Open Infrastructure FOUNDATION

INDONESIA OpenInfra Days

Securing Kubernetes **Service Using Oauth2-proxy**

Talk #1 IOID2021

Dewa Putu Prasta Maha Gangga

Cloud Engineer at **Boer Technology**

Jakarta, August 21, 2021

About Me

Dewa Putu Prasta Maha Gangga

Cloud Engineer at Boer Technology

@prastamaha

Prasta Maha

prasta@btech.id

Agenda

- What is oauth2-proxy ?
- How does it work?
- Implement oauth2-proxy on K8S
- How to configure?
- Quick Demo

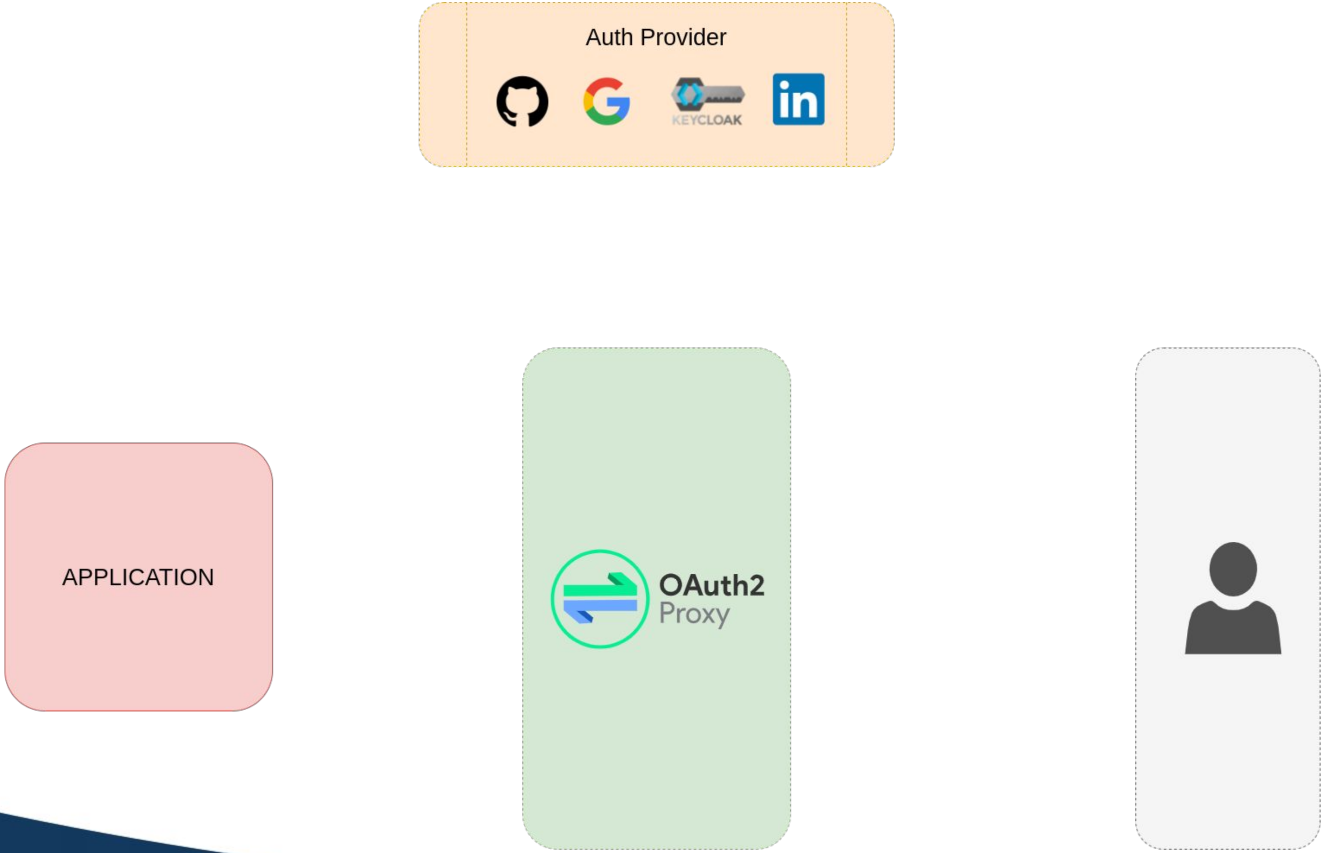
What is oauth2-proxy ?

oauth2-proxy

A reverse proxy that provides authentication using Providers (Google, GitHub, and others) to validate accounts by email, domain or group.

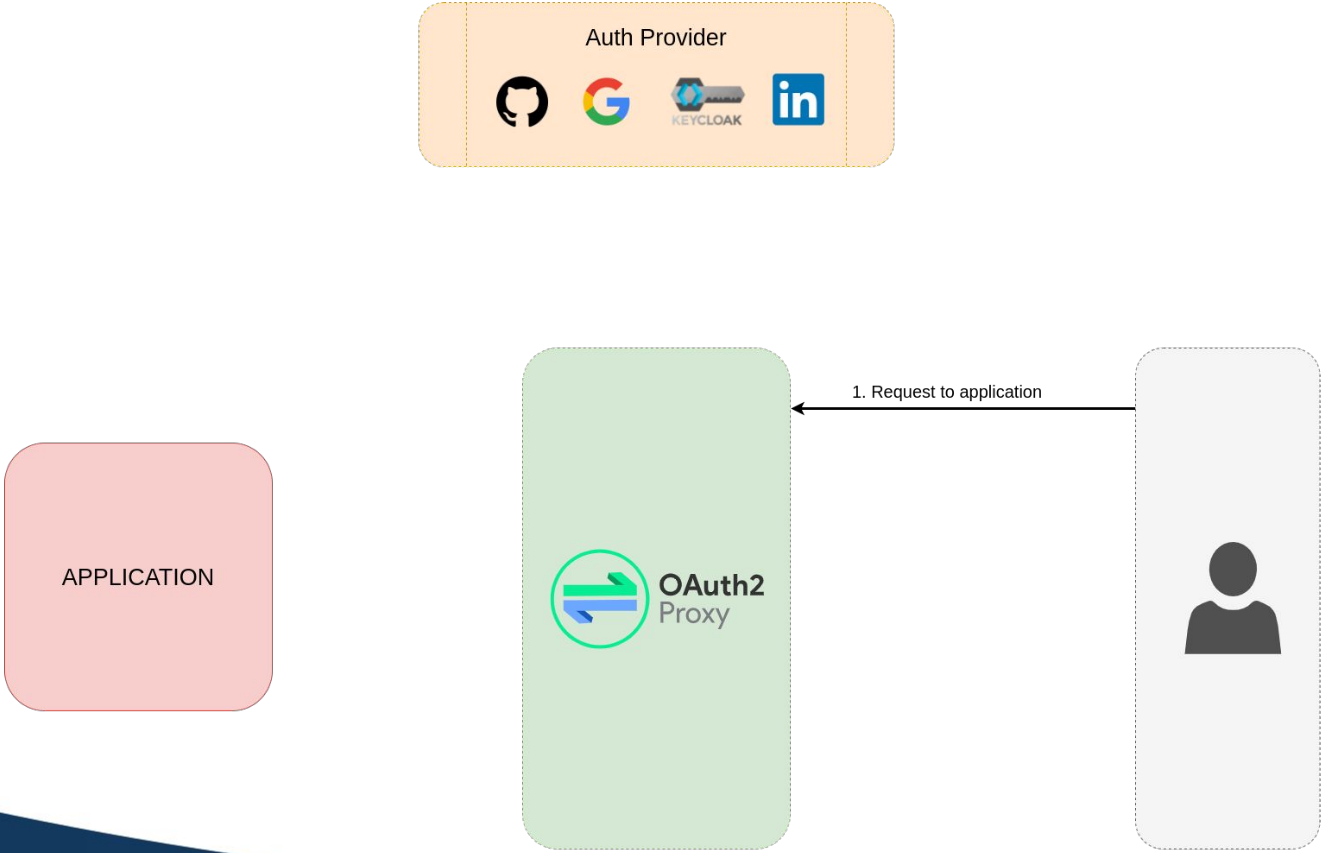
Source: https://oauth2-proxy.github.io/oauth2-proxy/

How it's works?



