HoneySwarm

Release 0.1

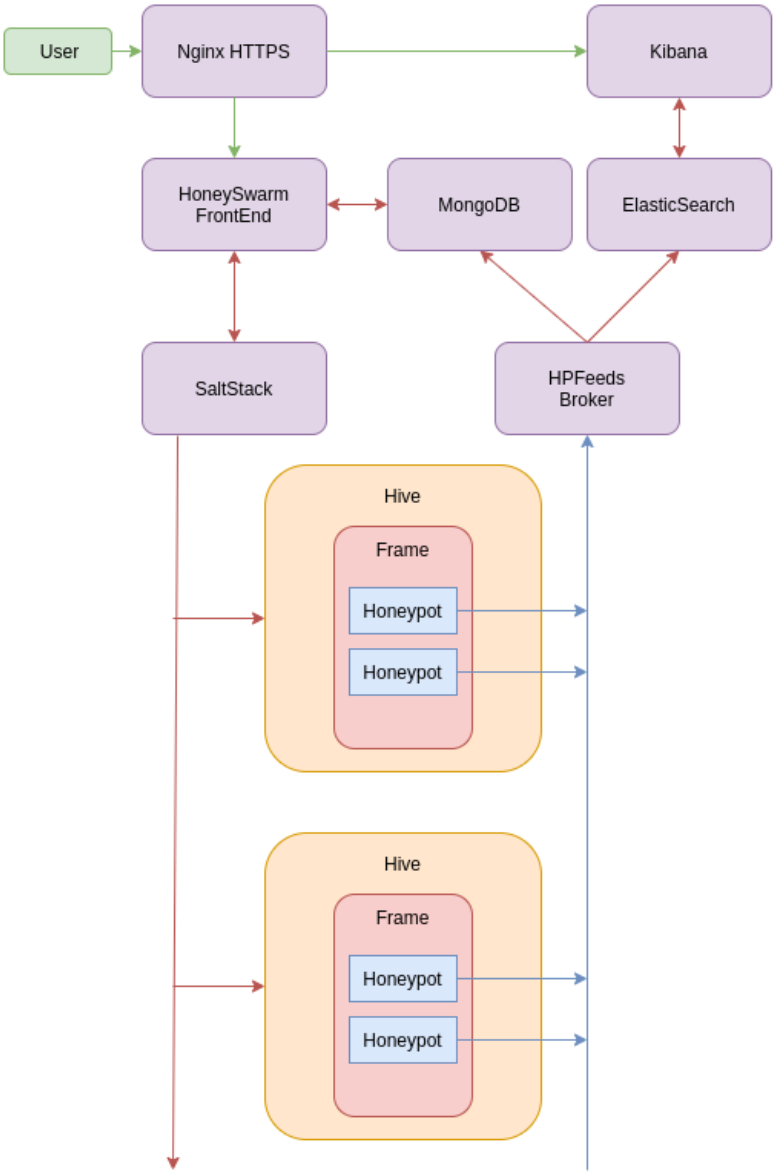
Oct 20, 2020

Contents:

CHAPTER 1

About

Honeyswarm is a Honeypot Orchestration and monitoring platform designed to make honeypots easy again


User
Nginx HTTPS
Kibana
HoneySwarm FrontEnd
MongoDB
ElasticSearch
SaltStack
HPFeeds Broker
Hive
Frame
Honeypot
Honeypot
Hive
Frame
Honeypot
Honeypot

CHAPTER 2

Installation

The officially supported installation process is to use the docker-compose that is shipped with the repo. Installing in a method other than docker-compose is left as an exercise to the user. Read the compose and the docker files for each container should give you a headstart.

