Architecture Dumbo

Victor Zimmermann Arthur Bruneel Antoine Caillet Clément Boutin
Baptiste Saclier

30 Juin 2019

Table des matières

1 Introduction

L'association Dumbo pour laquelle nous travaillons a pour but de mettre à disposition des associations du monde entier un support numérique solide pour tous leurs besoins tels que le stockage de fichiers, la communication interne, la messagerie mail et autres wiki de référence.

Dumbo a toujours une bonne réputation mais récemment les demandes se multiplient plsu que jamais et il est devenu difficile de répondre à toutes les demandes et l'infrastructure actuelle sature.

Dans ce but, il nous a été demandé de mettre en place une architecture de Cloud pour répondre aux besoins de toutes les associations utilisant les services de Dumbo.

Les technologies et l'architecture existante sont maintenant dépassées et ne répondent plus aux besoins actuels.

Nous avions ainsi pour but de mettre en place une architecture Cloud complète basée sur Docker, l'objectif étant d'avoir un système qui s'étende facilement à tout un datacenter ou même plusieurs distribués sur des sites différents en balance de charge, les containers se plaçant sur les nodes les moins chargés.

Nous avions ainsi l'accès au support de l'école pour pouvoir faire des demandes de créations de machines virtuelles qui représenteraient notre Cloud, avec un total de mémoire vive autorisée de 18Go, un total d'espace de stockage de 220Go et un total de 7vCPU pour faire tourner ces machines. Ayant une SLA de 2h sur la création ou modification de ces machines, il a fallu s'y prendre plsu tôt.

2 Innovation

2.1 Anticipation

Durant notre phase d'ingénierie, nous avons pu déduir les principaux problèmes suivants :
— Evolution trop rapide du système d'information
— Des projets d'évolution du système non pas abouti
— Trop de changement ont été effectués empêchant l'harmonisation des outils et pratiques
— Insatisfaction client croissante résultant en une perte financière
— Organisation non adaptée à l'évolution et le demande des besoins utilisateurs
— Objectifs d'efficacité, fiabilité, collaboration, résilience et de « time to market » non atteints.

Nous pouvons alors formuler des opportunitées que Dumbo peux saisir pour améliorer son infrastructure :
— Meilleur expérience utilisateur
— Maintenance simplifiée et plus rapide
— Montée en charge simplifiée et automatique
— Datacenters plus écologiques et moins consommateurs d'électricité
— Administration mutualisée et simplifiée
— Authentification unifiée des utilisateurs

2.2 Créativité

— Nous avons pris d'autres solutions de service que celles proposés de par nos connaissances ou en se renseignant de ce qui se faisait actuellement en entreprises.
— Nous avons utilisé un reverse proxy avec Traefik nous permettant ainsi de maitriser le port de l'host sur lequel écoutent le conteneur et avoir des informations en temps réel sur ce conteneur.
— Pour permettre aussi une bonne utilisation de notre solution d'un point de vue utilisateurs nous avons mis en place différents nom de domaine différents.

2.3 Clés du succès

— Une méthode de gestion de projet agile afin de se fixer un premier objectif et de s'adapter en fonction de la situation.
— Une bonne organisation au sein du projet comme une nomenclature précise,une répartition équitable du travail fournit.
— Une bonne communication au sein du groupe de projet pour voir où en ai le projet et respecter l'architecture du système.
— Une architecture Open Source qui permet donc une mise a disposition et une modification possible accessible à tous.

2.4 Innovation

Notre solution est innovante car elle utilise la conteneurisation Docker qui est une solution innovant en elle-même. Docker permet aujourd'hui de déployer n'importe quelle application dans n'importe quel environnement. Elle permet aux applications d'être indépendantes des infrastructures, et aux développeurs de se libérer des contraintes IT.

Comme en témoigne ce graphique l'innovation des composants Dockers ne cesse de croitre.

