Streaming data to track vulnerabilities and phishing campaigns

MCH2022

SCHUBERG PHILIS

WHOAMI

- Joey Dreijer
- Security Engineer @ Schuberg Philis
- Python hobbyist ☺

Twitter: joeydreijer

Mail: jdreijer@schubergphilis.com / jdreijer@d3vzer0.com

What we'll do

- Configure an RSS aggregator to track feeds related to security and vulnerabilities
- Use the twitter API to monitor CVE/vulnerability related tweets
- Set up tooling to perform NLP on text and extract products/companies/CVE id etc
- Store and visualise our results with elasticsearch + kibana!

- **Bonus**: We can use the same stack for other purposes... may organise another workshop to track phishing campaigns organised later ☺

Pre-Requirements

- **Accounts**
  - A twitter account with API key

- **Tools/Apps**
  - OpenSSH (for tunnelling)
  - Git
  - Python 3.8+
  - pip / pipenv / poetry

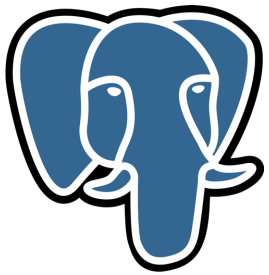
Optional: Docker but not necessarily needed

Components

- **Kafka:** Distributed event streaming platform. Ie. sending messages back and forth via topics
- **Elasticsearch:** Search and analytics engine
- **Kibana:** Front-end / visualisation platform for Elastic
- **Miniflux:** Open source RSS aggregrator
- **Postgres:** Open source SQL database
- **Faust:** Stream processing library for Python used with Kafka
- **Spacy:** Natural language processing toolkit

ElasticSearch

Kafka



Postgres

Kibana

Faust

Some of the spaCy features

Feature	Description
Tokenization	Segmenting text into words, punctuations marks etc.
Lemmatization	Assigning the base forms of words. For example, the lemma of “was” is “be”, and the lemma of “rats” is “rat”.
Named Entity Recognition (NER)	Labelling named “real-world” objects, like persons, companies or locations.
Rule-based Matching	Finding sequences of tokens based on their texts and linguistic annotations, similar to regular expressions.
Text Classification	Assigning categories or labels to a whole document, or parts of a document.

https://spacy.io/usage/spacy-101

Simplified process

Original: A critical issue has been discovered in GitLab version 1.2.3

After tokenization

0	1	2	3	4	5	6	7	8	9
A	critical	issue	has	been	discovered	in	GitLab	version	1.2.3

Lemmatization

0	1	2	3	4	5	6	7	8	9
a	critical	issue	**have**	**be**	**discover**	in	Gitlab	version	1.2.3
Tags	**DET**	**ADJ**	**NOUN**	**AUX**	**AUX**	**VERB**	**ADP**	**PROPN**	**NOUN**	**NUM**
Entities							**ORG**		

displaCy

Example hosted @ https://github.com/d3vzer0/mch2022-workshop-nlp inside the **examples** directory

