



**Nicolas Bidel**

Mars 2023



Sujet :

Upgrade architecture multi-instances Dolibarr



**nextformation**  
déjà 20 ans !



## VERSIONS DU DOCUMENT

Numéro	Date	Auteur	Action
1	10 octobre 2022	Nicolas BIDEL	Création du document
2	7 novembre 2022	Nicolas BIDEL	Ajout page de garde et tête/pieds de pages
3	15 novembre 2022	Nicolas BIDEL	Ajout des sections Introduction et remerciements
4	18 novembre 2022	Nicolas BIDEL	Ajout de la section partie prenante
5	1 <sup>er</sup> décembre 2022	Nicolas BIDEL	Ajout de la section Projet
6	22 décembre 2022	Nicolas BIDEL	Ajout du glossaire et de la bibliographie et Webographie
7	9 janvier 2023	Nicolas BIDEL	Ajout introduction, remerciements, parties prenantes
8	14 janvier 2023	Nicolas BIDEL	Ajout contexte méthodologie et expression des besoins
9	16 février 2023	Nicolas BIDEL	Ajout contexte technique
10	1 <sup>er</sup> mars 2023	Nicolas BIDEL	Ajout analyse fonctionnelle, GANNT, RACI, analyse de risques
11	12 mars 2023	Nicolas BIDEL	Ajout des illustrations du projet
12	15 mars 2023	Nicolas BIDEL	Fin de rédaction des parties techniques
13	17 mars 2023	Nicolas BIDEL	Ajout de la conclusion
14	18 mars 2023	Nicolas BIDEL	Ajout de la section Glossaire et Bibliographie / Webographie
15	22 mars 2023	Isabelle BIDEL	Révision du texte pour seconde correction

# 1. Table des matières

1.	Table des matières .....	3
2.	Table des illustrations .....	5
3.	Introduction .....	6
4.	Remerciements .....	8
5.	Parties prenantes (Acteurs) .....	9
5.1.	Centre de formation.....	9
5.2.	Entreprise .....	9
5.3.	Personnel .....	10
6.	Projet.....	11
6.1.	Contexte .....	11
6.2.	Méthodologie.....	11
6.3.	Expression des besoins .....	12
6.3.1.	Cahier des charges .....	12
6.3.2.	Echéances.....	13
6.4.	Contexte technique.....	13
6.4.1.	Infrastructure actuelle .....	13
6.5.	Réalisation.....	16
6.5.1.	Analyse fonctionnelle.....	16
6.5.2.	Planning prévisionnel - GANTT.....	19
6.5.3.	Répartition des tâches - RACI.....	19
6.5.4.	Analyse des risques .....	21
6.5.5.	Infrastructure prévisionnelle .....	22
6.5.6.	Mise en œuvre .....	25
6.5.6.1.	Hyperviseur .....	25
6.5.6.2.	Templating des VM .....	27
6.5.6.3.	Routage avec VyOS .....	28
6.5.6.4.	Machine virtuelle « Services » .....	29
6.5.6.5.	Supervision et métriques .....	33
6.5.6.6.	ERP/CRM avec Dolibarr .....	37
6.5.6.7.	Sécurité .....	38
7.	Conclusion.....	43
8.	Glossaire.....	44
9.	Bibliographie – Webographie .....	46
10.	Annexes.....	47

10.1.	Annexe 1 : Diagramme de GANTT .....	47
10.2.	Annexe 2 : Script « clean-up ».....	48
10.3.	Annexe 3 : Configuration de VyOS .....	48
10.4.	Annexe 4 : Configuration de WireGuard sur VyOS .....	50
10.5.	Annexe 5 : Configuration de bind9 .....	50
10.6.	Annexe 6 : Fichier docker-compose stack « service ».....	51
10.7.	Annexe 7 : Fichier docker-compose stack « supervision » .....	52
10.8.	Annexe 8 : Fichier prometheus.yml .....	52
10.9.	Annexe 9 : Fichier configuration alertmanager.yml.....	53
10.10.	Annexe 10 : Docker-file image Backend-Dolibarr .....	54
10.11.	Annexe 11 : Script création infrastructure Dolibarr.....	55
10.12.	Annexe 12 : Script configuration instance Dolibarr cliente .....	57

## 2. Table des illustrations

Figure 1 - Méthodologie AGILE	11
Figure 2 - Méthode SCRUM	12
Figure 3 - Versions Ubuntu	14
Figure 4 - Interface graphique Webmin	15
Figure 5 - Diagramme "bête à corne"	16
Figure 6 - Diagramme "pieuvre" des fonctions principales	17
Figure 7 - Diagramme "pieuvre" des fonctions de contraintes	18
Figure 8 - Diagramme de Gantt	19
Figure 9 - Matrice RACI	20
Figure 10 - Matrice de criticité	21
Figure 11 - Exemple de plan de gestion des risques	22
Figure 12 - Infrastructure matérielle prévisionnelle	23
Figure 13 - Infrastructure réseau globale	23
Figure 14 - Infrastructure réseau Baremetal	24
Figure 15 - ESXi Hôte sur DELL R620	26
Figure 16 - Machines virtuelles	26
Figure 17 - Mise en réseau	26
Figure 18 - Datastore de stockage	26
Figure 19 - Configuration du client Windows	29
Figure 20 - Conteneur vs machine virtuelle	30
Figure 21 - Ecosystème docker	30
Figure 22 - Interface Nginx Proxy Manager	31
Figure 23 - NPM Configuration d'un host	32
Figure 24 - Portainer et la gestion des conteneurs Docker	32
Figure 25 - Vue d'ensemble de la stack supervision	33
Figure 26 - Endpoint configurés dans Prometheus	34
Figure 27 - Métriques exposées par Node Exporter	34
Figure 28 - Le dashboard de Node Exporter sur Grafana	35
Figure 29 - ERP/CRM Dolibarr	37



### 3. Introduction

#### FRANÇAIS :

Ce mémoire a pour but principal de détailler le projet que j'ai pu mettre en place durant cette formation et mon stage auprès de la société InfraS, société de conseil en informatique de gestion (ERP/CRM).

Je décrirais donc rapidement dans ce document le centre de formation Nextformation et la société InfraS. Je présenterais ensuite le projet professionnel que j'ai pu construire, ainsi qu'une première ébauche de sa mise en service lors du stage de fin d'étude. La présentation se déroulera comme suit :

- Contexte du projet et méthodologie
- Présentation de l'environnement existant
- Axes de travail et mise en œuvre

La société InfraS est une société d'audit et de conseil en informatique de gestion qui existe depuis 2015. Elle accompagne les entreprises dans l'intégration et la personnalisation de leur outil de gestion. Elle est labellisée « Dolibarr Preferred Partner ». Elle agit donc sur la partie intégration/infogérance, développement et hébergement de cette solution de type SaaS.

L'infrastructure est basée sur un serveur sous Ubuntu 20.04 LTS, déployé autour d'une solution de type LAMP. Elle héberge actuellement environ 80 instances client Dolibarr (ERP/CRM), tous déployés à la main via un script bash. Les sauvegardes sont effectuées sur un second serveur et sur support physique. Les différents axes de travail sont dans un premier temps :

- Faire évoluer l'infrastructure
- Automatiser le déploiement d'instances ERP/CRM
- Superviser l'infrastructure

Dans un second temps, il sera prévu de faire encore évoluer l'infrastructure. Une attention particulière devra être faite sur la sauvegarde des données, scalabilité et la haute disponibilité de l'infrastructure pour garantir un plan de reprise d'activité (PRA) rapide.