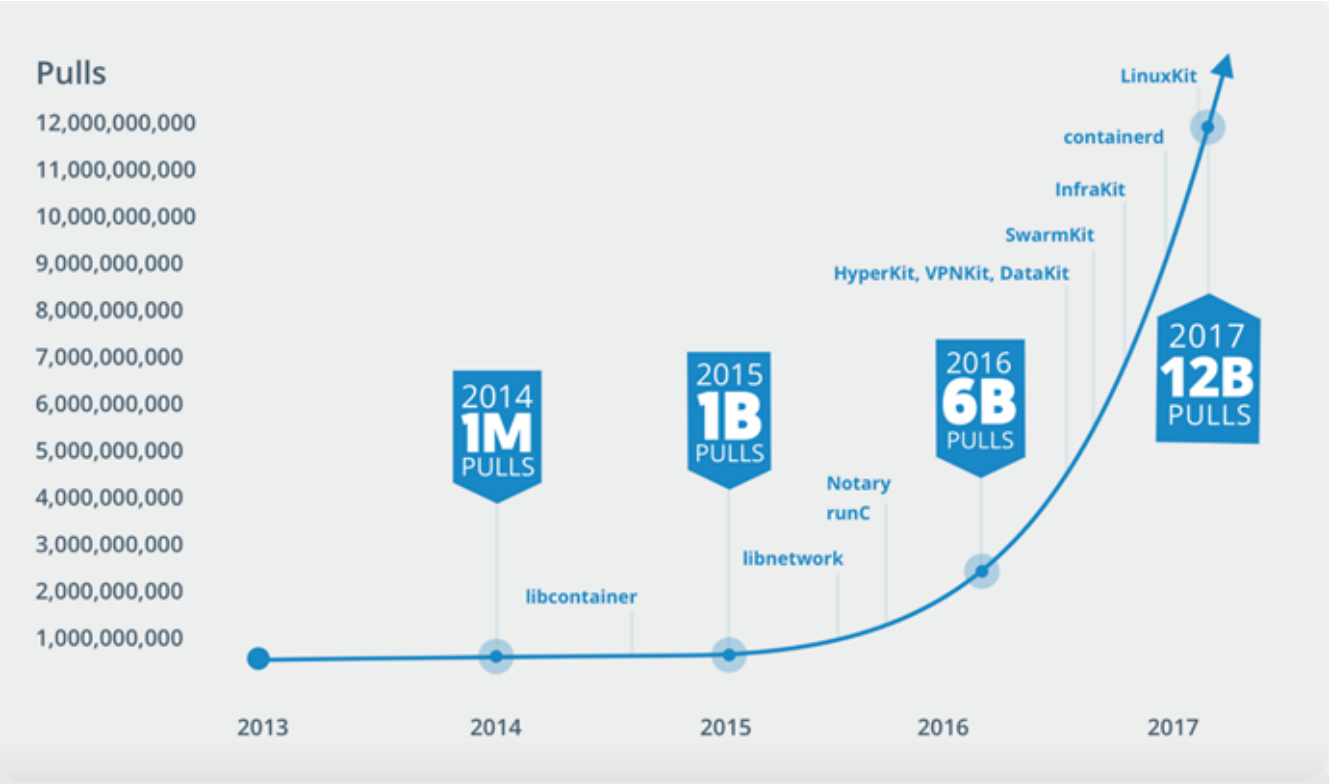


FIGURE 1 – Évolution des composants téléchargés depuis `hub.docker.com`

Il faut aussi savoir que les containers Docker, compte tenu de leur légèreté, sont portables de cloud en cloud.

Nous avons aussi utilisé que des solution Open Source ce qui nous a permis de pouvoir moduler nos services en fonction de nos besoins donc ceux du client.

2.5 Imagination

Nous avons imaginé une solution innovante et utilisant de nouvelles technologies. Nous avons tenté de pousser ses technologies tel que Docker, pour en obtenir une architecture innovante et sécurisée. Durant notre phase de conception nous avons cherché à créer une solution inédite aux problèmes de Dumbo. Nous avons séparés l'ensemble de nos services en de multiples réseaux permettant de regrouper les services en themes. Nous avons aussi utilisé Traefik et son système de reverse-proxy dynamique permettant de facilement changer les services disponibles et ainsi d'expérimenter sur de nouveaux services pour toujours plus satisfaire les clients de Dumbo.

3 Concevoir et créer

3.1 Besoins

L'association Dumbo est en plein remaniement de son architecture suite à une croissance fulgurante et il nous reviens de mettre en place une architecture complètement nouvelle, en utilisant docker. Il est en effet nécessaire de mettre en place des solutions qui permettront l'évolution et la gestion du nombre de clients qui ne cesse de

croître sans que celà n'impacte les utilisateurs existants. Un autre besoin est de concentrer les applications mises à dispositions des clients afin de faciliter leur accès et améliorer la productivité au sein des entreprises clientes. En effet des nos jours toutes les applications en Cloud se doivent de communiquer entre elles et leur interopérabilité est absolument nécessaire.

3.2 Enjeux et risques

Un tel remaniement implique de re-déployer une solution complète de A à Z, jusquà la répartition de l'utilisation du matériel. La nouvelle solution doit donc être testée de façon extensive avant d'être mise en production à la place de la solution existante car le montant de charge à supporter ne sera pas des moindres. Au-delà de ceci, l'utilisation de technologies de containerisation telles que Docker et d'orchestrateur tels que Docker Swarm nous permet de bien mieux gérer l'évolution de la demande et les différents pics d'utilisation de façon automatique.

3.3 Expérience utilisateur

Au-delà de la prouesse technique, notre but était également de fournir une expérience utilisateur plaisante et simplifiée.

Nous avons ainsi intégré des applications en chaînes autant que possible.

Par exemple, notre service Nextcloud est relié à un srevice LDAP pour l'authentification unifiée mais également à OnlyOffice pour l'édition de tableurs et documents textes à l'intérieur même de l'interface Nextcloud.

Dans un souci de clarté, nous avons également mis en place un système de DNS Inversé avec Traefik, ce qui en pratique permet aux utilisateurs d'accéder à l'application voulue en fonction du nom de sous-domaine.

Par exemple, se rendre sur chat.kore.sh nous donne accès au service Rocket.Chat, et ainsi pour chaque service.

3.4 Méthodologie/Travail collaboratif

Afin de s'organiser pour le travail en équipe, nous avons utilisé plusieurs outils de communication et de gestion de projets.

3.4.1 Discord

Discord est un client de chat type Slack que toute l'équipe utilisait et maitrisait déjà et nous avons donc crée des chaînes de discussions spécifiques au projet pour s'échanger des messages ou des ressources ou encore discuter de vive voix grâce aux chaînes vocales de Discord.

3.4.2 Trello

Dans le but d'organiser les différentes tâches du projet, les dates limites de rendu, etc. nous avons utilisé Trello en détaillant certains objectifs et en utilisant un code couleur pour chaque tâche afin d'avoir un suivi plus visuel de ce qui est fait et reste à faire.

3.4.3 Git

Afin de partager nos avancées tant au niveau de la configuration qu'au niveau de la rédaction des différents rapports, nous avons crée un dépôt GitHub où tout le monde pouvait ainsi pousser ses différents ajouts et modifications.

