



Discovery to Delivery

La sécurité dans une architecture microservices

# Coucou c'est nous



✉ f.garcia@ippon.fr

🐦 @lm\_flog

> Backend lover

> Speaker

♥ Tracing distribué

♥ Kafka

♥ Cloud



✉ vmaleze@ippon.fr

🐦 @vmaleze

> Architecte

> Tech friendly

♥ Spring

♥ React

♥ DevOps

# 1h sur de la sécu !



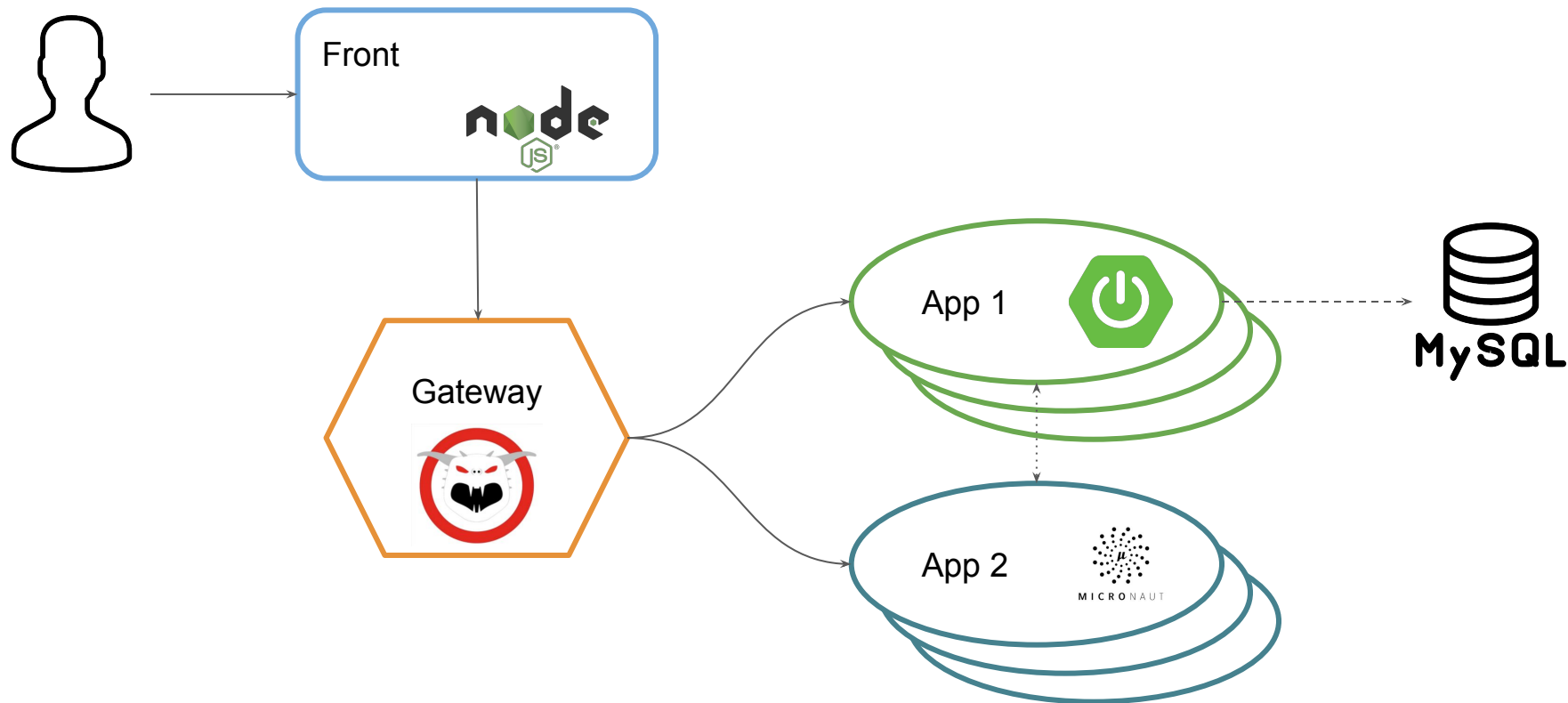
On y va pas à pas :

- Le besoin du client ?
- Les différentes solutions ?
- Comment on met tout ça en place ?

De quoi on va parler ?