## ANGLAIS :

The main purpose of this thesis is to detail the project that I was able to set up during this training and my internship with the company InfraS, a consulting company in management information technology (ERP/CRM).

I will therefore quickly describe in this document the Nextformation training center and the InfraS company. Finally, I will present the professional project that I was able to build, as well as a first draft of its commissioning during my end-of-study internship. The presentation will proceed as follows :

- Context of the project and methodology
- Presentation of the existing environment
- Areas of work and implementation

InfraS is an audit and management IT consulting company that exists since 2015, supporting companies in the integration and customization of their management tool. Sylvain Legrand is a Dolibarr specialist and is labeled "Dolibarr Preferred Partner". It therefore acts on the integration/outsourcing, development and hosting part of this SaaS-type solution.

The infrastructure is based on a simple server under Ubuntu 20.04 LTS, deployed around a LAMP type solution. It currently hosts about 80 Dolibarr client instances (ERP/CRM), all deployed by hand via a bash script. Backups are performed on a second server and on physical media. The different areas of work are initially:

- Develop the infrastructure
- Automate the deployment of instances
- Supervise the infrastructure

In a second step, it will be planned to further develop the infrastructure. Particular attention should be paid to data backup, the scalability and high availability of the infrastructure to guarantee a rapid disaster recovery plan (DRP).

## 4. Remerciements

---

La réalisation de ce mémoire a été possible grâce au concours de plusieurs personnes à qui je voudrais témoigner toute ma gratitude.

Je voudrais tout d'abord adresser ma reconnaissance à mon tuteur de stage, Sylvain LEGRAND, pour sa patience et sa disponibilité qui a contribué à alimenter ce projet.

Un remerciement également à tous les formateurs sélectionnés par NEXTFORMATION auprès desquels j'ai pu apprendre, et qui m'ont fourni les outils nécessaires à la réussite de cette reconversion. Je tiens à témoigner également mes remerciements à Yassine pour toute l'énergie qu'il a mise dans cette expérience éreintante !

Merci à l'équipe pédagogique de NEXTFORMATION pour leur suivi, spécialement à Daramony TEK qui est venue régulièrement nous visiter, ainsi que Emilie TOUZEAU et Marine BONTEMPS.

Je voudrais également saluer mes collègues qui ont apporté une belle humeur tout au long de ces six mois de cours en distanciel et pour leur soutien moral.

Un grand merci également à Léonard SUSLIAN et Erwan JESTIN pour tous leurs conseils avant et pendant cette reconversion, en termes de motivation et d'aide technique. Ils m'ont grandement facilité le travail.

Enfin, je tiens à remercier mon binôme de formation : Sanny. Merci pour toutes ces heures de recherche et de casse-tête !



## 5. Parties prenantes (Acteurs)

### 5.1. Centre de formation

Nextformation est un centre de formation présent sur la région parisienne. A travers trois espaces de formation situés en Île-de-France, à Paris et à Vincennes, il accompagne chaque année plus de 900 adultes dans leur projet de transition et d'évolution professionnelle.

Leur catalogue est uniquement composé de formations validées par un titre professionnel ou une certification professionnelle reconnu par le Ministère du Travail et de l'Emploi (RNCP). Leur ingénierie pédagogique s'emploie à former les apprenants à toutes les compétences et connaissances professionnelles requises pour mener à l'obtention de la certification professionnelle mais également pour exercer le métier visé. De plus, tous leurs parcours de formation peuvent se découper en certifications professionnelles et s'adapter à une expérience acquise. Chiffres clés de 2018 :

- 93% de réussite aux examens, dont 15% partiellement.
- 85% de stagiaires sont satisfaits de leur formation.
- 65% changent de métier dans l'année qui suit leur formation.

### 5.2. Entreprise

InfraS est une société d'audit et de conseil en informatique de gestion. Elle est un spécialiste Dolibarr, ERP/CRM de gestion. Elle s'occupe de l'hébergement, de l'infogérance et du développement de modules spécifiques à Dolibarr.

Portée par Sylvain LEGRAND, cette petite société est partenaire Dolibarr depuis plus de 8 ans maintenant en tant que « Dolibarr Preferred Partner ».



### 5.3. Personnel

De formation technico-commercial, j'ai toujours été attiré par l'informatique en général. Mais les aléas de la vie en ont plus fait une passion qu'une réelle vocation professionnelle. J'ai pourtant toujours tourné autour durant mes 20 ans de carrière.

Depuis 2008, je travaille chez différents OCEN, en tant que vendeur chez Orange, puis technicien itinérant chez Protelco (ILIAD), et enfin depuis 2017 comme chargé de déploiement, d'exploitation et de maintenance chez FREE Réseau (ILIAD).

La période COVID fut comme beaucoup une étape de remise en question, notamment sur la direction que je voulais donner à ma carrière professionnelle, mais aussi sur sa meilleure intégration dans ma vie familiale.

Et voilà le projet de reconversion professionnel lancée grâce à des connaissances qui m'ont beaucoup aidé et guidé dans les choix de carrière à étudier, à décortiquer, à analyser : réseau, système, cybersécurité, développement, « DevOps » ?

Cette formation pour avoir le titre professionnel AIS fut un bon choix. Repartir sur de bonnes bases et affirmer mes goûts pour la partie architecture système/cloud et automatisation des process était une nécessité. Le « DevOps » est clairement ce que je recherche !

## 6. Projet

### 6.1. Contexte

InfraS est une toute jeune société créée en janvier 2020. Mais son activité en tant que tel existe depuis bien plus longtemps. Son gérant Sylvain LEGRAND exerce en tant que FREELANCE depuis 2015. Son épouse la rejoint en 2020 car le nombre de client à gérer était en pleine expansion. Il gère à ce jour environ 80 clients dont la grande majorité en hébergement propre.

Cette augmentation rapide de client ne lui a pas permis de modifier l'approche technique de son infrastructure. Dans un contexte donc de forte croissance à son niveau (pour rappel il fait office de commercial, support technique, administrateur système, développeur...), il se devait de regarder pour optimiser son infrastructure.

De plus, la période COVID a soulevé des problématiques de sécurité jusqu'alors plus ou moins écarté dans toutes les petites structures :

- Intégrité : garantir que les données sont bien celles que l'on croit être.
- Disponibilité : maintenir le bon fonctionnement du système d'information.
- Confidentialité : rendre l'information inintelligible à d'autres personnes que les seuls acteurs d'une transaction.
- L'authentification : assurer que seules les personnes autorisées aient accès aux ressources.

### 6.2. Méthodologie

Pour réaliser ce projet, nous avons utilisé une méthode de gestion dite Agile : SCRUM. Son objectif est bien évidemment d'améliorer la productivité des équipes, même à distance. Elle permet aussi une optimisation du projet grâce à des feedbacks réguliers.

