Sandbox Instance with All-In-One Image - no Load Balancer

Overview

In our steps for setting up a Yellowfin sandbox, Yellowfin runs on the Kubernetes cluster, with 6GB of allocated RAM.

Before you start, use the following table of information to choose between LoadBalancer and NodePort.

Type	Description	Typical usage
LoadBalancer	This setting instructs Kubernetes to interact with a cloud provider to provision a load balancer to route traffic to the Yellowfin service.	Cloud
NodePort	Publish a port on the Kubernetes cluster that can be used to communicate with the Yellowfin instance. This setting can be used when an environment can't auto-provision load balancers.	On-premises

See the official Kubernetes documentation on the service types of LoadBalancer and NodePort for more information.

In our example, we've instructed Kubernetes to deploy Yellowfin behind a load balancer using the **type: LoadBalancer** attribute in the service definition, but we've provided details for NodePort too.

To deploy a self-contained instance with these defaults, follow the steps below.

1. Ensure Kubernetes is running
2. Copy the following text and paste it into your preferred text editor:

```
---
### Yellowfin All in one Service ###
apiVersion: v1
kind: Service
metadata:
  name: yellowfin-all-in-one
spec:
  ports:
  - name: "web"
    port: 8080
    targetPort: 8080
  selector:
    app: yellowfin-all-in-one
  type: LoadBalancer
status:
  loadBalancer: {}
---
### Yellowfin All in one Deployment ###
apiVersion: apps/v1
kind: Deployment
metadata:
  namespace: default
  labels:
    app: yellowfin-all-in-one
  name: yellowfin-all-in-one

spec:
  replicas: 1
  selector:
    matchLabels:
      app: yellowfin-all-in-one
  template:
    metadata:
      labels:
        app: yellowfin-all-in-one
    spec:
      containers:
        - env:
          - name: APP_MEMORY
            value: "6144"
          name: yellowfin-all-in-one
          image: yellowfinbi/yellowfin-all-in-one:<RELEASE_VERSION_GOES_HERE>
          ports:
            - name: web
              containerPort: 8080
```

3. Update `<RELEASE_VERSION_GOES_HERE>` with your release version (eg, 9.6.0)
4. If you don't wish to provision a load balancer via a cloud provider, replace **LoadBalancer** with **NodePort** for **Spec.Type**, then remove the line for **Service.Spec.Status**
5. Save the text to a YAML file called **yellowfin-all-in-one.yml**
6. Run the following command in a terminal to deploy Yellowfin and execute it in the background:
   `kubectl apply -f yellowfin-all-in-one.yml`
7. Start Yellowfin by typing your host URL on port 8080.

Section navigation
Current topic - Install in a Container

The page is part of the Install in a Container topic contains the following pages, split by Docker and Kubernetes:

Sandbox Instance with All-In-One Image - no Load Balancer

Install on Premises

Sandbox Instance with All-In-One Image - no Load Balancer

- Installation Steps

Install in the Cloud

Install in the Cloud

- Yellowfin for AWS
- Yellowfin for Azure
- Yellowfin for Google Cloud Platform

Install in a container

Install in a Container

- Docker
- Kubernetes
- Upgrading Yellowfin Container Deployment

Deploy Yellowfin

Deploy Yellowfin

- Logs and Logging
- Yellowfin Directory Structure
- User Welcome

Advanced Deployments

Advanced Deployments

- Clustering Guide
- Yellowfin Server Specification
- Automate Yellowfin Deployment on Linux
- SAML Bridge

- Standalone
  Configuration Tools