Objective

The objective of this workshop is to

a. Deploy an instance of Code Server and expose the application thru Ingress
b. Create a volume and mount the volume to Code Server so that your work can be persisted across redeployment
c. Use GitHub OAuth2 to protect with Code Server application by leveraging Ingress-Nginx's support of external authentication.

Setup

a. Assumed that you have a Docker Hub account. If not create one at https://hub.docker.com
b. This workshop will be using the following images
   i. https://hub.docker.com/r/linuxserver/code-server
   ii. https://hub.docker.com/r/bitnami/oauth2-proxy

Workshop

In this workshop you will be deploying Code Server, a browser-based version of Visual Studio Code.

The workshop consists of 2 parts

1. Deploy a Code Server without password configured
2. Deploy a OAuth2 proxy to authenticate users

Deploy Code Server

The following creates a Code Server container

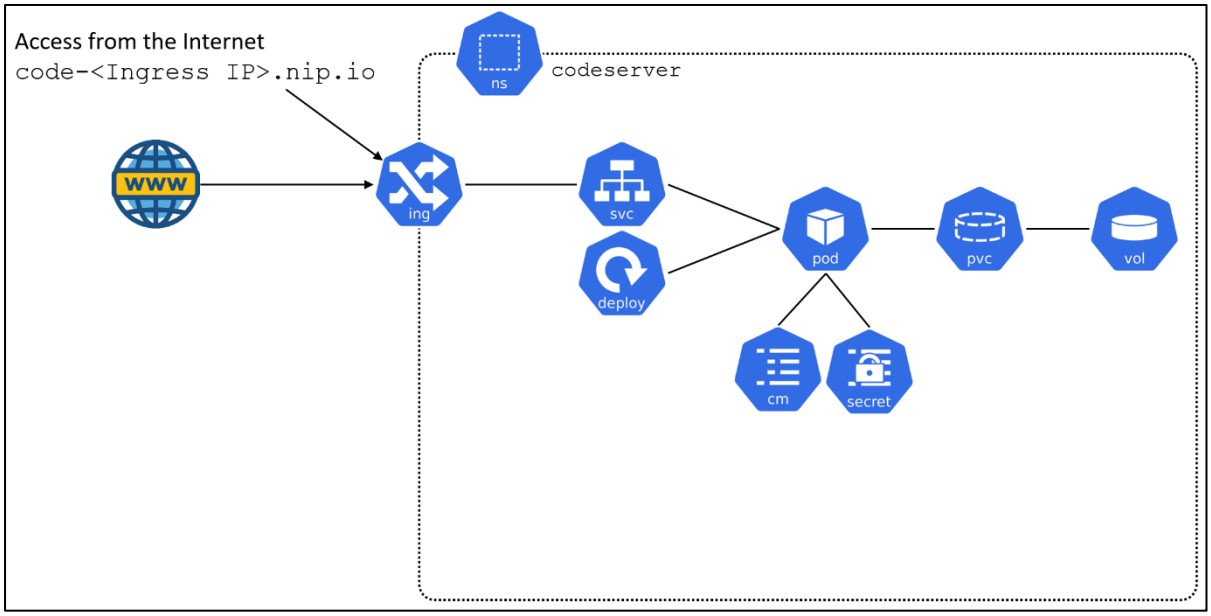
```
docker run -d \
  --name=code-server \
  -e TZ=Asia/Singapore \
  -e DEFAULT_WORKSPACE=/config/workspace \
  -p 8443:8443 \
  -e PROXY_DOMAIN=code-<IP>.nip.io \
  -v workspace-vol:/config/workspace \
  linuxserver/code-server:amd64-<version tag>
```

where

- `TZ` is the timezone
- `PROXY_DOMAIN` sets the domain for the Code Server instance
- `DEFAULT_WORKSPACE` sets the directory to use

See **Parameters** section in https://hub.docker.com/r/linuxserver/code-server for other parameter settings.