Auth Provider

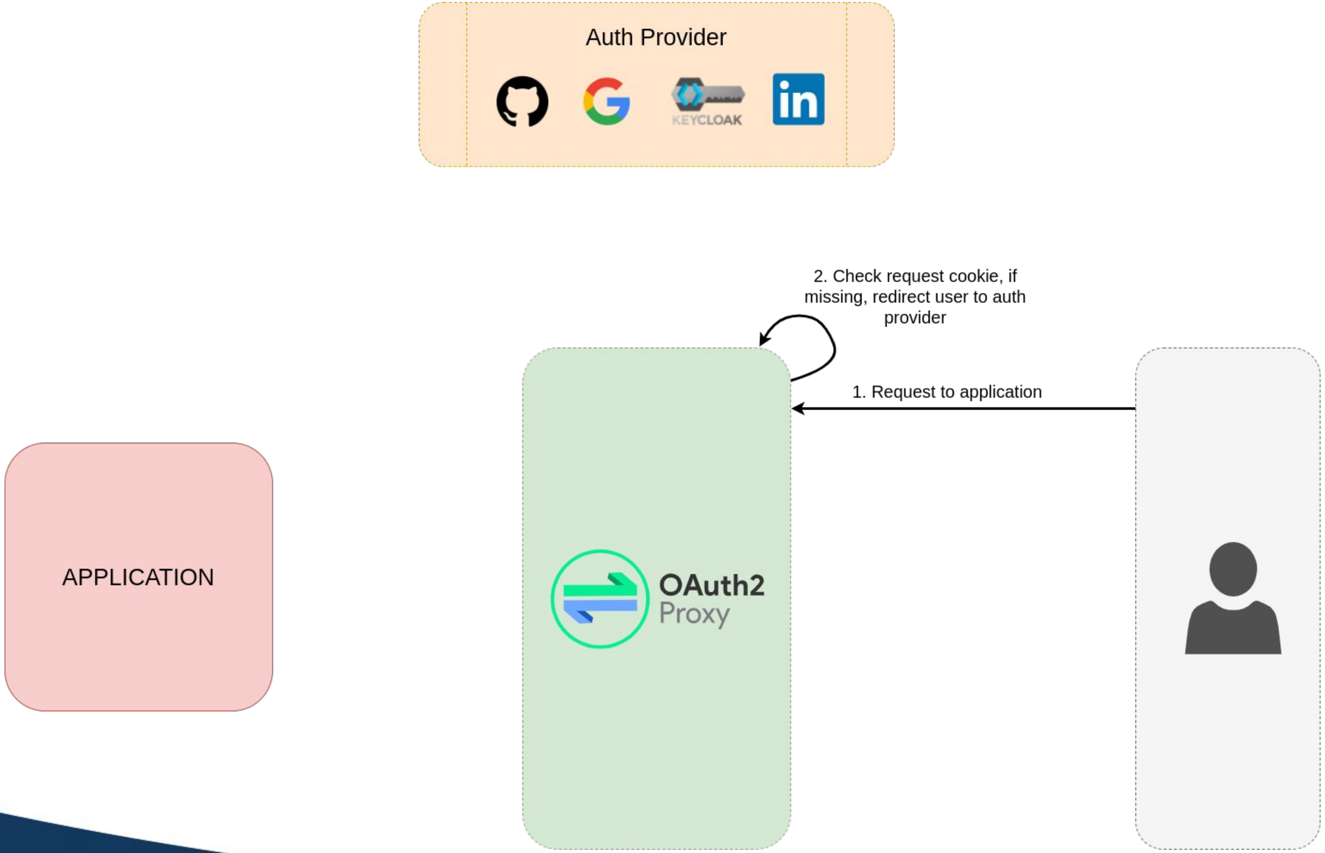
APPLICATION

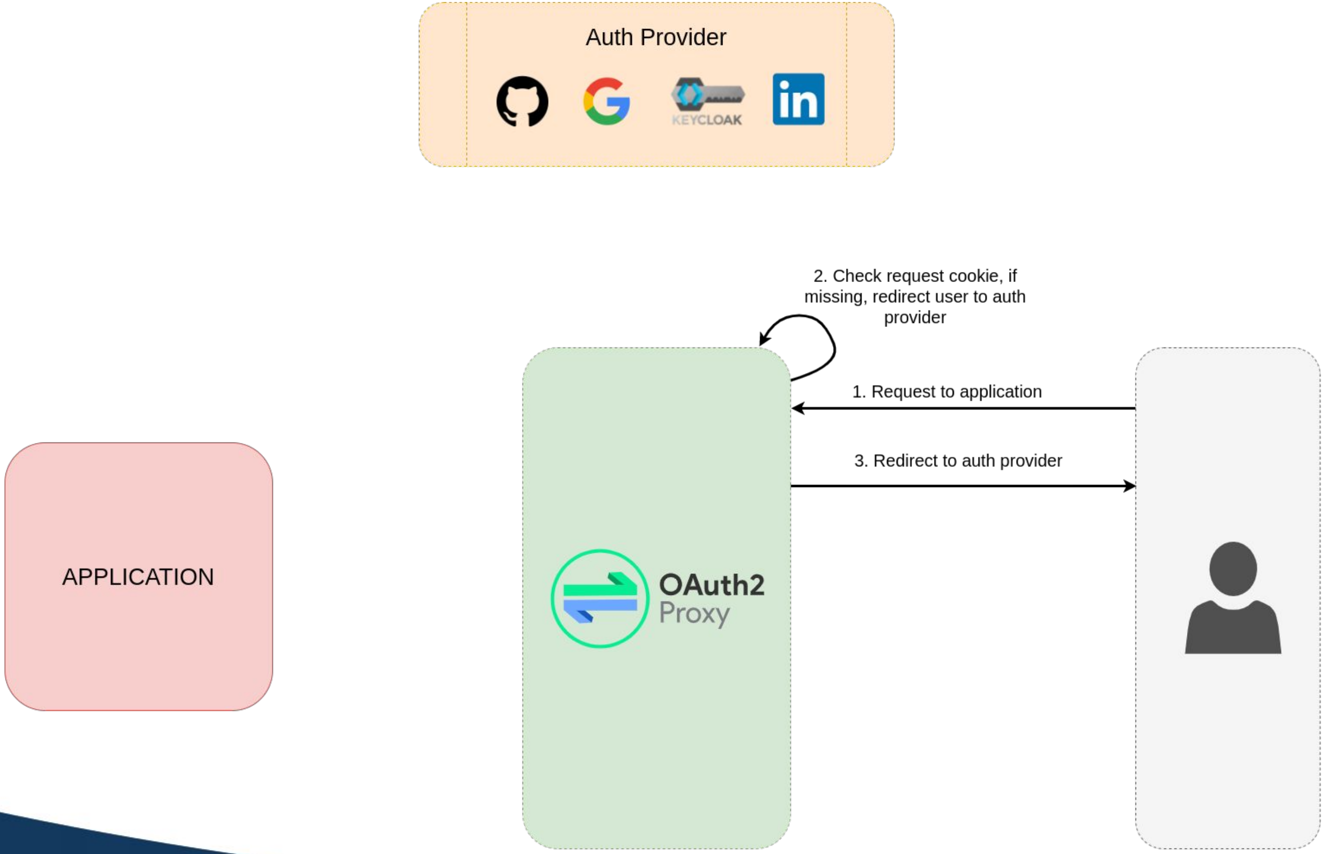


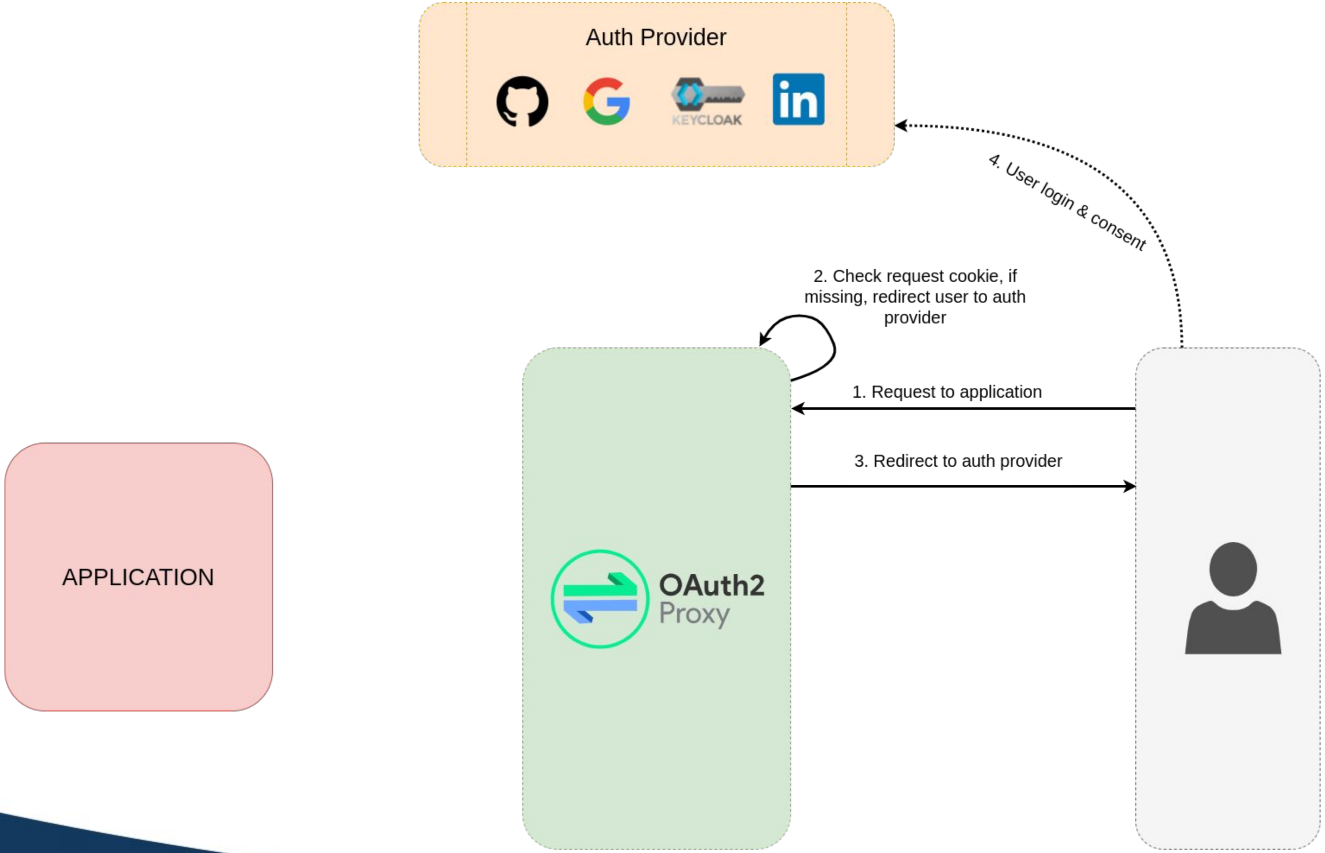
Auth Provider

1. Request to application

APPLICATION


Auth Provider
KEYCLOAK
in
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
APPLICATION
OAuth2
Proxy


Auth Provider
KEYCLOAK
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
APPLICATION
OAuth2 Proxy



Auth Provider

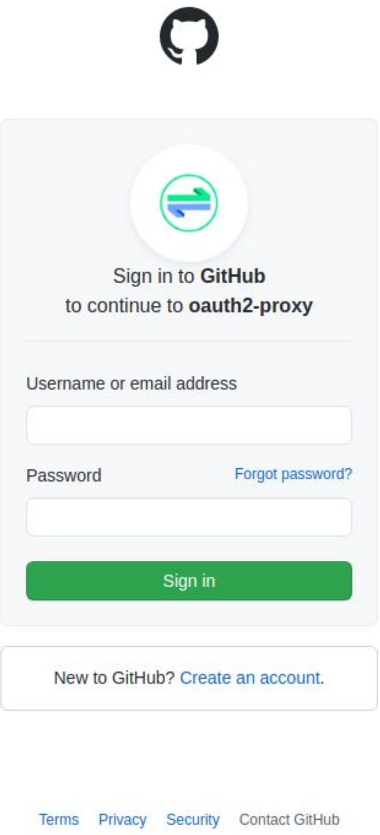
2. Check request cookie, if missing, redirect user to auth provider

1. Request to application

3. Redirect to auth provider

4. User login & consent

Login Page



Sign in to **GitHub**
to continue to **oauth2-proxy**

Username or email address

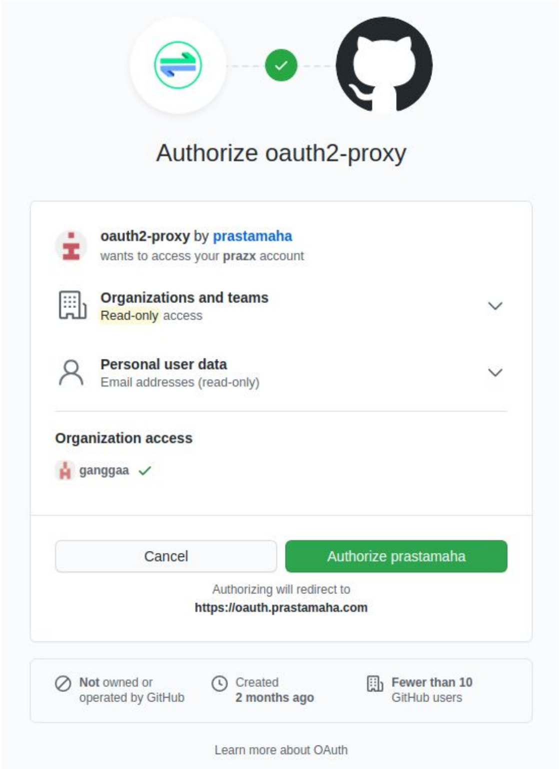
Password

Forgot password?

Sign in

New to GitHub? Create an account.

Terms Privacy Security Contact GitHub



Authorize oauth2-proxy

oauth2-proxy by **prastamaha**
wants to access your **prazx** account

Organizations and teams
Read-only access

Personal user data
Email addresses (read-only)

Organization access

ganggaa ✓

Cancel

Authorize prastamaha

Authorizing will redirect to
https://oauth.prastamaha.com

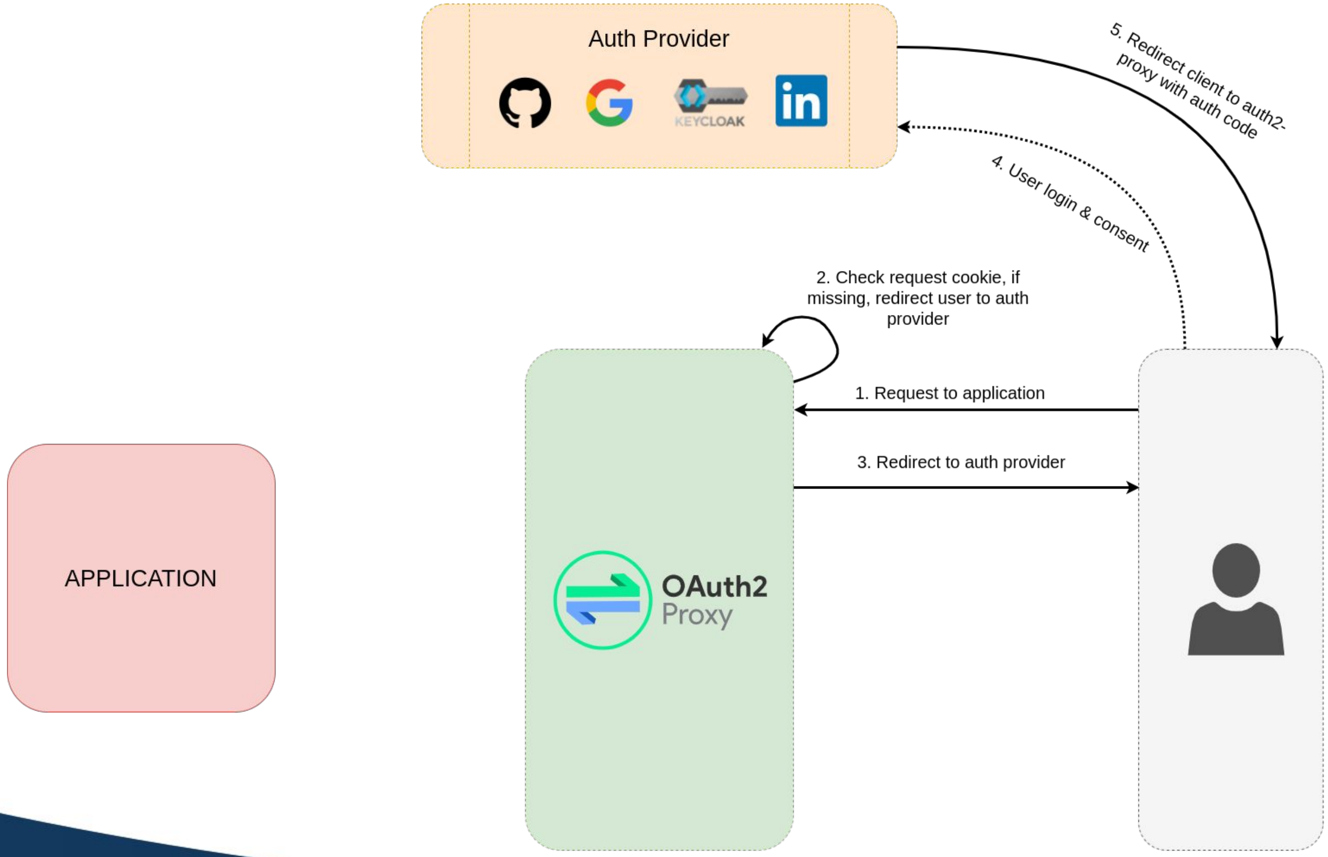
Not owned or operated by GitHub

Created **2 months ago**

Fewer than 10 GitHub users

Learn more about OAuth

Consent Page



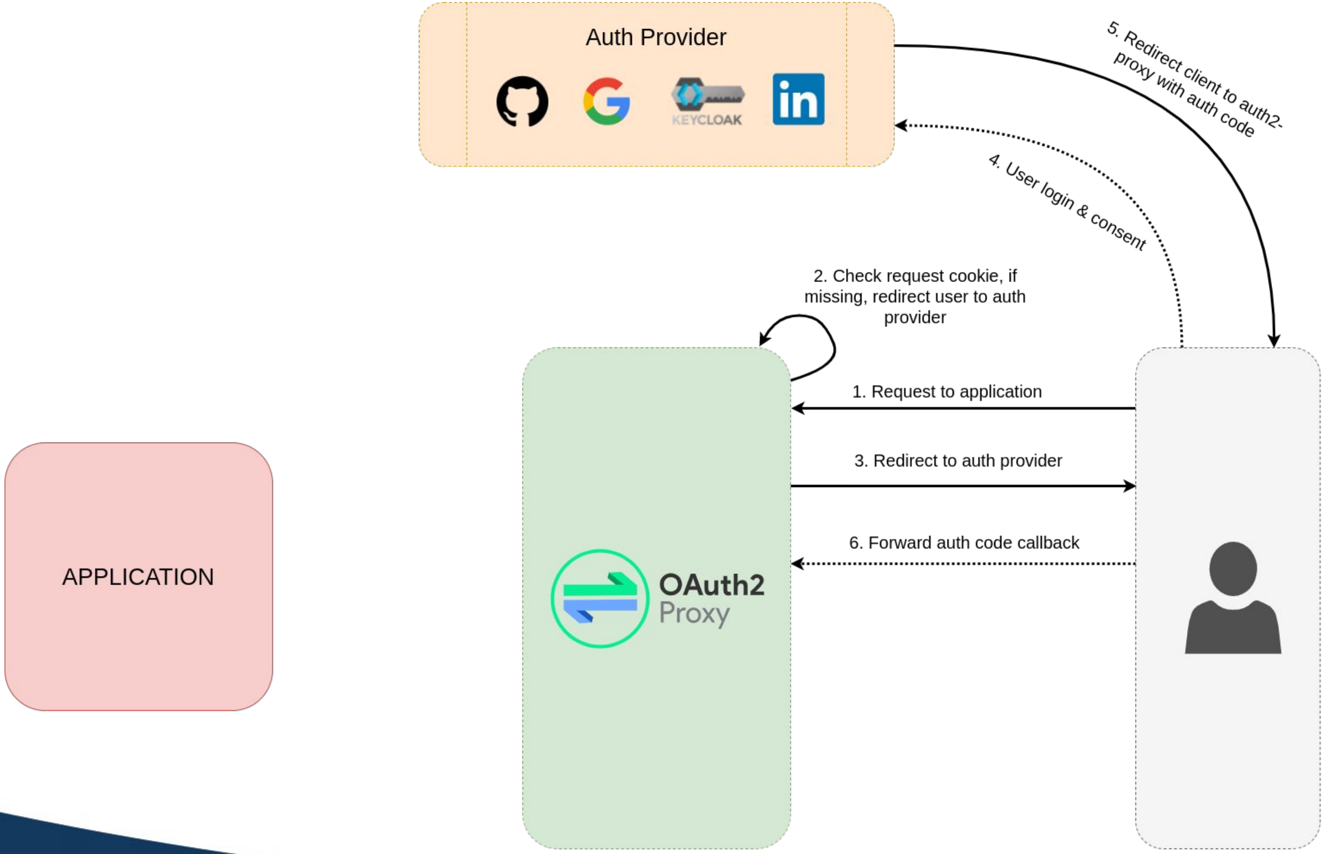
1. Request to application

2. Check request cookie, if missing, redirect user to auth provider

3. Redirect to auth provider

4. User login & consent

5. Redirect client to auth2-proxy with auth code



1. Request to application

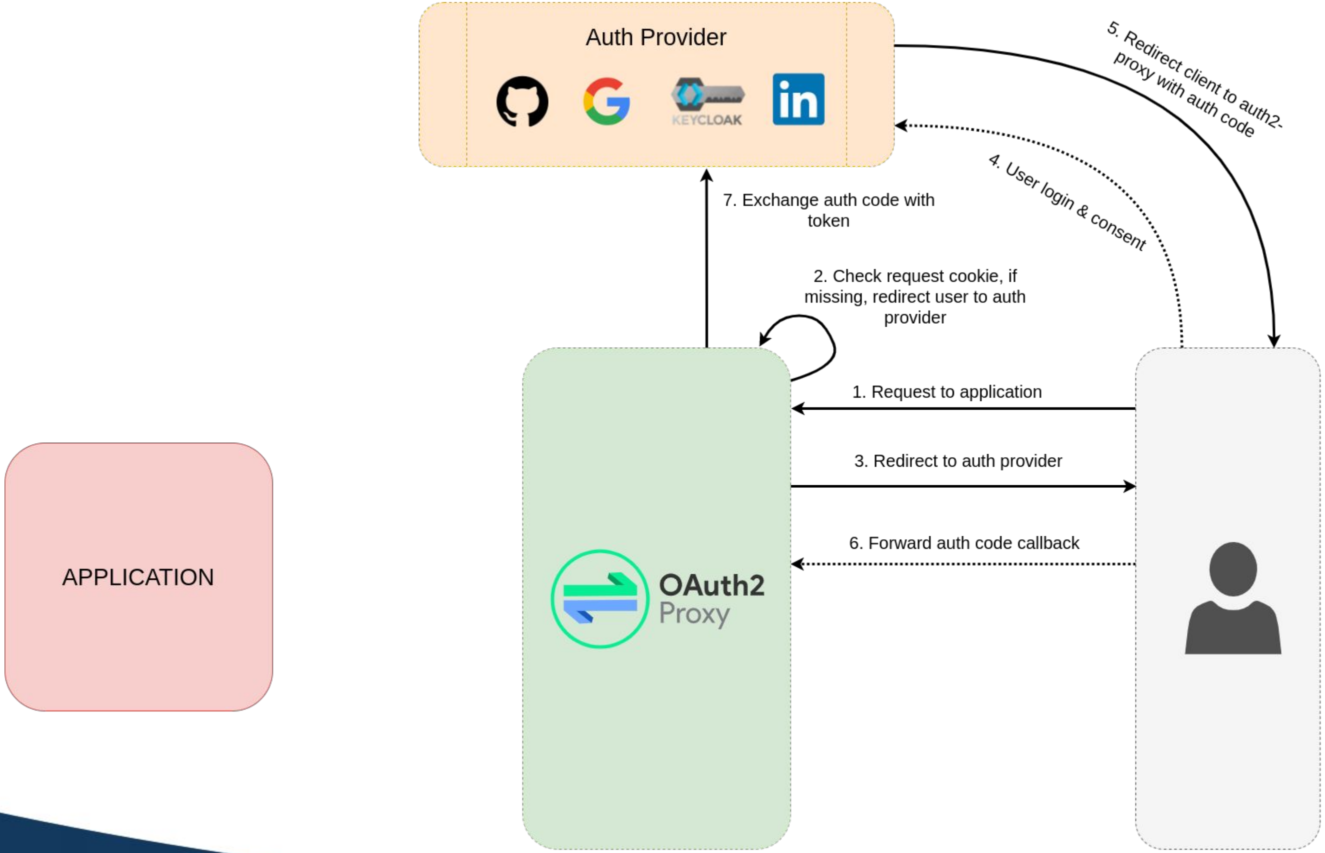
2. Check request cookie, if missing, redirect user to auth provider