- Authentication
- Authorization
- OAuth2
- JWT
- Sessions
- Spring security
- ...

# L'architecture



# Le(s) problème(s)

- Une architecture micro services, sans sécurité ...
  - Tout le monde peut tout faire !
- Gestion des comptes
  - Il y a un LDAP et ils veulent l'utiliser,
  - Mais il y a aussi des comptes google,
  - Et parfois des prestataires externes ...
- La complexité
  - Je dois identifier mon utilisateur dans chaque  $\mu s$  ?!
  - On peut pas mettre une session ?
  - Et le SSO ?
  - Comment j'implémente tout ça facilement ?



## Serveur d'identité / Gestion des accès ?

- Single Sign On
- Fédération d'identité (LDAP / Custom DB)
- Social login (Google)
- Gestion de rôles
- Facilité d'administration
- Intégration Spring, NodeJs
- Coûts (maintenance / SAAS)

okta



Auth0



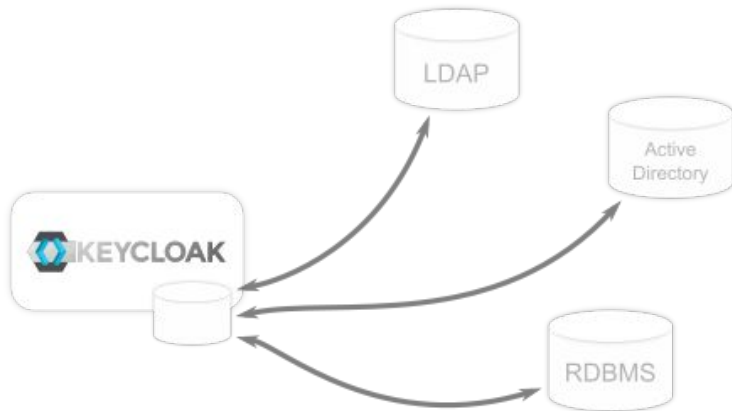
KEYCLOAK



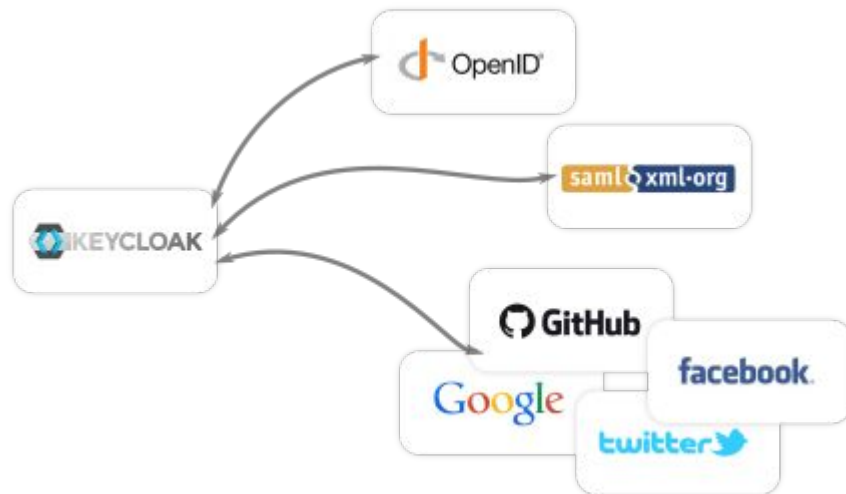