2.1 Docker and Compose

Install Docker and docker-compose using the offical guides at https://docs.docker.com/get-docker/

2.2 HoneySwarm

If you want to run the latest stable release use the following docker-compose file and pin a release tag

```
version: '3.7'
services:
honeyswarm:
   image: honeyswarm/honeyswarm
   container_name: honeyswarm
   env_file:
      - honeyswarm.env
   ports:
      - "8080:8080"
   networks:
      honeynet:
      ipv4_address: 10.1.0.101
   volumes:
      - "honeyswarmStates:/opt/honeystates/salt:rw"
   depends_on:
      - mongoserver
      - saltmaster
mongoserver:
```

(continues on next page)

(continued from previous page)

```
    image: mongo:latest
    container_name: honeyswarm_db
    env_file:
        - honeyswarm.env
    ports:
        - '27017:27017'
    networks:
        honeynet:
        ipv4_address: 10.1.0.102
    volumes:
        - "honeyswarmDB:/data/db"
saltmaster:
    image: "saltstack/salt:latest"
    container_name: honeyswarm_saltstack
    env_file:
        - honeyswarm.env
    ports:
        - "8000:8000"
        - "4505:4505"
        - "4506:4506"
    networks:
        honeynet:
        ipv4_address: 10.1.0.103
    volumes:
        - "honeyswarmPKI:/etc/salt/pki:rw"
        - "honeyswarmStates:/srv/salt:rw"
hpfeeds-broker:
    image: honeyswarm/honeyswarm_broker
    container_name: honeyswarm_broker
    ports:
    - "0.0.0.0:10000:10000"
    networks:
        honeynet:
        ipv4_address: 10.1.0.104
    env_file:
        - honeyswarm.env
    depends_on:
        - mongoserver

networks:
honeynet:
    driver: bridge
    ipam:
        driver: default
        config:
        - subnet: 10.1.0.0/24

volumes:
honeyswarmDB:
honeyswarmPKI:
honeyswarmStates:
```

If you prefer a development version then `git clone git@github.com:honeyswarm/honeyswarm.git`

Create a docker-compose.yml file on the host you want to operate as the HoneySwarm Controller. Add the contents of the compose file from above.

Create a honeyswarm.env file in the same directory as the docker-compose and add the following content. The latest

version can be found in the git repo.

```
# Salt Master details
SALT_USERNAME=salt
SALT_SHARED_SECRET=supersecretsaltstackmasterstring
SALT_HOST=https://127.0.0.1:8000

# Flask Shell
FLASK_APP=honeyswarm.py
PYTHONPATH=/opt/
SESSION_SECRET=MuhktUNBDthagZkY477ZWcXfM41x5dRuao8eEXZK

# Mongo Details
MONGODB_HOST=127.0.0.1
MONGODB_PORT=27017
MONGODB_USERNAME=admin
MONGODB_PASSWORD=admin
MONGODB_AUTH_SOURCE=admin
MONGODB_DATABASE=honeyswarm
MONGO_INITDB_ROOT_USERNAME=admin
MONGO_INITDB_ROOT_PASSWORD=admin

# HPFeeds
WAIT_HOSTS=127.0.0.1:27017
```

2.3 Configuration

Copy `honeyswarm_template.env` to `honeyswarm.env` and change the default passwords and tokens as per the list below.

- SALT_SHARED_SECRET
- MONGODB_USERNAME and MONGO_INITDB_ROOT_USERNAME
- MONGODB_PASSWORD and MONGO_INITDB_ROOT_PASSWORD

Please leave all the HOST names and ports as they are pre configured.

If you wish to change the external HTTP port from 8080 to something of your choice edit the docker-compose.yml file.

Once you have made your changes you will need to start the application and complete the first time setup.