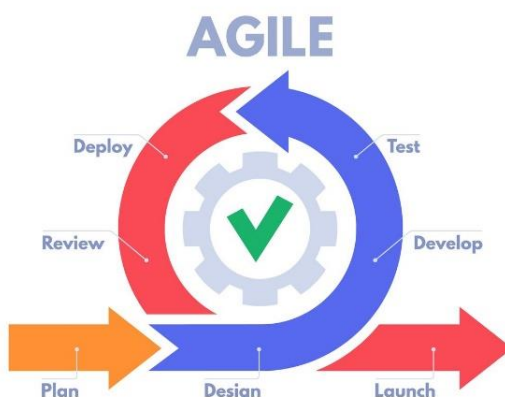


Figure 1 - Méthodologie AGILE

La méthode Scrum tire son nom du monde du rugby (scrum = mêlée), car les équipes agiles qui utilisent Scrum se réunissent le plus souvent possible afin de vérifier que le projet avance correctement,

toujours prêts à réorienter ce dernier au fil de son avancement. C'est donc une approche dynamique et participative de la conduite du projet.

Scrum est de nos jours l'approche agile la plus plébiscitée par les équipes de développeurs. Elle permet :

- La collaboration avec le client
- L'acceptation du changement
- L'interaction avec les personnes et des logiciels opérationnels



## La méthode Scrum

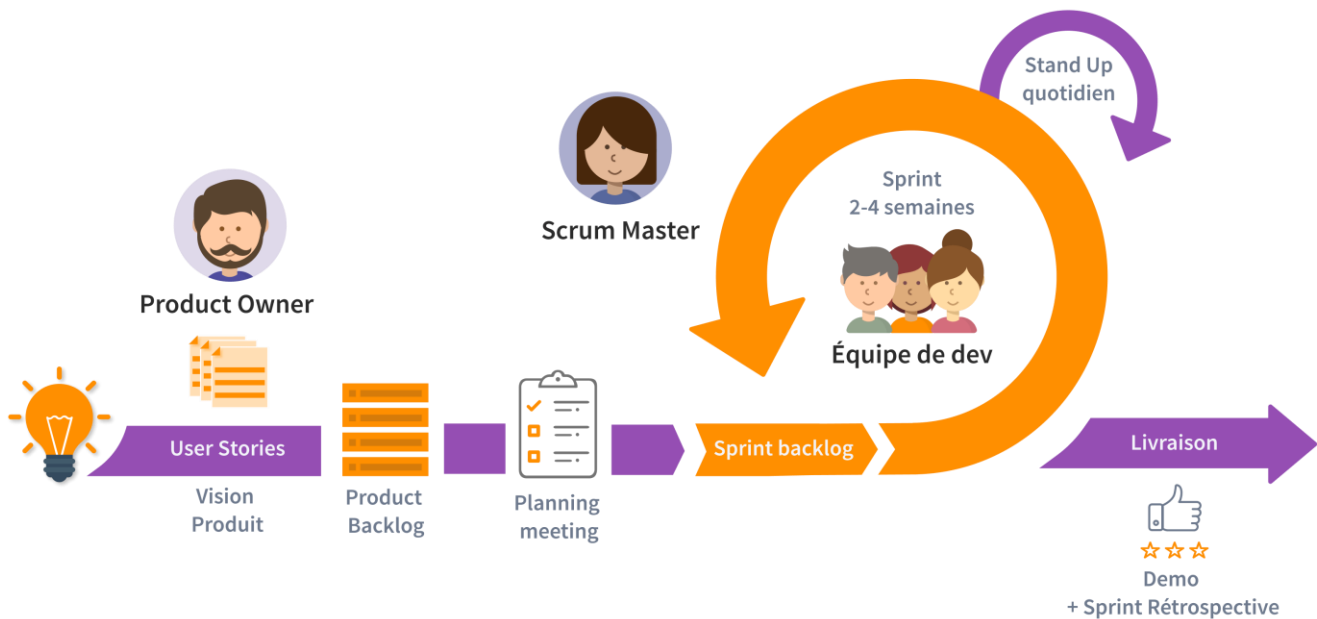


Figure 2 - Méthode SCRUM

## 6.3. Expression des besoins

### 6.3.1. Cahier des charges

Ce projet se présente dans un contexte de forte croissance chez InfraS. Le nombre d'instances de Dolibarr monte rapidement, passant de 30 en 2020 à plus de 80 à ce jour. Aux vues de la criticité de l'infrastructure, il y a donc besoin de revoir son architecture pour permettre un plan de reprise d'activité rapide en cas de pannes, qu'elles soient matérielles (chez l'hébergeur) ou logicielles.

Dans une première approche donc, il faut repenser l'infrastructure pour qu'elle soit plus rapidement redéployable, et ce à l'identique. Le développement de Dolibarr ne doit plus se faire sur la partie production pour éviter toute mise hors service des instances clientes existantes. Enfin, une première ébauche de supervision des points critiques de l'infrastructure devra être faite.

Les ressources humaines se limitant au gérant de la société, une attention particulière sera portée dans une deuxième étape à l'automatisation des tâches. Cette première restructuration permettra



également de préparer un probable passage sur des services managés. Il faudra se pencher également sur la gestion des LOG pour être en accord avec la loi RGPD.

### 6.3.2. Echéances

Le temps est assez limité à la fois de mon côté, mais également pour l'entreprise. Le projet doit donc commencer au plus vite, dès le mois d'octobre 2022 idéalement. Il devrait être finalisé dans une première phase lors du stage.

Ces impératifs temporels sont dictés par les disponibilités des parties prenantes à ce projet. Un planning prévisionnel a été établi (disponible en [section 6.5.2](#)).

## 6.4. Contexte technique

### 6.4.1. Infrastructure actuelle

InfraS utilise à l'heure actuelle les services de l'hébergeur OVHCloud. Elle a en sa possession deux serveurs dédiés de type baremetal Advance-1. Un petit VPS est également disponible.

#### Serveur Baremetal 1 & 2 :

Produit	Dénomination	Tarif / mois
Serveur dédié	Advance-1 Gen 2	85,49€ HT
Localisation datacenter	France (Strasbourg - SBG)	
Processeur	Intel Xeon-E 2386G - 6c/12t - 3.5GHz/4.7GHz	
Stockage	2x 512GB SSD NVMe + 2x 6TB HDD SATA Soft RAID	31.35€ HT
Mémoire	64GB DDR4 ECC 3200MHz	
Bande passante publique	1Gbit/s illimitée	
Bande passante privée	1Gbit/s illimitée et garantie	
Frais d'installation		Activation Gratuite

#### VPS :

Produit	Dénomination	Tarif / mois
Serveur privé virtuel	Vps2020-comfort-2-4-80	45.50€ HT
Localisation datacenter	France (Strasbourg - SBG)	
Processeur	4 vCore	
Stockage	160 Go SSD NVME + 500Go HDD SATA	
Mémoire	8Go	
Bande passante publique	1Gbit/s	
Frais d'installation		Activation Gratuite

Dolibarr est porté par une infrastructure de type LAMP (Linux, Apache, MariaDB, PHP). Voici donc les versions utilisées à ce jour sur le serveur Baremetal 1 :

## Ubuntu 20.04 LTS

L'infrastructure est basée sur une version 20.04 LTS de Ubuntu. Elle est sortie en première release en avril 2020, toujours soutenue malgré une version 22.04 aussi en LTS. Son soutien est assuré par Canonical pour une durée de 5 ans donc en Avril 2025. Pour cette raison, nous profiterons de la version 22.04 LTS qui sera elle soutenue jusqu'en 2027.