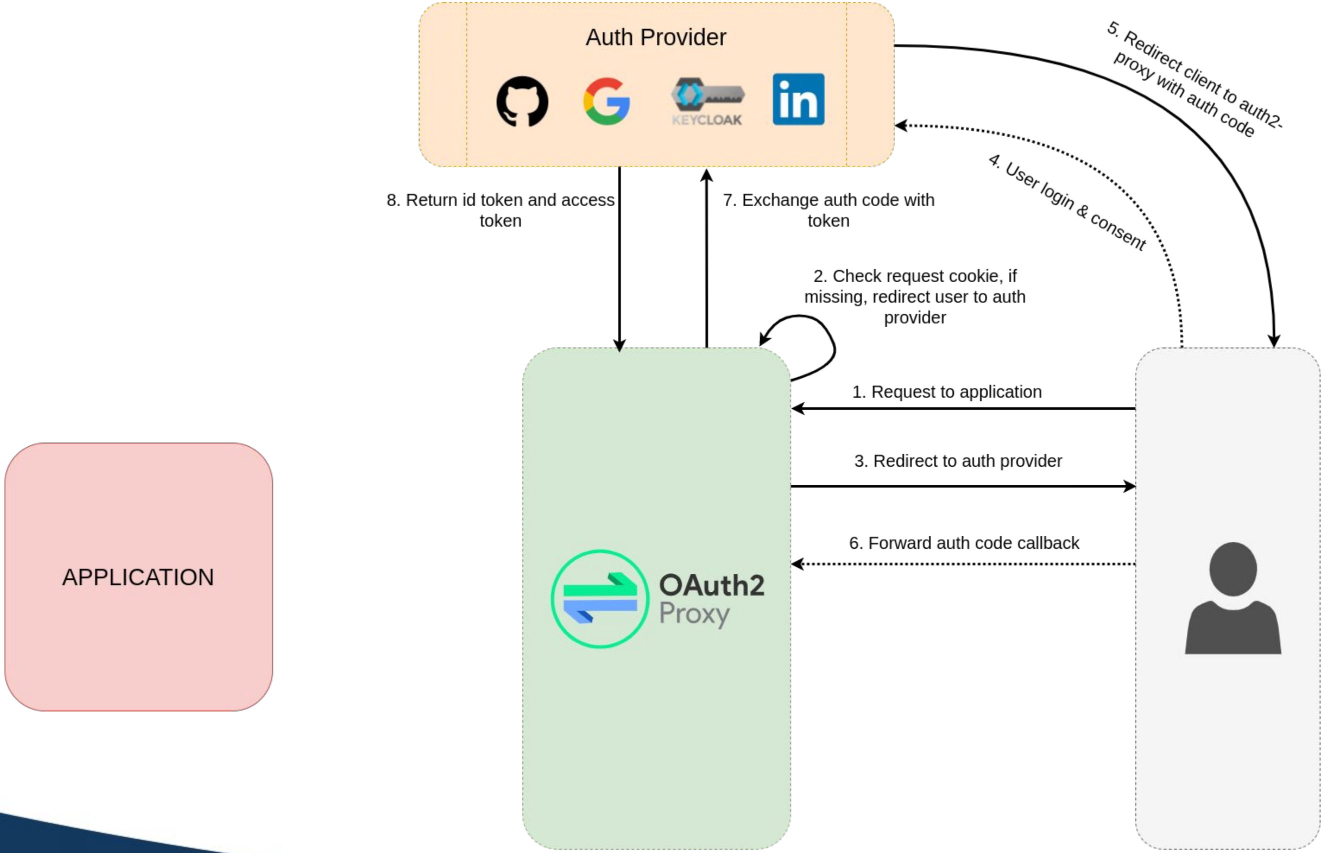
3. Redirect to auth provider

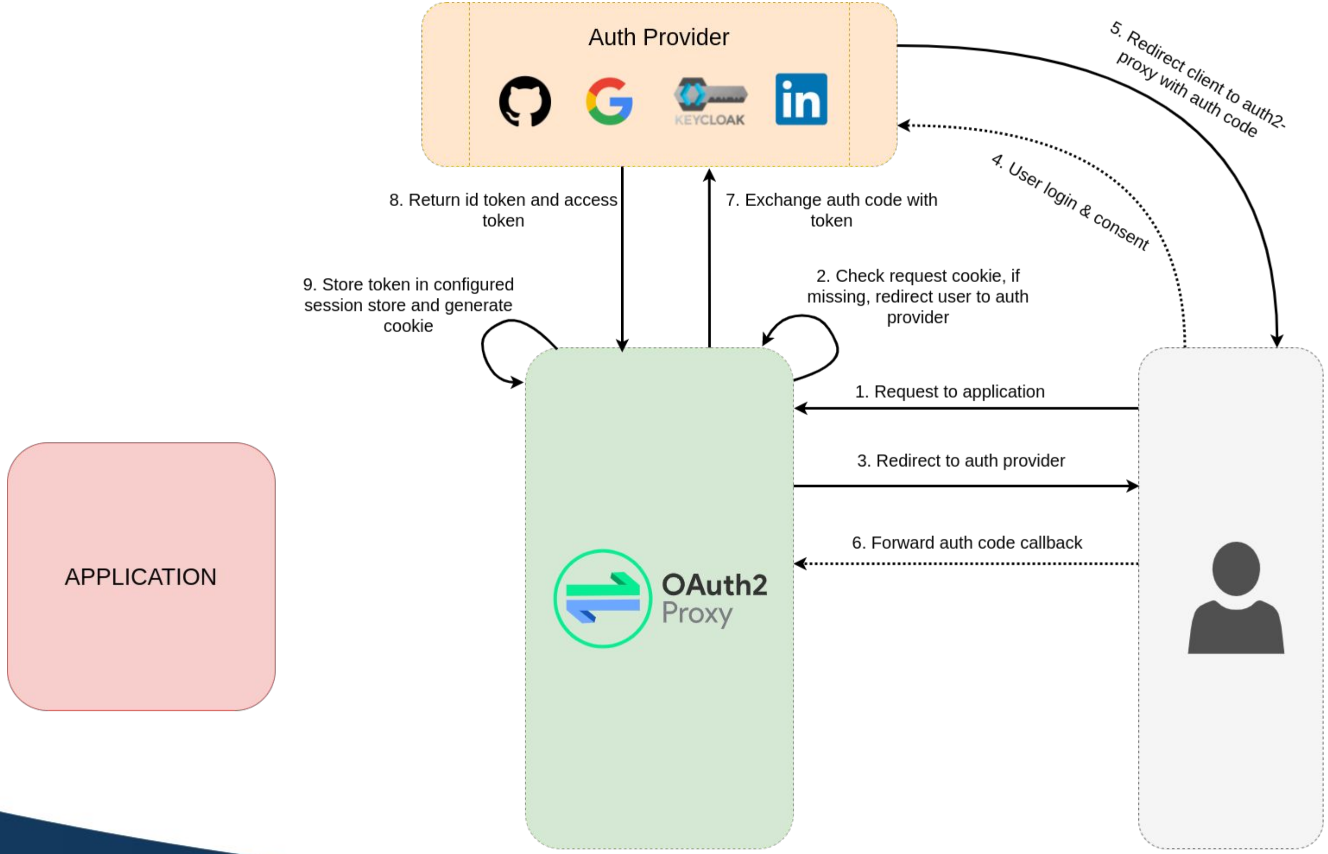
4. User login & consent

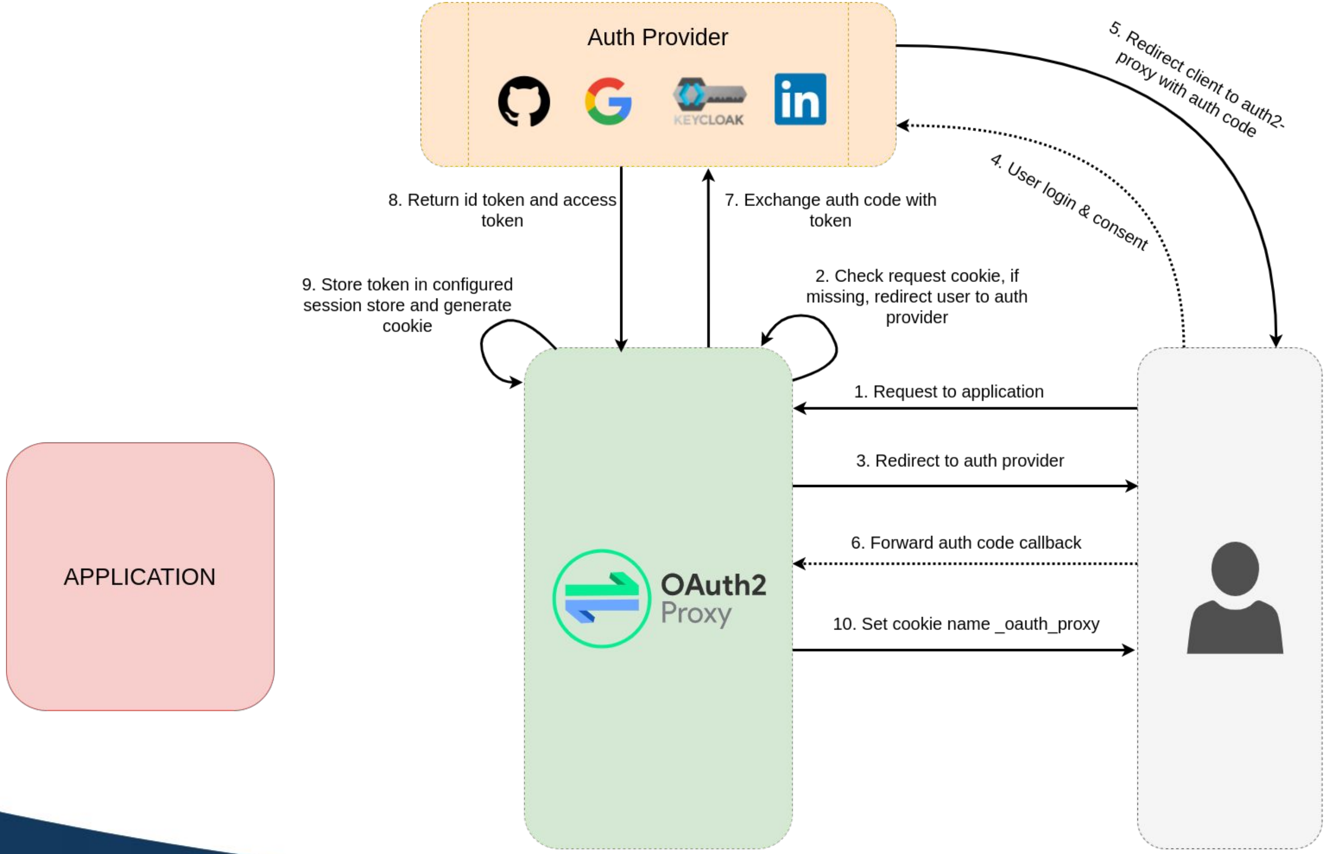
5. Redirect client to auth2-proxy with auth code

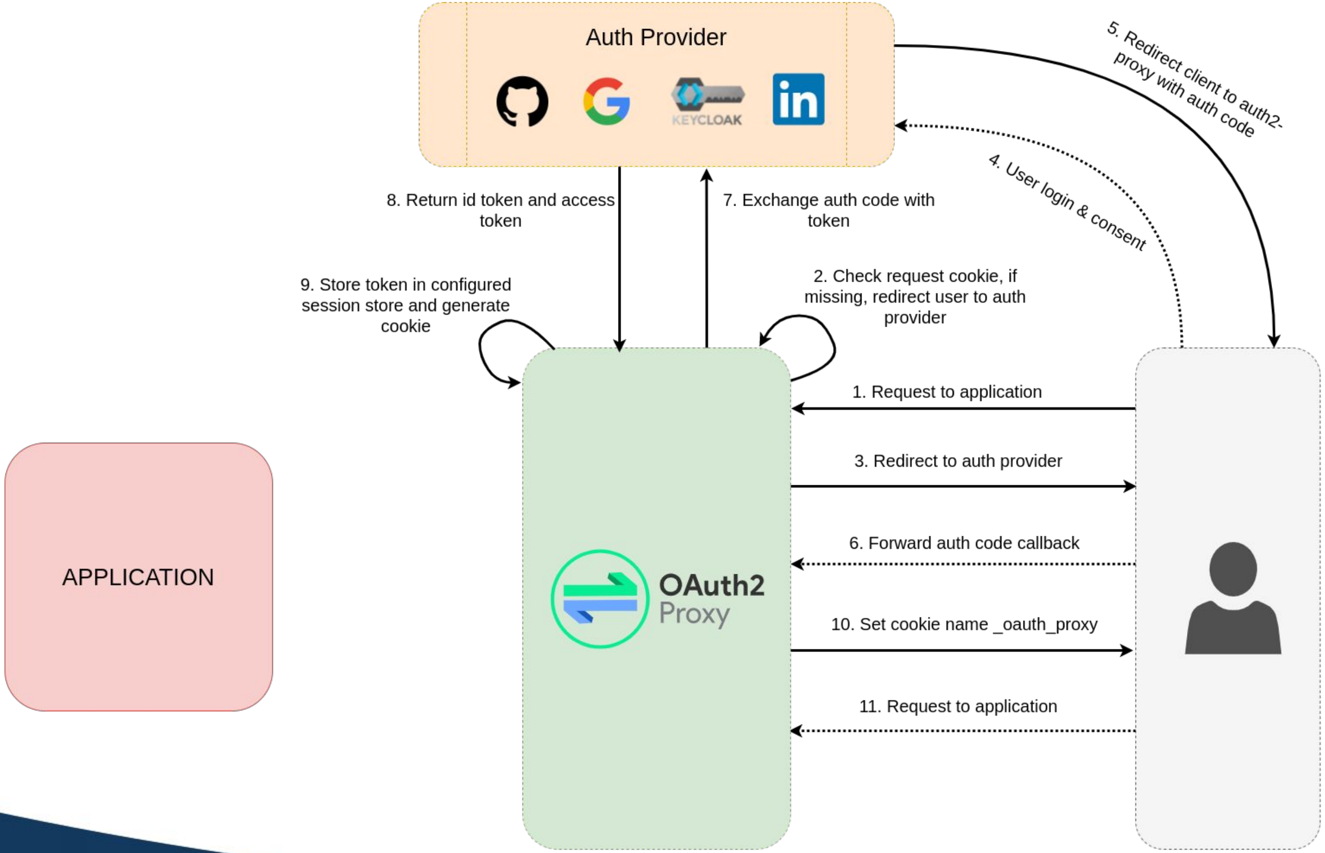
6. Forward auth code callback


Auth Provider
KEYCLOAK
7. Exchange auth code with token
5. Redirect client to auth2-proxy with auth code
4. User login & consent
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
6. Forward auth code callback
APPLICATION
OAuth2 Proxy