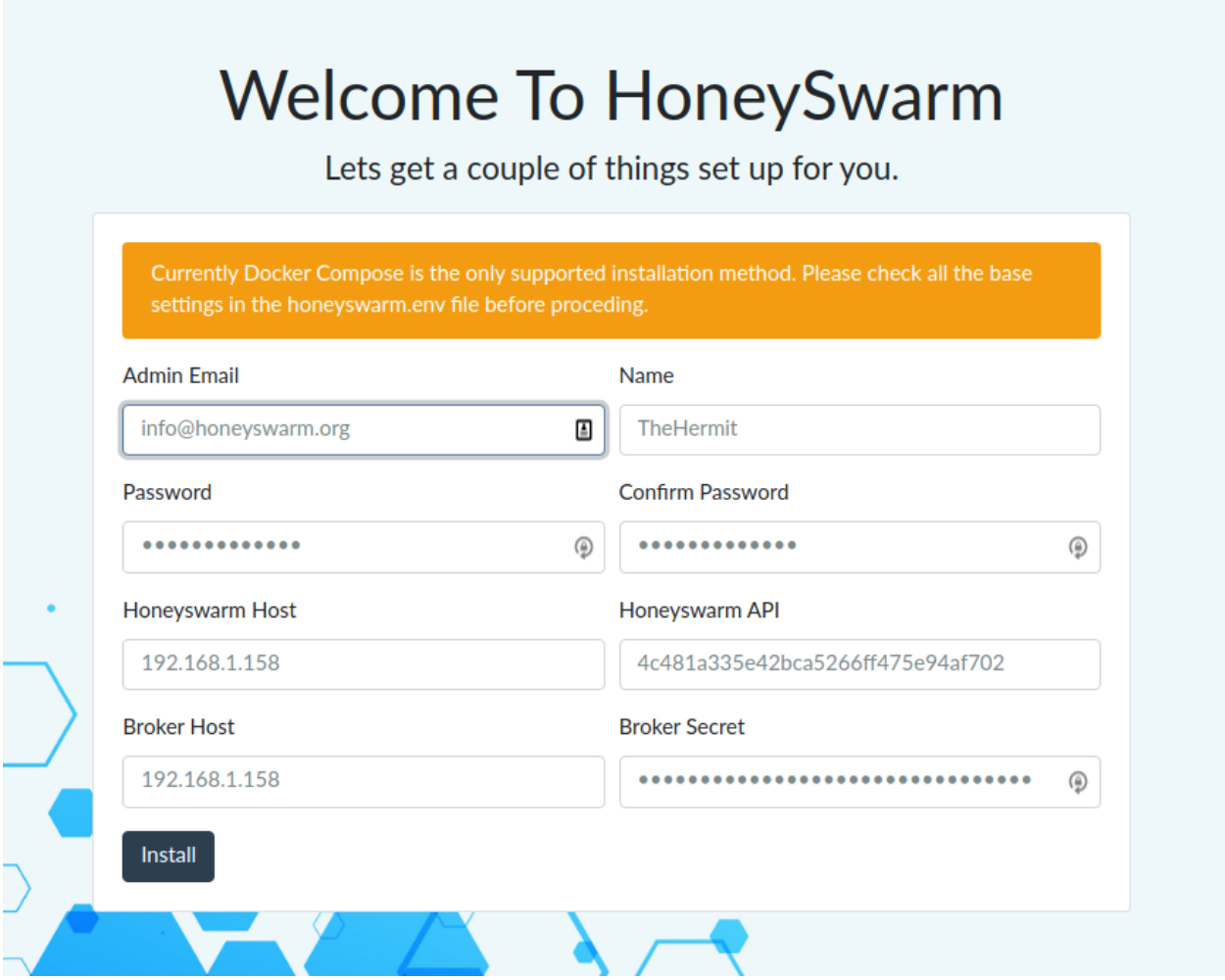
2.4 First Time Setup

Start Honeyswarm using the command `docker-compose up`. Refer to the Starting section for more details.

The first start will download all the required docker images and configure them as per the .env file.

Once you start your HoneySwarm instance for the first time you will need to run the initial installation. To start the install visit http://HONEYSWARMIP:8080/install

You should be presented with an installation form.



Fill all the required fields.

