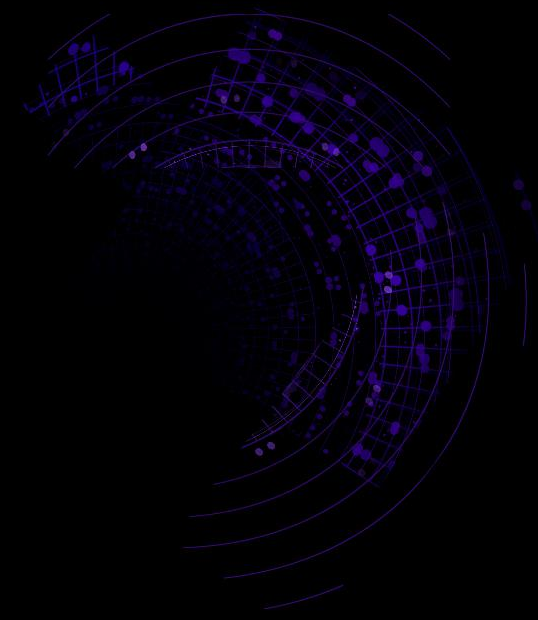




APACHE
APISIX



Show case

How to secure an app without any dev

Leverage the power of Keycloak and Apisix :)



APACHE
APISIX



Hello :)

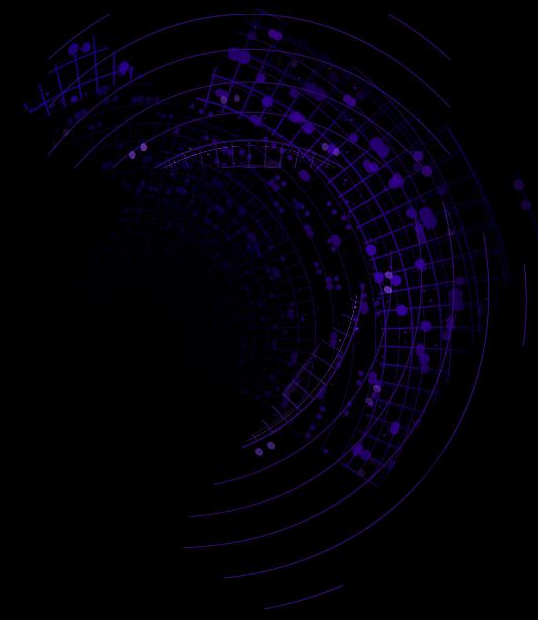
Using kubernetes , Apisix, Apisix Ingress Controller and Keycloak , learn how to secure a webapp without having to change a bit to your application.

This leverage the powerful connection with Apisix and Keycloak

Jean-Philippe Gouin
Contributor on Apache Apisix



APACHE
APISIX



CONTENT

01 Expectation

02 Architecture

03 How to secure an app /w Keycloak and Apisix

04 Demo !



APACHE
APISIX



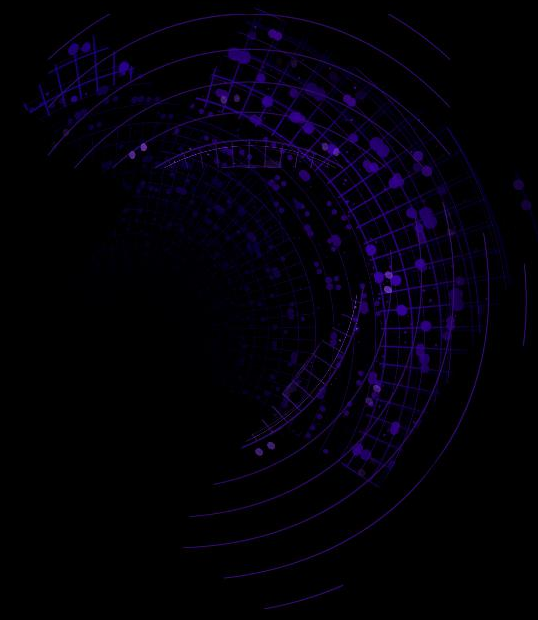
01

Expectation

What is the approach and why use Apisix



APACHE
APISIX



Expectation

Outcome :

