Cyphondock Documentation

Release 1.6.0

Leila Hadj-Chikh

Feb 22, 2018

Contents:

Cyphondock is a collection of Docker Compose files and configurations for running Cyphon.

Cyphondock allows you to easily deploy a stack of Docker containers for running Cyphon and its related services, including:

- PostgreSQL
- RabbitMQ
- Nginx
- Logstash
- Elasticsearch

It also contains example configurations for these services.

These files are here to help you get up and running quickly with Cyphon, but you should feel free to modify the Docker Compose files and configuration files to suit your particular needs.

CHAPTER 1

Server Configuration

To deploy Cyphon using Cyphondock, you'll need to install Docker and Docker Compose, and configure virtual memory on the host machine.

1.1 Minimum System Requirements

Cyphon requires that the server (either dedicated machine or virtual machine) hosting Docker has a minimum of 2 cores and at least 8GB of RAM. Docker-based installation will require at least 20GB of storage space. Expect an increase over time due to Docker's snapshot system. For production environments, you may want to map data folders to a separate volume with much larger storage capacity.

1.2 Install Docker

The following instructions are for installing Docker Community Edition (CE) x86_64 on Ubuntu. For other editions, versions, or operating systems, see Docker's documentation.

1.2.1 Unistall Older Versions

If you have an older version of Docker already installed, such as `docker` or `docker-engine`, uninstall it before proceeding:

```
$ sudo apt-get remove docker docker-engine
```

1.2.2 Extra packages for Trusty 14.04

If you're installing on Ubuntu 14.04 (Trusty), Docker strongly recommends installing the `linux-image-extra-*` packages:

```
$ sudo apt-get update

$ sudo apt-get install \
    linux-image-extra-$(uname -r) \
    linux-image-extra-virtual
```

1.2.3 Install Docker CE

To install Docker Community Edition x86_64:

```
$ sudo apt-get update
$ sudo apt-get install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common

$ curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
$ sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"

$ sudo apt-get update
$ sudo apt-get install docker-ce
```

See Docker's Ubuntu documentation for more information on this installation procedure. You can also refer to Digital Ocean's tutorial for Ubuntu 16.04.

1.3 Install Docker Compose

To install Docker Compose on Ubuntu, first install `pip`:

```
$ sudo apt-get install python-pip
```

Then install Docker Compose:

```
$ sudo pip install docker-compose
```

To install Docker Compose on other operating systems, see Docker's documentation.

1.4 Configure Virtual Memory

This project uses the official Elasticsearch docker repository. These images require a Linux host to allow a process to have at least 262,144 memory-mapped areas (see Elasticsearch's documentation for more info).

To increase the memory map count:

```
$ sudo sysctl -w vm.max_map_count=262144
```

Make this setting permanent by editing `/etc/sysctl.conf`:

```
$ echo "vm.max_map_count = 262144" | sudo tee -a /etc/sysctl.conf
```

1.5 Bash Script to Auto Install on Host

You can use the following bash script to auto install the project on an Ubuntu 16.04 server. To install on other operating systems, please modify it accordingly.

```
#!/bin/bash

# Uninstall older version of Docker if present
sudo apt-get remove docker docker-engine

# Update the apt package index
sudo apt-get update

# Install packages to allow apt to use a repository over HTTPS
sudo apt-get -y install \
    apt-transport-https \
    ca-certificates \
    curl \
    software-properties-common

# Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

# Set up the stable repository
sudo add-apt-repository \
    "deb [arch=amd64] https://download.docker.com/linux/ubuntu \
    $(lsb_release -cs) \
    stable"

# Update the apt package index
sudo apt-get update

# Install the latest version of Docker Community Edition
sudo apt-get -y install docker-ce

# Install pip
sudo apt-get -y install python-pip

# Install Docker Compose
sudo pip install docker-compose

# Increase the memory map count for Elasticsearch
```

```
sudo sysctl -w vm.max_map_count=262144

# Make this setting permanent by editing /etc/sysctl.conf
echo "vm.max_map_count = 262144" | sudo tee -a /etc/sysctl.conf

# Download the Cyphondock Git repository into the /opt/cyphon directory
sudo git clone https://github.com/dunbarcyber/cyphondock.git /opt/cyphon/cyphondock

# Copy generic settings to the `config` directory for this project instance
cd /opt/cyphon/cyphondock/
sudo cp -R config-COPYME config

# Set up Elasticsearch and PostgreSQL data directories
sudo mkdir -p /opt/cyphon/data/elasticsearch
sudo mkdir /opt/cyphon/data/postgresql
sudo chown -R 1000:1000 /opt/cyphon/data/elasticsearch
sudo chown -R 999:999 /opt/cyphon/data/postgresql

# Build production environment
sudo docker-compose up -d

echo "One minute... Waiting for services to start..."
sleep 60

echo "Loading configurations"
sudo docker exec -it cyphondock_cyphon_1 sh -c "python manage.py loaddata ./fixtures/
↪starter-fixtures.json"

echo "Create a user account for Cyphon"
echo "sudo docker exec -it cyphondock_cyphon_1 sh -c "python manage.py createsuperuser
↪""

echo "Note: To get Twitter working, don't forget to enable the reservoir!"
echo "Note: You will need to add the index pattern on http://ip_addr:5601/ (cyphon-*)"
```