- Honeyswarm Host: This should be set to an IP address that your honeypot hosts (Hives) can access.
- Honeyswarm API: This will autofil with a randomly generated API Key, but you can change it. This is the key that will be used to run the initial Hive setup
- BrokerHost: This is the HPFeeds Broker IP, if your using the default installation this should be set to match the HoneySwarm Host fields
- Broker Secret: This is the main auth key that will be used to Subscribe to ALL incoming honeypot Events.

As part of the installation HoneySwarm will download and install all the available frames and honeypots.

Once the installation has completed you will need to stop and restart the docker-compose to restart all the services with the new configuration.

CHAPTER 3

Starting

All commands must be executed from the honeyswarm directory.

To start the application in the background enter `docker-compose up -d` in a terminal. To start the application in the forground with visible logging enter `docker-compose up` in a terminal.

CHAPTER 4

Stopping

All commands must be executed from the honeyswarm directory.

```
docker-compose down
```

CHAPTER 5

Backup / Restore

5.1 Volumes

To maintain persistance of data HoneySwarm uses docker volumes. As long as you do not prune or destory these volumes you can start, stop and upgrade your HoneySwarm containers without losing data.

5.2 Backup

For details on backing up or restoring docker volumes please refer to the docker documentation.

CHAPTER 6

Update

If your using docker-compose you can update your installation by following these steps.

Note This will take your hpfeeds broker offline for a few minutes and you will not store any incoming events.

- `cd` to the honeyswarm directory
- `docker-compose pull`
- `docker-compose up --force-recreate --build -d`

CHAPTER 7

Quickstart

CHAPTER 8

Hives

8.1 What is a Hive

A Hive is a host device that is capable of running one or more honeypots. A hive can be a virtual machine a physical macine or an Amazon instance.

There are a few requirements that a hive must meet in order to deploy and run honeypots.

- Can connect to the internet.
- Can connect to the HoneySwarm controller
- Able to install and run docker containers
- Supports python >=3.7

8.2 Create a Hive

The first step is to initialise the Hive Host. We do this by installing a Salt Minion with some custom parameters. You can use the follwing examples to init a Linux or Windows Host replacing the IP address.

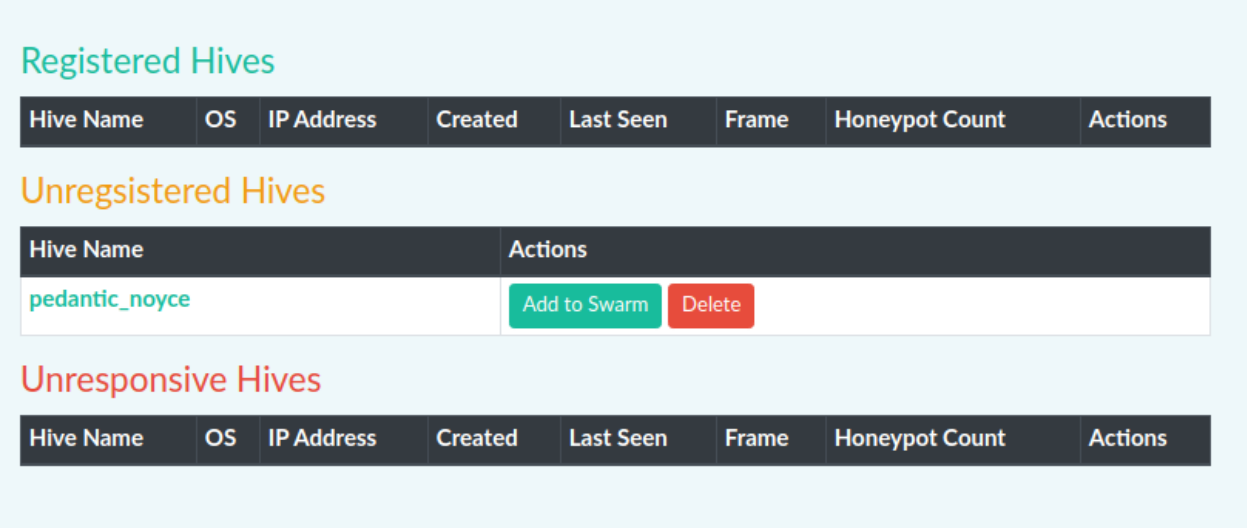
```
curl -H "Authorization:  APITOKEN" http://HONEYSWARMIP:8080/hives/api/hive/
register/linux | sudo sh
```

These command lines are also displayed in HoneySwarm on the hives page with your current API token.