Secure a web application and it's backend

Assumptions :

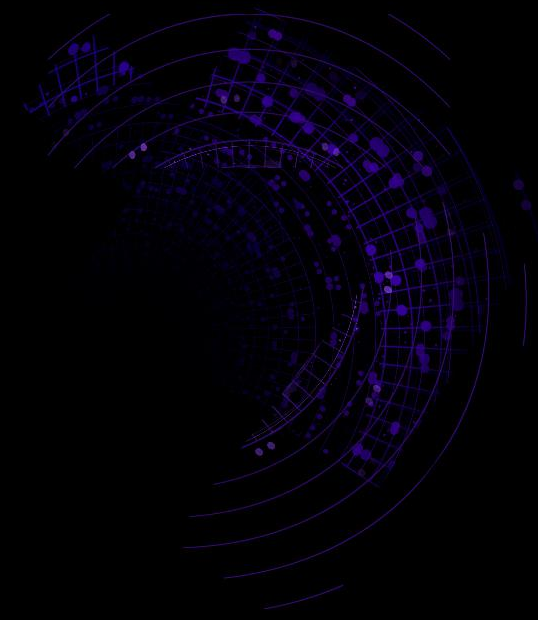
- When reusing application or legacy development , you can't always update the code and include an in depth security using modern protocol like OIDC.
- You don't want the developer to focus too much on the security and rather on the micro service architecture and rely on an external solution to implement the security
- You want to build a MVP and not put to much effort on the security at first , but still don't want to expose the application
- You want an easy solution to maintain and don't necessarily have the team for

Key result :

I have an application running and secured



APACHE
APISIX



02 Architecture

Preparation of the Demo
Walkthrough the configuration



APACHE
APISIX

Architecture

Description

This demo run on a Kind cluster with 80 and 443 exposition bound on Apisix.

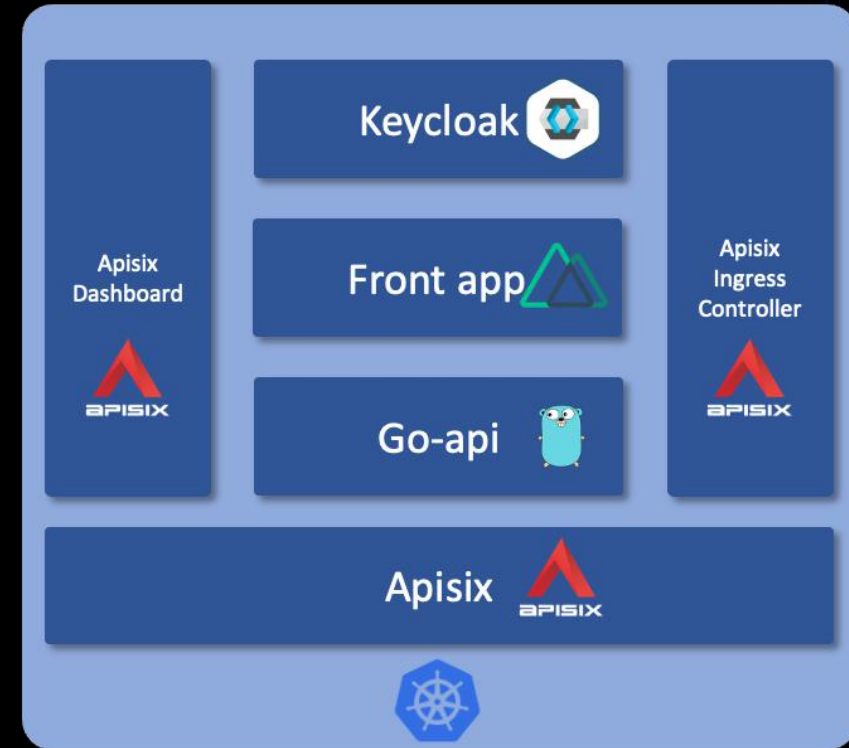
Apisix is deployed using the helm chart /w :