CHAPTER 2

Project Setup

2.1 Installation

To install your Cyphon project, first download the Cyphondock Git repository. For this example, we'll clone it into the `/opt/cyphon` directory:

```
$ git clone https://github.com/dunbarcyber/cyphondock.git /opt/cyphon/cyphondock
```

2.2 Configuration

Generic settings for the project are contained in the `./config-COPYME` directory. Copy this directory to a new directory called `config`, which will be used by your project instance:

```
$ cd /opt/cyphon/cyphondock/
$ cp -R config-COPYME config
```

The `config` directory contains settings for:

- *Environment Variables*
- *Django*
- *Cyclops*
- *Nginx*
- *Logstash*
- *Elasticsearch*

2.2.1 Environment Variables

The `./config/env/cyphon.env` file contains default settings for superusers created for various services. Use a text editor to change the usernames and passwords to more secure options.

You can read more about environment variables in Docker's documentation.

2.2.2 Django

Cyphon's backend is based on the Django web framework. Settings for the Cyphon Django project are found in the `./config/cyphon/settings` directory.

The `conf.py` file contains settings specific to your Cyphon instance. Using your favorite text editor, edit the `conf.py` file with the IP address and/or domain name for your host machine:

```
HOST_SETTINGS = {
    'ALLOWED_HOSTS': ['example.com', '127.0.0.1'],
    'CORS_ORIGIN_WHITELIST': ['example.com', '127.0.0.1'],
}
```

You should also change the Django `SECRET_KEY` to something unique. You can generate one here.

The `base.py`, `dev.py`, and `prod.py` files contain more general settings for Django and other third party apps used by Cyphon. At a minimum, you should configure the `BASE_URL` setting in the `dev.py` and `prod.py` files. See *Cyphon Configurations* for more info.

2.2.3 Cyclops

Cyclops is an optional frontend that helps to manage alerts and data from Cyphon in real time. This product is under a different license than Cyphon, found here. Make sure to take a quick look over it before using Cyclops in your use case.

Cyclops is configured with the `conf.py` file in the Cyphon settings:

```
CYCLOPS = {
    'ENABLED': True,
    'MAPBOX_ACCESS_TOKEN': '',
    'DEVELOPMENT_ENABLED': False,
    'DEVELOPMENT_URL': 'http://localhost:8080/',
}
```

The only thing to change in these settings is `MAPBOX_ACCESS_TOKEN`. This is required to show maps with locations and to get geolocation data. You can sign up for an access token here.

The next optional step would be to set up push notifications for Cyclops, which is explained in Cyphon's documentation.

2.2.4 Nginx

Settings for the Nginx web server are in the `./config/nginx/nginx.conf` file. You can use the default settings if you like. See *Nginx Configurations* for more info.

2.2.5 Logstash

Cyphon receives data from logs using Logstash and RabbitMQ. Logstash receives the logs and passes them to RabbitMQ, where Cyphon consumes them.

To configure Logstash, you can use the example configuration files in `./config/logstash`. See *Logstash Configurations* for more info. The Docker Compose files are already configured to create a Logstash container based on these configs.

> **Warning:** If you changed the RabbitMQ settings in the `cyphon.env` file, you will need to make sure your `logstash.conf` file reflects those changes.

2.2.6 Elasticsearch

Settings for the Elasticsearch search engine are in the `./config/elasticsearch/` folder. You can use the default settings if you like. See *Elasticsearch Configurations* for more info.

CHAPTER 3

Securing Cyphon

3.1 Disclaimer

The Cyphon project is provided as open-source software, and we encourage you to report any security bugs, configuration issues, or comments related to Cyphon, Cyclops, or Cyphondock. We're extremely thankful for responsible security researchers that report vulnerabilities to us. To make a report, please email us with the full details, including steps to reproduce the issue.

We recommend that you follow security best practices when running Cyphon. Please consult the official documentation of any open source or commercially available products that are used as a component of (or are integrated with) Cyphon, and follow their recommended security practices. This disclaimer also applies to outside APIs, operating systems, or virtualization technologies.

3.2 Credentials

3.2.1 Cyphon environment file

The `cyphondock/config/env/cyphon.env` file contains default usernames and passwords for:

- Cyphon
- PostgreSQL
- RabbitMQ

Please change these values to secure your instance.

3.2.2 Cyphon configuration file

The `cyphondock/config/cyphon/settings/conf.py` file contains default usernames and passwords for:

- PostgreSQL

- RabbitMQ

If you're not setting these values through environment variables in the *Cyphon environment file*, you should change the default values in `conf.py`.

You should also change the Django `SECRET_KEY` to something unique. See the instructions on *configuring Django* for details.

3.2.3 RSA keys

The `KEYS_DIR` setting in the *Cyphon configuration file* specifies the directory in which Django stores RSA keys associated with Passports. Public access to this directory should be denied. Check your `cyphondock/config/nginx/nginx.conf` file to confirm this.

