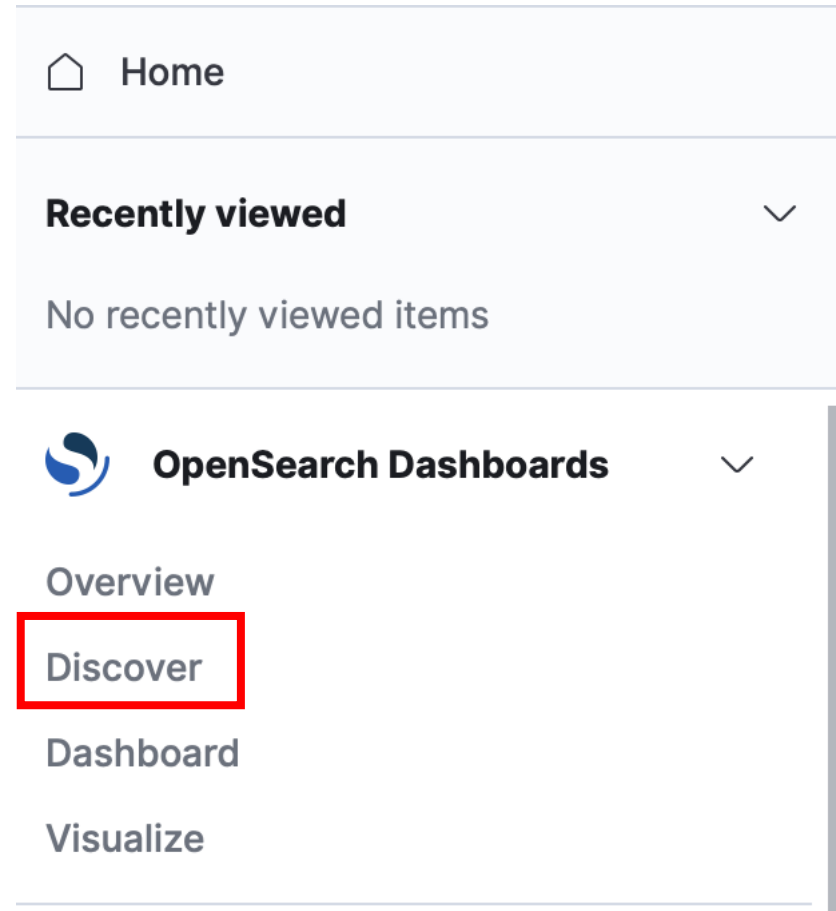


> [Show advanced settings](#)

< [Back](#)

[Create index pattern](#)

First steps: OpenSearch Dashboards



First steps: OpenSearch Dashboards