Nous avons également utilisé CircleCI afin d'automatiser la compilation de nos rapports rédigés en LaTeX.

4 Réalisation technique

Dans le cadre du projet Dumbo, nous devions réfléchir à une infrastructure permettant de déployer un grand nombre de services pour de nombreux clients. Cela pose de nombreux problèmes comme la gestion des différents espaces d'entreprise, la gestion de la charge au sein d'un cluster de machines et de multiple clients.

4.1 Infrastructure orientée containers

Nous avons déployé nos services au sein de containers Docker. Docker est une téchnologie permettant d'enfermer les applications déployées dans un système de fichiers isolé du reste du système. Cela permet aux différentes applications de s'éxécuter dans un environnement sécurisé indépendemment des autres applications qui pourraientt influencer le reste de l'installation. Cela permet aussi de séparer les différents services et de les redémarrer facilement en cas d'erreur. En effet, lors du crash d'un container, il suffit de le redémarrer ce qui à pour effet de réinitialiser son contenu et ainsi de reprendre sur une base saine. La persistance des données s'effectue par le montage de dossiers de stockage depuis la machine hôte. Ainsi, les données utilisateurs sont constantes et les services peuvent redémarrer à tout momment.

4.2 Orchestration multi-nodes

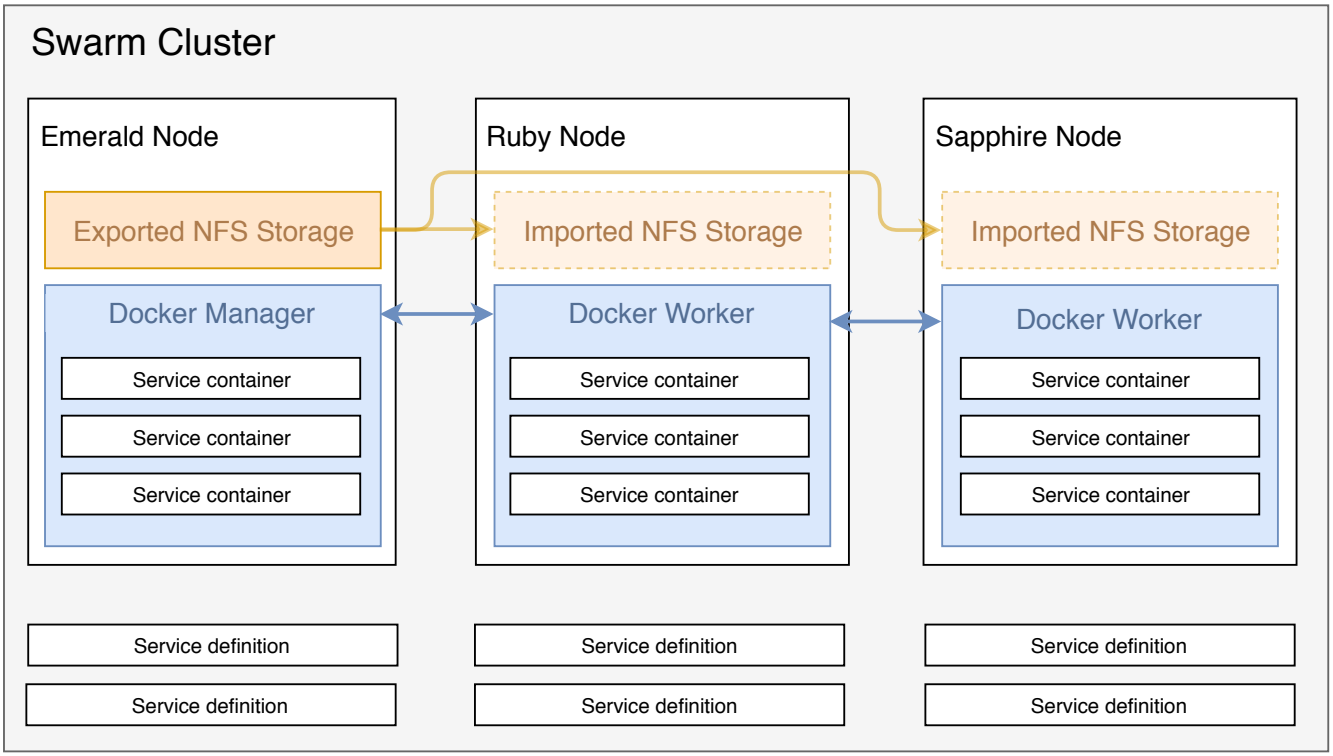


Figure 2 – Schéma de l'intrastructure d'orchestration

Notre infrastructure comprend plusieurs machines permettant de diffuser la charge sur l'ensemble d'un cluster de machines. On appelle l'ensemble de ces machines communiquantes, un Cluster et chaque membre de ce Cluster est appelé Noeud.

Le stockage est assuré par un disque présent sur le Noeud principal (Noeud Manager) disposant d'un grand espace de stockage. Ce disque est alors exporté via le protocole NFS vers les autres noeuds pour conserver un stockage synchronisé sur l'ensemble du cluster. Ainsi, les containers disposent d'un accès au stockage peu importe de leur emplacement dans le Cluster.

Un Cluster implique un système d'orchestration. Un tel système gère l'ensemble des containers exécutés sur chaque Noeud et définis quel container doit être démarré ou arrêté et sur quel machine. Dans notre cas, nous avons choisi Docker Swarm pour assurer cette Orchestration. Apres avoir constitué le cluster, il suffit de définir de de configurer les services qui seront déployés au sein de celui-ci.