3.3 Ports

Cyphondock's Docker Compose files for the *production environment* open the following ports on the host machine:

Port	Protocol	Service
80	TCP (HTTP)	Cyphon - Nginx proxy
443	TCP (HTTPS)	Cyphon - Nginx proxy
5000	UDP	Logstash
5044	TCP	Logstash
5045	TCP	Logstash
5601	TCP	Kibana - Nginx proxy
15672	TCP	RabbitMQ management - Nginx proxy

Please restrict external access to these ports.

3.4 Running Cyphon with SSL through Nginx

You may wish to run Cyphon with SSL to secure connections. Cyphondock makes this easy by providing an alternate Nginx configuration file at `cyphondock/config/nginx/nginx.conf.ssl` that can be used to get started quickly. Placeholder files for SSL certificates and private keys are also provided. You can choose to use self-signed certificates or use certificates signed by a trusted certificate authority.

Assuming you already have an SSL certificate and private key, copy their contents into the placeholder files provided:

```
$ cd /opt/cyphon/cyphondock/config/nginx
$ cp /path/to/your/certificate ssl.crt
$ cp /path/to/your/private/key ssl.key
```

Then, copy the provided `nginx.conf.ssl` to be the main Nginx configuration file:

```
$ cp nginx.conf.ssl nginx.conf
```

Finally, restart the Nginx Docker container or the entire Docker-Compose:

```
$ docker-compose restart
```

CHAPTER 4

Docker Compose files

Cyphondock contains several Docker Compose files used for deploying to development, testing, and production environments:

File name	Documentation
docker-compose.yml	*Base Configuration*
docker-compose.dev.yml	*Development Environment*
docker-compose.override.yml	*Production Environment*
docker-compose.travis.yml	*Testing Environment*
docker-compose.backends.yml	*Backends*

You can read more about Docker Compose files in Docker's official documentation.

4.1 Cyphon Images

4.1.1 Dev Image

Builds from custom cyphon.env file

Image	local build from `../cyphon`
Env File	*./config/env/cyphon.env*
Environment	`CYPHON_ENV: PROD DJANGO_SETTINGS_MODULE: cyphon.settings.prod`
Volumes	*./config/cyphon/custom*, *./config/cyphon/settings*, *./config/cyphon/fixtures*, cyphon, entrypoints

YAML:

```
cyphon-dev:
  build: ../cyphon
  restart: always
  env_file:
```

```
    - ./config/env/cyphon.env
  environment:
    CYPHON_ENV: DEV
    DJANGO_SETTINGS_MODULE: cyphon.settings.dev
  volumes:
    - ../cyphon/cyphon:/usr/src/app/cyphon
    - ../cyphon/entrypoints:/usr/src/app/entrypoints
    - ./config/cyphon/custom:/usr/src/app/cyphon/custom
    - ./config/cyphon/settings:/usr/src/app/cyphon/cyphon/settings
```

4.1.2 Production Image

Builds from custom cyphon.env file

Image	dunbar/cyphon
Env File	*./config/env/cyphon.env*
Environment	`CYPHON_ENV: PROD DJANGO_SETTINGS_MODULE: cyphon.settings.prod`
Volumes	*./config-COPYME/cyphon/custom*, *./config-COPYME/cyphon/settings*, *./config-COPYME/cyphon/fixtures* `../media`, `../keys`

YAML:

```
cyphon-prod:
  image: dunbar/cyphon:${CYPHON_VER}
  restart: always
  env_file:
    - ./config/env/cyphon.env
  environment:
    CYPHON_ENV: PROD
    DJANGO_SETTINGS_MODULE: cyphon.settings.prod
  volumes:
    - ../media:/usr/src/app/media
    - ../keys:/usr/src/app/keys
    - ./config/cyphon/custom:/usr/src/app/cyphon/custom
    - ./config/cyphon/settings:/usr/src/app/cyphon/cyphon/settings
    - ./config/cyphon/fixtures:/usr/src/app/cyphon/fixtures
```

4.1.3 Testing Image

Builds from default cyphon.env file

Image	dunbar/cyphon
Env File	*./config-COPYME/env/cyphon.env*
Environment	`CYPHON_ENV: PROD DJANGO_SETTINGS_MODULE: cyphon.settings.prod`
Volumes	*./config-COPYME/cyphon/custom*, *./config-COPYME/cyphon/settings*, *./config-COPYME/cyphon/fixtures*