- Apisix
- Apisix Ingress Controller
- Apisix Dashboard

A mockup app is deployed

- A front end to manage books and authors in a library
- A golang backend to store informations

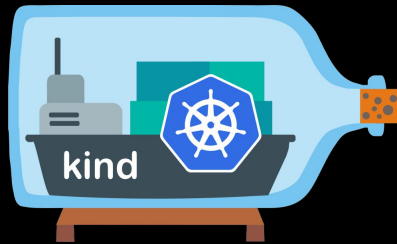
Keycloak is deployed using the Bitnami helm chart.



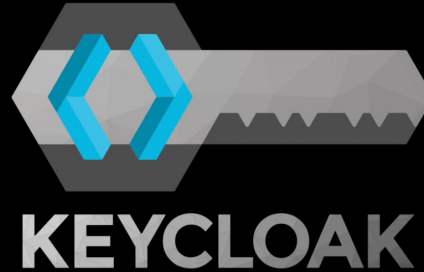


APACHE
APISIX

Walkthrough the configuration



Standard configuration with 3 nodes and extraPortMapping to expose Apisix




KEYCLOAK

Setup Client poc-apisix
Users creation Franck and John and 2 groups author and editor
Setup the authorization on the client

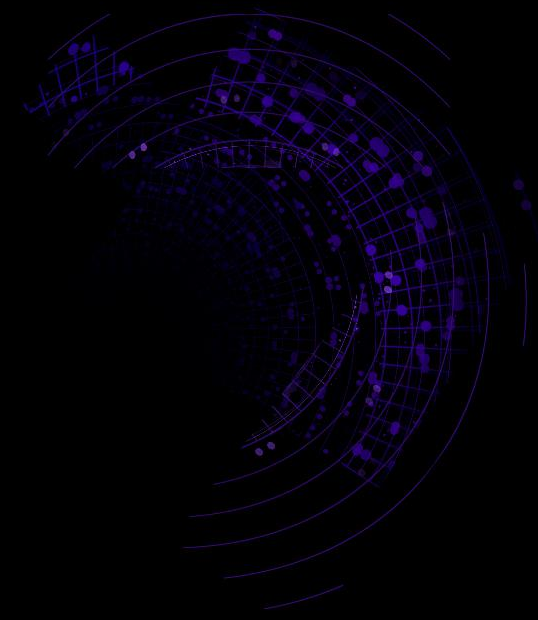


Standard deployment with NodePorts exposition mapped on Kind configuration
Setup TLS certificate to handle HTTPS

All resources available  [here](#)



APACHE
APISIX



03

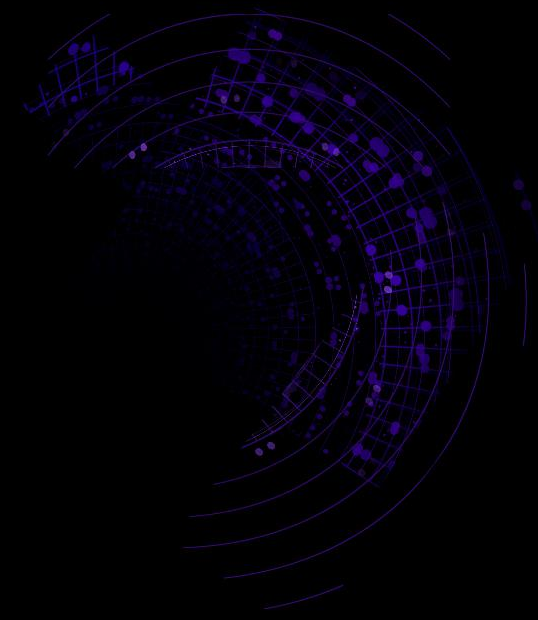
Demo !