This should install the base and register the hive with HoneySwarm. The next step is to approve the registration

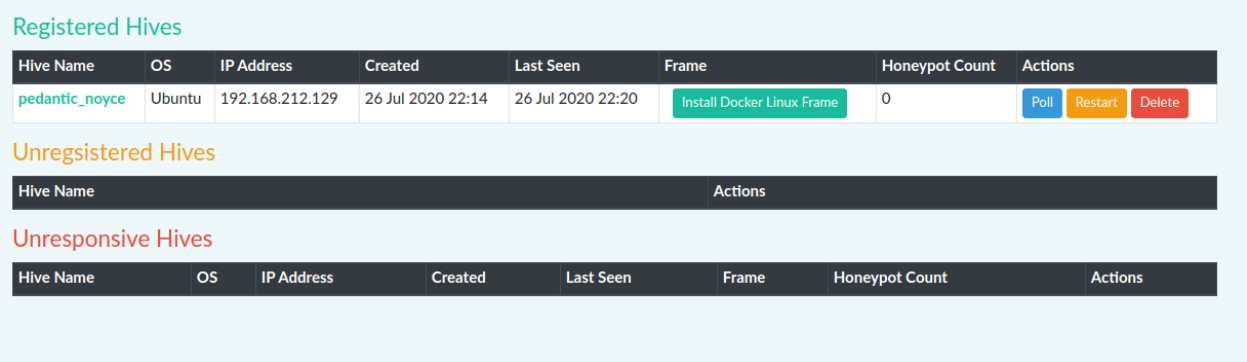
8.3 Add hive to HoneySwarm

Once a Hive has been initialised we need to approve it in to the swarm. This prevents rogue hosts from connecting to us. You will only see the dropdown action once the Minion has started and sent its key to the master. Once availiable Just select the 'Add to swarm' button under actions



After the Hive is authenticated to the swarm you need to add a Frame. For more details on Frames see the Frames page.

All availaible frames will be displayed for Hives. To install a Frame click the Install button



Frame installation can take several minutes depending on OS and internet speeds. You can track the installation under the /jobs page

With a frame installed we can now deploy some honeypots.

CHAPTER 9

Frames

9.1 What is a Frame

ToDo

9.2 Create a Frame

ToDo

Now you have a frame install some honeypots :)