YAML:

```
cyphon-test:
  image: dunbar/cyphon:${CYPHON_VER}
  env_file:
    - ./config-COPYME/env/cyphon.env
  environment:
    CYPHON_ENV: PROD
    DJANGO_SETTINGS_MODULE: cyphon.settings.prod
    DJANGO_LIVE_TEST_SERVER_ADDRESS: cyphon:8081-8100
    FUNCTIONAL_TESTS_ENABLED: ${FUNCTIONAL_TESTS_ENABLED}
    FUNCTIONAL_TESTS_DRIVER: docker
    FUNCTIONAL_TESTS_HOST: selenium
    FUNCTIONAL_TESTS_PORT: 4444
    FUNCTIONAL_TESTS_BROWSER: chrome
  volumes:
    - ./config-COPYME/cyphon/custom:/usr/src/app/cyphon/custom
    - ./config-COPYME/cyphon/settings:/usr/src/app/cyphon/cyphon/settings
    - ./config-COPYME/cyphon/fixtures:/usr/src/app/cyphon/fixtures
```

4.2 Base Configuration

The docker-compose.yml file contains base settings for use in both the *Production Environment* and *Development Environment*. The file includes the following services:

- *Cyphon services*
- *PostgreSQL*
- *RabbitMQ*
- *Nginx*
- *GeoIP*
- *Elastic stack*
- *MongoDB*
- *Logspout*

4.2.1 Cyphon services

Cyphon

This service is based on the *Cyphon production image*. It links to *PostgreSQL*, which it uses as its Django backend database service for storing Cyphon admin settings and alerts. It also links to *Elasticsearch* and (optionally) *MongoDB* for storing data.

The service creates a volume for *Cyphon fixtures*, which can be used to load settings. It also uses the GeoLite2 database from the *GeoIP service* to assign geolocations to IP addresses.

Extends	*Cyphon production image*
Links	*PostgreSQL*, *Elasticsearch*, (optionally) *MongoDB*
Dependency	*PostgreSQL*
Shared Volume	*GeoIP*

YAML:

```
cyphon:
  extends:
    file: common-services.yml
    service: cyphon-prod
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
    - postgres
  depends_on:
    - postgres

    serialized_results = [AlertDetailSerializer(alert) for alert in queryset]
```

Celerybeat

This service runs a celery beat scheduler for periodic tasks, such as checking and processing email. It is similar to the *Cyphon service*, but it runs a different command on start up, and it has a link to *RabbitMQ*, where it sends the scheduled tasks.

Extends	*Cyphon production image*
Command	run_celerybeat.sh
Links	*PostgreSQL*, *RabbitMQ*, *Elasticsearch*, (optionally) *MongoDB*
Dependencies	*PostgreSQL*, *RabbitMQ*
Shared Volume	*GeoIP*

YAML:

```
celerybeat:
  extends:
    file: common-services.yml
    service: cyphon-prod
  restart: always
  command: ../entrypoints/run_celerybeat.sh
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
    - postgres
    - rabbit
  depends_on:
    - cyphon
    - rabbit
```

Celeryworker

This service runs a celery worker to handle periodic tasks sent to *RabbitMQ* by *Celerybeat*. It is similar to the *Celerybeat service*, but it runs a different command on start up.

Extends	*Cyphon production image*
Command	run_celeryworker.sh
Links	*PostgreSQL*, *RabbitMQ*, *Elasticsearch*, (optionally) *MongoDB*
Dependencies	*PostgreSQL*, *RabbitMQ*
Shared Volume	*GeoIP*

YAML:

```
celeryworker:
  extends:
    file: common-services.yml
    service: cyphon-prod
  restart: always
  command: ../entrypoints/run_celeryworker.sh
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
    - postgres
    - rabbit
  depends_on:
    - cyphon
    - rabbit
```

LogChutes

This service creates a queue consumer for log messages sent to *RabbitMQ* from *Logstash*. The messages are sent to Cyphon's LogChutes for processing.

Image	*Cyphon production image*
Command	run_receiver.sh `logchutes`
Links	*PostgreSQL*, *RabbitMQ*, *Elasticsearch*, (optionally) *MongoDB*
Dependencies	*PostgreSQL*, *RabbitMQ*
Shared Volume	*GeoIP*

YAML:

```
logchutes:
  extends:
    file: common-services.yml
    service: cyphon-prod
  restart: always
  command: ../entrypoints/run_receiver.sh logchutes
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
    - postgres
    - rabbit
  depends_on:
    - cyphon
    - rabbit
```

Monitors

This service creates a queue consumer for JSON messages sent to *RabbitMQ* from *Logstash*. These messages are inspected by Cyphon's Monitors.

Image	*Cyphon production image*
Command	run_receiver.sh `monitors`
Links	*PostgreSQL*, *RabbitMQ*, *Elasticsearch*, (optionally) *MongoDB*
Dependencies	*PostgreSQL*, *RabbitMQ*
Shared Volume	*GeoIP*

YAML:

```
monitors:
  extends:
    file: common-services.yml
    service: cyphon-prod
  restart: always
  command: ../entrypoints/run_receiver.sh monitors
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
    - postgres
    - rabbit
  depends_on:
    - cyphon
    - rabbit
```

Watchdogs

This service creates a queue consumer for JSON messages sent to *RabbitMQ* from *Logstash*. These messages are inspected by Cyphon's Watchdogs.