Un service est une abstraction d'un container. Un service définis le squelette d'un container mais aussi le nombre de container qu'il faut déployer simultanément et d'autres paramètre permettant, notemment, de configurer le Reverse Proxy. Une fois les services définis, Docker Swarm se charge de lui même de démarrer les containers sur les différents Noeuds et de les connecter ensemble.

4.3 Services proposés

Pad Un Pad est un fichier texte très simple permettant l'utilisation d'une mise en forme basique tel que la mise en gras, la mise en italique, certains titres, etc. Un Pad peut aussi être partagé pour une édition en collaboration, ainsi plusiuers perosnnes de l'entreprise peuvent le lire et l'éditer en simultanée. Dans le cadre d'une entrperise, cela permet de partager rapidement des informations sans envoyer d'email ou de fichier Word. Cette techologie est très utile dans le cadre d'une rédaction de document en équipe, cela permet aux divers memebres de l'équipe de débattre et de modifier le texte sans se soucier de la mise en forme mais de se concentrer sur le fond. Pour la mise en place de ce service sur `pad.kore.sh`, nous avons choisi la technologie *Etherpad* qui est la plus avancée actuellement et qui propose de nombreuses fonctionnalités.

Gestion des fichiers La gestion des fichiers permet le stockage et le partage de fichiers dans un espace personnel pour chaque utilisateur. Chaque utilisateur dispose de ses propres fichiers qu'il peut partager, diffuser et modifier en ligne. Pour mettre a disposition des utilisateurs ce service sur `files.kore.sh`, nous avons déployé un serveur *NextCloud* permettant de partager et de gèrer des fichiers au travers du protocole WebDAV. En plus de ce serveur de fichiers, nous avons mis en place une solution *OnlyOffice* permettant d'éditer en ligne et à plusieurs les documents de la suite Office Microsoft. cela comprend un tableur, un traitement de texte mais aussi un créateur de présentations. On peut aussi noter que les fichiers NextCloud sont accessibles depuis un téléphone au travers de l'application NextCloud permettant ainsi un accès aux données même en mobilité.

Messagerie La messagerie est un élément important d'une entrperise, elle permet la communication avec l'ensemble des collaborateurs mais aussi avec les clients et toute autre personne. Nous avons mis en place deux éléments pour la gestion de cette messagerie. Dans un premier temps, le serveur des messagerie *Postfix* et *Dovecot* permettent d'envoyer et de recevoir les messages au travers de protocols SMTP et IMAP. Dans un deuxième temps, nous avons déployé une application de messagerie Web *SOGo* accessible à `mail.kore.sh/SOGo` gèrant à la fois les messages par la connexion au services précédemment mis en place mais aussi un Agenda et un carnet d'adresses.

Wiki Un Wiki est un logiciel permettant à chacun de créer et modifier une base de connaissance. Ce wiki est ouvert à tout, même aux modifications anonymes et permet de constituer une connaissance de groupe accessible en tout temps. Un système wiki est très utile pour gèrer les problèmes récurrents d'une entreprise tant au niveau des système d'information que d'informations générales sur le fonctionnement de l'entreprise. Nous avons utilisé le système *Gitit* sur `wiki.kore.sh` pour ce service. Gitit dispose d'une intrface web de modification et gère son historique de modification dans un repository git, cela rend son utilisation très simple et permet aussi une modification hors ligne en clonant ce repository sur sa propre machine.

Chat Le chat d'entreprise permet de communiquer efficacement entre membres d'une équipe, d'un service et même d'une entreprise. L'application Rocket.chat permet de disposer de canaux de communication entre tout les membres d'une équipe permettant une communication fluide et bien plus rapide que l'adresse Email. Ces systèmes de communication se sont généralisés dans les entreprise et permettent bien plus de fonctionnalités qu'un simple échange d'email. Pour mettre en place cette application sur `chat.kore.sh`, nous avons utilisé le système Rocket.chat. Ce système est plus avancé des systèmes de chat d'entreprise open source et est aussi disponible sur mobile.

Base de données La base de données de l'entreprise permet un stockage de données arbitraires et leur recherche pour une utilisation ultérieur. Pour cette utilisation, nous avons choisi le logiciel *CouchDB* permettant un stockage de données sans schéma défini. CouchDB dispose d'une API REST permettant de communiquer facilement avec un client arbitraire comme un script d'entreprise, des capteurs IoT, etc. CouchDB dispose aussi d'une interface utilisateur sur `db.kore.sh/_utils` permettant une consultation des données par des utilisateur humains.

Annuaire Un service d'annuaire permet de centraliser la gestion des utilisateurs de l'entreprise. Nous avons utilisé cette technologie pour garantir qu'un collaborateur créé dans la base de données de l'annuaire dispose d'un compte sur le Gestionnaire de fichiers, le Chat, la Messagerie, le réseau social et le forum. La base de données gère les noms d'utilisateurs et mots de masse de manière centralisée permettant un modification bien plus aisée des comptes utilisateurs mais aussi une sécurité plus importante car les mots de passes des utilisateur ne sont pas présents en plusieurs endroits, potentiellement vulnérables. Pour la mise en place de cet annuaire, nous avons utilisé le logiciel OpenLDAP qui est une référence dans le domaine.

