Gestion de l'authentification et des autorisations via la mise en place d'une "gateway" applicative

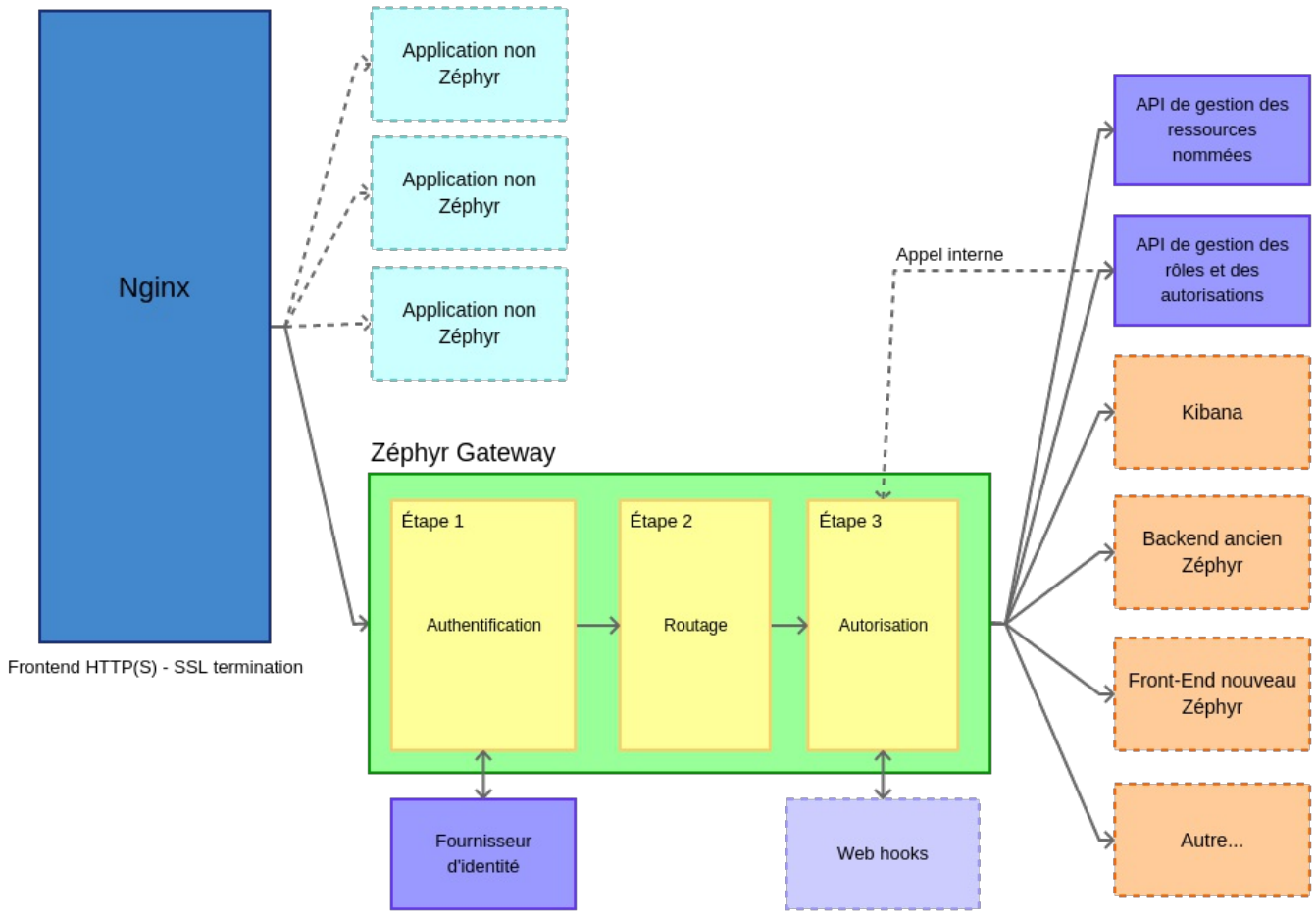
Définitions

- **Environnement Zéphyr** Ensemble des applications/services qui constitue le socle fonctionnel et technique du nouveau Zéphyr
- **Ressource** Une entité abstraite représentée par le tuple `(type,clé)`. Exemple: `(URL, /kibana/dashboard/1)`
- **Autorisation** Une autorisation est une association d'une **action** (représenté par un verbe) à une **ressource** (ou un filtre de ressource). Exemple: `GET -> (URL, /kibana/dashboard/1)`

Cas d'usage

- En tant que développeur/intégrateur, je veux pouvoir **authentifier les utilisateurs** puis les **autoriser à accéder à des applications web tierces** qui constituent l'environnement Zéphyr.
- En tant qu'administrateur, je veux pouvoir gérer les autorisations des utilisateurs de mon environnement Zéphyr à un endroit unique.

Schéma général



Fonctionnalités de la "gateway"

- Gestion de l'authentification avec des fournisseurs d'identité multiples (Clé d'API, OAuth, CAS, OpenID Connect, PAM...).
- Gestion des autorisations via un système d'accès basé sur les rôles et l'identification de ressources.
- Tranfert de l'identité aux applications tierces via cookie, entête `X-Auth` ou jeton JWT.
- Mécanisme d'interception du processus d'autorisation afin d'intégrer des briques logiques tierces.
- Routage (reverse proxy) vers des applications web tierces, intégré au mécanisme d'autorisation: **une URL est une ressource**.
- Déclaration des rôles, des autorisations, des ressources nommées et des règles de routage via une API REST.
- Gestion de "web hooks" au niveau des API de la gateway afin de rendre possible l'intégration des applications tierces.

Briques logicielles existantes

- Tyk - API Gateway
- OpenResty - Nginx + moteur LUA
- LemonLDAP::NG - Gestionnaire d'identité + Proxy + RBAC