CHAPTER 10

Honeypots

10.1 What is a Honeypot

ToDo

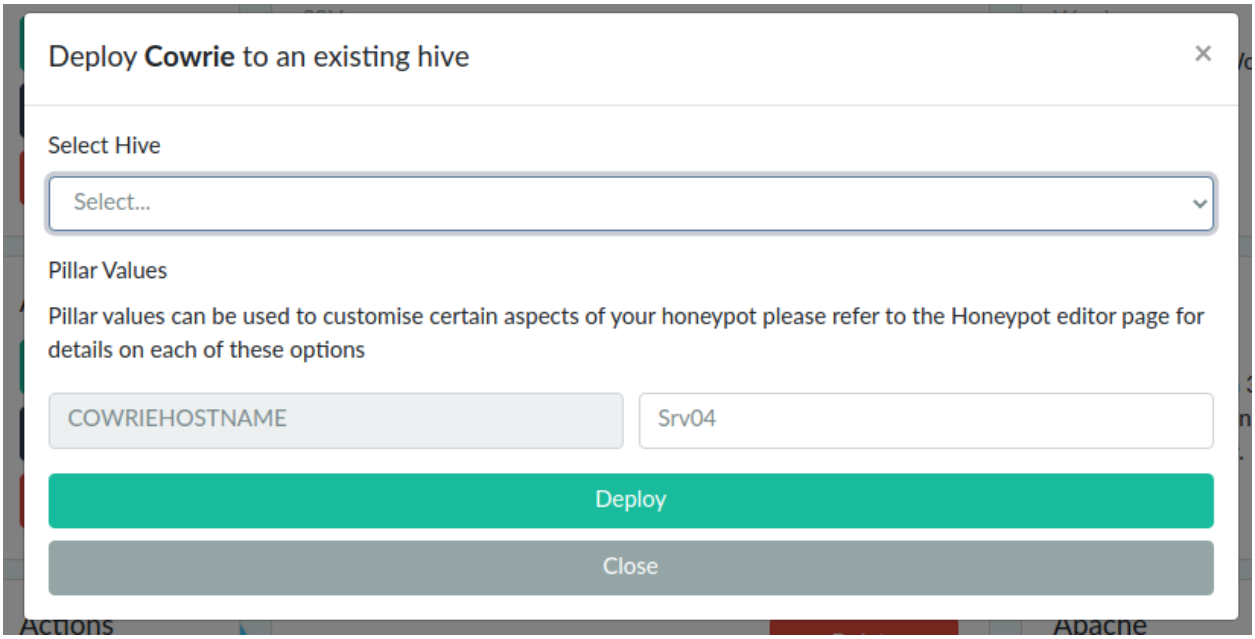
10.2 Existing Honeypots

- Apache
- Conpot
- Cowrie
- ElasticSearch
- SaltStack
- WordPress
- PortScans

10.3 Deploying Honeypots

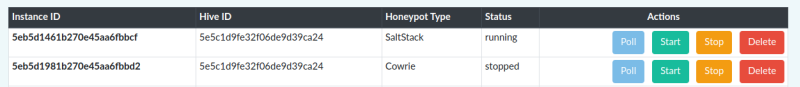
To deploy a honeypot navigate to the honeypots page and from the Available Honeypots section Click *deploy* on the honeypot you wish to load.

From this pop up box you can configure any customisable options and then select the Hive you wish to deploy the honeypot to and Click the Deploy. In the background you should see a notification for a scheduled deployment. The Deployment pop up will stay active until you select close to enable multiple deployments.



10.4 Controlling Honeypots

You can Start, Stop and Delete individual Honeypots from teh Honeypots page.



10.5 Editing Honeypots

ToDo

10.6 Creating New Honeypots

ToDo

CHAPTER 11

ElasticSearch

11.1 Installation

There is a customer docker-compose that you can use to additionaly launch a single node ElasticSearch and Kibana stack. You will need to modify the default password to ensure that your Kibana instance is secured against unathorised access.

11.2 Migration

If you already have data in your HoneySwarm you can add this data to your ElasticSearch using the following steps.

1. Connect to the HoneySwarm docker container
2. Start `flask shell`
3. Run the following python code.

```
import json
import datetime
from elasticsearch import Elasticsearch
elastic_client = Elasticsearch("honeyswarm_es01", http_auth=("elastic", "HoneySwarm"))
from honeyswarm.models import HoneypotEvents
events = HoneypotEvents.objects()
for event in events:
# If you already have data in your elastic index use this datetime filter to avoid␣
↪duplicates.
if event.date < datetime.datetime.strptime("2020-07-31 21:23:42.000000", '%Y-%m-%d %H:
↪%M:%S.%f'):
   try:
      event_entry = json.loads(event.to_json())
      event_entry["event_id"] = str(event_entry['_id'])
      del event_entry['_id']
      event_entry['date'] = event.date
```

(continues on next page)

(continued from previous page)

```
        instance_id = event_entry['honeypot_instance_id']
        index_name = "honeyswarm-{0}".format(instance_id)
        elastic_client.index(index=index_name,body=event_entry)
    except Exception as err:
        print(err)
```

CHAPTER 12

Indices and tables

- genindex
- modindex
- search