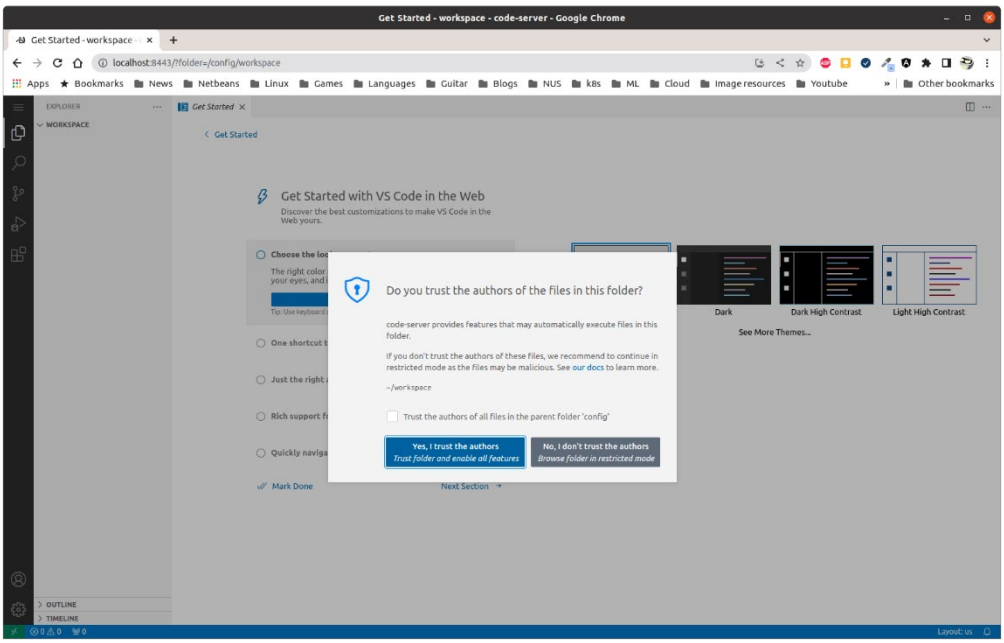
Deploy Code Server according to the following Kubernetes deployment architecture diagram.



The following are some requirements of the deployment

- Each deployment of Code Server should be isolated from other instances
- There should be only a single instance of Code Server running consuming 256 MB of memory and 200 milli cores of CPU. This should be the upper and lower limit
- User's work should be safe and recoverable if the Code Server pod crashes or is rescheduled to another node
- The entire deployment should be flexible to accommodate configuration changes
- Assign a domain name to each instance of Code Server. Use `nip.io` to simulate a domain name eg `code-<loadbalancer ip>.nip.io`

Test your Deployment by opening the assigned domain name on your browser and you should see the following



Deploy OAuth2 Proxy

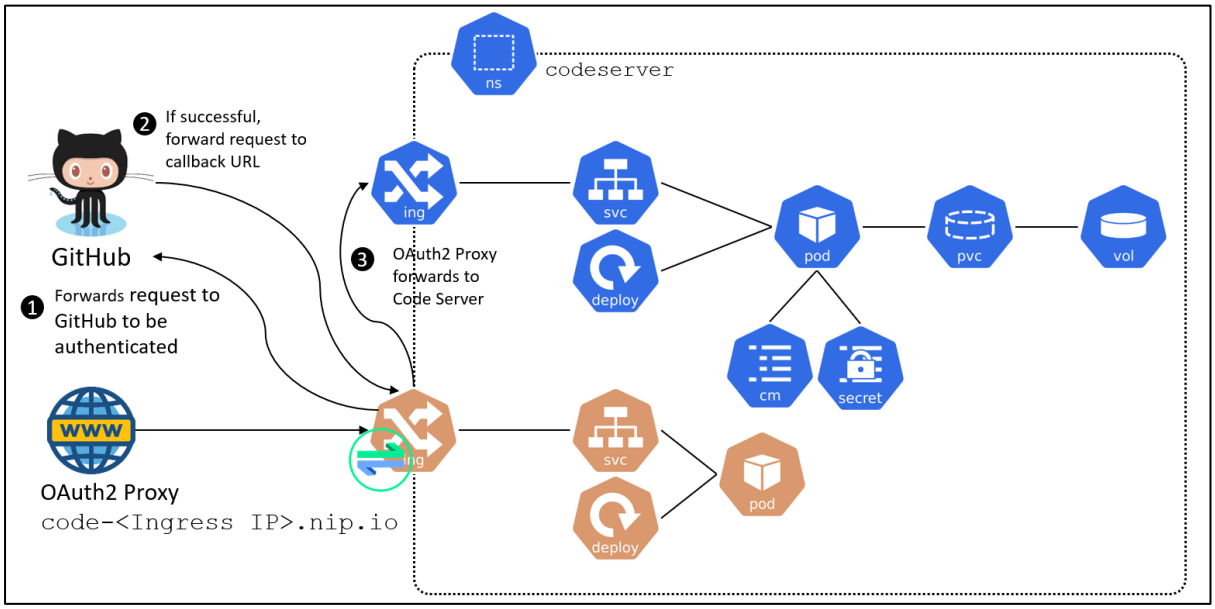
OAuth2 Proxy is a reverse proxy that provides authentication using configurable list of providers like Google, GitHub, GitLab, etc.

