Infrastructure

État de l'infrastructure de ChaTalK à ce jour

Présentation générale

- Déploiement initial avec Ansible = reproductibilité
- Utilisation de conteneurs Docker
  - architecture en microservices
  - pratique pour construire les différentes images
  - facilité de déploiement et de mise à jour
- Utilisation de Kubernetes
  - standard reconnu et très utilisé
  - montée en charge (*scaling*)
  - répartition de charge (*load-balancing*)
  - surveillance
  - déployé avec un *playbook* Ansible (Kubespray)
- Multi-site
  - un site à Illkirch
  - un site à Esplanade

Présentation du site d'Illkirch (cluster1)

- 4 machines virtuelles
  - 1 VM (chatalk-dumas) pour de l'utilitaire
    - 4vCPU, 8Go de RAM, 75Go de disque, Ubuntu Server 18.04
    - hébergement d'un stockage compatible S3 (Minio)
    - *runners* GitLab pour la CI
    - *registry* Docker privé
  - 3 VM (chatalk-balzac, chatalk-camus et chatalk-zola) pour le cluster Kubernetes avec 3 *masters*
    - 4vCPU, 4Go de RAM, 50Go de disque, Ubuntu Server 18.04

Présentation du site d'Esplanade (cluster2)

- 4 machines virtuelles (4vCPU, 4Go de RAM, 50Go de disque, Ubuntu Server 18.04)
  - 1 VM (chatalk4) inutilisée, potentiellement pour de l'utilitaire
  - 3 VM (chatalk1, chatalk2 et chatalk3) pour le cluster Kubernetes avec 3 *masters*

Ce qui est déployé sur les clusters

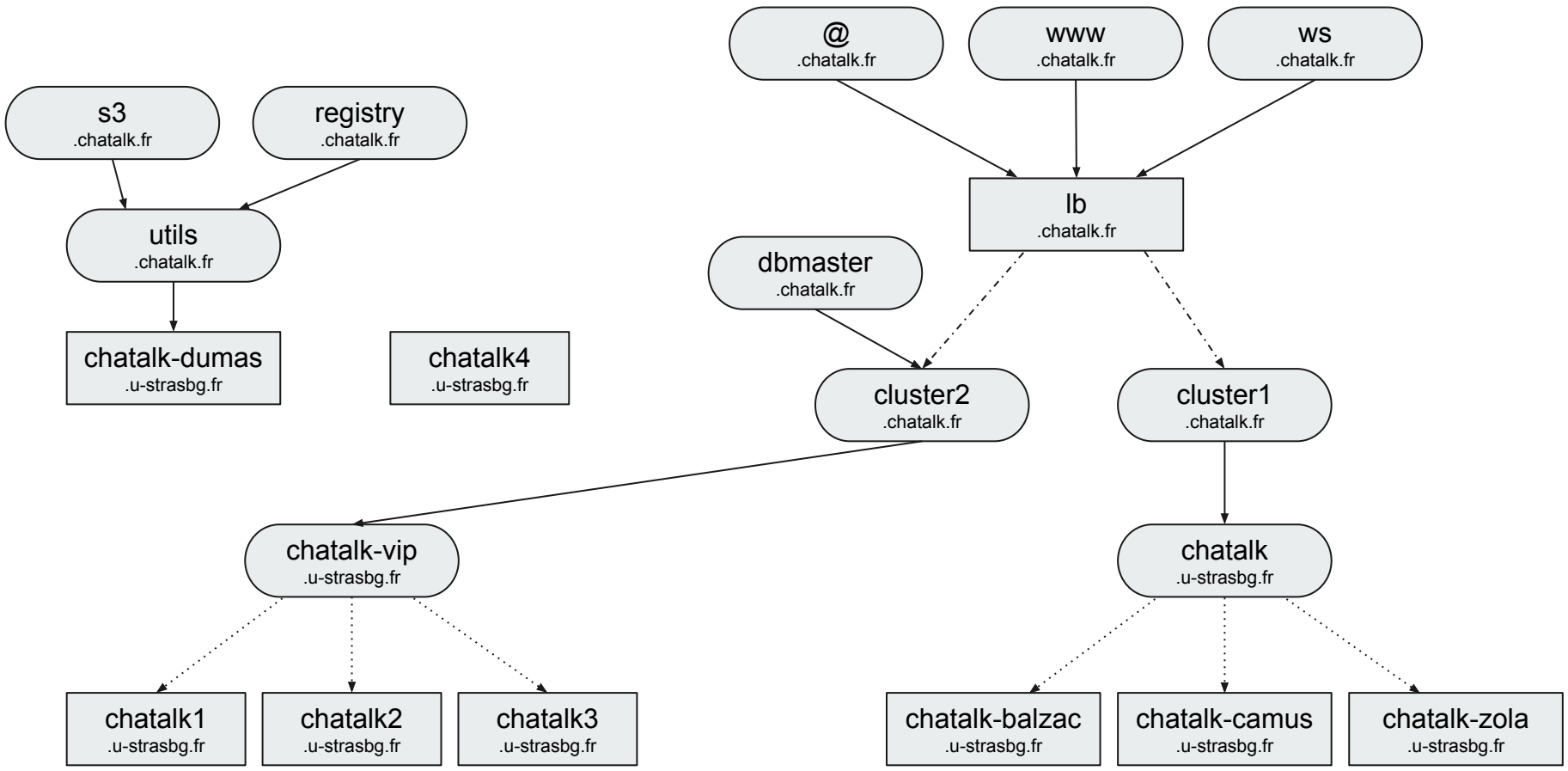
- un contrôleur d'*ingress* (*nginx-ingress-controller*) : rediriger le trafic entrant vers les bons services
- une solution d'*IP failover* (*metallb*)
  - une IP supplémentaire pour chaque site (*chatalk* pour Illkirch et *chatalk-vip* pour Esplanade)
  - l'IP sera «attribuée» à l'un des nœuds du cluster
  - si le nœud en question tombe, l'IP sera «attribuée» à un autre nœud
  - là où on fait pointer les entrées DNS
  - mécanisme déjà en place chez les *Cloud Providers* avec de vrais *load-balancer*
- bus de données (*nats / nats-streaming*)
  - léger
  - permet d'empiler les messages en cas de monté en charge soudaine
- gestionnaire de base de données Postgres (*stolon*)
  - haute disponibilité
  - fédération entre sites
- une stack de *monitoring (en cours)* avec Prometheus et Grafana
- notre application