# Keycloak



SSO



User federation


















Identity Brokering



Standard protocols

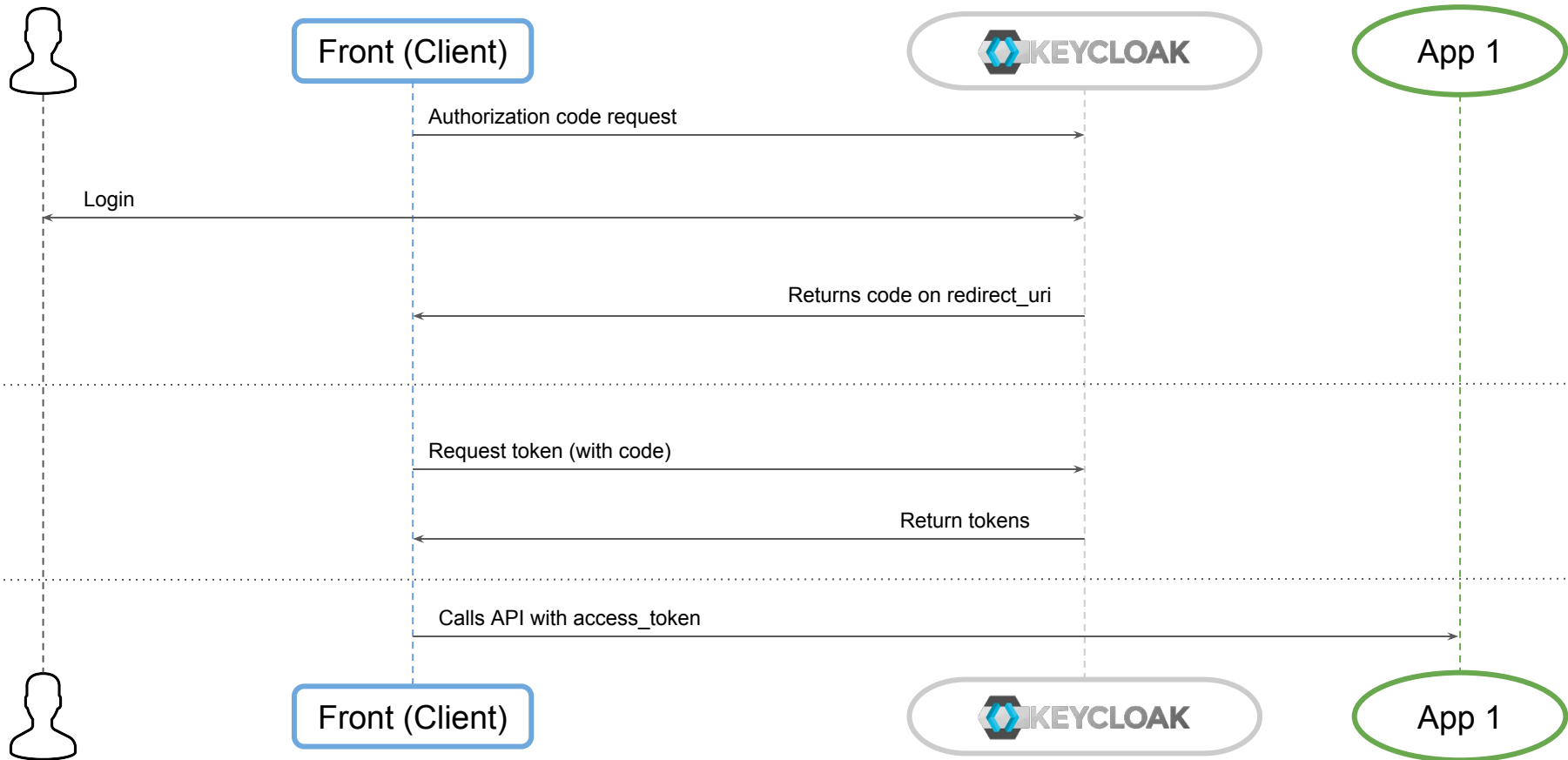
# L'authentification

---

Basic	OpenID	SAML
 Simplicité	 OAuth2	 Identity provider
 Base 64	 SSO	 SSO
 Intégration avec des tiers	 JWTs	 Mature
	 “Nouveau” standard	 Verbeux
	 Complexe	 Complexe
	 HTTPS	 “Ancien” Standard



# Oauth2



# OpenId en action

---

<https://github.com/ImFlog/microservices-security>

# Comment ça marche

---

- Les différents tokens
  - **id** => Contient les informations de l'utilisateur.
  - **access** => Il contient toutes les informations nécessaires à l'application afin de déterminer les accès de l'utilisateur.
  - **refresh** => Spécifiquement dédié à la demande d'un nouvel access\_token.
- Tous sont signés pour assurer leur intégrité
- 2 validations possibles
  - Public Key
  - Endpoint Keycloak