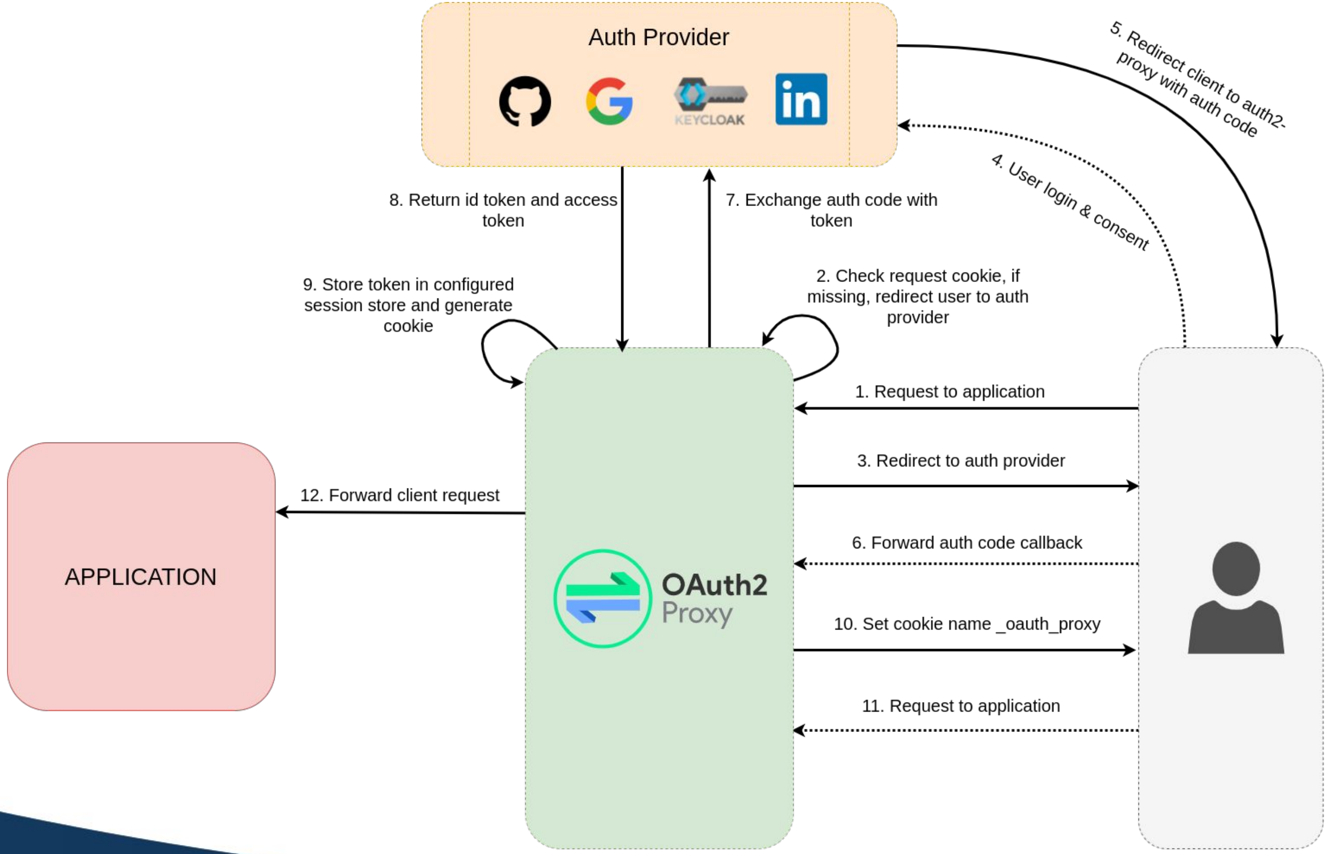

Auth Provider
KEYCLOAK
8. Return id token and access token
7. Exchange auth code with token
5. Redirect client to auth2-proxy with auth code
4. User login & consent
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
6. Forward auth code callback
APPLICATION
OAuth2 Proxy


Auth Provider
KEYCLOAK
8. Return id token and access token
7. Exchange auth code with token
5. Redirect client to auth2-proxy with auth code
4. User login & consent
9. Store token in configured session store and generate cookie
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
6. Forward auth code callback
APPLICATION
OAuth2
Proxy


Auth Provider
KEYCLOAK
5. Redirect client to auth2-proxy with auth code
4. User login & consent
8. Return id token and access token
7. Exchange auth code with token
9. Store token in configured session store and generate cookie
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
6. Forward auth code callback
10. Set cookie name _oauth_proxy
APPLICATION
OAuth2 Proxy



1. Request to application
2. Check request cookie, if missing, redirect user to auth provider
3. Redirect to auth provider
4. User login & consent
5. Redirect client to auth2-proxy with auth code
6. Forward auth code callback
7. Exchange auth code with token
8. Return id token and access token
9. Store token in configured session store and generate cookie
10. Set cookie name _oauth_proxy
11. Request to application



Auth Provider

1. Request to application

2. Check request cookie, if missing, redirect user to auth provider

3. Redirect to auth provider

4. User login & consent

5. Redirect client to auth2-proxy with auth code

6. Forward auth code callback

7. Exchange auth code with token

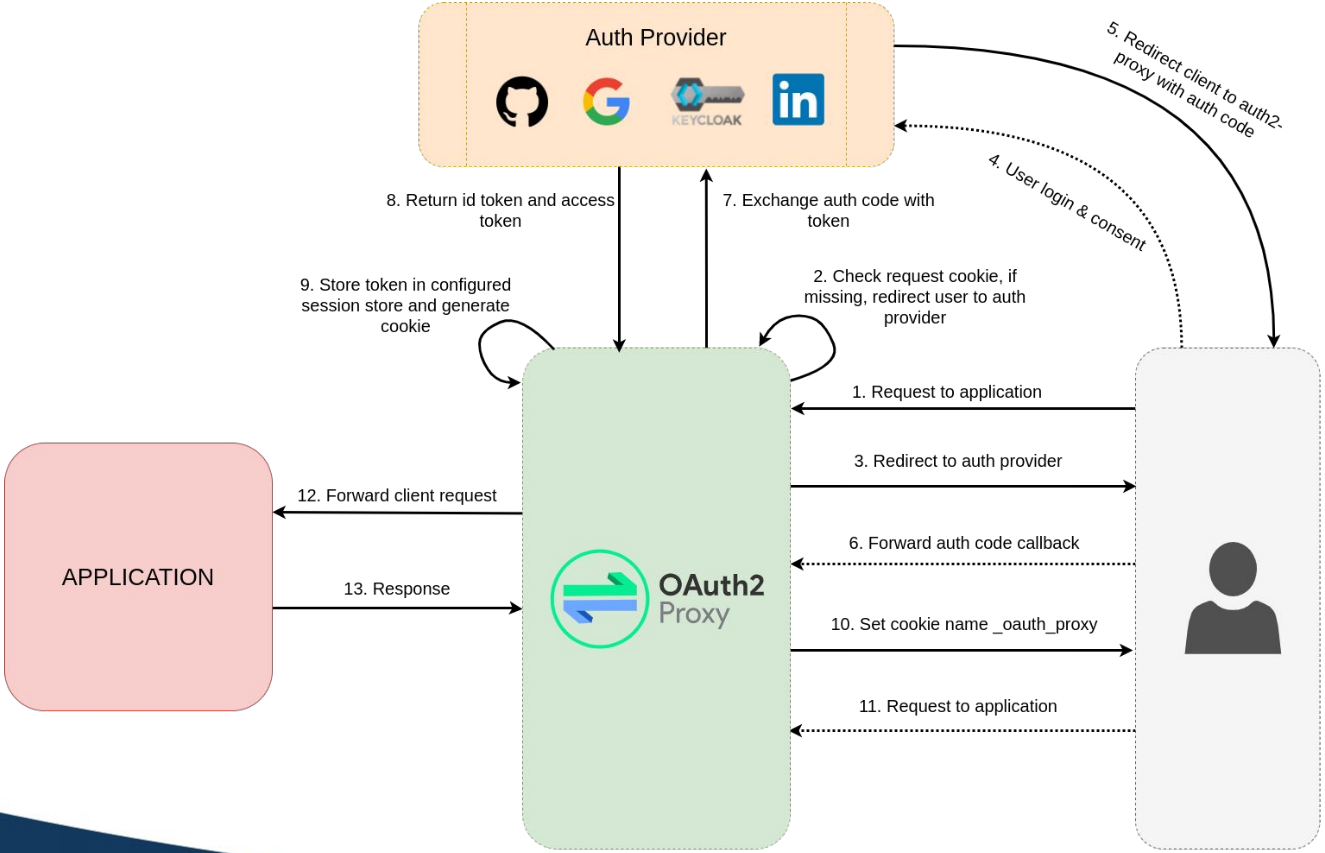
8. Return id token and access token

9. Store token in configured session store and generate cookie

10. Set cookie name _oauth_proxy

11. Request to application

12. Forward client request



Auth Provider

KEYCLOAK

OAuth2 Proxy

APPLICATION

1. Request to application

2. Check request cookie, if missing, redirect user to auth provider

3. Redirect to auth provider

4. User login & consent

5. Redirect client to auth2-proxy with auth code

6. Forward auth code callback

7. Exchange auth code with token

8. Return id token and access token

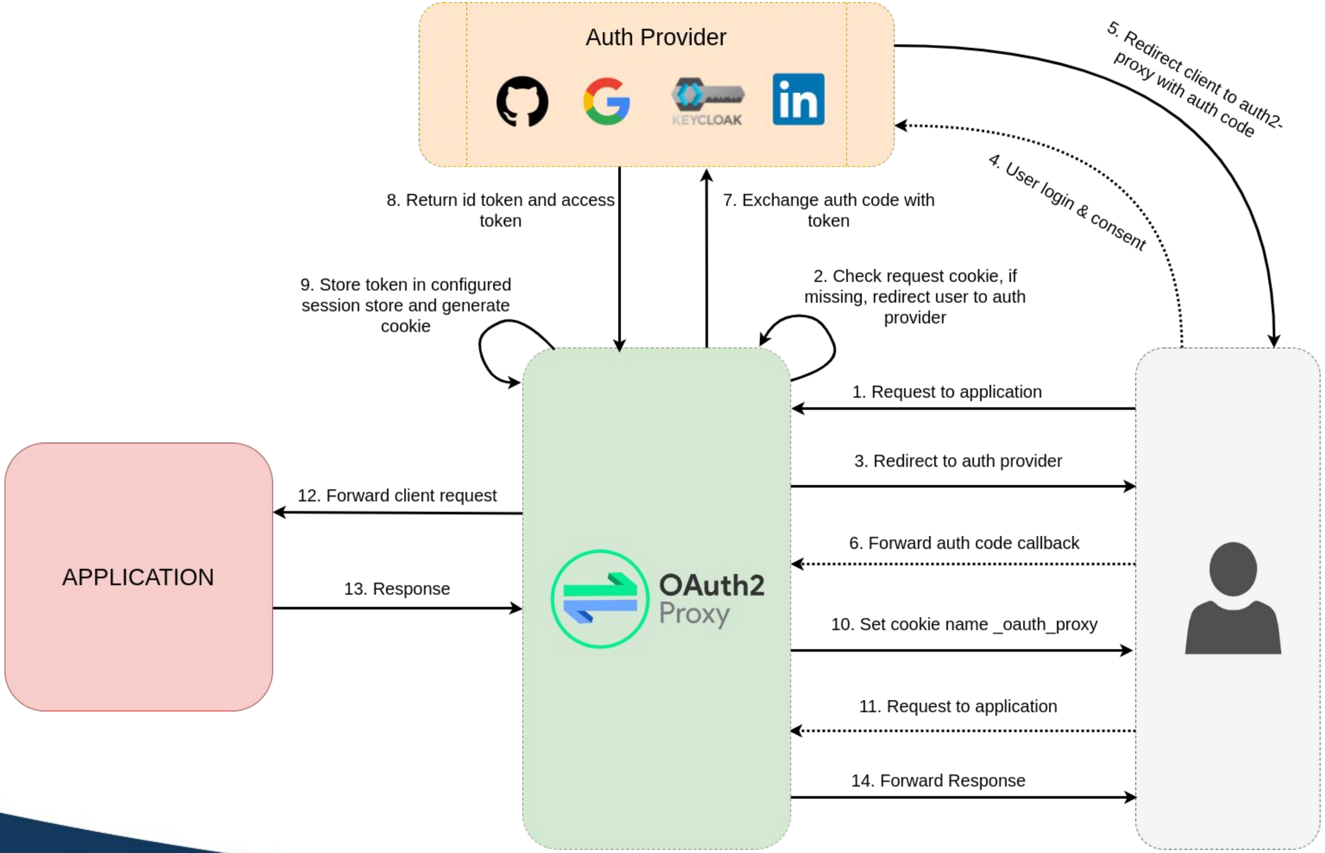
9. Store token in configured session store and generate cookie