Image	*Cyphon production image*
Command	run_receiver.sh `watchdogs`
Links	*PostgreSQL*, *RabbitMQ*, *Elasticsearch*, (optionally) *MongoDB*
Dependencies	*PostgreSQL*, *RabbitMQ*
Shared Volume	*GeoIP*

YAML:

```
watchdogs:
  extends:
    file: common-services.yml
    service: cyphon-prod
  restart: always
  command: ../entrypoints/run_receiver.sh watchdogs
  volumes_from:
    - geoip
  links:
    - elasticsearch
    # - mongo
```

```
    - postgres
    - rabbit
  depends_on:
    - cyphon
    - rabbit
```

4.2.2 PostgreSQL

This service creates a PostGIS database for saving Cyphon configurations and alerts. It is used as the database backend for GeoDjango. The host name, database name, username, and passowrd are determined by settings in the *Environment Variables*.

Image	mdillon/postgis
Env File	*cyphon.env*

YAML:

```
postgres:
  image: mdillon/postgis:${POSTGRES_VER}
  restart: always
  env_file:
    - ./config/env/cyphon.env
```

4.2.3 RabbitMQ

This service creates a RabbitMQ message broker for *Logstash* and *Cyphon services*. Defaults for the host name, virtual host, username, and password are determined by settings in the *Environment Variables*.

Image	rabbitmq
Env File	*cyphon.env*

YAML:

```
rabbit:
  image: rabbitmq:${RABBITMQ_VER}
  restart: always
  env_file:
    - ./config/env/cyphon.env
```

4.2.4 Nginx

This service creates a web service for *Cyphon*. It shares volumes from *Cyphon*, including directories for static assets and media files.

Image	nginx
Links	*Cyphon*, *RabbitMQ*, *Kibana*
Dependencies	*Cyphon*
Shared Volume	*Cyphon*

YAML:

```
nginx:
  image: nginx:${NGINX_VER}
  restart: always
  volumes:
    - ./config/nginx/nginx.conf:/etc/nginx/nginx.conf:ro
    - ./config/nginx/ssl.crt:/etc/nginx/ssl.crt:ro
    - ./config/nginx/ssl.key:/etc/nginx/ssl.key:ro
    - /www/static
  volumes_from:
    - cyphon
  links:
    - cyphon
    - kibana
    - rabbit
  depends_on:
    - cyphon
```

4.2.5 GeoIP

This service provides GeoLite2 databases for Cyphon's GeoIP package.

Image	geoip

YAML:

```
geoip:
  image: dunbar/geoip
  restart: always
```

4.2.6 Elastic stack

Elasticsearch

This service provides an Elasticsearch backend for Cyphon's Warehouses. It's also used to store data from *Logstash*.

The host name and port are determined by settings in the *Environment Variables*.

Image	nginx
Environment	`http.host=0.0.0.0` `transport.host=127.0.0.1`
Links	*Cyphon*, *RabbitMQ*, *Kibana*
Volumes	*elasticsearch.yml*, *jvm.options*, *log4j2.properties*

YAML:

```
elasticsearch:
  image: docker.elastic.co/elasticsearch/elasticsearch:${ELASTIC_VER}
  restart: always
  environment:
    - http.host=0.0.0.0
    - transport.host=127.0.0.1
  volumes:
    - ./config/elasticsearch/elasticsearch.yml:/usr/share/elasticsearch/config/
↪elasticsearch.yml:ro
    - ./config/elasticsearch/jvm.options:/usr/share/elasticsearch/config/jvm.
↪options:ro
    - ./config/elasticsearch/log4j2.properties:/usr/share/elasticsearch/config/log4j2.
↪properties:ro
```

Logstash

This service ingests and parses logs, and sends them to *Elasticsearch* and *RabbitMQ*.

Image	docker.elastic.co/logstash/logstash
Command	`logstash -f /usr/share/logstash/pipeline --config.reload.automatic`
Links	*Elasticsearch*, *RabbitMQ*
Dependencies	*Elasticsearch*, *RabbitMQ*
Volumes	*config*, *patterns*, *pipeline*

YAML:

```
logstash:
  image: docker.elastic.co/logstash/logstash:${ELASTIC_VER}
  restart: always
  command: logstash -f /usr/share/logstash/pipeline --config.reload.automatic
  volumes:
    - ./config/logstash/config:/usr/share/logstash/config:ro
    - ./config/logstash/patterns:/usr/share/logstash/patterns:ro
    - ./config/logstash/pipeline:/usr/share/logstash/pipeline:ro
  links:
    - elasticsearch
    - rabbit
  depends_on:
    - elasticsearch
    - rabbit
```

Filebeat

This optional service can be used to monitor logs and send them to *Logstash*. You can use it for local testing of your *Filebeat Configurations*.

Image	docker.elastic.co/beats/filebeat
Links	*Logstash*
Dependencies	*Logstash*
Volumes	*filebeat.yml*, `./log`

YAML:

```
# filebeat:
#   image: docker.elastic.co/beats/filebeat:${ELASTIC_VER}
#   restart: always
#   links:
#     - logstash
#   depends_on:
#     - logstash
#   volumes:
#     - ./config/beats/filebeat/filebeat.yml:/usr/share/filebeat/filebeat.yml:rw
#     - ./log:/var/log
```

Kibana

This service provides a dashboard for viewing *Elasticsearch* data.