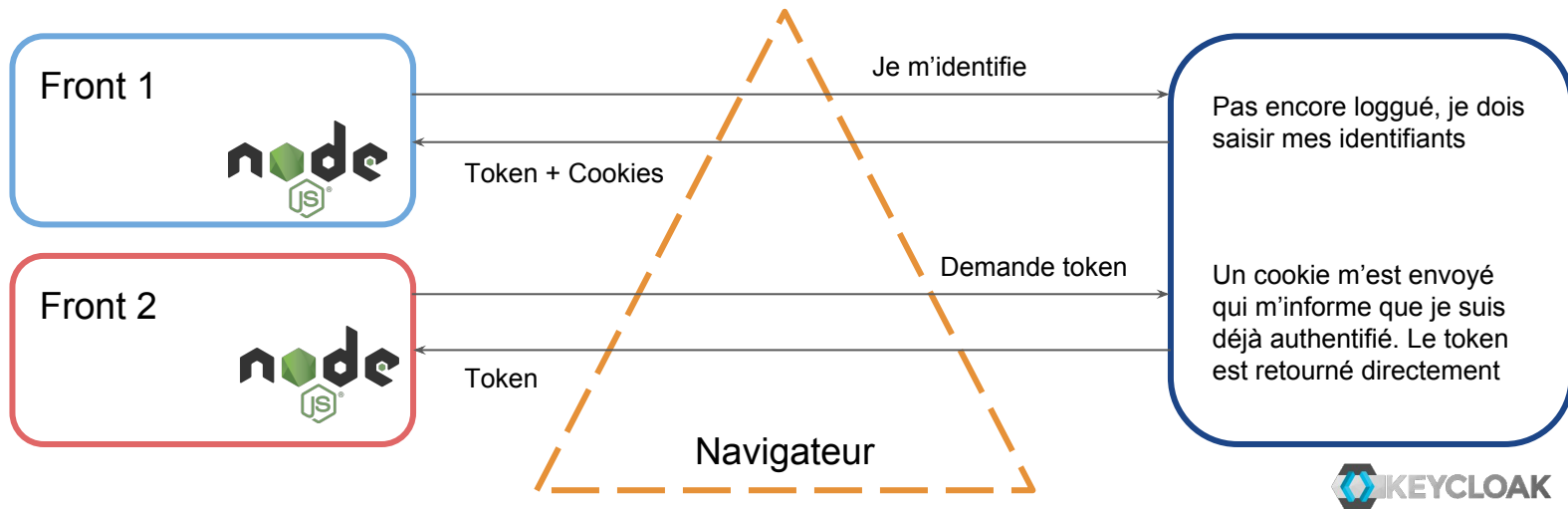
# Gestion des sessions ?

---

- Côté front
  - Local storage
  - Cookie
- Côté back
  - Stateless !
- Durée de vie des sessions
  - Short lived token
  - Long living refresh token
- Et l'invalidation ?!
  - Token introspection chaque appel
  - Révocation de sessions
- Attention à ce que vous stockez



# Le web SSO en détail



A grayscale photograph of two football players from behind, standing on a field. The player on the left wears a jersey with the number 86, and the player on the right wears a jersey with the number 17. Both are wearing helmets and shoulder pads. The background is a blurred field and sky.

# Frontend

---

# Gestion des rôles

---

- Utilisateurs
- Groupes
- Rôles
- Peuplé par LDAP / IDP / Services externes
- Attribution rôles
  - En fonction d'attributs de l'utilisateur
  - Selon des règles (javascript, Drools)
  - Applicable au niveau group / user
- Liés à un royaume



# L'autorisation côté back

---

- On valide le token jwt
  - Signature, expiration, issuer
  - Utilisation de la clé publique du serveur d'authentification (cachable)
  - Techno agnostique
- Rôle dans le JWT
- Attribution d'un contexte en fonction de l'appel
- On sécurise route par route pour plus de granularité