Audioconférence et VoIP Les systèmes de VoIP permettant aux collaborateurs de communiquer par voie vocal ou vidéo entre eux. Cela passe par les protocol SIP et RTC que nous avons déployés avec l'application open source *Asterisk*. Chaque collaborateur peut alors utiliser une application de VoIP (appelée SoftPhone) sur smartphone ou sur PC. Il est aussi possible de configurer des téléphones physiques pour utiliser le système de VoIP mis en place.

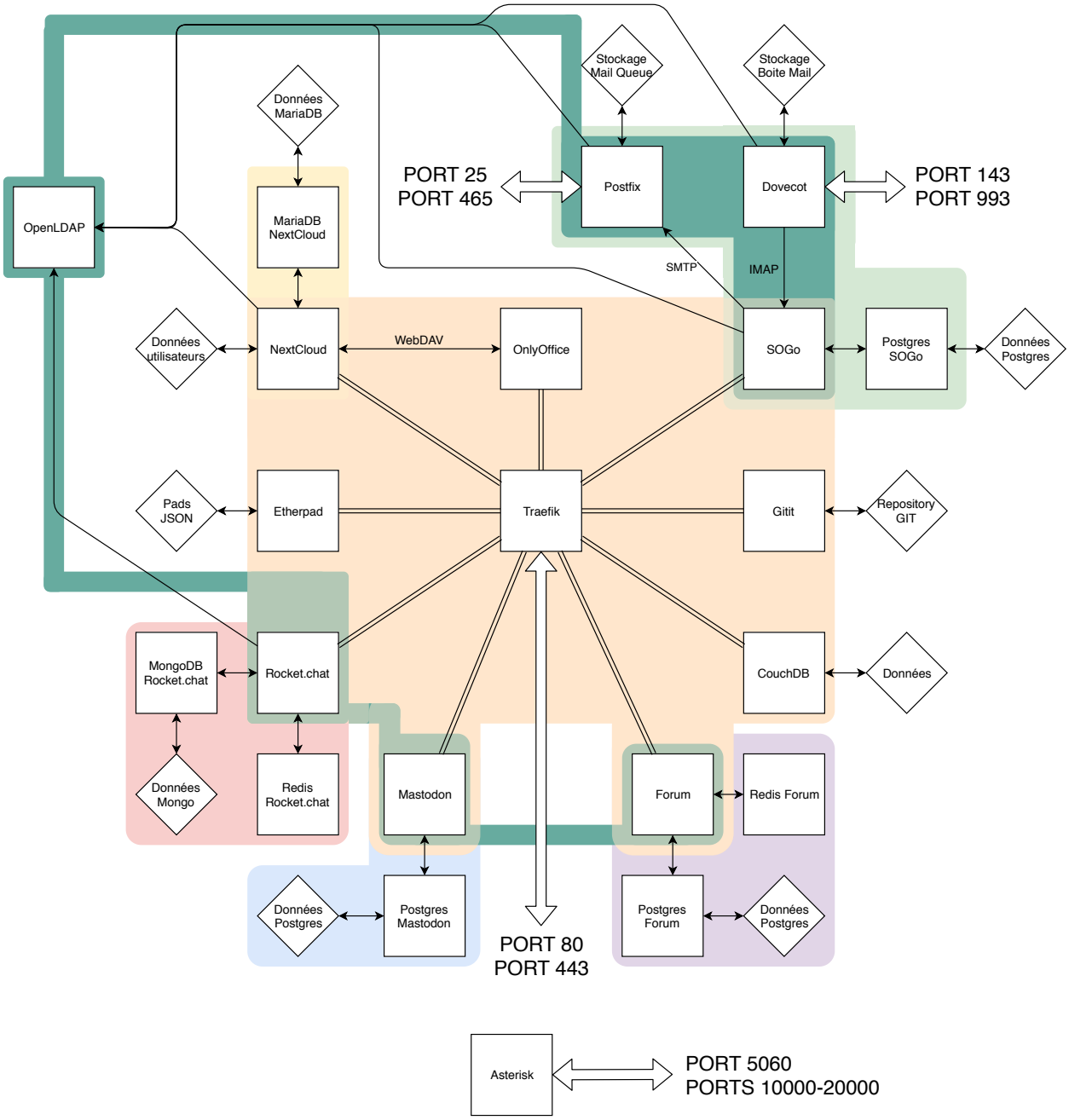
Réseau social Un réseau social d'entreprise permet une communication interne de moindre importance mais permattant aux collaborateurs de partager sur des sujets divers de manière convival et informel. Nous avons inclus le réseau social d'entreprise dans nos services sur `social.kore.sh`.

Forum Un forum est un site web permettant aux collaborateur de discuter autours de sujets définis. Ce service est très utile en entreprise pour encourager l'entraide entre collaborateurs. Cela permet aussi de faciliter les dialogues avec le support technique sur les difficultés que peuvent rencontrer les utilisateurs lors de leur utilisation du système d'information. Nous avons déployé un forum sur `forum.kore.sh`.

Exposition des services La majorité de ces services sont accessibles sur le Web. Le port définis pour la communication HTTP étant 80 et 443 pour sa version sécurisée, il est impossible d'exposer plusieurs applications sur la même addresses IP. Nous avosn donc mis en place un Reverse proxy se positionnant en entrée de toutes les communications vers les services web et distribuant les requètes aux services concernés sur la base du nom de domaine utilisé pour y accéder. Nous avons utilisé le reverse proxy *Traefik* permettant une configuration dynamique par l'observation du cluster. En effet, apres avoir déployé un service web, il suffit d'ajouter 4 labels à ce service pour configurer son acces depuis l'extérieur au travers de traefik. Cela rend le déploiement de services très simple et dynamique. Enfin, Trafik dispose d'un système de load balancing permettant de diffuser la charge des requètes sur plusieurs containers d'un même service.