Proxy (lb)

- Solution pour contourner la non-disponibilité d'IPv6
- VM hébergée à l'AIUS
- Point critique
- haproxy qui pour le moment redirige tout le trafic des ports 80 et 443 vers cluster1 (à terme fera du load-balancing entre cluster1 et cluster2)
- A une IPv4 et une IPv6
- Accessible publiquement (pas besoin de VPN pour accéder à notre application)

Problème induit : l'IP réelle du client ne sera pas remontée aux clusters

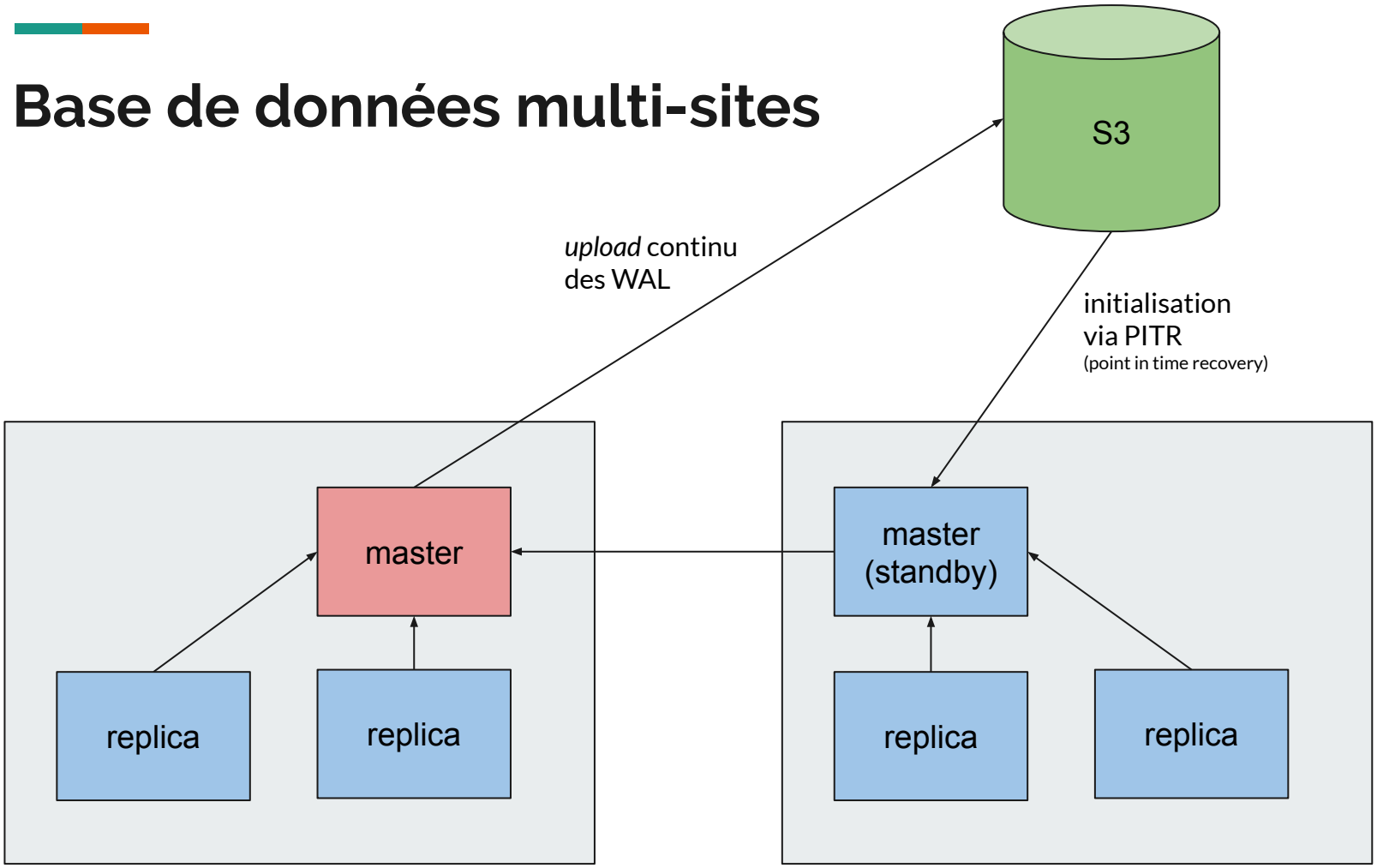
Vue d'ensemble



Base de données multi-sites

- Utilisation de Stolon
- Vraiment un problème complexe
- Choix de désigner un site comme étant le master (dans notre cas : cluster2)
- Mise en place d'un mode dégradé :
  - Si le cluster1 tombe, aucun impact concernant les accès à la base de données
  - Si le cluster2 tombe, plus aucune écriture ne peut être faite, mais les lectures restent possibles
- Les lectures se font depuis les réplicas