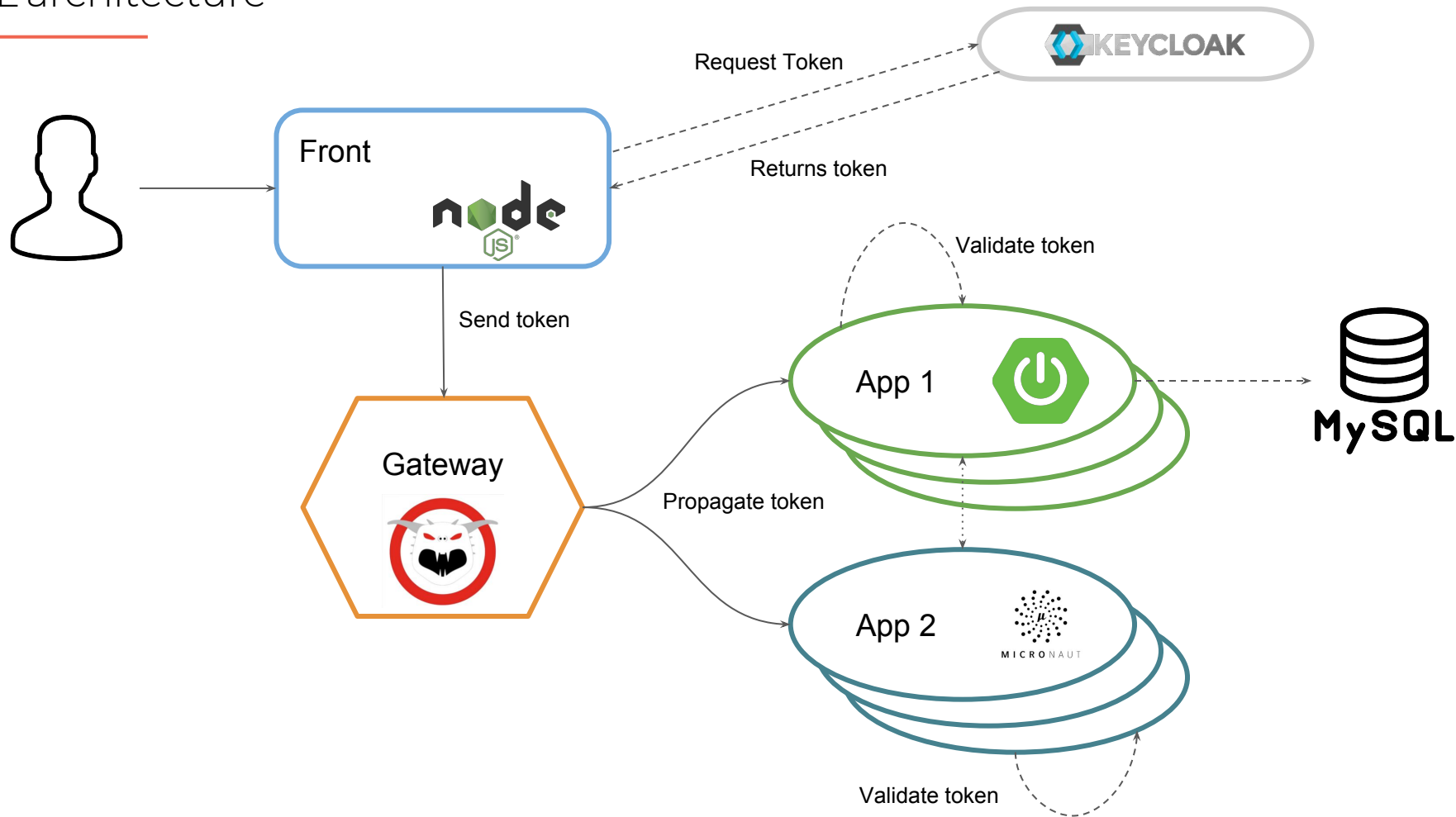


# Passer le token à ton voisin

- Pas besoin de s'authentifier à chaque fois
  - Une session est disponible
  - Transfert du token dans les headers
  - Validation à chaque saut
- Client credentials pour les applications
- Spring :
  - Spring cloud
  - OAuth2RestTemplate
  - Feign



# L'architecture



A dark, grainy background featuring a silhouette of a person surfing. The surfer is in a crouched position, riding a wave. The overall tone is moody and atmospheric.

# Backend

---

# Conclusion

---

- ▲ La sécurité, c'est plus aussi compliqué
- ▲ On peut gérer beaucoup plus facilement les droits
- ▲ Les standards sont bien ancrés
- ▲ Enfin des systèmes protégés



- ▼ Il faut prendre le temps de bien poser son archi
- ▼ La mise en place reste coûteuse
- ▼ Il faut sensibiliser les devs
- ▼ Maintenance/scalabilité de l'outil Keycloak



Discovery to Delivery

**Ippon.fr**

contact@ippon.fr

+33 1 46 12 48 48



@IpponTech