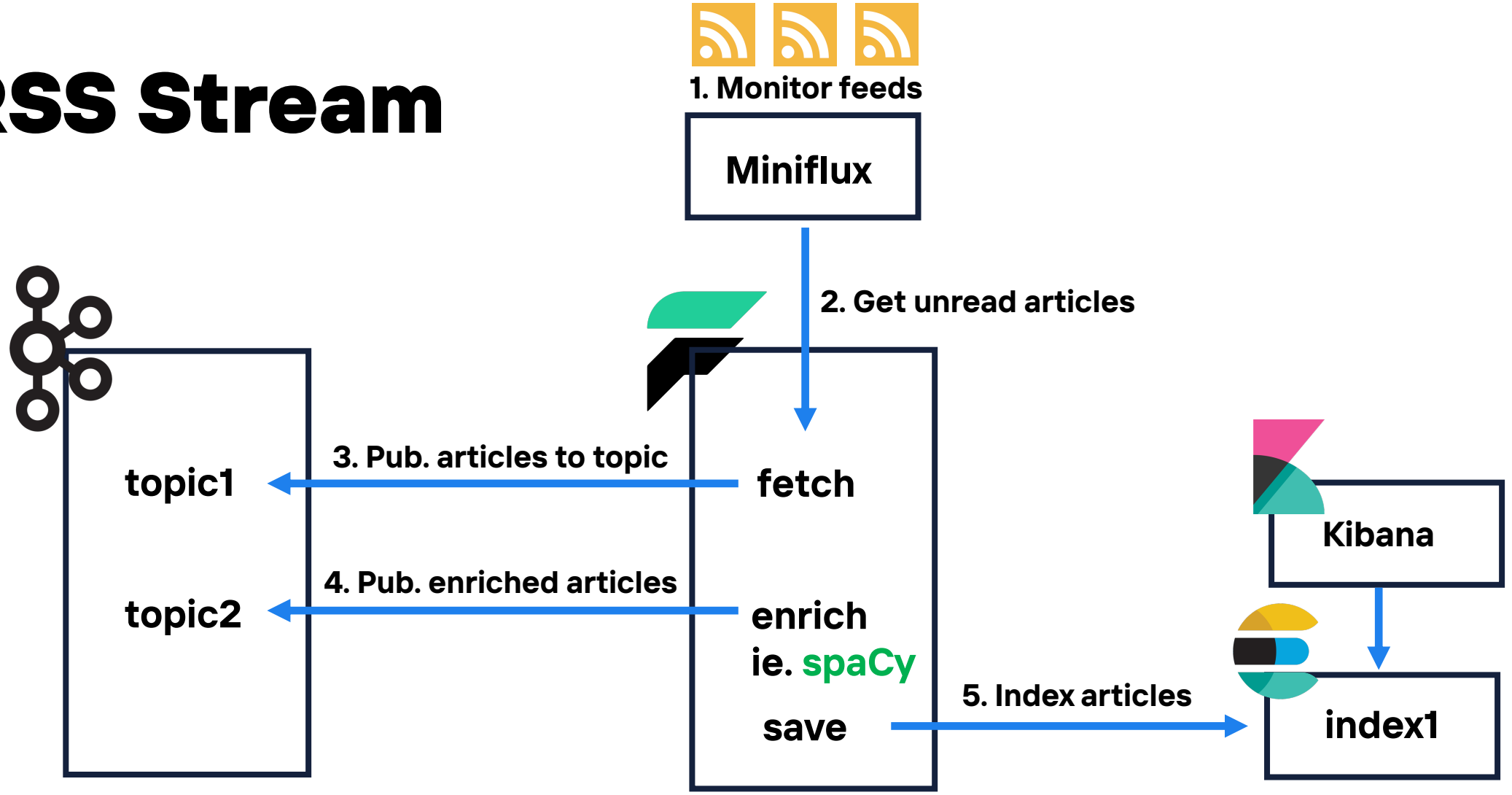
You can play around with the visualisation options

- style=**deb** (syntactic dependencies)
- style=**ent** (parsed entities)