4.4 Structure des services

Au sein de notre cluster, nous avons déployé ces services et leur dépendances avec une forte séparation garantissant une disponibilité même en cas de panne d'un service. L'infrastructure est aussi divisé en de nombreux réseaux permettant de séparer les usages de de chaque service.


Stockage Mail Queue
Stockage Boite Mail
Données MariaDB
PORT 25
PORT 465
Postfix
Dovecot
PORT 143
PORT 993
OpenLDAP
MariaDB NextCloud
SMTP
IMAP
Données utilisateurs
NextCloud
WebDAV
OnlyOffice
SOGo
Postgres SOGo
Données Postgres
Pads JSON
Etherpad
Traefik
Gitit
Repository GIT
MongoDB Rocket.chat
Rocket.chat
CouchDB
Données
Données Mongo
Redis Rocket.chat
Mastodon
Forum
Redis Forum
Données Postgres
Postgres Mastodon
Postgres Forum
Données Postgres
PORT 80
PORT 443
Asterisk
PORT 5060
PORTS 10000-20000
Stockage de données
Service
Communication bidirectionnel
Communication unidirectionnel
Exposition de port
Proxy Web
Réseau "Ldap"
Réseau "Web"
Réseau "Forum"
Réseau "Social"
Réseau "Chat"
Réseau "Files"
Réseau "Mail"

5 Mise en oeuvre et exploitation

5.1 Sécurisation

Afin de proposer un environnement de travail sécurisé, nous avons identifié deux risques potentiels concernant la messagerie et nous les avons réglés.

Expédier ses emails sans cryptage SSL revient à envoyer un courrier en langage clair, non chiffré. Ces emails non sécurisés peuvent être facilement interceptés et lus par un individu non autorisé. À travers l'utilisation du protocole de transmission TLS, les échanges de courriers électroniques peuvent être chiffrés. Si l'utilisateur appuie sur le bouton « envoyer », le message est transmis de manière codée et sera de nouveau rendu visible chez le destinataire à l'aide de la clé appropriée.

Le volume considérable d'emails peut affecter les ressources informatiques. 90 % des e-mails entrants sont constitués de spams, supprimer les messages indésirables avant qu'ils n'atteignent le réseau permet de libérer des ressources informatiques et de réduire les chances d'être infecté par un cybercriminel. C'est pourquoi nous avons opté pour une messagerie, sur laquelle nous avons configuré une protection antispam aux e-mails entrants.

5.2 Procédures d'exploitation

5.2.1 Ajouter un cluster

Ouvrez un terminal sur la machine sur laquelle vous voulez exécuter votre nœud.

La commande « docker swarm join » va ajouter le nœud sur lequel vous vous trouvez, dans le cluster spécifié via le token (*Le Token ID est généré par le nœud manager)* entré en argument.

```
1 $ docker swarm join <Swarm Token ID>
```

5.2.2 Déployer et/ou mettre à jour un service

Ouvrez un terminal sur la machine sur laquelle vous voulez exécuter votre nœud. Puis créer/modifier le fichier « docker-compose.yml » dans lequel figure la configuration du service.

```
1 $ vim docker-compose.yml
```

Une fois le fichier enregistré, utilisez cette commande afin de déployer le service, en spécifiant le nom du service. Cette commande va redémarrer seulement les services modifiés/créés

```
1 $ docker stack deploy -c docker-compose.yml <Service Name>
```

Vérifier l'état de vos services en temps réel.

```
1 $ watch docker service ls
```

5.2.3 Monter en charge

Ouvrez un terminal sur la machine sur laquelle votre nœud est exécuté.

La commande scale vous permet de mettre à l'échelle un ou plusieurs services répliqués, en augmentant ou en diminuant le nombre souhaité de répliques.

```
1 $ docker service scale <SERVICE-ID>=<NUMBER-OF-TASKS>
```

Vérifiez les informations de votre service.

```
1 $ docker service ps <SERVICE-ID>
```

```
Every 2,0s: docker service ls                                                                                    LV0201-Emerald: T

ID             NAME                          MODE         REPLICAS  IMAGE                                PORTS
ft6bno5mkhnn   kore-chat_mongo               replicated   1/1       mongo:latest
tvw92sddrlnz   kore-chat_reseau-social-app    replicated   1/1       koehn/diaspora:latest
xl14ejaz46ym   kore-chat_rocketchat           replicated   1/1       rocket.chat:latest
acynlv6uluux   kore-chat_social-postgres      replicated   1/1       postgres:10-alpine
o1zhbfcugjeh   kore-chat_social-redis         replicated   1/1       redis:latest
zbifvuuu6zs4   kore-db_couchDB               replicated   1/1       couchdb:latest
fvzd5oj7hdib   kore-files_db                 replicated   1/1       mariadb:latest
rxktmp8s3hcf   kore-files_nextcloud          replicated   1/1       nextcloud:latest
dd6u4b82r969   kore-files_onlyoffice         replicated   1/1       onlyoffice/documentserver:latest
god46vye7tuj   kore-forum_discourse          replicated   1/1       bitnami/discourse:2
uyy4cohbky6i   kore-forum_postgresql         replicated   0/1       bitnami/postgresql:latest
vnesbt49rqi8   kore-forum_redis              replicated   0/1       bitnami/redis:latest
dgs7qu8abeef   kore-forum_sidekiq            replicated   1/1       bitnami/discourse:2
2di5ljzznkl6   kore-ldap-openldap            replicated   1/1       osixia/openldap:latest               *:389->389/tcp
c4izl14d2az9   kore-mail_mail                replicated   1/1       tvial/docker-mailserver:latest       *:25->25/tcp, *:143->143/tcp, *:587->587/tcp, *:993->993/tcp
zq1cwe0i4eek   kore-mail_sogo                replicated   1/1       jenserat/sogo:latest
bll5a6k8g5kk   kore-mail_sogodb              replicated   1/1       postgres:latest
093h4f98s1ww   kore-pad-etherpad             replicated   1/1       etherpad/etherpad:latest
qw4nvh4ljgig   kore-social_social-app         replicated   1/1       koehn/diaspora:latest
rxgc7azz0jv6   kore-social_social-postgres    replicated   1/1       postgres:10-alpine
yld9yja4f89p   kore-social_social-redis       replicated   1/1       redis:latest
jqyi6590xgo0   kore-traefik                  replicated   1/1       traefik:latest                       *:80->80/tcp, *:3389->8080/tcp, *:8080->8080/tcp
54ca54pqkfrp   kore-wiki_wiki                replicated   1/1       bringnow/gitit:latest
```