10. Set cookie name _oauth_proxy

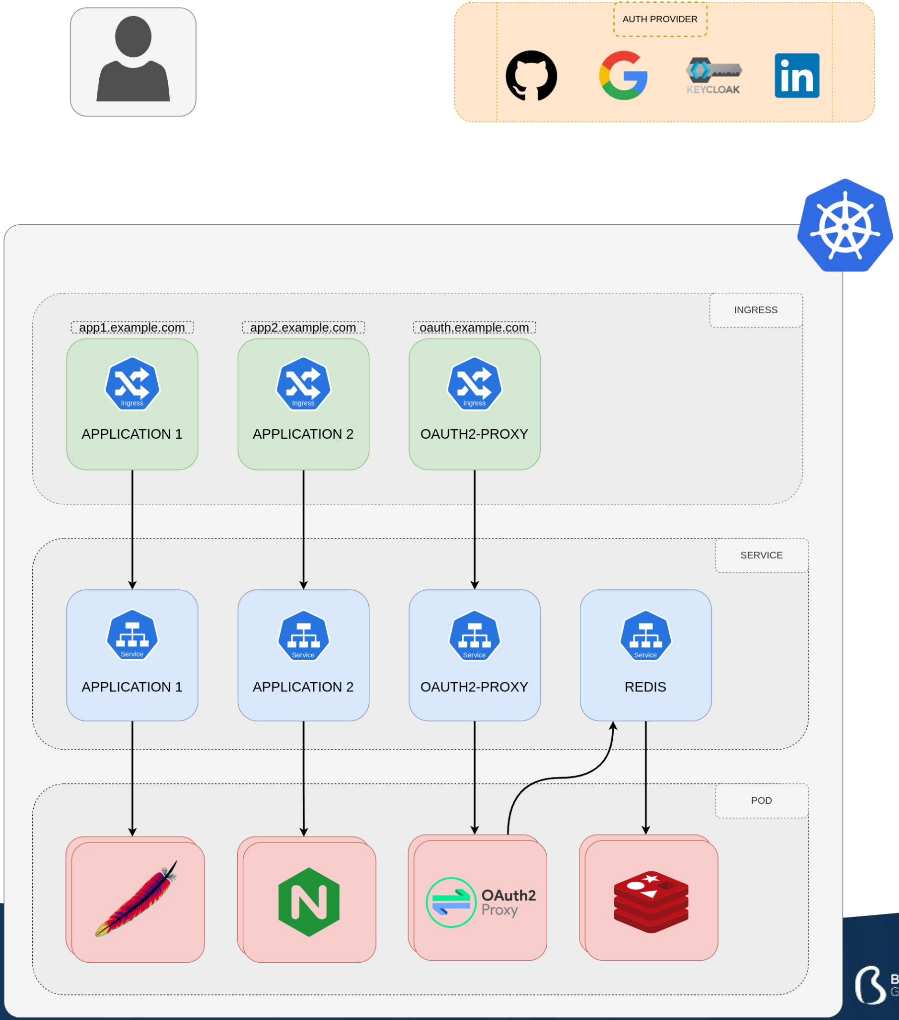
11. Request to application

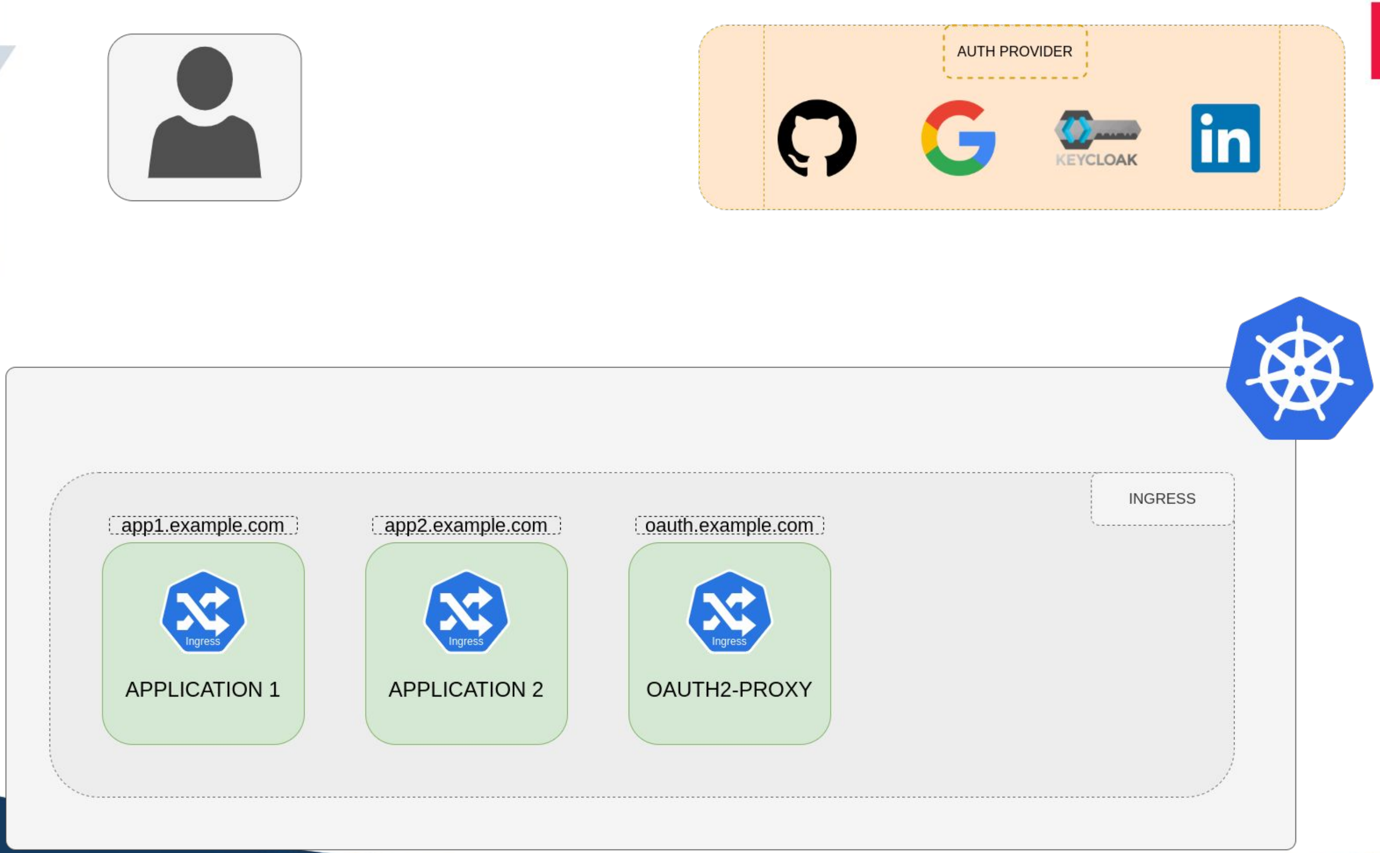
12. Forward client request

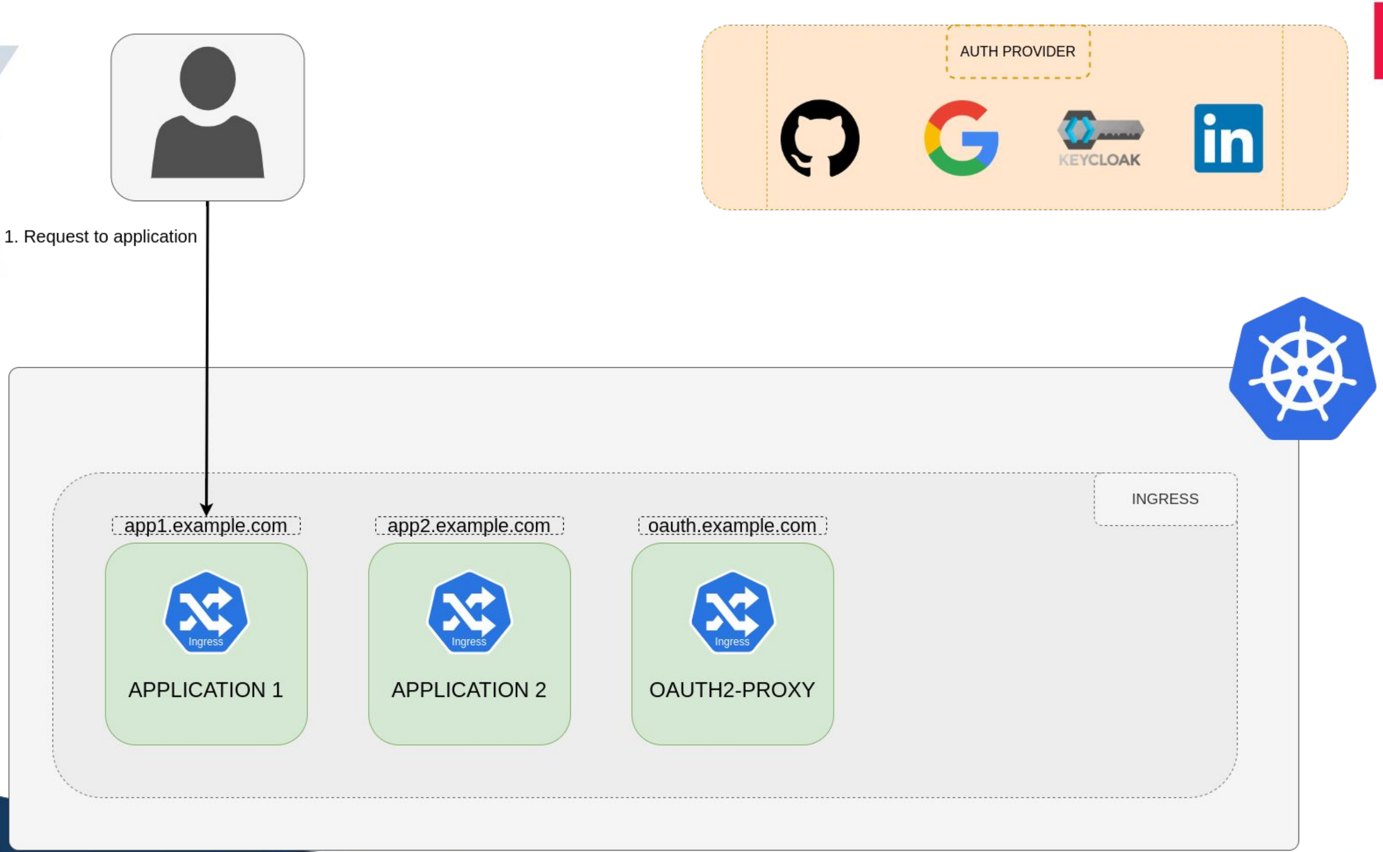
13. Response

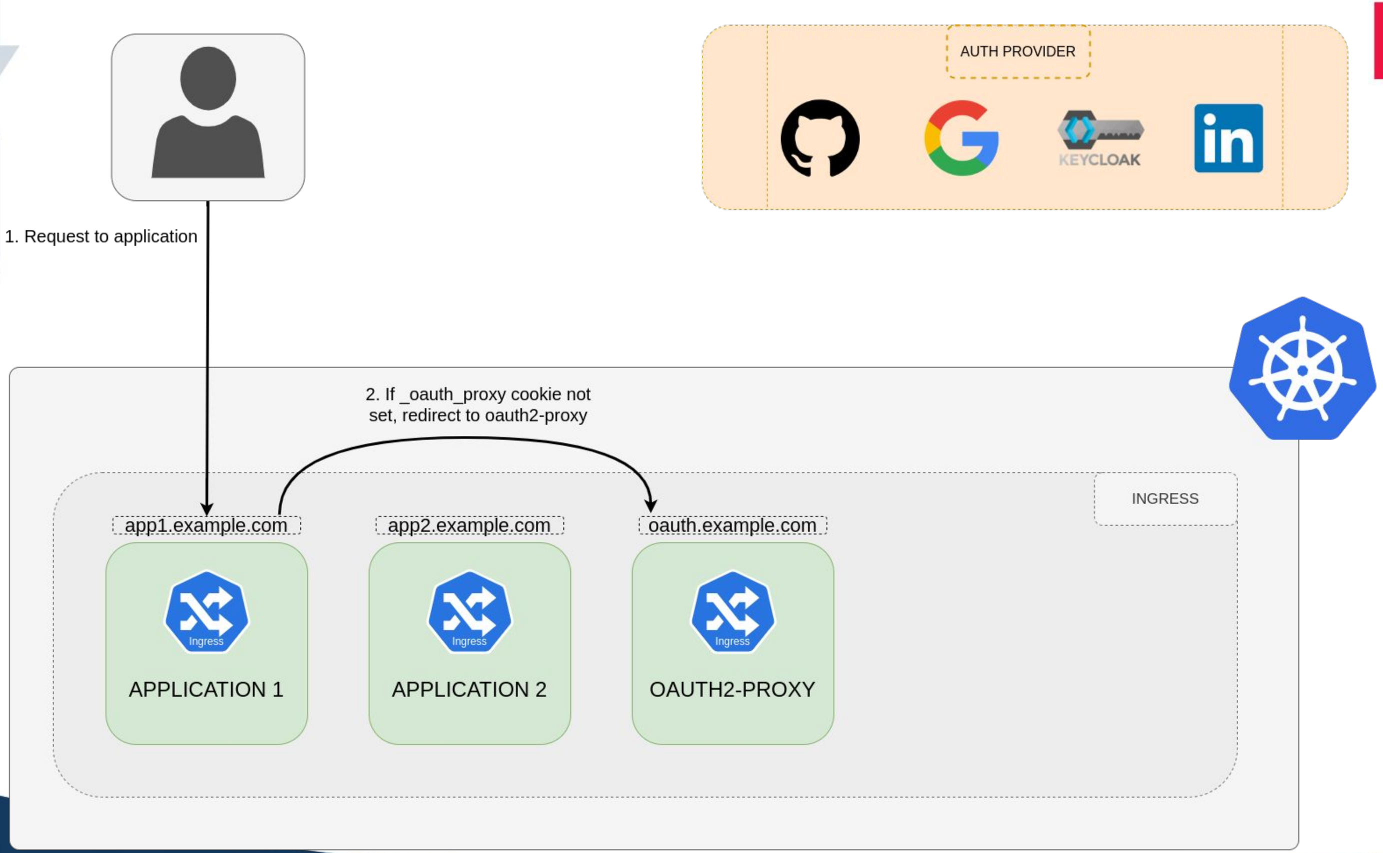

Auth Provider
KEYCLOAK
in
8. Return id token and access token
7. Exchange auth code with token
5. Redirect client to auth2-proxy with auth code
4. User login & consent
9. Store token in configured session store and generate cookie
2. Check request cookie, if missing, redirect user to auth provider
1. Request to application
3. Redirect to auth provider
APPLICATION
12. Forward client request
13. Response
OAuth2
Proxy
6. Forward auth code callback
10. Set cookie name _oauth_proxy
11. Request to application
14. Forward Response

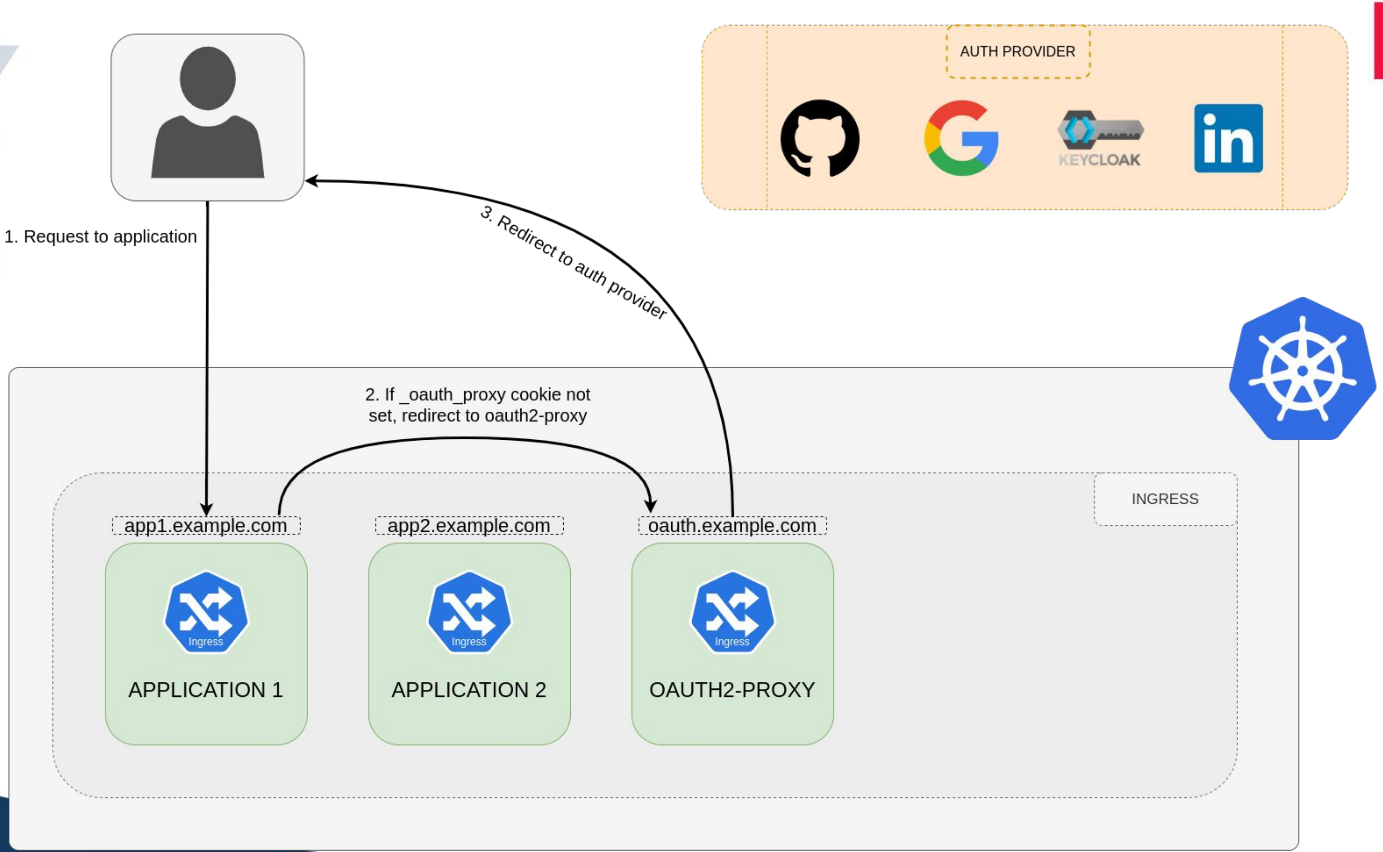
Oauth2-proxy on K8S


AUTH PROVIDER
KEYCLOAK
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
APPLICATION 2
OAUTH2-PROXY
SERVICE
Service
APPLICATION 1
APPLICATION 2
OAUTH2-PROXY
REDIS
POD
OAuth2
Proxy


AUTH PROVIDER
KEYCLOAK
INGRESS
app1.example.com
Ingress
APPLICATION 1
app2.example.com
Ingress
APPLICATION 2
oauth.example.com
Ingress
OAUTH2-PROXY


AUTH PROVIDER
KEYCLOAK
1. Request to application
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY