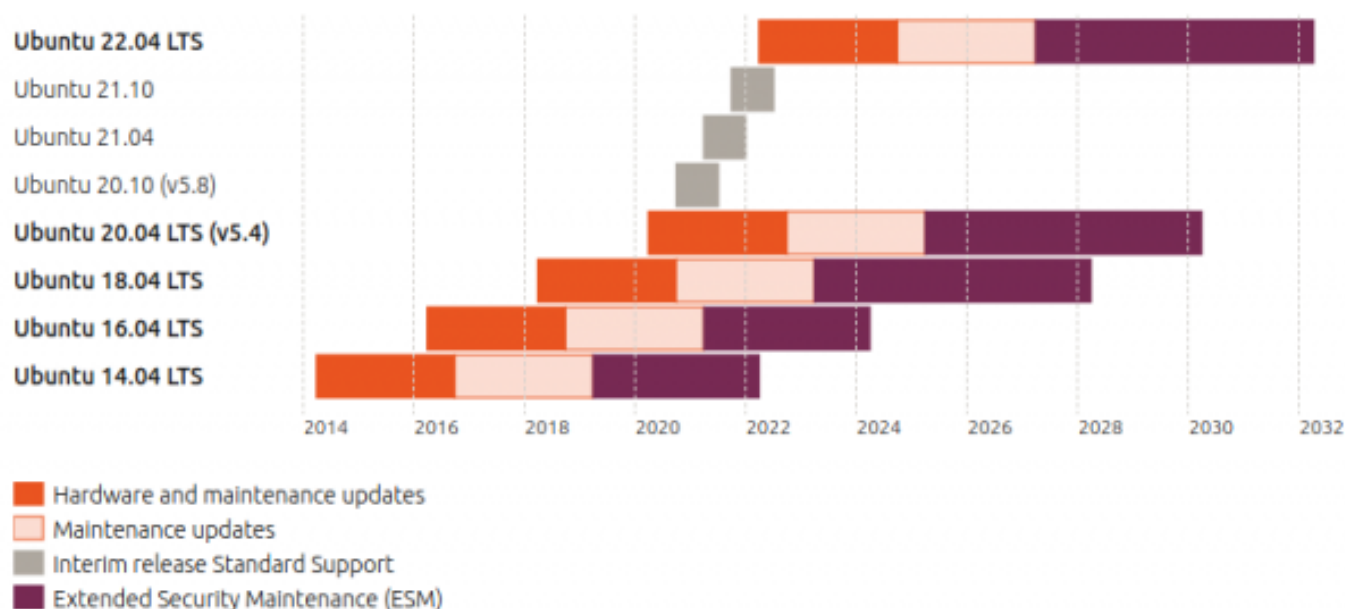


Figure 3 - Versions Ubuntu

## Apache 2.4.56

Apache est un logiciel de serveur web gratuit et open-source lancé en 1995. Il est utilisable sur les plateformes modernes de type Unix et Windows. Son objectif est de fournir un serveur web sécurisé, efficace et extensible. Il alimente aujourd'hui presque un site web sur deux à travers le monde.

## PHP FPM

PHP est un langage de scripts généraliste et Open Source, spécialement conçu pour le développement d'applications web dynamiques. Il est principalement utilisé pour produire des pages Web dynamiquement via un serveur HTTP, mais peut également fonctionner comme n'importe quel langage interprété de façon locale.

C'est un langage orienté objet. Il peut être intégré facilement à d'autres langages car il génère du code (HTML, XHTML, CSS par exemple) et des données (JPEG, GIF, PNG) pouvant être interprété par nos navigateurs web. Il est considéré comme une des bases de la création de sites web dits dynamiques mais également des applications web.

PHP dans sa version FPM est utilisé pour ses performances lors de fortes charges. Contrairement au serveur PHP, il est fourni avec son propre daemon.

## MariaDB 10.6.12

MariaDB est un système de gestion de base de données relationnelles édité sous licence GPL. Il s'agit d'un embranchement communautaire de MySQL suite au rachat de ce dernier par Sun Microsystems.

Il s'agit donc d'un fork plus communautaire et ouvert, et 100% compatible MySQL. Il s'avère aussi plus performant selon certaines études. MariaDB est utilisé comme serveur MySQL par défaut sur Debian. Sur Ubuntu cependant, c'est toujours MySQL qui est proposé.

## ProFTPd

ProFTPd est un serveur FTP sous licence GNU GPL. Il est puissant et parfaitement sécurisé. ProFTPd est bien documenté et la plupart des configurations sont assez proches de celles fournis comme exemple avec le logiciel.

Son paramétrage tourne autour d'un unique fichier de configuration, proftpd.conf. Il utilise une syntaxe similaire à celle d'Apache permettant ainsi d'homogénéiser les fichiers de configuration. Le logiciel permet de configurer plusieurs serveurs FTP virtuels et a la possibilité d'être utilisé dans un environnement dédié. Il peut être lancé comme un démon ou comme service inetd.

Il est principalement utilisé pour donner accès aux clients à leurs sauvegardes.

## Webmin 2.013

Webmin est un outil d'administration système en interface Web pour les serveurs et les services de type Unix. Il est possible d'y configurer les composants internes du système d'exploitation tels que les utilisateurs, les quotas de disque, les services ou les fichiers de configuration, ainsi que de modifier et de contrôler les applications telles que BIND DNS Server, Apache HTTP Server, PHP, MySQL, et d'autres.