Source and more info
https://spacy.io/usage/visualizers

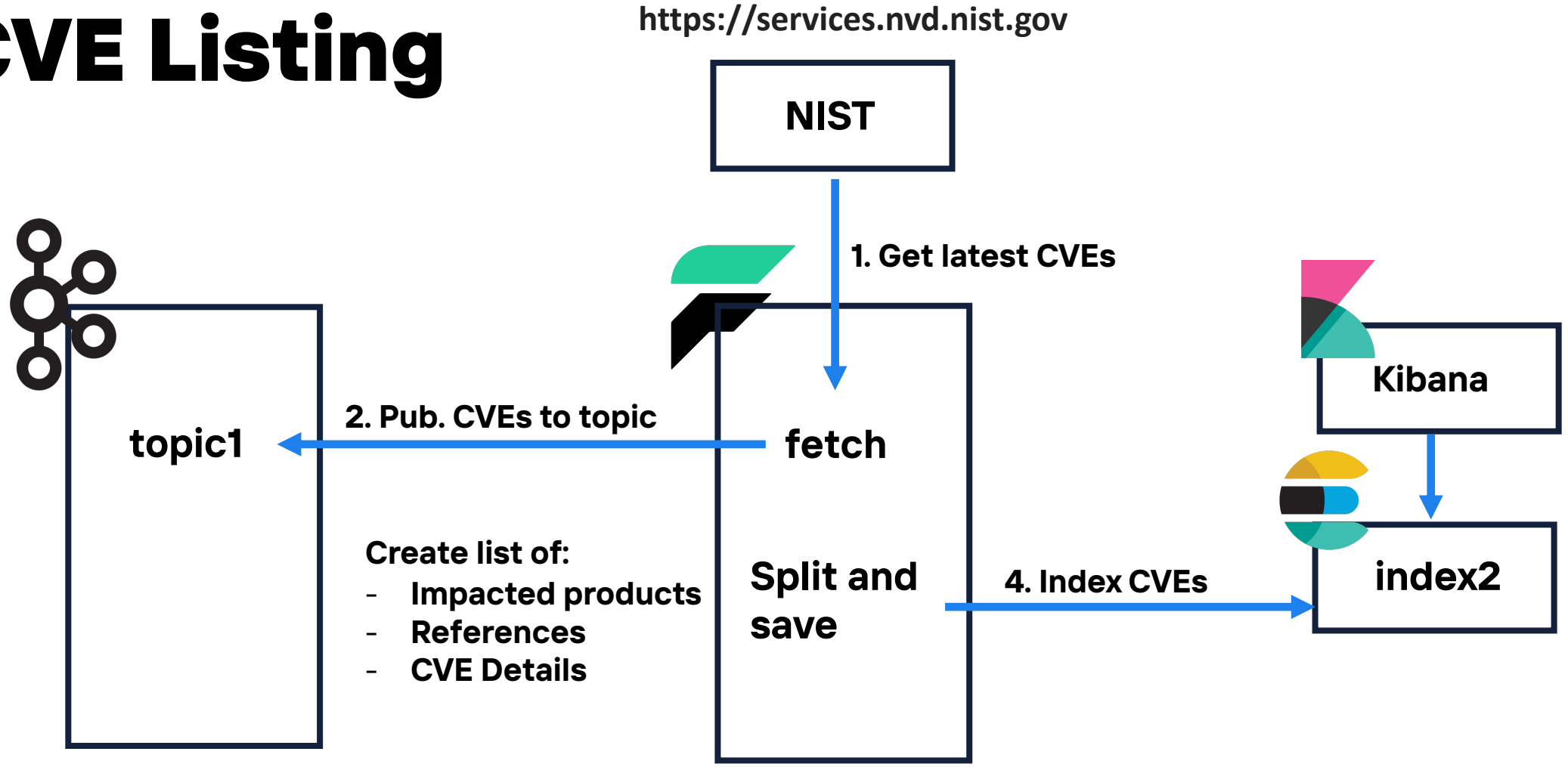
How the components connect

RSS Stream



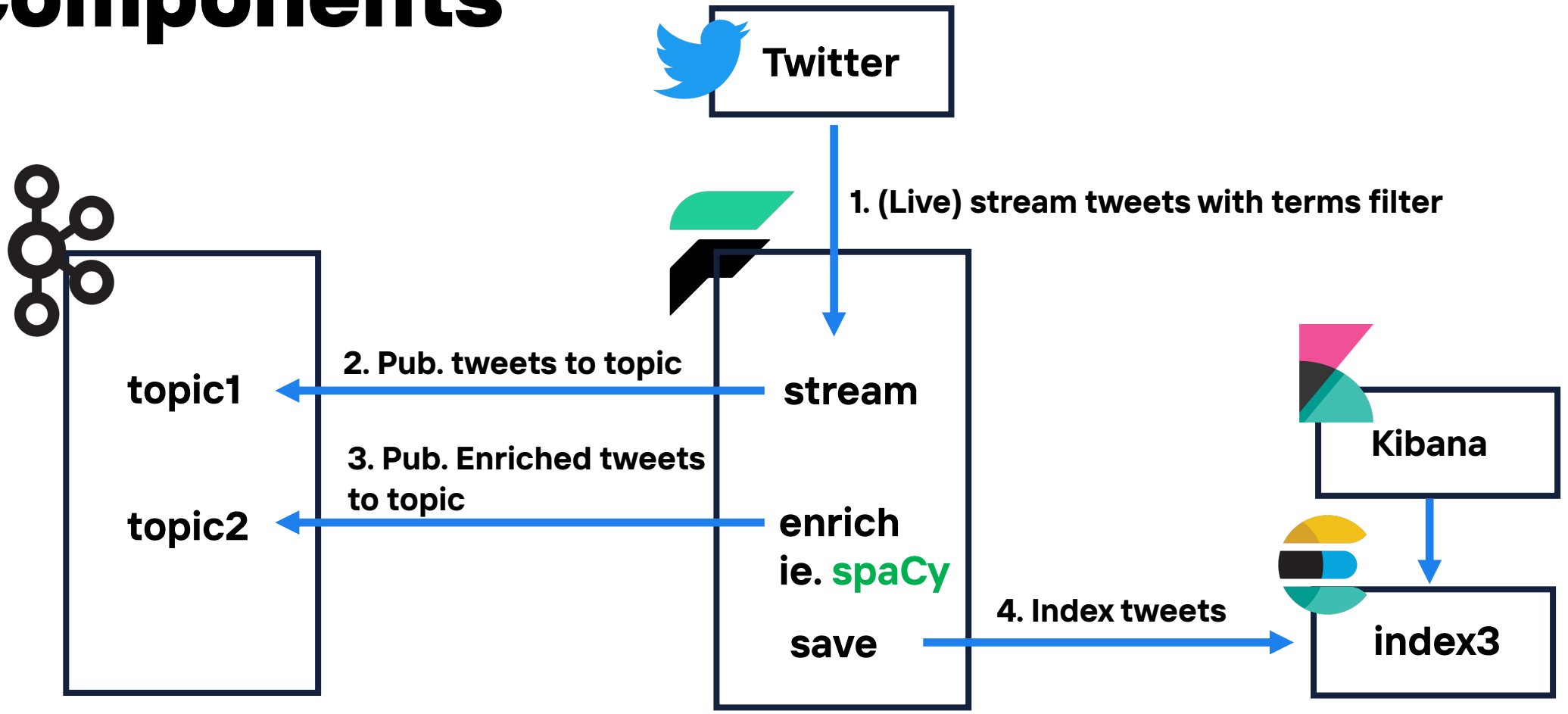
How the components connect

CVE Listing



How the components connect

Components

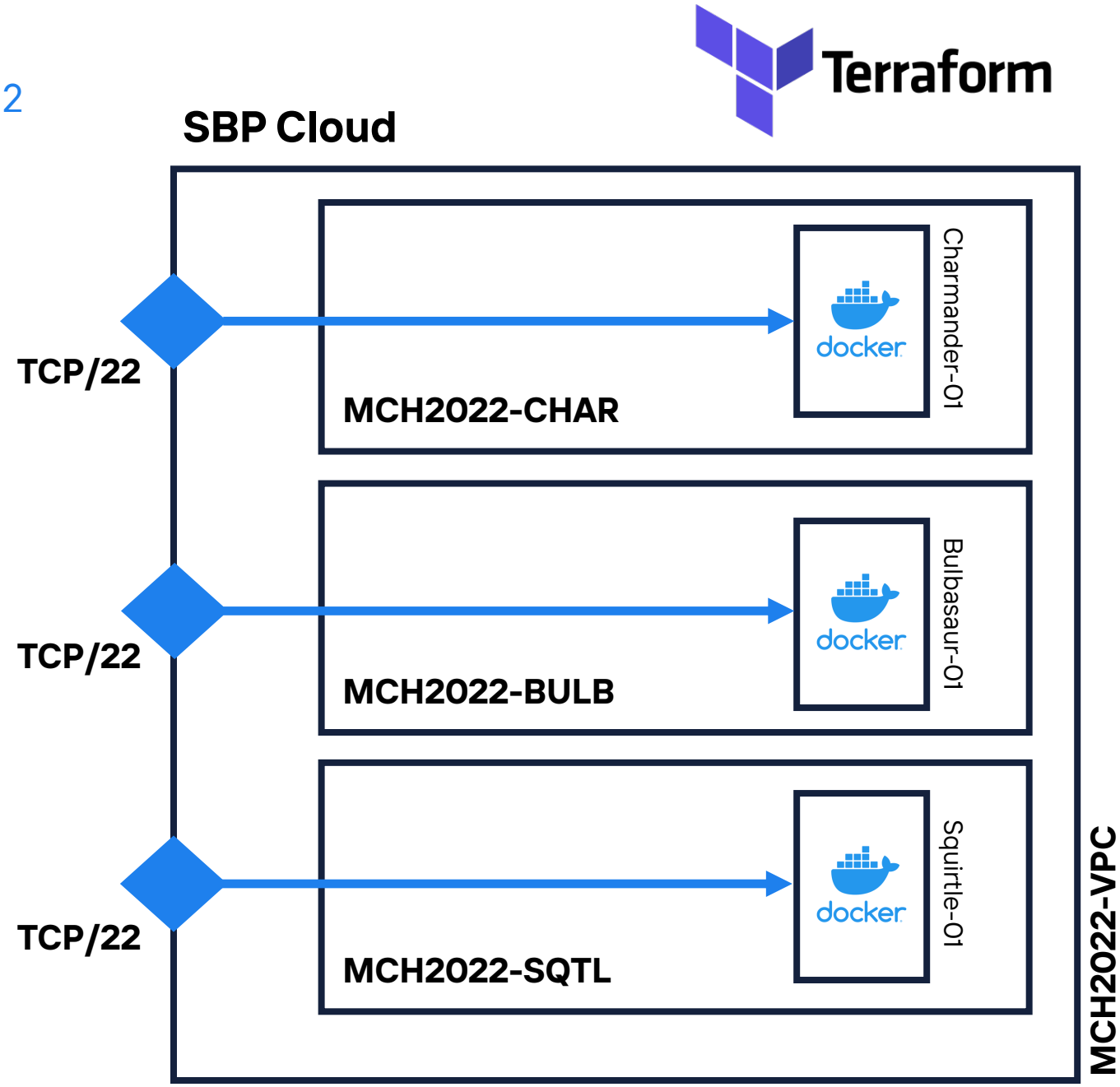


Infra preprovisioned for MCH2022

Infrastructure

- VMS pre-provisioned with containers
  - ElasticSearch
  - Kibana