FIGURE 3 – Kore services

5.3 Cycle de vie

Notre infrastructure possède une certaine tolérance aux pannes, grâce à Docker Swarm, nous avons définie 3 nœuds manager ce qui permet au logiciel d'avoir une tolérance aux pannes d'une machine sans impacter les possibilités d'administration.

Malgré cela un incident peut toujours arriver et c'est pourquoi, la DSI de l'association Dumbo a à sa disposition tous les fichiers « Docker-compose.yml » comportant la configuration des différents services de l'environnement numérique. Cela permet par exemple de revenir à une version antérieure d'un service après une mise à jour problématique.

De plus le Docker Swarm que nous avons mis en place permet à la DSI de redéployer ses services et de les partager de façon rapide et efficace pour pouvoir garantir une reprise d'activité rapide de la production après un incident non souhaité.

Cette configuration permet également une amélioration continue des services en permettant une mise à jour simple des services pour tous les utilisateurs.

Afin de remettre en état ou de mettre à jour un service, référez-vous à la partie procédures d'exploitation.

FIGURE 4 – Cycle de vie d'un service

5.4 Amélioration et évolution

Service de monitoring Afin de répondre le plus efficacement possible aux besoins de l'association et de prévoir d'éventuels problèmes de surcharge, il serait intéressant d'intégrer une solution de monitoring tel que Prometheus qui permet de collecter des mesures de plusieurs cibles préalablement définies.

Système de fichier décentralisé Cette évolution permettra de garantir un accès aux documents partagés sur le réseau pour les utilisateurs, même en cas d'arrêt du serveur master procédant à la synchronisation des fichiers grâce à une solution du type Ceph.

Adapter les applications aux besoins des utilisateurs Dans une optique d'amélioration continue il serait intéressant de demander l'avis des utilisateurs concernant les différentes applications. Ceci permettra de répondre au mieux aux besoins utilisateurs et d'augmenter la qualité d'expérience.

6 Rôle et responsabilité de l'ingénieur.

6.1 Responsabilité Juridique

Incontestablement, Docker s'adresse à trois grands profils de clients : les early adopters, les utilisateurs qui découvrent cet environnement et ceux qui ne l'ont pas encore utilisé. Pour autant, quel que soit son niveau de maturité, la sécurité reste toujours un obstacle à franchir.

Concrètement, lorsque l'on utilise des « containers », la sécurité traditionnelle ne suffit pas. On ne sécurise pas une infrastructure « containérisée » comme on sécurise son infrastructure virtualisée. Il est donc important de bien comprendre les enjeux de la sécurité dans une infrastructure « containérisée », ainsi que les risques associés pour les intégrer efficacement dans sa politique de sécurité.

Evaluation de l'infrastructure containerisée afin de déterminer les différences de sécurisation de la solution.

Plus de composants Dans une infrastructure virtualisée, les flux sont plus au moins maitrisés. Dans une infrastructure « containérisée », cela se complique au regard des architectures complexes type « microservices », où le code est éclaté sur différents composants. Pour les équipes sécurité, il faut donc s'adapter à cette multitude de flux.

Durée de vie courte des containers En comparaison avec les machines virtuelles, un container a une durée de vie plus courte. Les équipes de développement vont donc constamment recréer des containers.

Quels changements au niveau des risques ? Les risques sont pratiquement identiques à ceux des infrastructures virtualisées ou classiques. Cependant, certains points sont intéressants à préciser :

Risques sur les configurations par défaut Garder les configurations par défaut amène un risque important dans une infrastructure conteneurisée. Nous avons donc personnalisé chacune de nos configuration afin de fournir un service unique et sécurisé répondant à chacun des besoins de votre entreprise.

Le top ten owasp reste valable Avec les containers, on utilise toujours les mêmes protocoles de communication : les attaques applicatives classiques sont donc courantes. Nous avons donc créé notre infrastructure en prenant en compte les dix attaques les plus fréquentes de cette années.

Attaque réseau Si un pirate prend la main sur un container, il pourra lancer un scan réseau et faire des déplacements latéraux. Comme il y a un manque de visibilité, les équipements de sécurité ne vont pas détecter ce scan, et le pirate pourra facilement passer d'un container à l'autre.