Base de données multi-sites



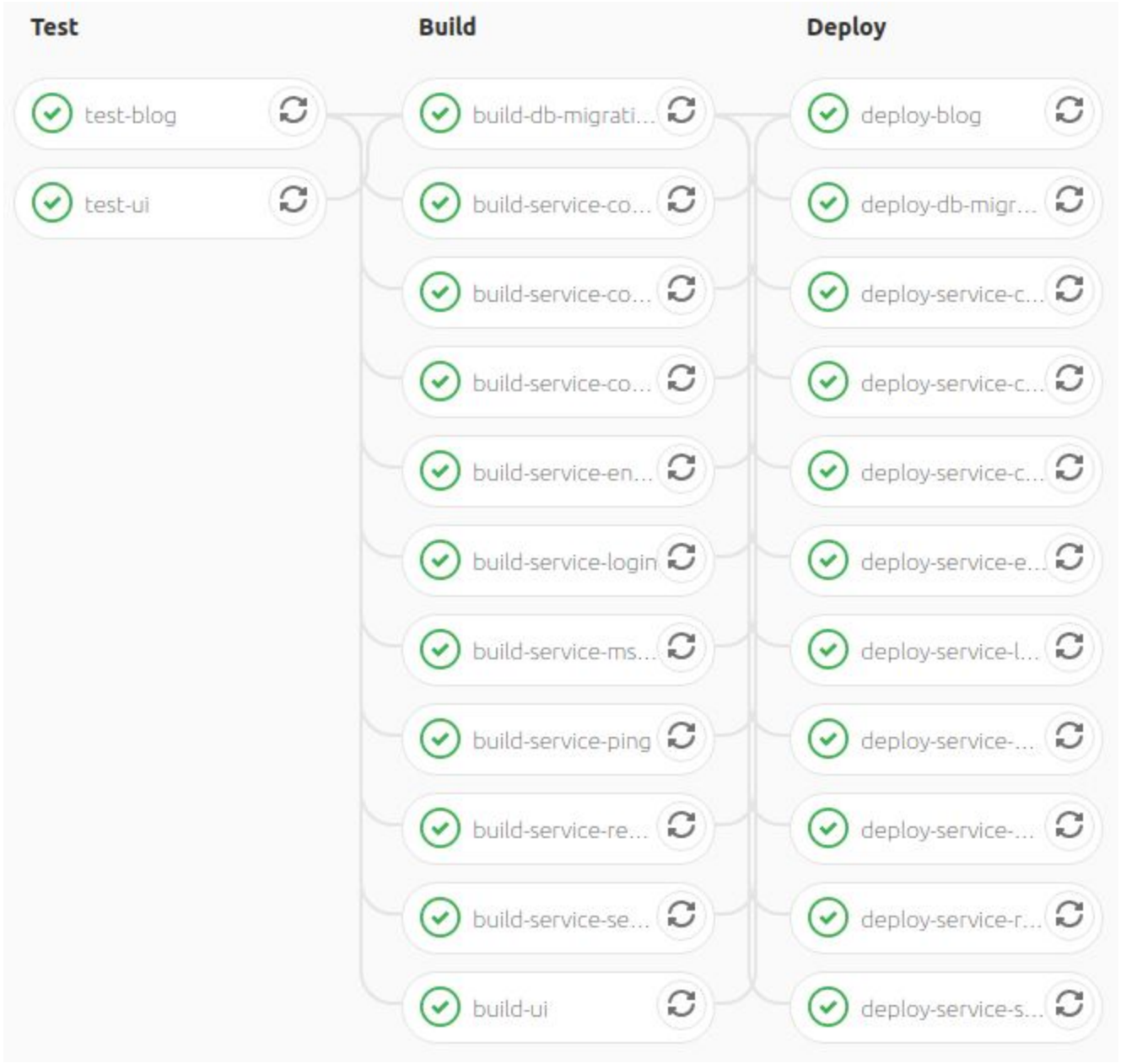
Intégration et déploiement continu

Utilisation de la CI de GitLab pour :

- tester si le *frontend* et le *blog* se lancent sans crasher directement
- construire les image Docker de chaque service
- déployer le blog
- prochainement : déployer l'application en production (presque prêt)

Optimisations :

- ne lancer un tâche seulement si le service a changé
- utilisation d'images Docker basée sur Alpine pour réduire la taille


Test
Build
Deploy
test-blog
test-ui
build-db-migrati...
build-service-co...
build-service-co...
build-service-co...
build-service-en...
build-service-login
build-service-ms...
build-service-ping
build-service-re...
build-service-se...
build-ui
deploy-blog
deploy-db-migr...
deploy-service-c...
deploy-service-c...
deploy-service-c...
deploy-service-e...
deploy-service-l...
deploy-service-...
deploy-service-...
deploy-service-r...
deploy-service-s...

Contraintes techniques

- pas d'IPv6 en interne :
  - les réseaux de la DNUM n'en propose pas (là on on est)
  - le dual-stack dans Kubernetes est en alpha : instable et trop contraignant
- contraintes propres aux sites
  - Illkirch : derrière un firewall géré par la DNUM, à chaque fois faire des demandes pour ouvrir des ports
  - Esplanade : doit faire attention à bien configurer moi-même le firewall (ufw)
- difficultés à faire des tests d'infrastructure
  - nécessite de prévenir bien à l'avance pour faire des tests de montée en charge, possible que de nuit
  - impossible pour le moment d'allumer les VM en cas d'extinction (doit attendre qu'on nous les rallume)

Chantiers en cours et à venir

- Intégration/déploiement continu
- Monitoring
  - collecter des métriques
  - faire de l'*auto-scaling*
  - voir l'état des différents services (https://status.chatalk.fr/)
- Fédérer le bus de données entre les sites (en attente de l'ouverture d'un port…)
- Load-balancer entre cluster1 et cluster2 depuis la VM de l'AIUS

Feedback ?

Est-ce que ça vous convient en termes de :

- Redondance
- Multi-sites
- Dual-stack
- Résistance aux pannes

Des questions ?