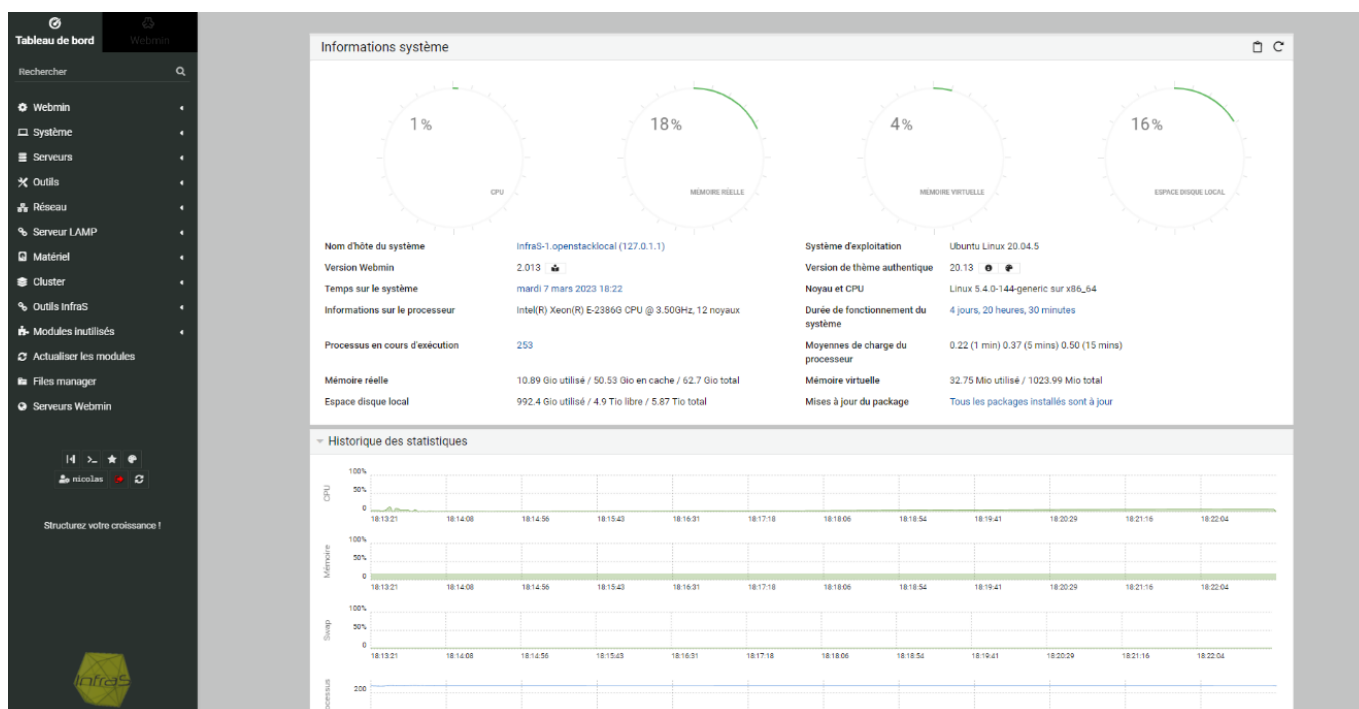


Figure 4 - Interface graphique Webmin

## 6.5. Réalisation

### 6.5.1. Analyse fonctionnelle

L'analyse fonctionnelle a pour but de créer ou d'améliorer un produit. Cela permet de connaître toutes les caractéristiques de l'objet en question et de déterminer ses limites. Les diagrammes « bête à corne » et « pieuvre » sont des outils de représentation graphique de cette analyse fonctionnelle, ils rendent de façon visuelle et plus simple tout ou partie du cahier des charges. Ce sont donc de bons outils d'analyse pour représenter les fonctions essentielles et secondaires d'un produit, et voir comment ces fonctions réagissent avec le milieu extérieur.

Avant de détailler cette analyse fonctionnelle, rappelons via le diagramme des prestations ou diagramme « bête à cornes » l'objectif de notre produit/service et de savoir si ce dernier est viable par rapport aux besoins de l'utilisateur final. Ce diagramme répond aux questions suivantes :

- A qui le produit/service s'adresse-t-il ?
- Sur quoi agit-il ?
- Quel est l'objectif du produit/service ?