  - Kafka
  - Postgres
  - Miniflux

- Access via SSH tunnel ☺



Terraform + Ansible code @ https://github.com/d3vzer0/mch2022-workshop-cloud

The most important part

Choose your starter pokemon






Access to infra

Pokemon	External IP
Charmander (mch2022-charmander-01)	95.142.96.38
Bulbasaur (mch2022-bulbasaur-01)	195.43.158.39
Squirtle (mch2022-squirtle-01)	195.43.158.44
Cyndaquil (mch2022-cyndaquil-01)	195.43.158.48
Chikorita (mch2022-chikorita-01)	195.43.158.74
Totodile (mch2022-totodile-01)	195.43.158.102

Access to infra

Create a new public/private keypair (or use an existing one)

```
ssh-keygen –t ed25519
```

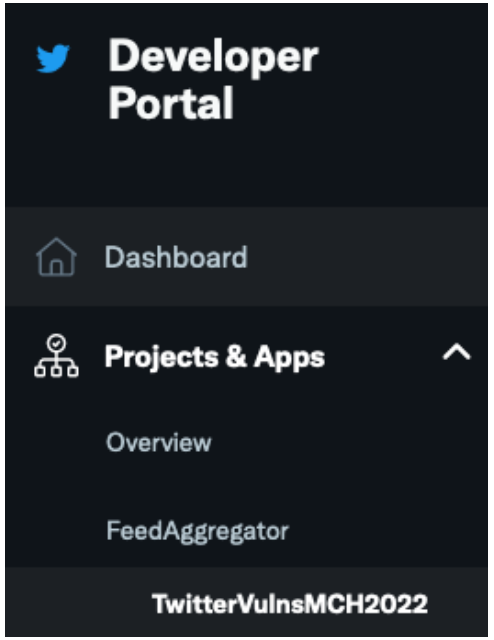
Use $channel to share your public key, saved in the previously selected directory and ends with **.pub**. Double check that you **don’t send your private key (this happened before)**!