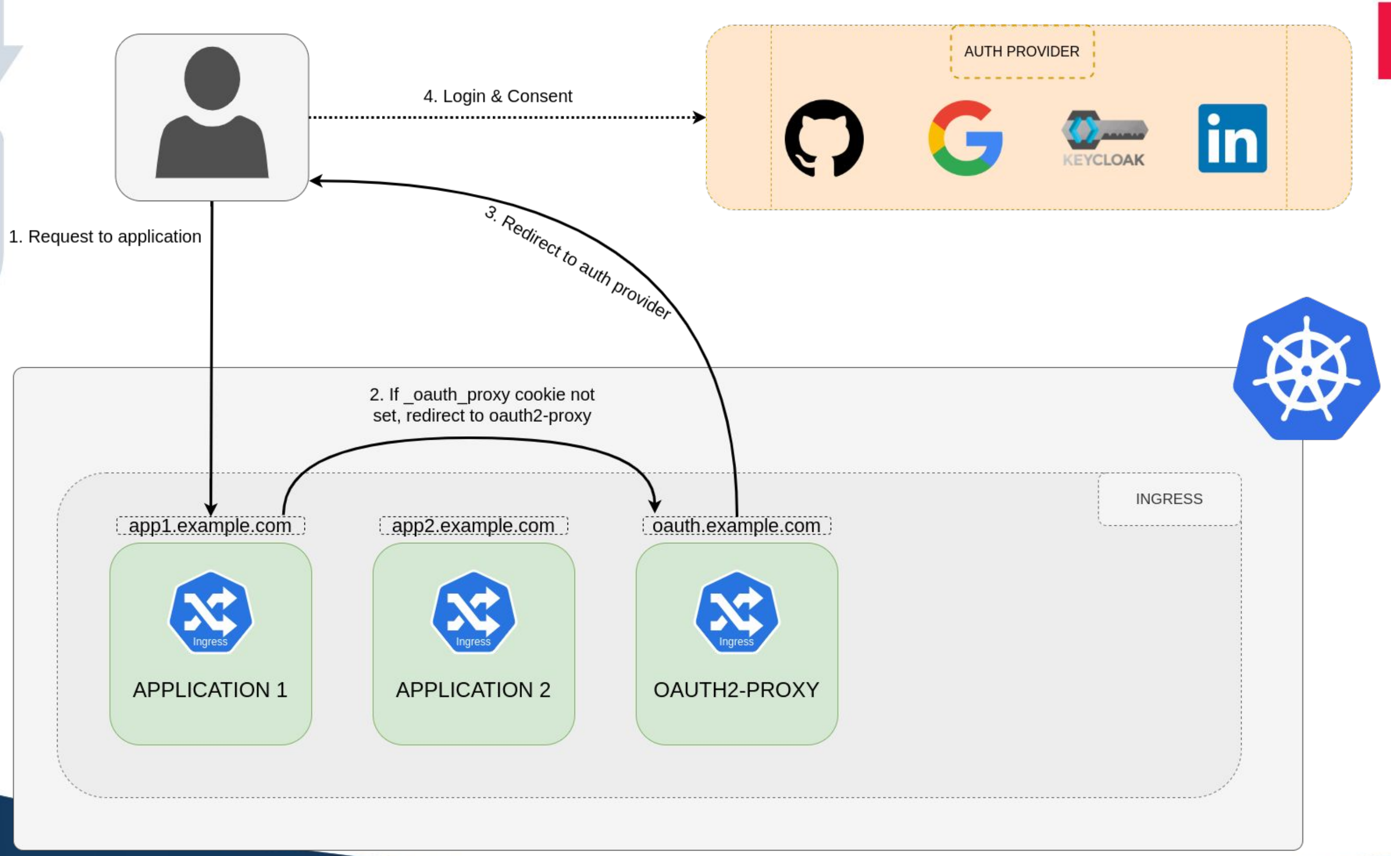

AUTH PROVIDER
KEYCLOAK
1. Request to application
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY

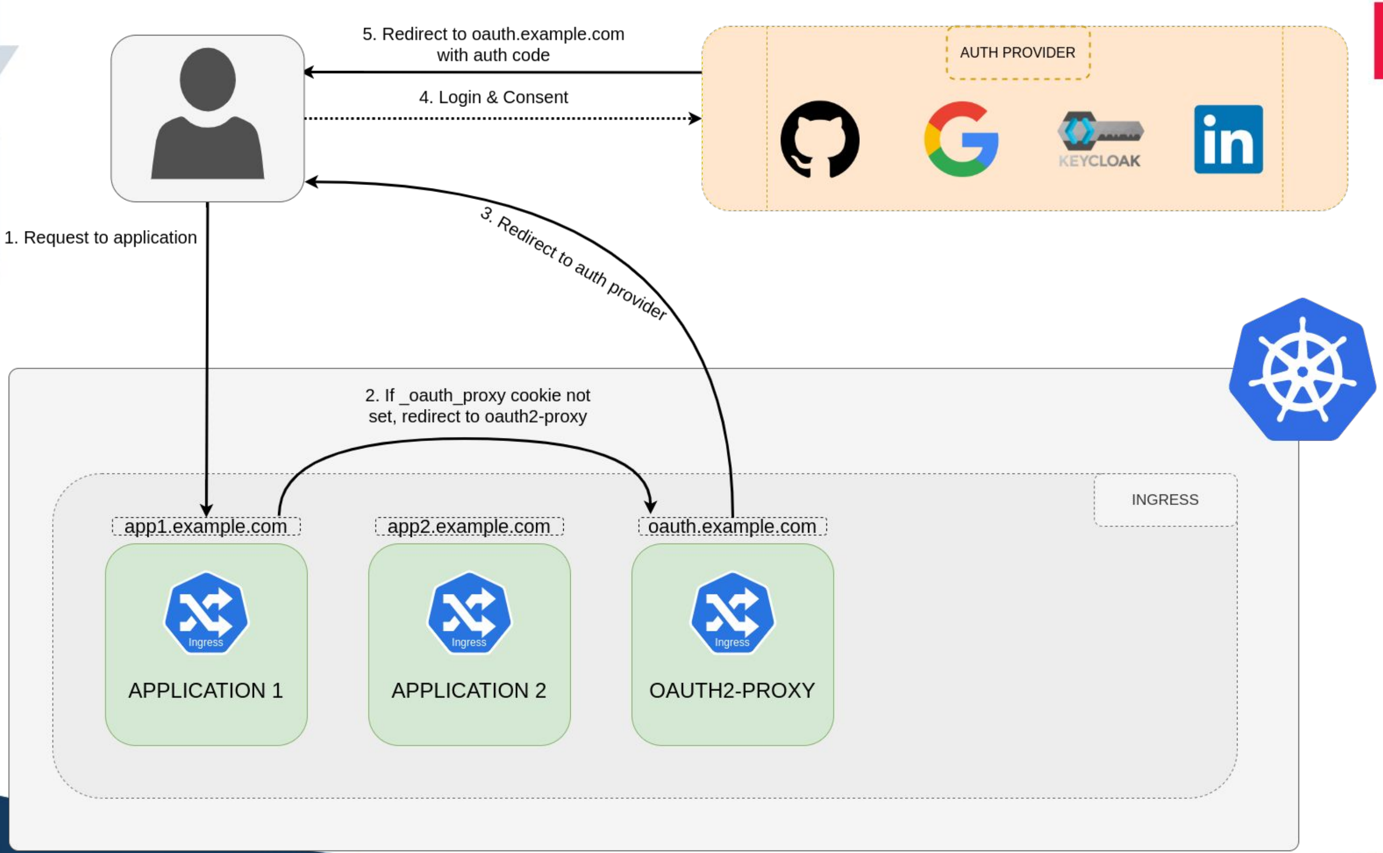


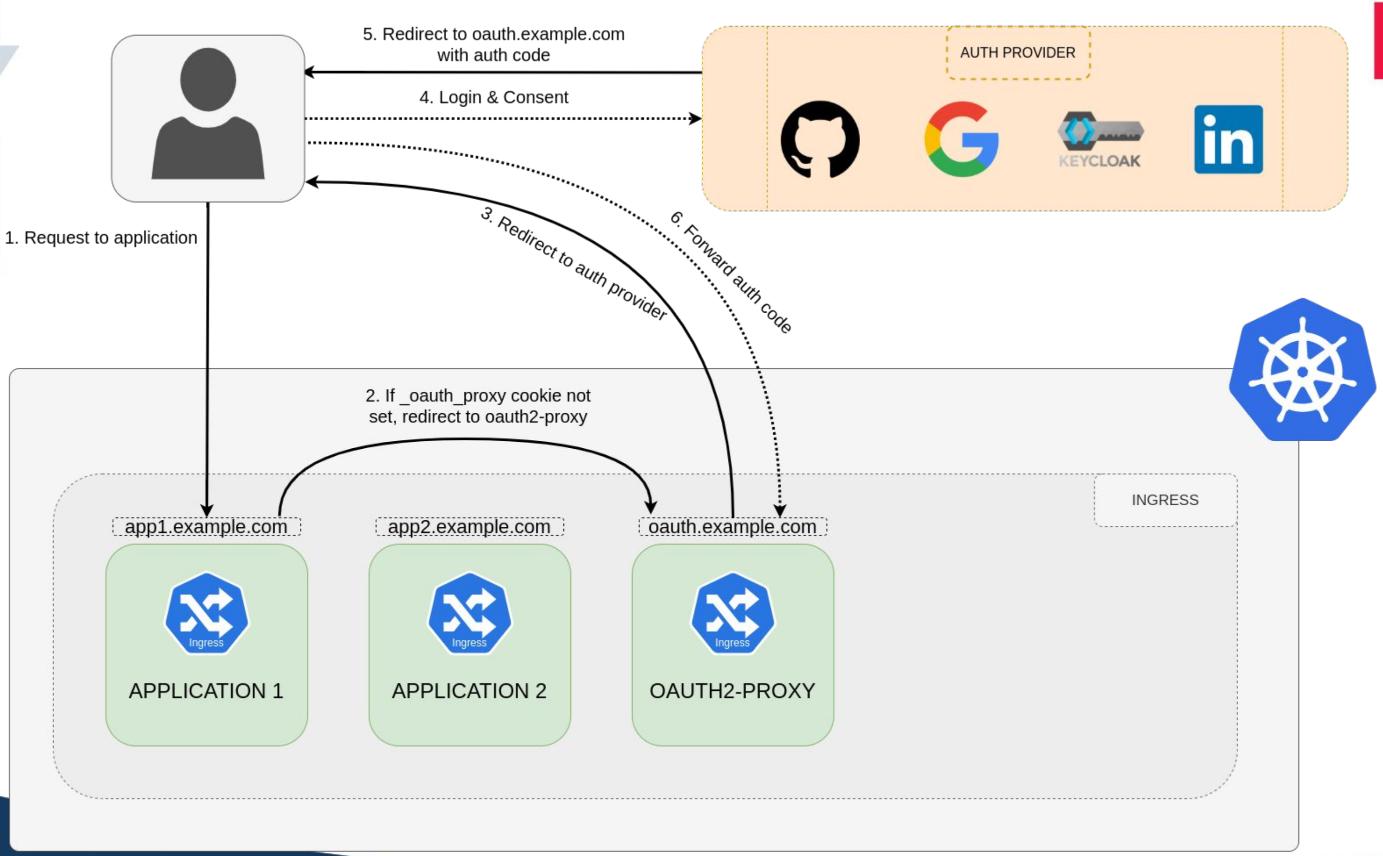
1. Request to application

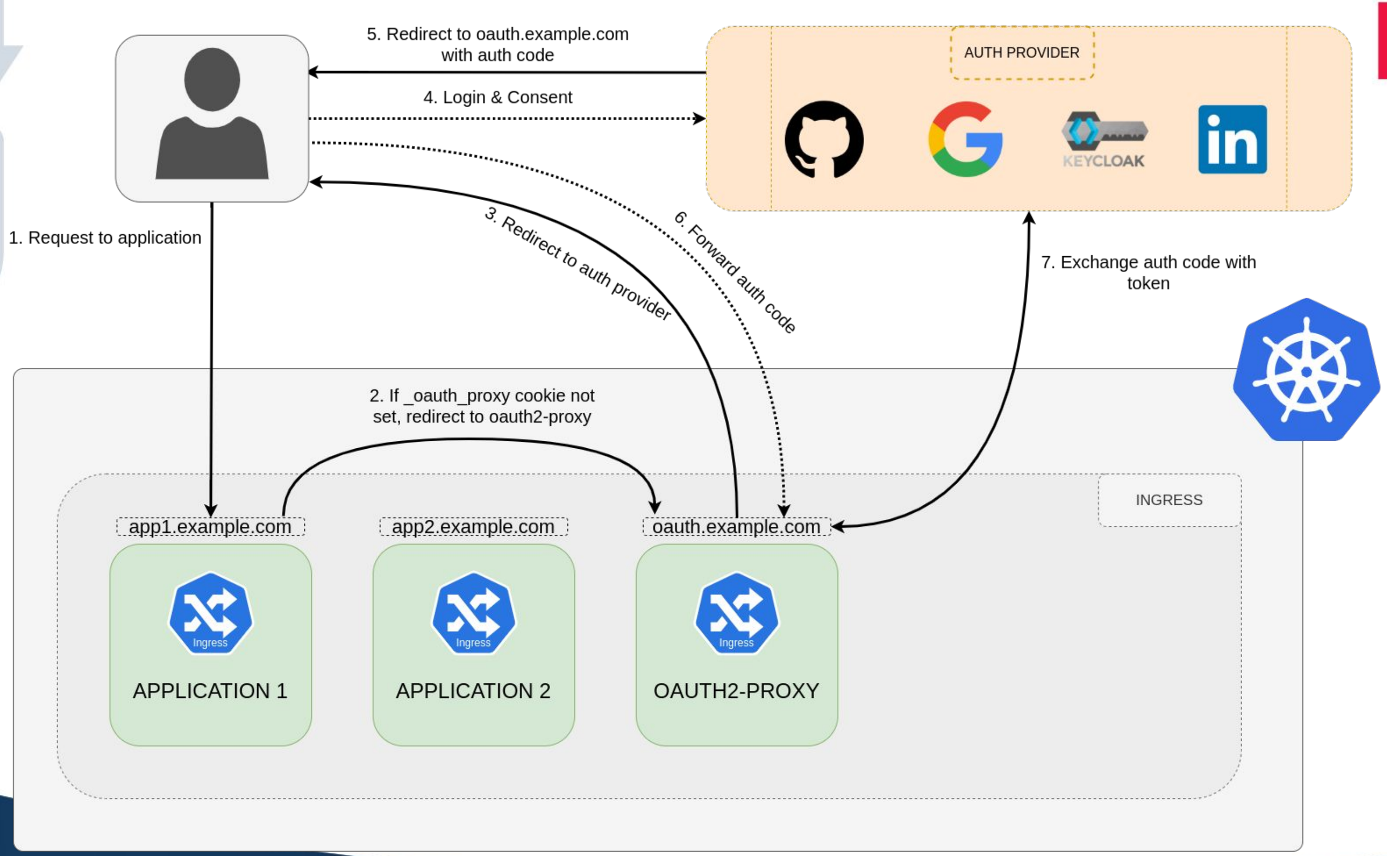
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy