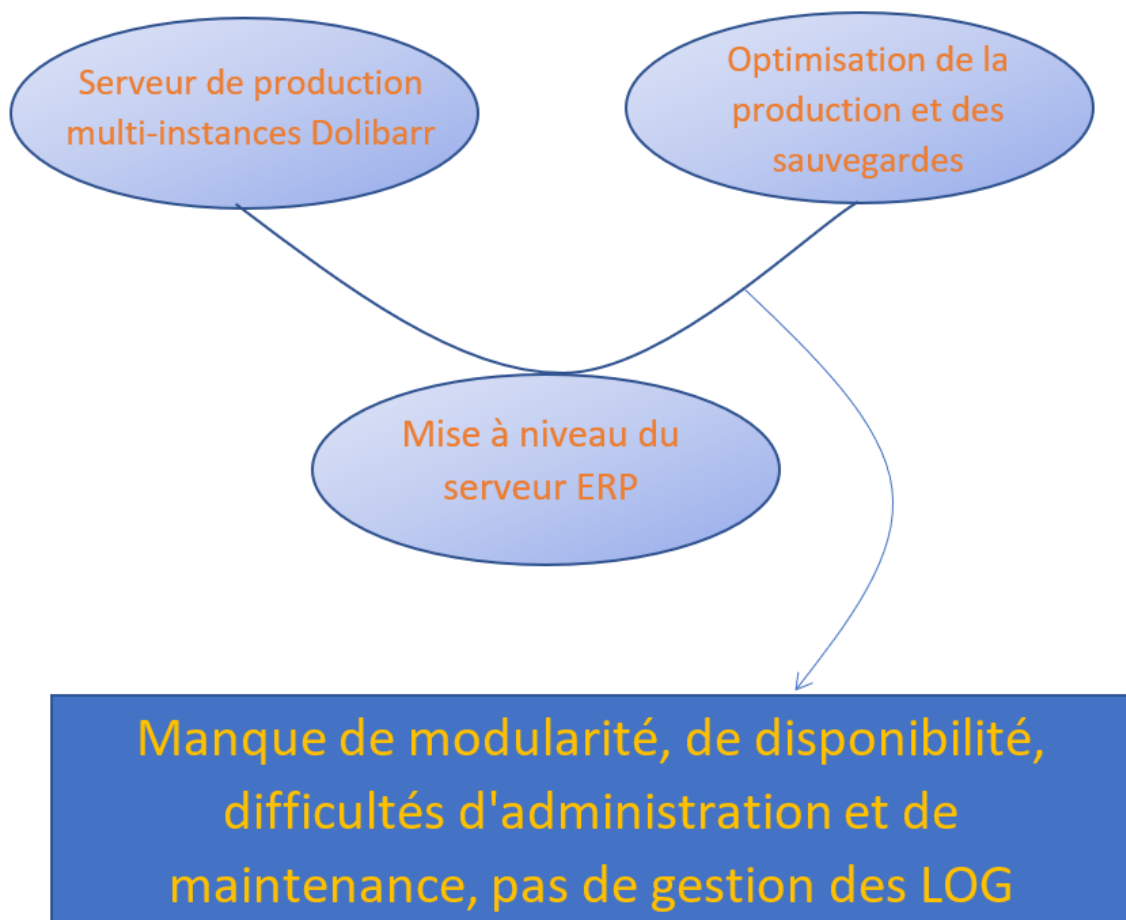


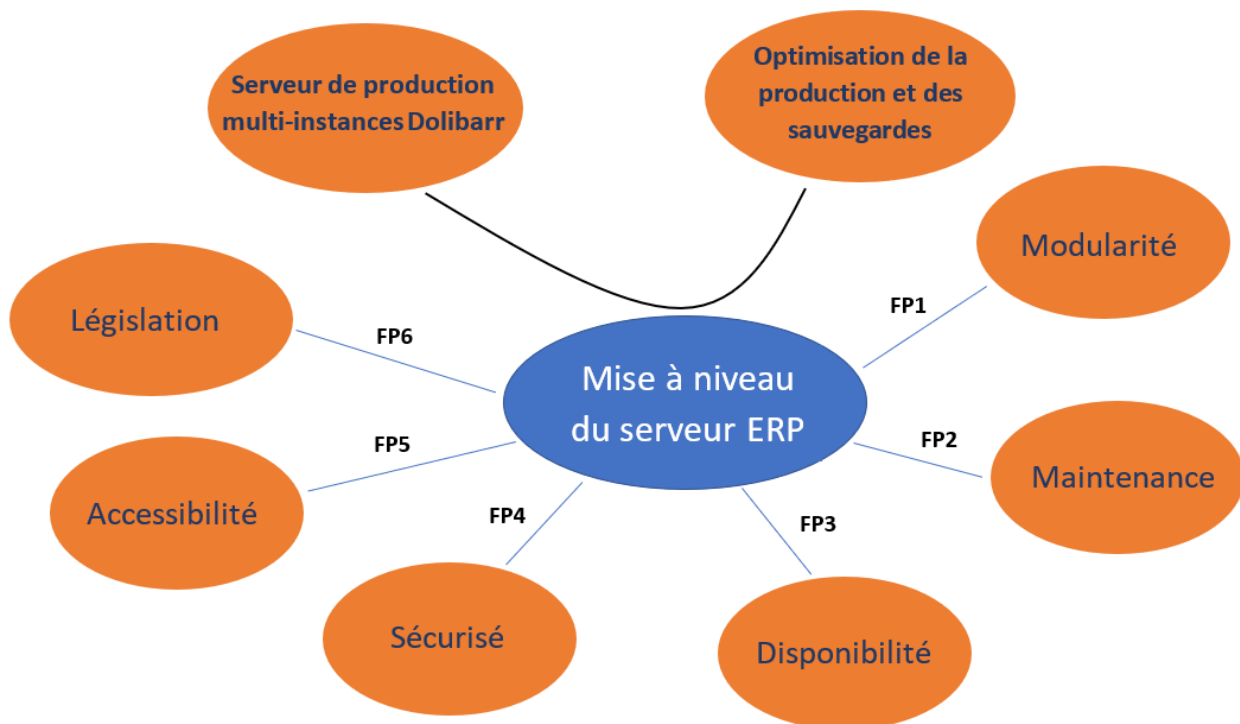
Figure 5 - Diagramme "bête à corne"



Ensuite vient le deuxième outil de cette méthode, le cahier des charges fonctionnel ou diagramme pieuvre. Comme il met en relation les fonctions du produit avec son environnement extérieur, nous devons représenter à la fois les fonctions dites principales, et les fonctions de contraintes.

### Fonctions principales :

Ces fonctions principales répondent à la question du « pourquoi le produit est là, quel est son but ». Dans notre cas, nous retrouvons six fonctions principales qui vont caractériser se projet :



FP 1 : S'adapte facilement à la taille de la base client

FP 2 : Permet une maintenance en HO sans interruptions client

FP 3 : Disponibilité sous condition de SLA

FP 4 : Les données utilisateurs sont sauvegardés régulièrement

FP 5 : Système accessible et bien documenté

FP 6 : Les solution mises en œuvre répondent aux législations en vigueur sur les sauvegardes de LOG (RGPD)

Figure 6 - Diagramme "pieuvre" des fonctions principales

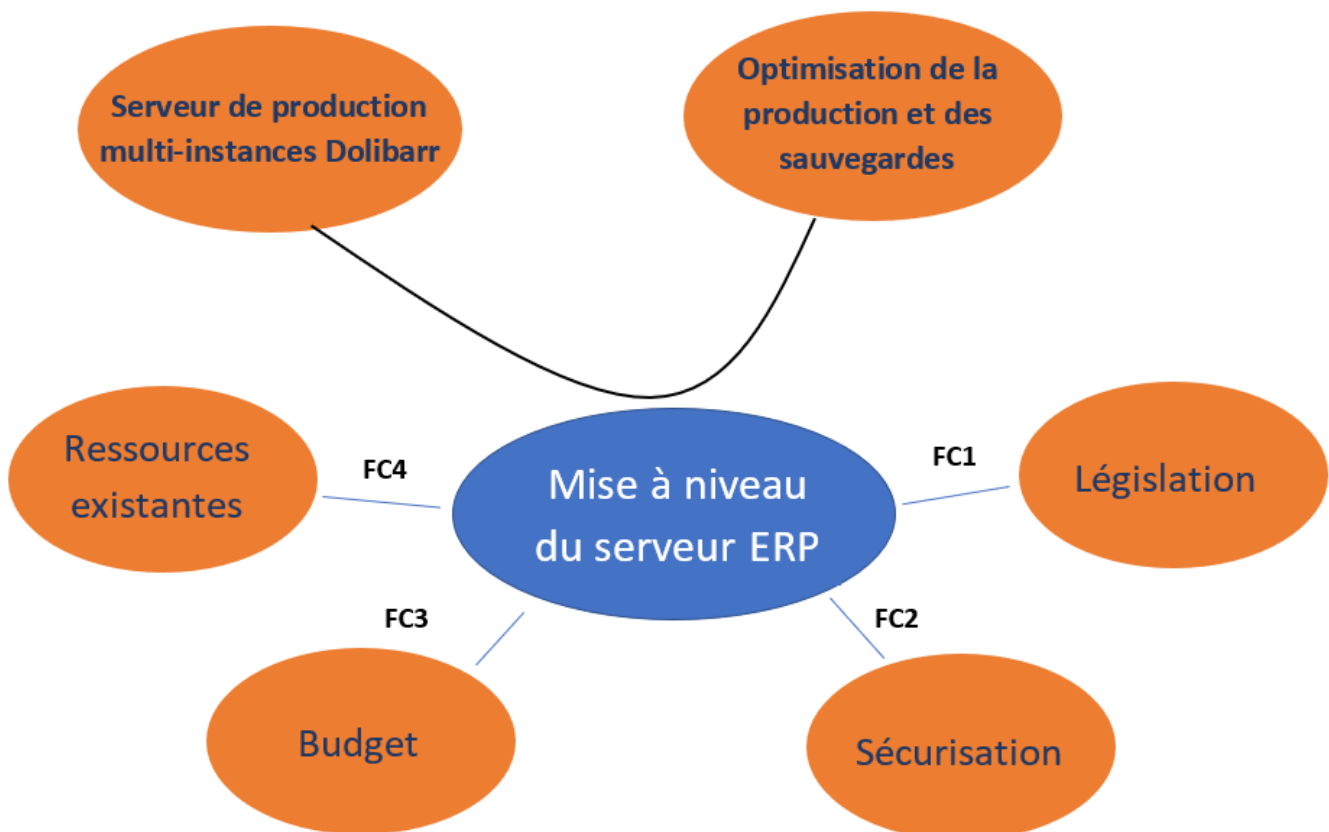
### Fonctions de contraintes :

Les fonctions de contraintes sont comme leurs noms l'indiquent des impératifs liés au produit, ce qui aura des effets sur lui directement, ses limites ou limitations. Le produit n'a pas été créé pour celles-ci, mais le fait d'exister lui impose d'assurer certaines fonctions « par contrainte », en relation avec les éléments extérieurs.

Ce sont des fonctions qui traduisent les contraintes liées à l'adaptation du produit à son environnement et à la prise en compte de ce dernier. De faite, ce sont des obligations à satisfaire.

On retrouve des contraintes de plusieurs catégories :

- Fonctionnelles (nécessaires pour remplir la fonction principale).
- Ergonomiques : faciliter l'utilisation.
- Esthétiques : le rendre attrayant.
- De sécurité : l'utiliser en toute sécurité-> normes
- Environnementales : être conçu et utilisé dans un souci de développement durable.
- Economiques/commerciales : coût correspondant au service, au budget, à la publicité.