1st scenario -> Expose the application

2nd scenario -> Secure the application



APACHE
APISIX



Expose the application

First thing first, we need to expose the application to outside our cluster.

We are using Apisix Ingress Controller to help us with that.

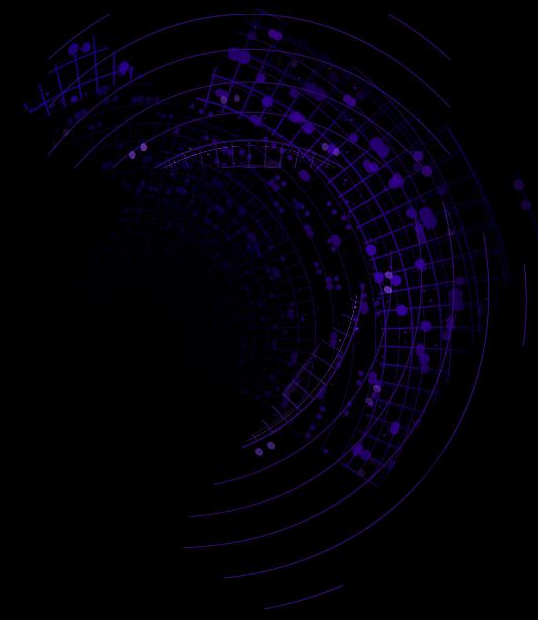
Instead of having to curl Apisix and create the **Upstream** and the **Route**, we are going to use some **CRD** from Apisix Ingress Controller that will do the magic for us !

To allow the connection between our backend and front-end we are also going to use the **cors** plugin

CORS = Cross Origin Request Sharing -> our front end is accessing a back-end that is not provided by the same server.



APACHE
APISIX



Secure the application

It's time to secure the access to the application !

For that we are going to protect our front-app using the **openid-connect** plugin that will handle the generation of the token against Keycloak by **keycloak** UI to login.

Then we are going to protect our go-api using the amazing **authz-keycloak** plugin.

The key here is to use **http_method_as_scope** along **lazy_load_paths** to leverage the Authorization tab from Keycloak.

We are now going to play a little with the app and the Authorization configuration :)



APACHE
APISIX



Reach out

Feel free to reach out if you have any question :)

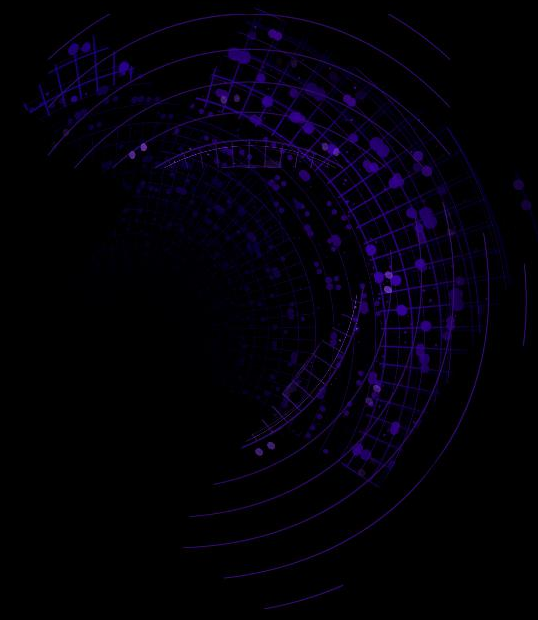
LinkedIn : <http://www.linkedin.com/in/jean-philippe-gouin-2492aa16a>

Github : <https://github.com/jp-gouin?tab=repositories>

All resources available here :
<https://github.com/jp-gouin/apisix-summit-material>



APACHE
APISIX



THANKS

APACHE APISIX CONNECTS THE WORLD