Image	docker.elastic.co/kibana/kibana
Environment	`LOGSPOUT: ignore`
Links	*Elasticsearch*
Dependencies	*Elasticsearch*
Volumes	*kibana.yml*

YAML:

```
kibana:
  image: docker.elastic.co/kibana/kibana:${ELASTIC_VER}
  restart: always
  environment:
    LOGSPOUT: ignore  # don't send Kibana's logs to Logspout
  links:
    - elasticsearch
  depends_on:
    - elasticsearch
  volumes:
    - ./config/kibana/kibana.yml:/usr/share/kibana/config/kibana.yml:ro
```

4.2.7 MongoDB

This optional service provides a MongoDB backend for Cyphon's Warehouses. It can also be used to store data from *Logstash* when used with Logstash's MongoDB output plugin.

The host name and port are determined by settings in the *Environment Variables*.

Image	mongo

YAML:

```
# mongo:
#   image: mongo:${MONGODB_VER}
#   restart: always
```

4.2.8 Logspout

This service collects logs from the other Docker containers and sends them to *Logstash*. From there, they can be stored in *Elasticsearch* and viewed in *Kibana*.

Image	gliderlabs/logspout
Command	`syslog://logstash:5000`
Expose	5000/udp
Links	*Logstash*
Dependencies	*Logstash*
Volumes	`/var/run/docker.sock`

YAML:

```
logspout:
  image: gliderlabs/logspout:${LOGSPOUT_VER}
  restart: always
  expose:
    - "5000/udp"
  volumes:
    - /var/run/docker.sock:/var/run/docker.sock:ro
  command: syslog://logstash:5000
  links:
    - logstash
  depends_on:
    - logstash
```

4.3 Development Environment

Unlike the *production environment*, the dev environment builds the images for Cyphon using local files. You'll need to download the Cyphon GitHub repository into the `/opt/cyphon/cyphon` directory so these files are available to Docker Compose:

```
$ git clone https://github.com/dunbarcyber/cyphon.git /opt/cyphon/cyphon
```

Settings for the dev environment are contained in the `docker-compose.dev.yml` file. To use the dev environment, you must specify both the base configuration file and the dev configuration file when starting the containers:

```
$ cd /opt/cyphon/cyphondock
$ sudo docker-compose -f docker-compose.yml -f docker-compose.dev.yml up
```

Use the `-d` flag if you want to start the containers in the background.

4.3.1 Superuser

The dev environment will create a Cyphon superuser using the `CYPHON_USERNAME` and `CYPHON_PASSWORD` settings defined in your .config/env/cyphon.env file, if that user does not already exist.

4.3.2 Ports

The dev environment maps the exposed ports in the containers to random free ports on the host. This prevents any problems with ports already in use on the host. Once the docker containers are running, you can list the containers to

see which ports on the host are being used for the various services:

```
$ sudo docker ps -a
```

For example, the Kibana container exposes port 5601, but that port will be mapped to a random port on the host machine. If the port mapping for the Kibana container is `0.0.0.0:32775->5601/tcp`, you would visit `http://localhost:32775` in your web browser to access Kibana.

4.3.3 Data Volumes

Unlike the *production environment*, the dev environment does not assign any specific host directories to store data from PostgreSQL, Elasticsearch, or MongoDB. Instead, it allows those services to automatically configure the locations for their data volumes. This prevents any confusion of dev and production data if the production environment is used at a later time. See Docker's documentation to learn more about data volumes.

4.3.4 Container Overrides

Cyphon

This override for the *base Cyphon service* will create a Docker container using the *Cyphon dev image*. It will also create a superuser using the settings in the *Cyphon environment file*, and load starter settings from *Fixtures*.

YAML:

```
cyphon:
  extends:
    file: common-services.yml
    service: cyphon-dev
  environment:
    CYPHON_SUPERUSER: "YES"
    LOAD_EXAMPLE_FIXTURES: "YES"
  ports:
    - "8000"
```

Celerybeat

This override for the *base Celerybeat service* will create a Docker container using the *Cyphon dev image*.

YAML:

```
celerybeat:
  extends:
    file: common-services.yml
    service: cyphon-dev
```

Celeryworker

This override for the *base Celeryworker service* will create a Docker container using the *Cyphon dev image*.

YAML:

```
celeryworker:
  extends:
    file: common-services.yml
    service: cyphon-dev
```

LogChutes

This override for the *base LogChutes service* will create a Docker container using the *Cyphon dev image*.

YAML:

```
logchutes:
  extends:
    file: common-services.yml
    service: cyphon-dev
```

Monitors

This override for the *base Monitors service* will create a Docker container using the *Cyphon dev image*.

YAML:

```
monitors:
  extends:
    file: common-services.yml
    service: cyphon-dev
```

Watchdogs

This override for the *base Watchdogs service* will create a Docker container using the *Cyphon dev image*.