6.2 Sécurisation des opérations sur Docker

Voici les quelques bonnes pratiques que nous avons mis en place :
— Créer une partition physique séparée pour Docker
— Maintenir le système et les images à jour
— Éviter le stockage de secrets au sein d'une image
— Ne pas utiliser n'importe quel registre, nous n'utilisons que le hub docker.

— Contrôler les communications entre containers : nous avons mis en place des réseaux spécifiques pour chaque liaison. De cette manière, vos services peuvent uniquement communiquer avec leurs propres applications.
— Vérifier les sources de construction des images utilisées par les conteneurs et assurer un suivi de leurs évolutions.

6.3 Sécurisation des communications

Nous proposons la mise en place du protocole de communication HTTPS afin de sécuriser les échanges entre nos serveurs et le client. Grâce à cette technologie nous pouvons garantir au client que celui-ci communique uniquement avec nos serveurs lors de l'échange de données confidentielles. L'HyperText Transfer Protocol Secure : HTTPS, est la combinaison du HTTP avec une couche de chiffrement TLS. HTTPS permet au visiteur de vérifier l'identité du site web auquel il accède, grâce à un certificat d'authentification émis par une autorité de certification : une autorité tierce, réputée fiable et faisant généralement partie de la liste blanche des navigateurs internet. Il garantit théoriquement la confidentialité et l'intégrité des données envoyées par l'utilisateur et reçues du serveur.

6.4 Sécurisation des données personnelles : RGPD

La protection des données personnelles nécessite de prendre des mesures techniques et organisationnelles appropriées afin de garantir un niveau de sécurité adapté au risque.

Sensibiliser les utilisateurs Faire prendre conscience à chaque utilisateur des enjeux en matière de sécurité et de vie privée.

Authentifier les utilisateurs Reconnaître ses utilisateurs pour pouvoir ensuite leur donner les accès nécessaires. Par exemple dans le cas de notre solution nous utilisons LDAP qui est un protocole permettant l'interrogation et la modification de notre service d'annuaire.

Gérer les habilitations Limiter les accès aux seules données dont un utilisateur a besoin.

Tracer les accès et gérer les incidents Journaliser les accès et prévoir des procédures pour gérer les incidents afin de pouvoir réagir en cas de violation de données (atteinte à la confidentialité, l'intégrité ou la disponibilité).

Protéger le réseau informatique interne Autoriser uniquement les fonctions réseau nécessaires aux traitements mis en place.

Sécuriser les serveurs Renforcer les mesures de sécurité appliquées aux serveurs.

Sauvegarder et prévoir la continuité d'activité Effectuer des sauvegardes régulières pour limiter l'impact d'une disparition non désirée de données.

Archiver de manière sécurisée Archiver les données qui ne sont plus utilisées au quotidien mais qui n'ont pas encore atteint leur durée limite de conservation, par exemple parce qu'elles sont conservées afin d'être utilisées en cas de contentieux.

Encadrer la maintenance et la destruction des données Garantir la sécurité des données à tout moment du cycle de vie des matériels et des logiciels.

Gérer la sous-traitance Encadrer la sécurité des données avec les sous-traitants.

Sécuriser les échanges avec d'autres organismes Renforcer la sécurité de toute transmission de données à caractère personnel.

Protéger les locaux Renforcer la sécurité des locaux hébergeant les serveurs informatiques et les matériels réseaux.

Chiffrer, garantir l'intégrité ou signer Assurer l'intégrité, la confidentialité et l'authenticité d'une information.

6.5 L'impact sur l'environnement

Le groupe doit analyser l'impact de la solution sur la société et l'environnement en s'appuyant par exemple sur une politique Green IT ou de développement durable

- Les matériels qui ne demandent pas un fonctionnement permanent sont systématiquement éteints et non mis en veille. Grâce à la containerisation il est possible lors de perte de charge de limiter l'utilisation des machines afin de réduire la consommation d'énergie.
  - Objectifs :
    - Permettre une réduction de la consommation énergétique supérieure à celle relevée lorsque les appareils sont en veille
    - Adopter un comportement responsable
- Les calculs et les tâches sont répartis sur différents serveurs dédiés en fonction de la nature de leurs tâches (données, mails, vidéos etc.)
  - Objectifs :
    - Maîtriser l'impact sur la consommation énergétique
    - Réduire l'impact écologique des échanges d'information
- Dans le cadre de travaux collectifs, l'usage d'outils collaboratifs en ligne est privilégié à l'échange de courriels.
  - Objectifs :
    - Réduire l'impact écologique des échanges d'information
    - Sensibiliser les personnels
    - Responsabiliser les personnels
    - Maîtriser la politique GreenIT de l'entreprise
- Anti-SPAM : Tous les spams sont supprimés
  - Objectifs :
    - Limiter l'impact écologique du stockage de l'information
    - Induire des comportements responsables

7 Conclusion

Nous avons installé docker et tenté de mettre en place quelques services avec Rancher, puis avons finalement choisi d'utiliser Docker Swarm comme orchestrateur.

Nous avons pu mettre en place la grande majorité des services demandés malgré certains retards techniques.

Certains ports étant bloqués nous n'avons pas pu mettre en place des certificats sur chaque sous-domaine afin d'avoir toutes nos applications en HTTPS et nous n'avons également pas pu implémenter de solution de VoIP.

Les utilisateurs des associations peuvent néanmoins rédiger des documents en diverses formats, les partager, discuter sur un chat, partager des informations générales sur le Wiki, échanger des mails et agencer leurs calendriers et même réaliser des appels et visioconférences à l'aide de Nextcloud Talk.