Run an SSH tunnel and forward the appropriate ports needed to run the project

```
ssh -L 9200:localhost:9200 -L 9092:localhost:9092 -L 5601:localhost:5601 -L
8080:localhost:80 <your_pokemon>@<your_ip>
```

Create a Twitter API key

- Navigate to https://developer.twitter.com and login with your Twitter account
- Navigate to the **Projects and Apps** page on the left
- Click on **Add App**
  - Select **Development** for the environment and press next
  - Give your new app a name (ex. TwitterVulnsMCH2022) and press next
  - Save/copy the **Bearer Token**, this should remain a ***secret***!



Add some feeds to Miniflux

Cyber Security news

- Oday Initiative: https://www.zerodayinitiative.com/blog/?format=rss
- NCSC (English): https://feeds.english.ncsc.nl/news.rss
- HackerNews: http://feeds.feedburner.com/TheHackersNews?format=xml
- Schneier on security: https://www.schneier.com/blog/atom.xml
- Krebs: https://krebsonsecurity.com/feed/
- RedHat: https://www.redhat.com/en/rss/blog/channel/security
- MS: https://msrc-blog.microsoft.com/feed
- LinuxSec: https://linuxsecurity.com/linuxsecurity_articles.xml
- Bleeping Computers: https://www.bleepingcomputer.com/feed/
- Cyberscoop: https://www.cyberscoop.com/feed
- Packetstorm: https://rss.packetstormsecurity.com/files/
- ThreatPost: https://threatpost.com/feed