FC 1 : La législation RGPD doit être appliquée

FC 2 : L'infrastructure, les logiciels et les moyens de communication doivent être sécurisés

FC 3 : Le budget doit être respecté

FC 4 : Réutiliser une partie du code et du matériel existant

Figure 7 - Diagramme "pieuvre" des fonctions de contraintes

## 6.5.2. Planning prévisionnel - GANTT

Le diagramme de Gantt est un outil utilisé en gestion de projet. Il permet de visualiser dans le temps les tâches diverses qui le compose. Il permet donc de déterminer les dates de réalisation, les marges temporelles existantes sur certaines tâches, de visualiser d'un seul coup d'œil le retard ou même l'avancement des travaux.

La démarche pour son utilisation est simple. Sont représentées en abscisse les unités de temps (exprimées en mois, en semaines ou en jours). En ordonnées on y trouve les différentes tâches ou postes de travail.

Voici représentés de façon non détaillée les principaux postes de travail

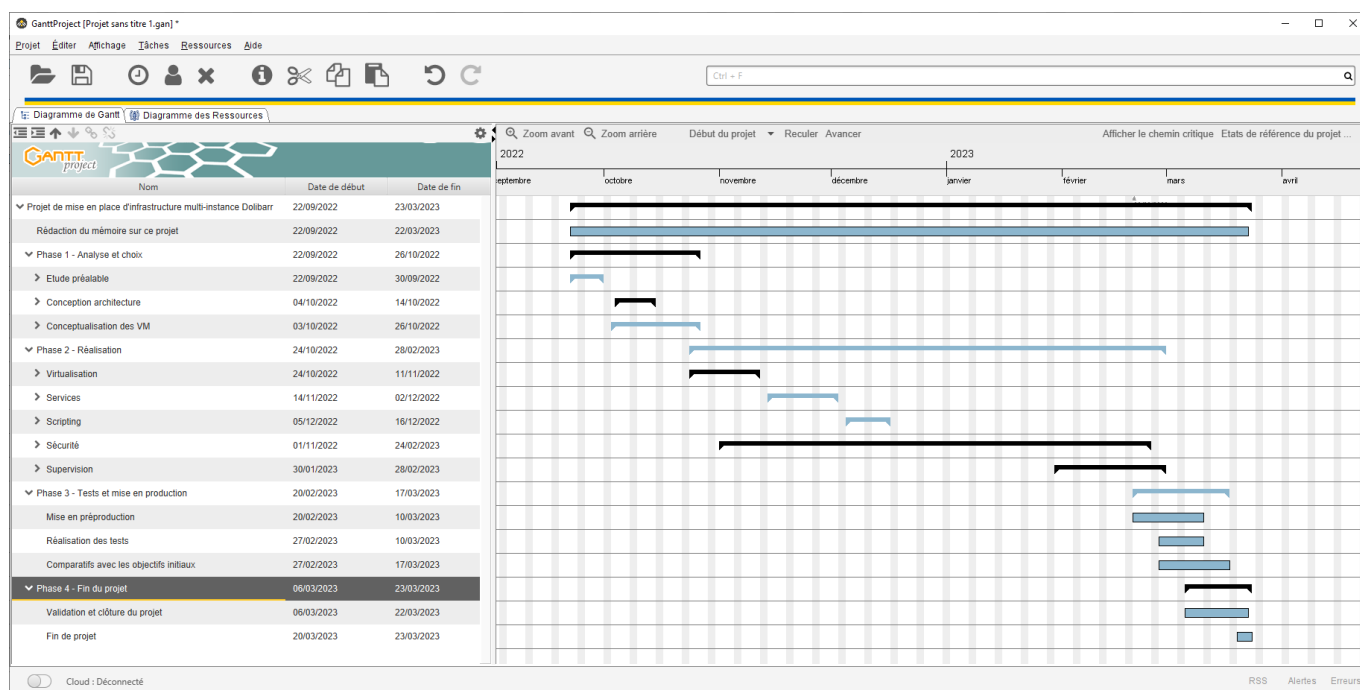


Figure 8 - Diagramme de Gantt

La version détaillée du diagramme de GANTT est disponible en [annexe 1](#).

## 6.5.3. Répartition des taches - RACI

La matrice RACI désigne une matrice des responsabilités. L'acronyme RACI signifie **R**esponsable, **A**cteur, **C**onsulté et **I**nmomé. Elle indique donc les rôles et les responsabilités des intervenants au sein de chaque processus et activité du projet.

R : Qui est chargé de mener à bien le projet ?

A : Qui valide ?

C : Qui peut aider, apporter son expertise ou son avis ?

I : Qui doit être tenu informé de l'avancée du projet ?

Grâce à cet outil, nous avons une vision simple et claire mais suffisamment détaillée de qui fait quoi, tout en permettant d'éviter une redondance des rôles ou une dilution des responsabilités.

Une variante de la matrice RACI est la RASCI, le S faisant intervenir un « Support », c'est-à-dire une personne qui vient en soutien pour aider à réaliser la tâche.

Matrice RACI			
<b>Processus: Mise en place d'une infrastructure multi-instances Dolibarr</b>			
Revu par : Nicolas BIDEI			
Le : 16/03/20023			
Activité	Dpt Informatique		
	Chef de projet	Nicolas BIDEI	Client Final
Rédaction du mémoire sur ce projet	C	R	
<b>Phase 1 - Analyse et choix</b>			
Etude préalable	R	A	C
Conception architecture	R	A	
Conceptualisation des VM	A	R	
<b>Phase 2 - Réalisation</b>			
Virtualisation	I	R	
Services	I	R	
Scripting	I	R	
Sécurité	C	R	I
Supervision	C	R	I
<b>Phase 3 - Tests et mise en production</b>			
Mise en pré-production	A	R	I
Réalisation des tests	C	R	I
Comparatifs avec les objectifs initiaux	A	R	I
<b>Phase 4 - Fin du projet</b>			
Validation et cloture du projet	I	I	R
Fin de projet	I	I	R

R : Responsable  
A : Acteur  
C : Consulté ou contributeur  
I : Informé

Figure 9 - Matrice RACI



### 6.5.4. Analyse des risques

L'objectif du changement de type d'infrastructure est de pouvoir remédier à une multitude de risques déjà énoncés plus haut. Toutefois des risques inerrants aux matériels et réseaux de l'opérateur d'infrastructure (hébergeur du serveur Baremetal) existent toujours. S'il n'est pas toujours possible d'éliminer une défaillance, diminuer son impact est en revanche un objectif atteignable.

Pour cela, il est impératif de faire une analyse des modes de défaillance, de leurs effets et de leur criticité (AMDEC). Cette méthode est utilisée pour identifier les défaillances ou risques possibles d'un système et en évaluer les effets probables. Elle permet aux équipes de conception d'instaurer divers contrôles de risques dans le but d'empêcher ou d'atténuer les impacts, sur la production par exemple.