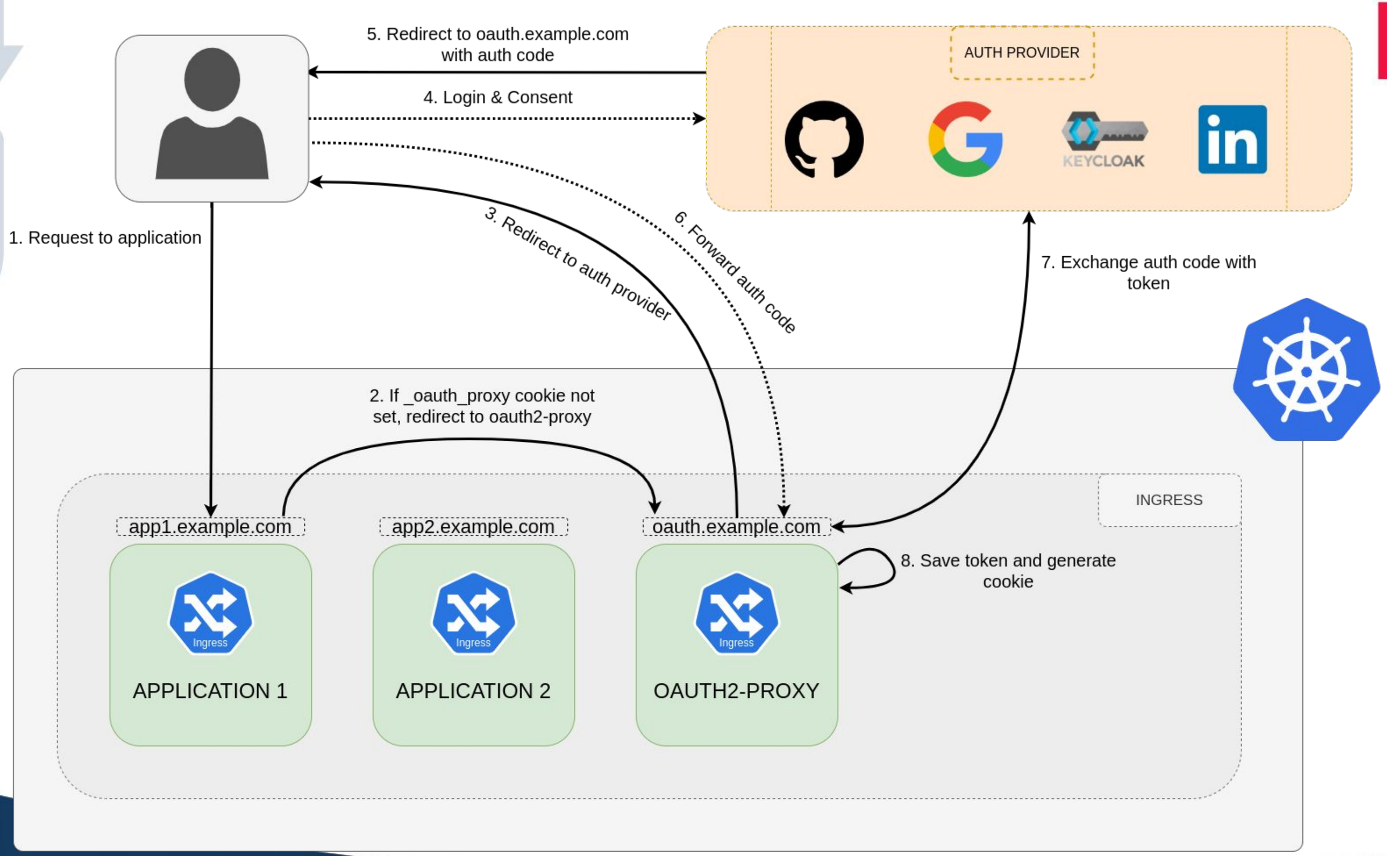
3. Redirect to auth provider

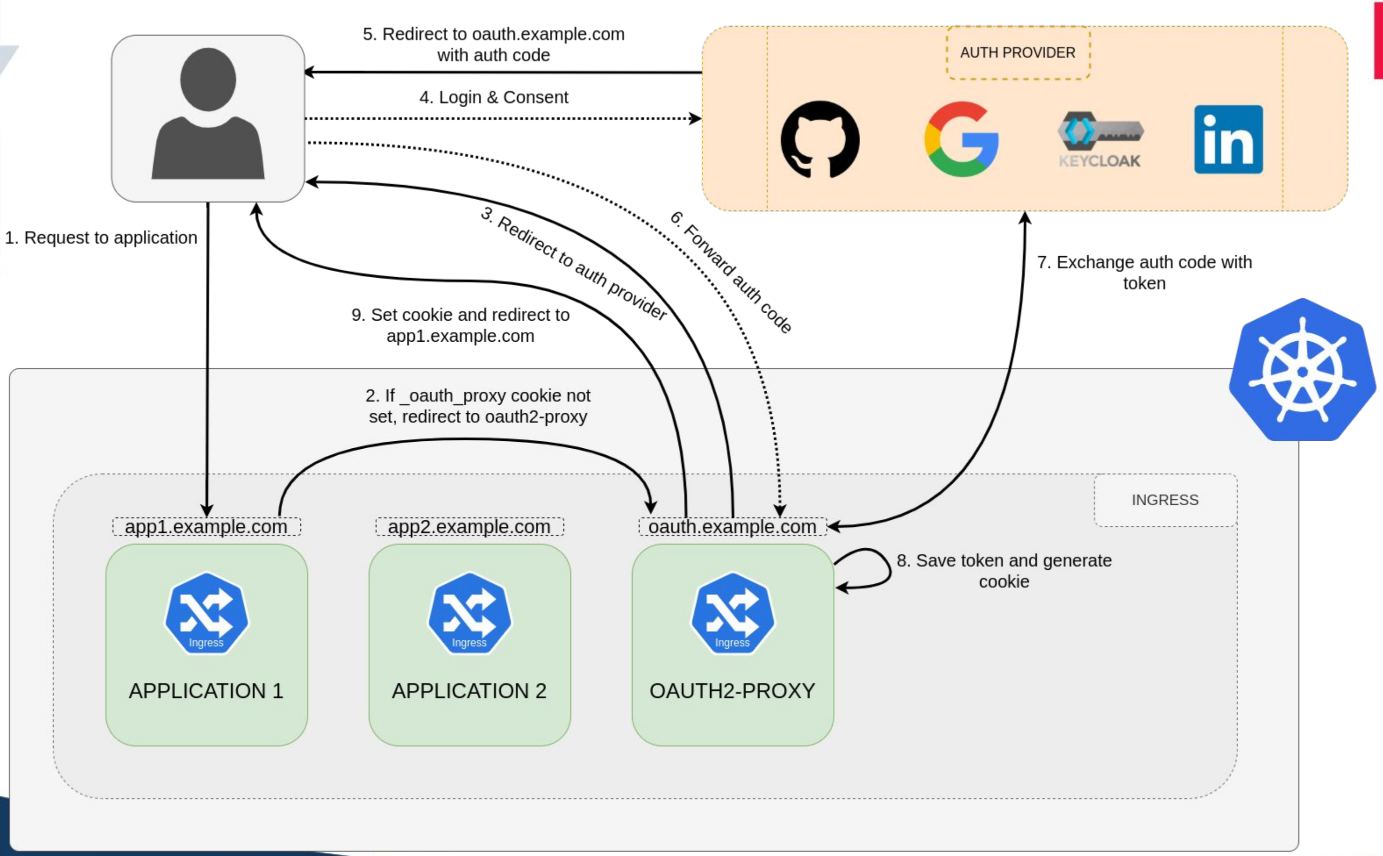

4. Login & Consent
AUTH PROVIDER
KEYCLOAK
1. Request to application
3. Redirect to auth provider
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY


5. Redirect to oauth.example.com with auth code
4. Login & Consent
AUTH PROVIDER
KEYCLOAK
1. Request to application
3. Redirect to auth provider
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY


5. Redirect to oauth.example.com
with auth code
4. Login & Consent
AUTH PROVIDER
KEYCLOAK
1. Request to application
3. Redirect to auth provider
6. Forward auth code
2. If _oauth_proxy cookie not
set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY


5. Redirect to oauth.example.com with auth code
4. Login & Consent
AUTH PROVIDER
KEYCLOAK
1. Request to application
3. Redirect to auth provider
6. Forward auth code
7. Exchange auth code with token
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY


5. Redirect to oauth.example.com with auth code
4. Login & Consent
AUTH PROVIDER
KEYCLOAK
1. Request to application
3. Redirect to auth provider
6. Forward auth code
7. Exchange auth code with token
2. If _oauth_proxy cookie not set, redirect to oauth2-proxy
INGRESS
app1.example.com
app2.example.com
oauth.example.com
8. Save token and generate cookie
Ingress
APPLICATION 1
Ingress
APPLICATION 2
Ingress
OAUTH2-PROXY



1. Request to application

2. If _oauth_proxy cookie not set, redirect to oauth2-proxy

3. Redirect to auth provider

4. Login & Consent

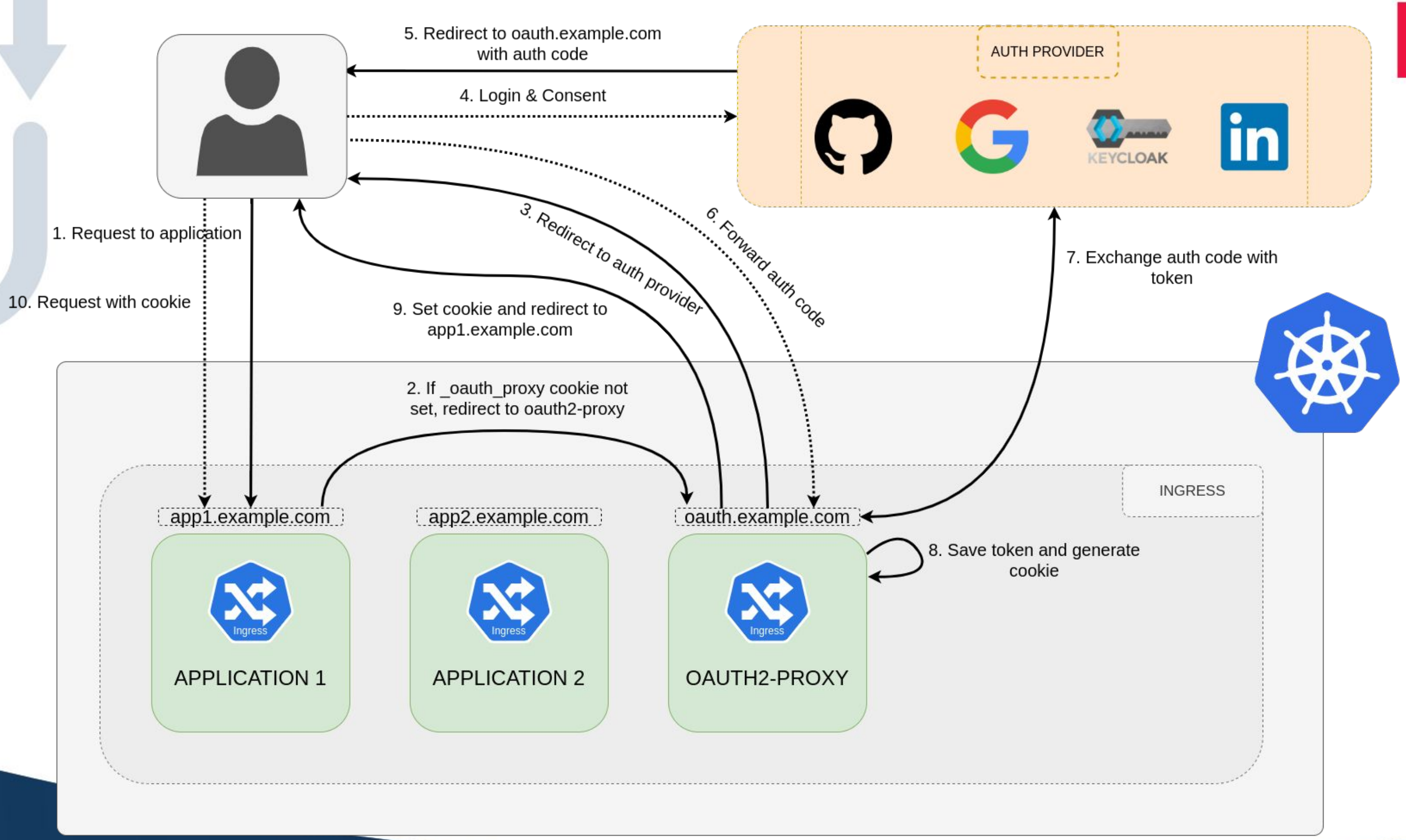
5. Redirect to oauth.example.com with auth code

6. Forward auth code

7. Exchange auth code with token

8. Save token and generate cookie

9. Set cookie and redirect to app1.example.com



1. Request to application

2. If _oauth_proxy cookie not set, redirect to oauth2-proxy

3. Redirect to auth provider

4. Login & Consent

5. Redirect to oauth.example.com with auth code

6. Forward auth code

7. Exchange auth code with token

8. Save token and generate cookie

9. Set cookie and redirect to app1.example.com

10. Request with cookie

Useful Endpoints

/oauth2/sign_in the login page

/oauth2/sign_out logout (clear session cookie)

/oauth2/userinfo return user’s from the session in JSON

/oauth2/callback oauth callback url

/ping health check (return 200 OK)

/oauth2/auth check if the user is authenticated

/oauth2/start redirect to start the OAuth cycle

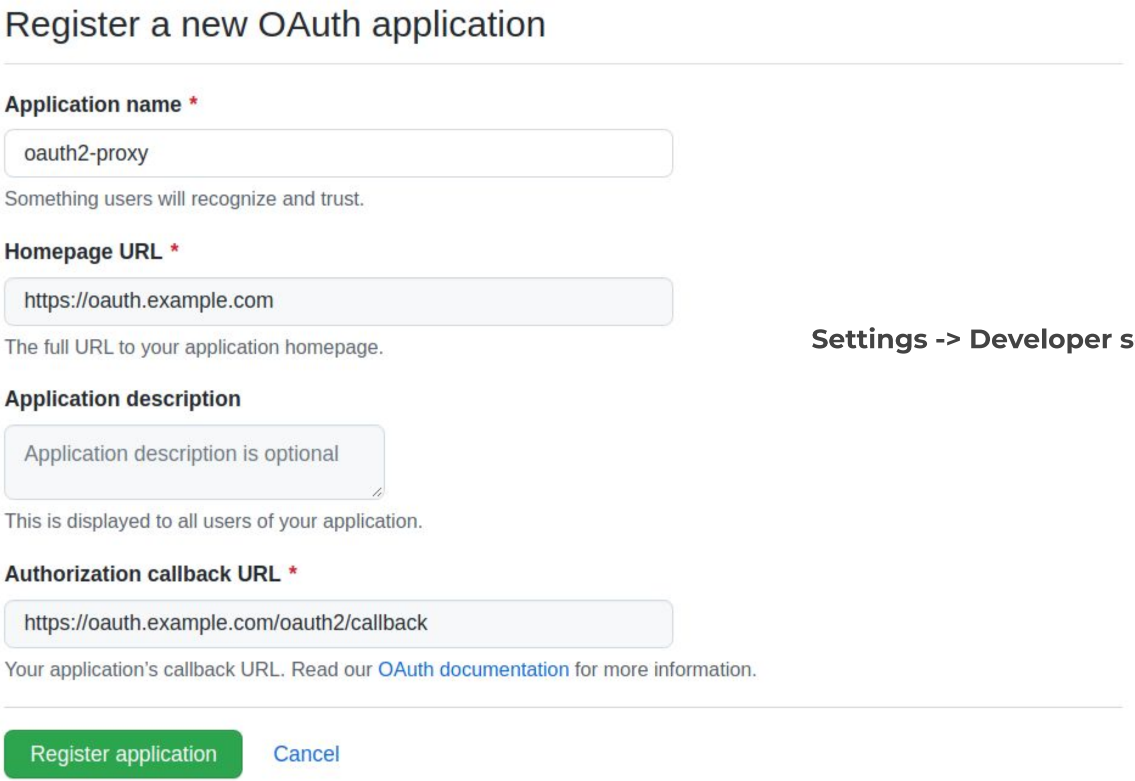
OAuth2 Proxy

Source: https://oauth2-proxy.github.io/oauth2-proxy/docs/features/endpoints

How to Configure?

https://medium.com/prastamaha

Implement Oauth2-proxy on Kubernetes



Register a new OAuth application

Application name *

oauth2-proxy

Something users will recognize and trust.

Homepage URL *

https://oauth.example.com

The full URL to your application homepage.

