Installing using Docker/Podman

Installation · https://2yetis.net/docs/installing-in-a-docker-environment/

You can install uvuyo in a docker or podman environment on a machine. Be aware that the installation described here will install all components on one host. This will not provide you with high availability and the throughput is limited by the resources of the machine you are installing on. This is why this type of installation is only recommended for test or development environments which don't have the need for high availability and which handle limited amount of data. As a rule of thumb you would calculate 1 CPU per pipeline. So if you have a dedicated host for uvuyo with 8 CPUs you could run 8 pipelines (a pipeline are two connected endpoints for example an SNMP adapter sending data to a Helix Event Manager.)

Architecture

The core of the uvuyo product is composed of the uvuyo node which runs as a micro service in a docker container and the common services. The common services are kafka in combination with zookeeper which is used to send data between the uvuyo nodes and endpoints. The elasticsearch database is used to persist data.

All microservices can be loaded from docker hub.

Preparation

In order to run uvuyo in a docker environment you should have docker installed on the machine you want to install the product. You should be running a version of docker which is still officially supported.

Note
The commands listed in this article are Unix commands. We recommend you to use a Unix System to run a docker or podman environment. If you choose to use a windows System, you would need to change the command appropriately so that the paths are set correctly.
Also you should make sure, that you can access the systems you want to connect from this machine. Most of the inbound endpoints (collectors) use a push mechanism to retrieve data from the source. This means that the connection is going to be opened from the source system. Some endpoints - like the prometheus metric collector - use a pull mechanism to collect data. So the

connection is open from the uvuyo endpoint to the source system.

Dependent if you have an inbound or outbound connection you will need to open the firewall to your machine appropriately.

Containers

As mentioned in the introduction the minimum uvuyo installation consists of four microservices. The uvuyo node and the common services kafka for the communication between the endpoints, zookeeper is currently still required to run kafka and elasticsearch as a database to persist data.

We will learn in a later section, how to setup these services.

Network

The uvuyo micro services will use a network to communicate with each other. You should create a specific docker network in your docker environment. We recommend to call the network uvuyo.

We will learn in a later section, how to setup the network

Volumes

The micro services will need a docker volumes to persist their data. Elasticssearch will need a volume for example to store all the data uvuyo wants to persists. We need volumes to ensure that the data stored while the micro services are running is still available after the service is restarted.

Uvuyo normaly stores everthing in the elasticsearch database. Nevertheless core configuration can also be stored in the file system. Core configuration data is for example the logging level the microservice runs with. Also some endpoints store configuration data in the volume. The SNMP adapter for example loads MIB files from volumes. This es why we need a specific volume for uvuyo.

Kafka uses volumes to store topics. Topics are used to send messages between endpoints.

We will learn in a later section how to setup the volumes.

Setting up the network

To setup the network on your machine use the following command:

- Docker
- Podnan

```
docker network create -d bridge uvuyo
```

```
podman network create -d bridge uvuyo
```

Setting up the volumes

To create the uvuyo volume first create a directory which you would like to use on your local machine. We recommend to call the directory /opt/2yetis/uvuyo

```
sudo mkdir -p /opt/2yetis/uvuyo \
sudo mkdir -p /opt/2yetis/uvuyo/etc \
sudo mkdir -p /opt/2yetis/uvuyo/mibs
```

This will create the directory including all the parent directories if they don't exist. Make sure that directory can be accessed by docker.

- Docker
- Podman

```
docker volume create elastic_data ;\
docker volume create zookeeper_data ;\
docker volume create kafka_data
```

```
podman volume create elastic_data ;\
podman volume create zookeeper_data ;\
podman volume create kafka_data
```

These commands will create the volumes in docker.

Setting up the Containers

Uvuyo

The following command will create a container running an uvuyo node

- Docker

- Podman

```
docker create --network uvuyo \
--name uvuyo \
--volume /opt/2yetis/uvuyo/etc:/app/uvuyo/etc \
--volume /opt/2yetis/uvuyo/mibs:/app/uvuyo/mibs \
-e UVUYO_TENANTID=uvuyo \
-e UVUYO_HOME=/app/uvuyo \
-e UVUYO_NODEID=uvuyo \
-e UVUYO_GROUPID=uvuyo \
-e UVUYO_ELASTIC_URL=http://elastic:9200 \
-e UVUYO_KAFKA_BOOTSTRAP_SERVERS=kafka:9092 \
-e SERVER_PORT=443 \
-p 443:443 \
the2yetis/uvuyo:1.4.0
```

```
podman create --network uvuyo \
--name uvuyo \
--volume /opt/2yetis/uvuyo/etc:/app/uvuyo/etc \
--volume /opt/2yetis/uvuyo/mibs:/app/uvuyo/mibs \
-e UVUYO_TENANTID=uvuyo \
-e UVUYO_HOME=/app/uvuyo \
-e UVUYO_NODEID=uvuyo \
-e UVUYO_GROUPID=uvuyo \
-e UVUYO_ELASTIC_URL=http://elastic:9200 \
-e UVUYO_KAFKA_BOOTSTRAP_SERVERS=kafka:9092 \
-e SERVER_PORT=443 \
-p 443:443 \
docker.io/the2yetis/uvuyo:1.4.0
```

Note

If you don't have root privileges you might have problems binding to port 443. You might consider to change the line from -p 443:443 to -p 8443:443 to expose the port 8443 instead.