The OAuth2 Proxy works by intercepting unauthenticated request to a domain; when the user have been authenticated with the OAuth2 provider, the request will be forwarded to the domain.

For this workshop, we will leverage Ingress Nginx proprietary feature to setup an OAuth2 authentication for the provisioned Code Server. See https://kubernetes.github.io/ingress-nginx/examples/auth/oauth-external-auth

For this workshop we will be using GitHub.

The following diagram illustrates how Ingress Nginx works with OAuth2 Proxy



Use the following steps to setup an OAuth2 Proxy to protect the Code Server

Create a GitHub OAuth2 Application

Go to GitHub and provision an OAuth2 application. Configure the following

- Homepage URL – this should be the Code Server domain name eg. `https://code-<loadbalancer ip>.nip.io`
- Authorization callback URL – the endpoint for GitHub to invoke when the authentication is successful. In this workshop, this will be the OAuth2 Proxy endpoint. Use `https://code-<loadbalancer ip>.nip.io/oauth2`

Note the Client ID and the Client secret; you will need to generate a Client secret.

Deploy OAuth2 Proxy

The following Docker command creates a OAuth2 Proxy using GitHub as the OAuth2 provider

```
docker run -d -p 4180:4180 bitnami/oauth2-proxy:7.2.1 \
  --provider=github \
  --email-domain=* \
  --upstream=file:///dev/null \
  --http-address=0.0.0.0:4180 \
  --cookie-secret=<32-byte seed string for secure cookie> \
  --client-id=<GitHub client ID> \
  --client-secret=<GitHub client secret>
```

The cookie secret is a random hex string of 32 bytes long.

See https://oauth2-proxy.github.io/oauth2-proxy/docs/configuration/overview/ for other command line options

All command line options can be replaced with an equivalent environment variable by prefixing it with `OAUTH2_PROXY_` and replacing dash (−) with underscore (_) eg. `OAUTH2_PROXY_CLIENT_ID`.

Write a Deployment, Service and Ingress resources to deploy an instance of OAuth2 Proxy.

The Ingress should be created according to the following

- Host/domain name name should be the same as Code Server
- The path should be `/oauth2`

Update Code Server Ingress to Use OAuth2 Proxy

Add the following annotations to Code Server's Ingress resource

```
ngnix.ingress.kubernetes.io/auth-url:
"https://$host/oauth2/auth"
```

```
ngnix.ingress.kubernetes.io/auth-signin:
"https://$host/oauth2/ start?rd=$escaped_request_uri"
```

See https://kubernetes.github.io/ingress-nginx/examples/auth/oauth-external-auth/#key-detail

Test the OAuth2 by navigating to the Code Server domain; you should now be prompted with the GitHub login. Once you have successfully authenticated with GitHub, you will be redirected to Code Server.

Warning: the way this workshop has configured GitHub OAuth2 is NOT SECURE because it allows anyone with a valid GitHub login to access your Code Server. If you are planning on using GitHub consider creating an GitHub organization and restrict access to those within that organization.

Submission

Create a Git repo for this course if you have not done so. Clone the Git repo. This repo will be used for all the assignment for this course. This should the same repos as you used for previous workshops.

Create a directory called `workshop03` inside your repo. Placed all the file for this workshop inside `workshop03` directory.