Application description

Application description is optional

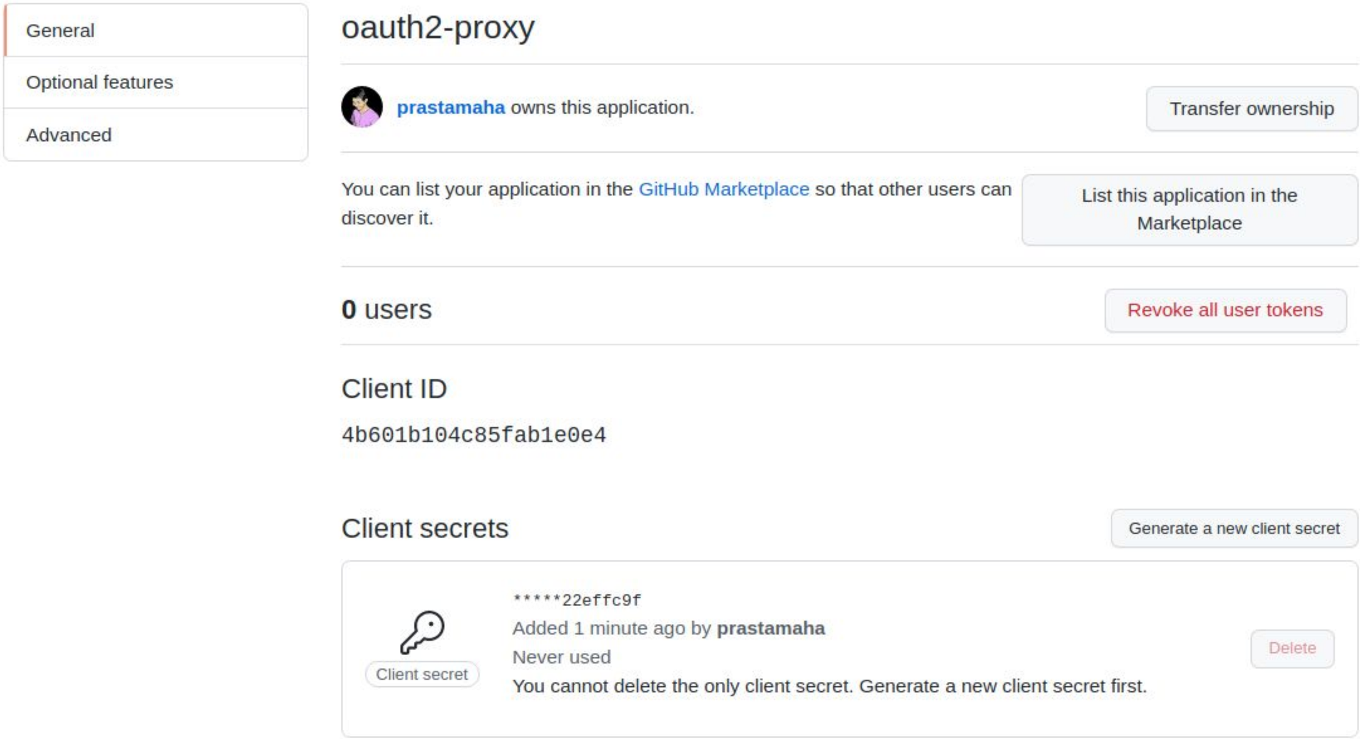
This is displayed to all users of your application.

Authorization callback URL *

https://oauth.example.com/oauth2/callback

Your application's callback URL. Read our OAuth documentation for more information.

Settings -> Developer settings -> New Oauth app



oauth2-proxy

prastamaha owns this application.

You can list your application in the GitHub Marketplace so that other users can discover it.

0 users

Client ID

```
4b601b104c85fab1e0e4
```

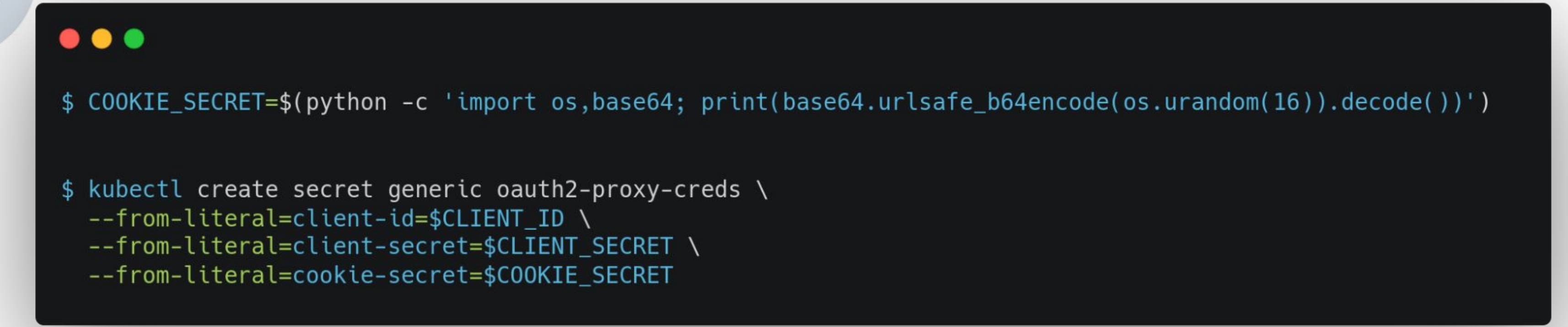
Client secrets

```
*****22effc9f
```

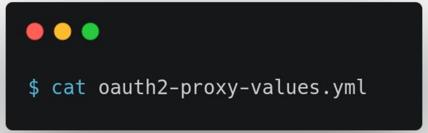
Added 1 minute ago by **prastamaha**

Never used

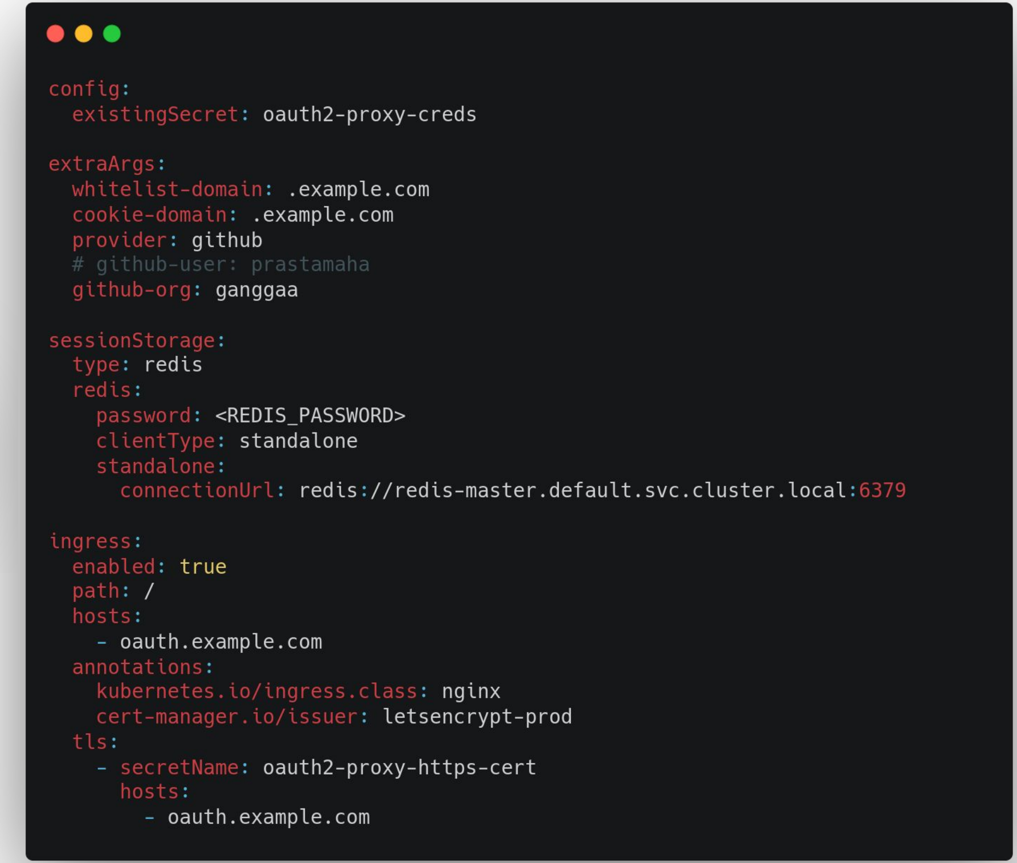
You cannot delete the only client secret. Generate a new client secret first.



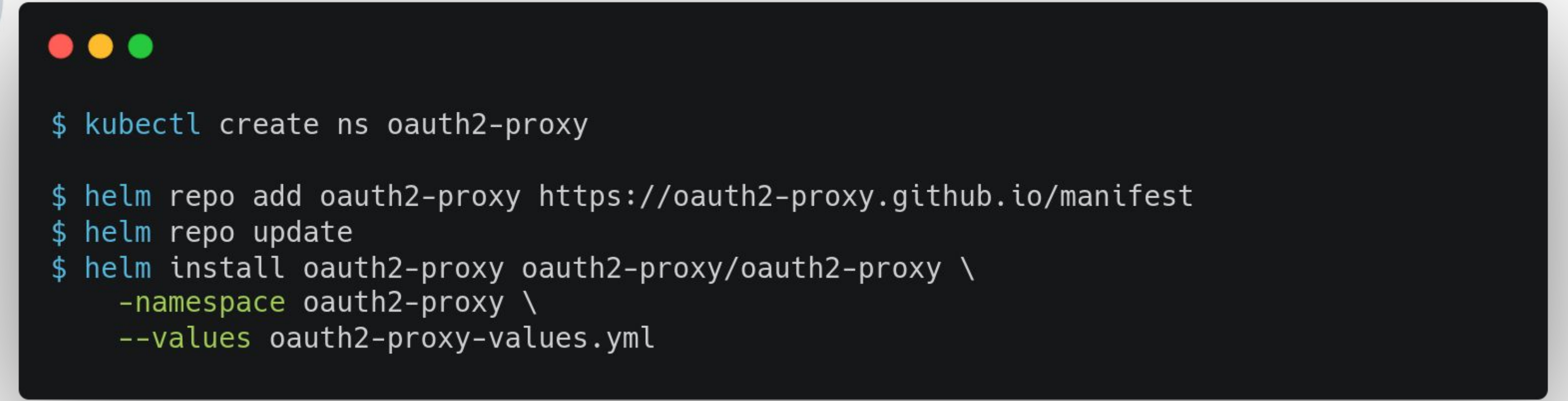
```
$ COOKIE_SECRET=$(python -c 'import os,base64; print(base64.urlsafe_b64encode(os.urandom(16)).decode())')


$ kubectl create secret generic oauth2-proxy-creds \
  --from-literal=client-id=$CLIENT_ID \
  --from-literal=client-secret=$CLIENT_SECRET \
  --from-literal=cookie-secret=$COOKIE_SECRET
```



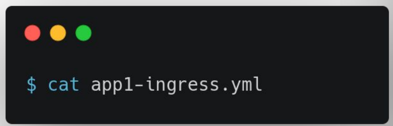
```
$ cat oauth2-proxy-values.yml
```



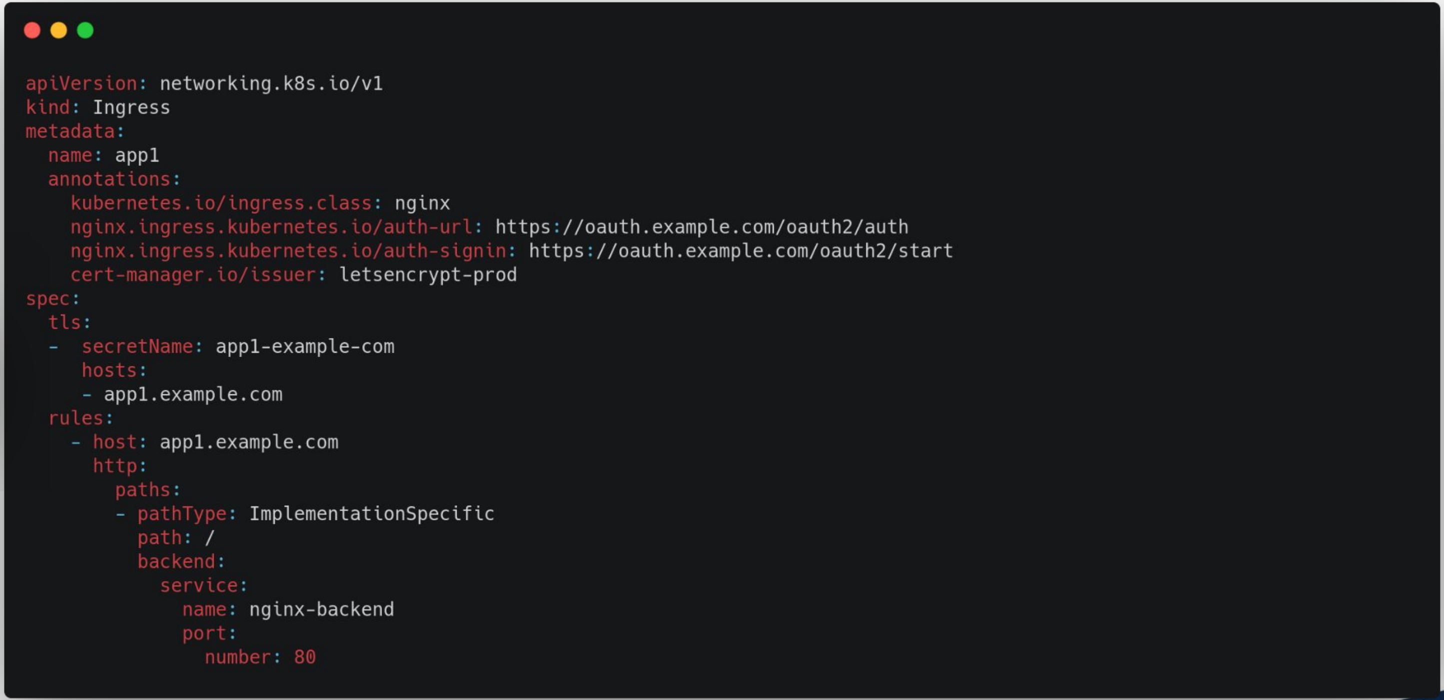
```
config:
  existingSecret: oauth2-proxy-creds

extraArgs:
  whitelist-domain: .example.com
  cookie-domain: .example.com
  provider: github
  # github-user: prastamaha
  github-org: ganggaa

sessionStorage:
  type: redis
  redis:
    password: <REDIS_PASSWORD>
    clientType: standalone
    standalone:
      connectionUrl: redis://redis-master.default.svc.cluster.local:6379

ingress:
  enabled: true
  path: /
  hosts:
    - oauth.example.com
  annotations:
    kubernetes.io/ingress.class: nginx
    cert-manager.io/issuer: letsencrypt-prod
  tls:
    - secretName: oauth2-proxy-https-cert
      hosts:
        - oauth.example.com
```



```
$ kubectl create ns oauth2-proxy

$ helm repo add oauth2-proxy https://oauth2-proxy.github.io/manifest
$ helm repo update
$ helm install oauth2-proxy oauth2-proxy/oauth2-proxy \
    -namespace oauth2-proxy \
    --values oauth2-proxy-values.yml
```



```
$ cat app1-ingress.yml
```



```
apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: app1
  annotations:
    kubernetes.io/ingress.class: nginx
    nginx.ingress.kubernetes.io/auth-url: https://oauth.example.com/oauth2/auth
    nginx.ingress.kubernetes.io/auth-signin: https://oauth.example.com/oauth2/start
    cert-manager.io/issuer: letsencrypt-prod
spec:
  tls:
  -  secretName: app1-example-com
     hosts:
     - app1.example.com
  rules:
    - host: app1.example.com
      http:
        paths:
        - pathType: ImplementationSpecific
          path: /
          backend:
            service:
              name: nginx-backend
              port:
                number: 80
```

Quick Demo

Sponsored by:

Open Infrastructure
FOUNDATION

NVIDIA®

boer
technology

Biznet
GioCloud

EasyStack
open cloud computing

BRI

Open Networking
Indonesia

imtek

INDOCENTER

Hosted by:

OPENSTACK USER GROUP INDONESIA

OpenStack Indonesia

Indonesia OpenStack Foundation Community

www.openstack.id

Community Partners:

sdX Indonesia

openSUSE ID

KLAS

gimpscape ID

CLOUD NATIVE INDONESIA

Adaptive Network Laboratory

NETWORK EVOLUTION

Open Infrastructure FOUNDATION

INDONESIA OpenInfra Days

Thanks!

Do you have any questions?

prasta@btech.id
+6281236437805
btech.id

Platinum sponsor :

NVIDIA

Gold sponsor :

boer technology

Biznet GioCloud

EasyStack

Silver sponsor :

BRI

Custom sponsor :

Open Networking Indonesia

imtek

INDOCENTER