Zookeeper

The following command will create a container running zookeeper which is needed for the kafka service

- Docker
- Podman

```
docker create --network uvuyo \
--name zookeeper \
--volume zookeeper_data:/bitnuami \
-e ZOO_SERVER_ID=0 \
-e ZOO_PORT_NUMBER=2181 \
-e ALLOW_ANONYMOUS_LOGIN='yes' \
-e ZOO_SERVERS=zookeeper:2888:3888::0 \
-p 2181:2181 \
-p 2888:2888 \
-p 3888:3888 \
bitnami/zookeeper:3.8
```

```
podman create --network uvuyo \
--name zookeeper \
--volume zookeeper_data:/bitnuami \
-e ZOO_SERVER_ID=0 \
-e ZOO_PORT_NUMBER=2181 \
-e ALLOW_ANONYMOUS_LOGIN='yes' \
-e ZOO_SERVERS=zookeeper:2888:3888::0 \
-p 2181:2181 \
-p 2888:2888 \
-p 3888:3888 \
docker.io/bitnami/zookeeper:3.8
```

Kafka

The following command will create a container running kafka

- Docker
- Podman

Add your content here...

```
docker create --network uvuyo \
--name kafka \
--volume kafka_data:/bitnuami \
-e KAFKA_BROKER_ID=0 \
-e KAFKA_CFG_ZOOKEEPER_CONNECT=zookeeper:2181 \
-e KAFKA_CFG_LISTENER_SECURITY_PROTOCOL_MAP=INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT \
-e KAFKA_CFG_INTER_BROKER_LISTENER_NAME=INTERNAL \
-e KAFKA_CFG_LISTENERS=INTERNAL://:29029,EXTERNAL://:9092 \
```

```
-e KAFKA_CFG_ADVERTISED_LISTENERS=INTERNAL://kafka:29029,EXTERNAL://kafka0:9092 \
-e KAFKA_CFG_OFFSETS_TOPIC_REPLICATION_FACTOR='1' \
-e KAFKA_MIN_INSYNC_REPLICAS='1' \
-e KAFKA_CFG_DEFAULT_REPLICATION_FACTOR='1' \
-e KAFKA_CFG_NUM_PARTITIONS='1' \
-e ALLOW_PLAINTEXT_LISTENER='yes' \
-p 9092:9092 \
bitnami/kafka:3.3
```

```
podman create --network uvuyo \
--name kafka \
--volume kafka_data:/bitnuami \
-e KAFKA_BROKER_ID=0 \
-e KAFKA_CFG_ZOOKEEPER_CONNECT=zookeeper:2181 \
-e KAFKA_CFG_LISTENER_SECURITY_PROTOCOL_MAP=INTERNAL:PLAINTEXT,EXTERNAL:PLAINTEXT \
-e KAFKA_CFG_INTER_BROKER_LISTENER_NAME=INTERNAL \
-e KAFKA_CFG_LISTENERS=INTERNAL://:29029,EXTERNAL://:9092 \
-e KAFKA_CFG_ADVERTISED_LISTENERS=INTERNAL://kafka:29029,EXTERNAL://kafka0:9092 \
-e KAFKA_CFG_OFFSETS_TOPIC_REPLICATION_FACTOR='1' \
-e KAFKA_MIN_INSYNC_REPLICAS='1' \
-e KAFKA_CFG_DEFAULT_REPLICATION_FACTOR='1' \
-e KAFKA_CFG_NUM_PARTITIONS='1' \
-e ALLOW_PLAINTEXT_LISTENER='yes' \
-p 9092:9092 \
docker.io/bitnami/kafka:3.3
```

Elasticsearch

The following command will create a container running the elasticsearch database service

- Docker
- Podman

```
docker create --network uvuyo \
--name elastic \
--volume elastic_data:/usr/share/elasticsearch/data \
-e xpack.security.enabled=false \
-e discovery.type=single-node \
-p 9200:9200 \
-p 9300:9300 \
--ulimit memlock=-1:-1 \
--ulimit nofile=65536:65536 \
```

```
--cap-add=IPC_LOCK \
elasticsearch:8.10.2
```

```
podman create --network uvuyo \
--name elastic \
--volume elastic_data:/usr/share/elasticsearch/data \
-e xpack.security.enabled=false \
-e discovery.type=single-node \
-p 9200:9200 \
-p 9300:9300 \
--ulimit memlock=-1:-1 \
--ulimit nofile=65536:65536 \
--cap-add=IPC_LOCK \
docker.io/elasticsearch:8.10.2
```