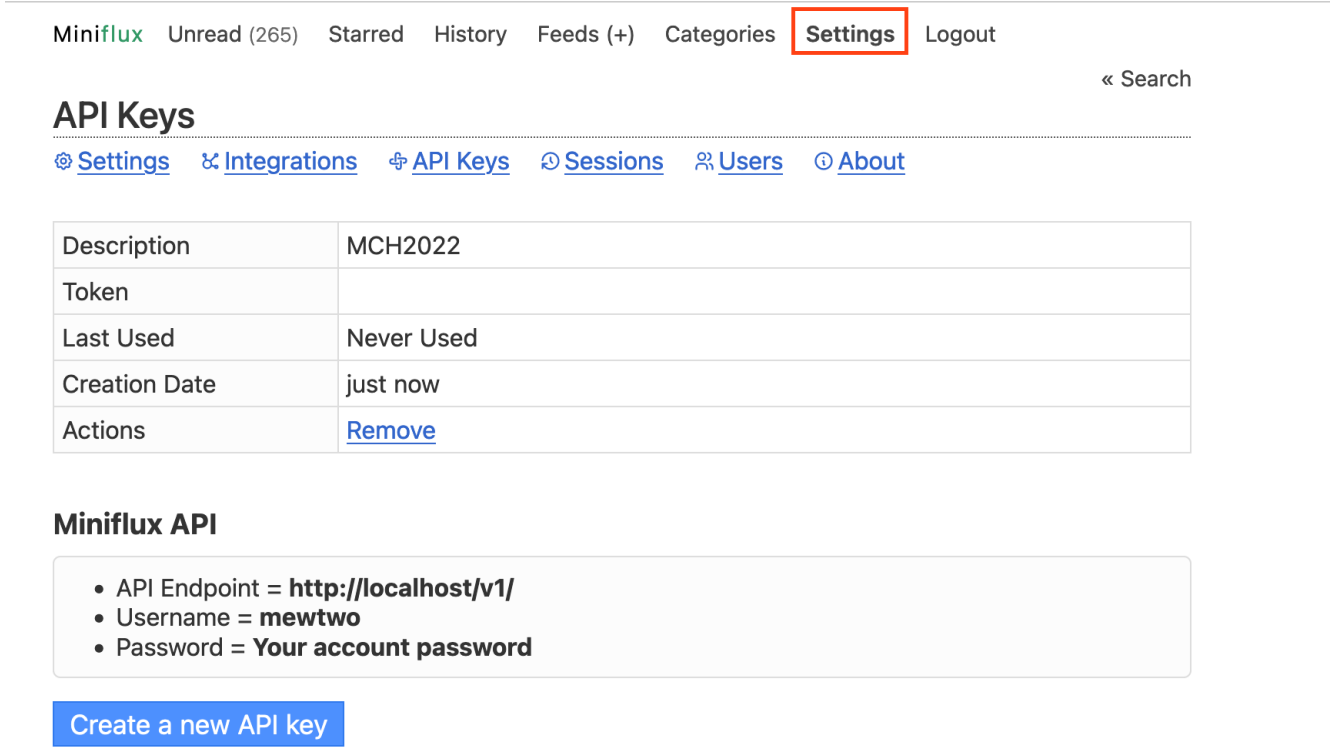
Add some feeds to Miniflux

Cyber Security advisories

- 0day Upcoming: https://www.zerodayinitiative.com/rss/upcoming/
- 0day published: https://www.zerodayinitiative.com/rss/published/
- CISA: https://www.cisa.gov/uscert/ncas/current-activity.xml
- CIS: https://www.cisecurity.org/feed/advisories
- LinuxSec: https://linuxsecurity.com/linuxsecurity_advisories.xml
- Oracle: https://www.oracle.com/ocom/groups/public/@otn/documents/webcontent/rss-otn-sec.xml

Create miniflux API key

- After setting up the tunnel navigate toL **http://localhost:8080/unread**
- Navigate to **Settings -> API Keys** and select **Create a new API Key**



Download and install Python

MacOS

- Via the official installer: https://www.python.org/
- Homebrew: `brew install python3.8`

Linux

- Ubuntu: `sudo apt-get install python3 python3-pip`
- CentOS/Rhel/Rocky: `dnf install python3 python3-pip`

Windows

- Via the official installer: https://www.python.org/
- WSL2: `sudo apt-get install python3 python3-pip`

Download and install Git (cli)

MacOS

- Homebrew: `brew install git`

Linux

- Ubuntu: `sudo apt-get install git`
- CentOS/Rhel/Rocky: `dnf install git`

Windows

- WSL2: `sudo apt-get install git`

Download and install virtual envs

Install pipx for easier isolation of Python applications

```
pip install pipx
```

Use pipx to install virtual environment (pipenv or poetry) of your choosing

```
pipx install poetry
pipx install pipenv
```

pipenv: https://pipenv.pypa.io/en/latest/
poetry: https://python-poetry.org/
pipx: https://github.com/pypa/pipx

Getting the code

Clone the code repository

`git clone https://github.com/d3vzer0/mch2022-workshop-streaming`

When inside the code directory:

- **poetry**: `poetry shell && poetry install`
- **pipenv**: `pipenv shell && pipenv install`

Environment variables (typically used with Docker)

Set the configuration

Set which streaming app to run (set either nvd, twitter or miniflux)

```
export STREAM_TYPE=<streaming.nvd (default) || streaming.twitter ||
streaming.miniflux>
```

Set your Twitter API Bearer / token

```
export  TWITTER_BEARER=<your_twitter_bearer_token>
```

Set your Miniflux API key:

export MINIFLUX_KEY=<your_api_key>

Set the elasticsearch URL which contains your random password

export ELASTIC_URI=https://elastic:<your_elastic_pass>@localhost:9200

What I'll share with you

Set the configuration

Username/password for the RSS aggregator

Username/password to your Elasticsearch cluster + Kibana

The CA needed to trust your connection with ElasticSearch
  Copy this file to the root of the project directory (ie. http_ca.crt)

Environment variables (typically used with Docker)

Run and $$ profit

You may need to run this command twice since the Kafka topics are not pre-created:

```
faust -A streaming.main worker
```

When you configured the twitter stream, you'll need to start a second process that watches for new tweets with your own filters, such as:

```
faust -A streaming.main get-tweets --filters cve,vulnerability
```

Navigate to http://localhost:5601 and login with your elastic credentials. Click around to see what data is available ☺


Always have been
Wait, we're data engineers now?
imgflip.com

Sources and refs

- TextBlob: https://textblob.readthedocs.io/en/dev/
- spaCy: https://spacy.io/
- Faust: https://faust.readthedocs.io/en/latest/

Grts.

SCHUBERG PHILIS