YAML:

```
watchdogs:
  extends:
    file: common-services.yml
    service: cyphon-dev
```

Nginx

This override for the *base Nginx service* will map the ports used by *Cyphon*, *Kibana*, and *RabbitMQ* to random ports on the host machine.

YAML:

```
nginx:
  ports:
    - "80"     # Cyphon
    - "443"    # Cyphon
    - "5601"   # Kibana
    - "15672"  # RabbitMQ management plugin
```

RabbitMQ

This override for the *base RabbitMQ service* will map port 15672 of the Logstash container to a random port on the host machine. This port is used for the RabbitMQ management web interface.

YAML:

```
rabbit:
  ports:
    - "15672"  # RabbitMQ management plugin
```

Logstash

This override for the *base Logstash service* will map ports 5044-5045 of the Logstash container to random ports on the host machine.

YAML:

```
logstash:
  ports:
    - "5044"
    - "5045"
```

Kibana

This override for the *base Kibana service* will map port 5601 of the Kibana container to a random port on the host machine.

YAML:

```
kibana:
  ports:
    - "5601"
```

4.4 Production Environment

The production environment uses a Cyphon image on Docker Hub as the basis for its Cyphon container.

Settings for the production environment are contained in the docker-compose.override.yml file. Docker Compose automatically reads both the docker-compose.yml and docker-compose.override.yml files when it builds an environment, unless other files are specified. So in this case, Docker will deploy the production environment by default.

To start up the (default) production environment:

```
$ cd /opt/cyphon/cyphondock
$ sudo docker-compose up -d
```

4.4.1 Superuser

To create a superuser for the production environment, enter the shell of the Django container:

```
$ sudo docker exec -it cyphondock_cyphon_1 /bin/sh
```

Then create the superuser:

```
$ python manage.py createsuperuser
```

You can use this user to log in to Cyphon and create other user accounts.

4.4.2 Ports

The production environment will map the exposed ports in the containers to same ports on the host machine. This makes it straightforward to use the endpoints of the various services.

You can access Cyphon and some of its related services through your web browser. The following services can be accessed by visiting your host domain on the appropriate port (e.g., `http://example.com:5601`):

Service	Port
Cyphon	80
Kibana	5601
RabbitMQ management	15672

4.4.3 Data Volumes

The production environment uses the subdirectories in `/opt/cyphon/data` as data volumes for PostgreSQL, Elasticsearch, or MongoDB. This makes the data from those services easy to locate and backup.

4.4.4 Container Overrides

PostgreSQL

This override for the *base PostgreSQl service* will store data in the `./data/postgresql` directory on the host machine.

YAML:

```
postgres:
  volumes:
    - ../data/postgresql:/var/lib/postgresql/data
```

Nginx

This override for the *base Nginx service* will map the ports used by *Cyphon*, *Kibana*, and *RabbitMQ* to the same ports on the host machine.

YAML:

```
nginx:
  ports:
    - "80:80"        # Cyphon
    - "443:443"      # Cyphon
    - "5601:5601"    # Kibana
    - "15672:15672"  # RabbitMQ management plugin
```

Elasticsearch

This override for the *base Elasticsearch service* will store data in the `./data/elasticsearch` directory on the host machine.

YAML:

```
elasticsearch:
  volumes:
    - ../data/elasticsearch:/usr/share/elasticsearch/data
```

Logstash

This override for the *base Logstash service* will allow Logstash to listen on ports 5044-5045 of the host machine.

YAML:

```
logstash:
  ports:
    - "5044:5044"
    - "5045:5045"  # you can configure Filebeat to send Nessus files here
```

MongoDB

This override for the *base MongoDB service* will store data in the `./data/mongodb` directory on the host machine.

YAML:

```
# mongo:
#   volumes:
#     - ../data/mongodb:/data/db
```

4.5 Testing Environment

The docker-compose.travis.yml file is used to run tests in Travis CI using the default configs in the config-COPYME directory.

4.6 Backends

The docker-compose.backends.yml file can be used to provide the backends needed to run Cyphon outside of a Docker container. This can be useful for local development and testing. To start the backend services:

```
$ cd /opt/cyphon/cyphondock
$ sudo docker-compose -f docker-compose.backends.yml up
```

4.6.1 PostgreSQL

This service will create a PostgreSQL server using environment settings in the original config-COPYME/env/cyphon.env file (as opposed to customized settings in the `config` directory you may have created for your own use). The server will be accessible on port 5432 of your host machine.

YAML:

```
postgres:
  image: mdillon/postgis:${POSTGRES_VER}
  env_file:
    - ./config-COPYME/env/cyphon.env
  ports:
    - "5432:5432"
```

4.6.2 Elasticsearch

This service will create an Elasticsearch server using Elasticsearch and environment settings in the original config-COPYME directory (as opposed to customized settings in the `config` directory you may have created for your own use). The server will be accessible on port 9200 of your host machine.

YAML:

```
elasticsearch:
  image: docker.elastic.co/elasticsearch/elasticsearch:${ELASTIC_VER}
  env_file:
    - ./config-COPYME/env/cyphon.env
  ports:
    - "9200:9200"
  volumes:
    - ./config-COPYME/elasticsearch/elasticsearch.yml:/usr/share/elasticsearch/config/
↪elasticsearch.yml:ro
    - ./config-COPYME/elasticsearch/jvm.options:/usr/share/elasticsearch/config/jvm.
↪options:ro
    - ./config-COPYME/elasticsearch/log4j2.properties:/usr/share/elasticsearch/config/
↪log4j2.properties:ro
```

4.6.3 MongoDB

This service will create a MongoDB server using the environment settings in the original config-COPYME/env/cyphon.env file (as opposed to customized settings in the `config` directory you may have created for your own use). The server will be accessible on port 27017 of your host machine.

YAML:

```
mongo:
  image: mongo:${MONGODB_VER}
  env_file:
    - ./config-COPYME/env/cyphon.env
  ports:
    - "27017:27017"
```

CHAPTER 5

Configuration files

Cyphondock contains example configurations for:

- Cyphon
- Nginx
- Elasticsearch
- Logstash
- Kibana
- Filebeat

5.1 Environment Variables

The env/cyphon.env file contains environment settings for *Cyphon services*, *PostgreSQL*, *RabbitMQ*, *Elasticsearch*, and *MongoDB*.

5.2 Cyphon Configurations

5.2.1 Fixtures

The cyphon/fixtures/starter-fixtures.json file contains some generic configurations to use when starting your Cyphon project. You can read more about fixtures in Django's documentation.

5.2.2 Custom

The cyphon/custom directory provides a place to add custom modules for Cyphon admin pages and filter backends. See the documentation for the Django admin site and Django REST Framework filters to learn more.

5.2.3 Settings

The cyphon/settings directory contains configurations for Cyphon's Django-based web framework, as well Cyphon-specific configurations. You can read more about Django settings in Django's documentation.

conf.py

The conf.py file contains settings specific to your Cyphon instance.

At a minimum, should edit the `conf.py` file with the IP address and/or domain name for your host machine:

```
HOST_SETTINGS = {
    'ALLOWED_HOSTS': ['example.com', '127.0.0.1'],
    'CORS_ORIGIN_WHITELIST': ['example.com', '127.0.0.1'],
}
```

You should also change the Django `SECRET_KEY` to something unique. You can generate one here.

base.py

The base.py file contains Django settings common to both the *Production Environment* and *Development Environment*.

dev.py

The dev.py file contains Django settings specific to the *Development Environment*. At a minimum, you should configure the `BASE_URL` setting.

prod.py

The prod.py file contains Django settings specific to the *Production Environment*. At a minimum, you should configure the `BASE_URL` setting.

default.py

The default.py file contains generic settings that can be used when running services that require Cyphon apps, such as Cyphon's message receiver.

5.3 Nginx Configurations

5.3.1 nginx.conf

The nginx.conf file configures the Nginx web server for accessing Cyphon. See Nginx's documentation for details.

5.3.2 nginx.conf.ssl

The nginx.conf.ssl file can be optionally used to run Cyphon with SSL. The SSL server certificate should be copied into ssl.crt and the corresponding private key should be copied into ssl.key. See Nginx's documentation on configuring other SSL options.

5.4 Elasticsearch Configurations

5.4.1 elasticsearch.yml

The elasticsearch.yml file contains Elasticsearch configuration flags. See Elasticsearch's documentation for more info.

If you don't have X-Pack, be sure to include the setting `xpack.security.enabled: false`. See Elastic's documentation for more info.

5.4.2 jvm.options

The jvm.options file is used to set JVM heap size. See Elasticsearch's documentation for more info.

5.4.3 log4j2.properties

The log4j2.properties file is used to configure Elasticsearch logging. See Elasticsearch's documentation for more info.

5.5 Logstash Configurations

5.5.1 config

The logstash/config directory contains Logstash settings files.

logstash.yml

The logstash.yml file contains Logstash configuration flags. See Logstash's documentation for more info.

If you don't have X-Pack, be sure to include the setting `xpack.monitoring.enabled: false: false`. See Elastic's documentation for more info.

jvm.options

The jvm.options file contains JVM configuration flags. It can be used to set JVM heap size and the locale for Logstash.

log4j2.properties

The log4j2.properties file is used to configure Logstash logging. See Logstash's documentation for more info.

5.5.2 patterns

The logstash/patterns directory contains grok patterns that can be used by Logstash's grok filter plugin.

5.5.3 pipeline

The logstash/pipeline directory contains files that define a Logstash pipeline, including inputs, filters, and outputs. See Logstash's documentation for more info.

5.6 Kibana Configurations

5.6.1 kibana.yml

The kibana.yml file contains Kibana configuration flags. See Kibana's documentation for more info.

If you don't have X-Pack, be sure to include the setting `xpack.security.enabled: false`. See Elastic's documentation for more info.

5.7 Filebeat Configurations

5.7.1 filebeat.yml

This filebeat.yml file contains example Filebeat configurations for sending logs to Logstash. See Filebeat's documentation for configuring Filebeat and available configuration options.