L'AMDEC se repose sur une hiérarchisation des défaillances et les plans d'action associés. Voici un tableau de synthèse et d'évaluation des risques :

Exemple de rendu : Tableau de synthèse et d'évaluation d'une installation				
Fréquence	Rare	Possible	Fréquent	Total
Gravité				
Faible	7	3	8	18
Moyenne	11	5	0	16
Importante	10	5	0	15
TOTAL	28	13	8	
	Les défaillance remarquables (événements possibles ou fréquents ayant une gravité notable)			
	Les défaillance relatives (événements intermédiaires dont la gravité est relativisée par la fréquence)			
	Les défaillances ayant un impact limité sur les installations (événements peu fréquents ayant une gravité minime)			

Figure 10 - Matrice de criticité

Méthode d'évaluation des risques :

1. Trouver les éléments pouvant être endommagés par des menaces : serveur physique, réseau, applications, système d'exploitation, sauvegarde, données clientes ...
2. Identifier les conséquences potentielles : pertes d'accès, pertes de données, conséquences juridiques (RGPD) ...
3. Identifier les menaces et leurs niveaux : Catastrophes naturelles, défaillances hébergement, interférences humaines accidentelles (mauvaise manipulation, suppression de fichier), humains malveillants ...
4. Identifier les vulnérabilités et évaluer la probabilité de leur exploitation : DDOS, cryptolock et ransomware ...
5. Evaluer la gravité des risques : Faible, moyenne, importante.



A l'aide des données recueillies il est possible de créer un plan de gestion de risque comme celui présenté ci-dessous à titre d'exemple :

Menace	Vulnérabilité	Actif et Impact	Risque	Recommandations de contrôle
Défaillance système – Surchauffe dans la salle de serveurs <b>Élevée</b>	Le système de climatisation a dix ans <b>Élevée</b>	Serveurs. Tous les services (site Web, messagerie, etc.) seront indisponibles pendant au moins trois ans <b>Critique</b>	<b>Élevée</b> Perte potentielle de 45 000 € par occurrence	Achat d'un nouveau climatiseur 2 700 €
Attaque DDoS par des humains malveillants (interférence) <b>Élevée</b>	Le pare-feu est correctement configuré et dispose d'une bonne atténuation des attaques DDoS <b>Faible</b>	Site Web. Les ressources du site Web seront indisponibles <b>Critique</b>	<b>Moyen</b> Perte potentielle de 8 900 € par heure d'indisponibilité	Surveiller le pare-feu
Catastrophes naturelles – inondation <b>Moyen</b>	La salle de serveurs se trouve au deuxième étage <b>Faible</b>	Serveurs. Tous les services seront indisponibles <b>Critique</b>	<b>Faible</b>	Aucune action requise
Interférence humaine accidentelle – suppressions accidentelles de fichiers <b>Élevée</b>	Les autorisations sont correctement configurées, un logiciel d'audit informatique est en place, des sauvegardes sont réalisées régulièrement <b>Faible</b>	Fichiers sur un partage de fichiers. Des données critiques seront peut-être perdues, mais pourront presque certainement être restaurées depuis une sauvegarde <b>Moyen</b>	<b>Faible</b>	Continuer à surveiller les modifications apportées aux autorisations, les utilisateurs privilégiés et les sauvegardes

Figure 11 - Exemple de plan de gestion des risques

### 6.5.5. Infrastructure prévisionnelle

#### Le matériel :

Aucun changement ne sera apporté en premier lieu sur la partie physique, l'objectif est ici d'optimiser le matériel pour les évolutions logicielles futures.

Les serveurs étant déjà en location et les contrats se terminant fin 2023, nous les garderons en l'état. Si le besoin se fait sentir par la suite à cause d'un trop grand nombre d'instance Dolibarr, une augmentation de la mémoire vive sera effectuée sur les serveurs Baremetal (passage à 128Go).

Une attention particulière sera toutefois à porter sur les performances du petit VPS. Le stockage risque de vite devenir problématique.

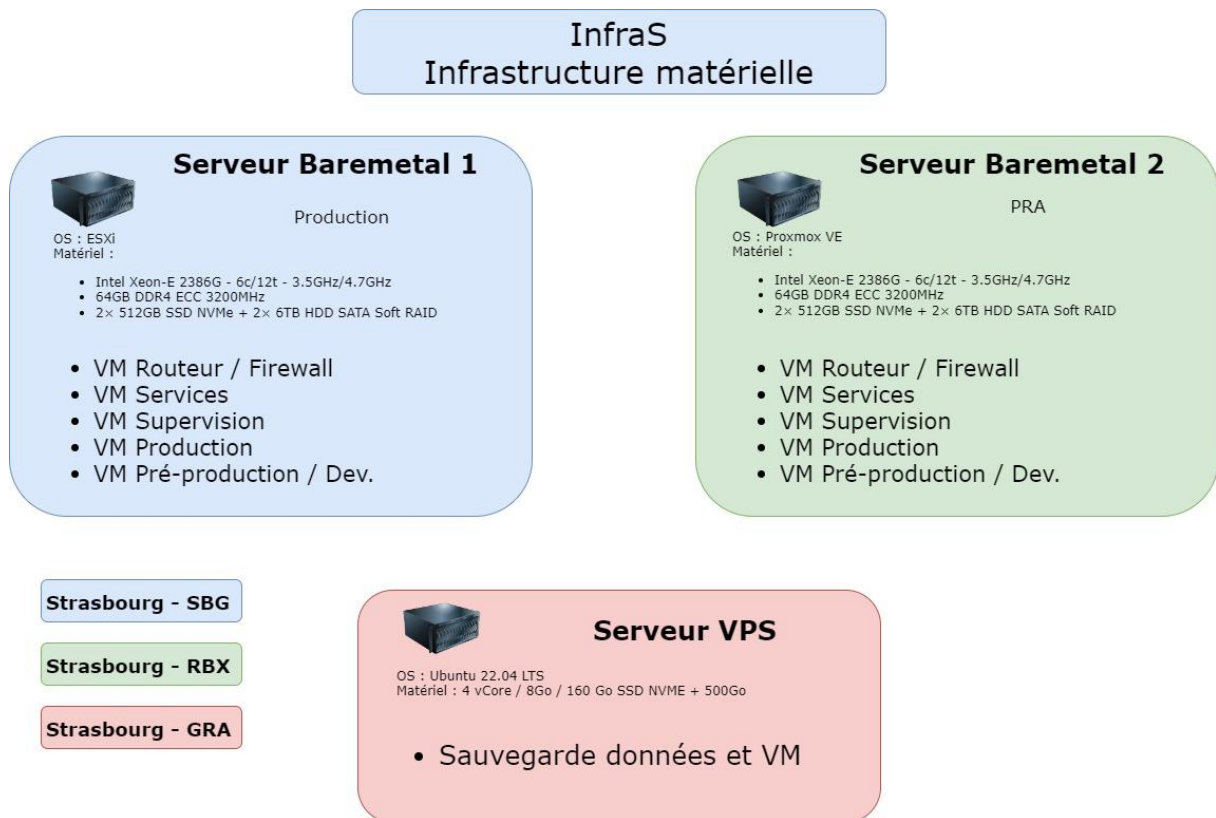


Figure 12 - Infrastructure matérielle prévisionnelle

### Le réseau :

Il est à noter que les deux serveurs baremetal disposent d'une interface réseau privée. Nous ne l'avons pas intégré pour le moment dans ce projet, car la partie interconnexion et transfert des diverses données de sauvegarde sera abordée plus tard dans l'année lors de l'évolution du projet.

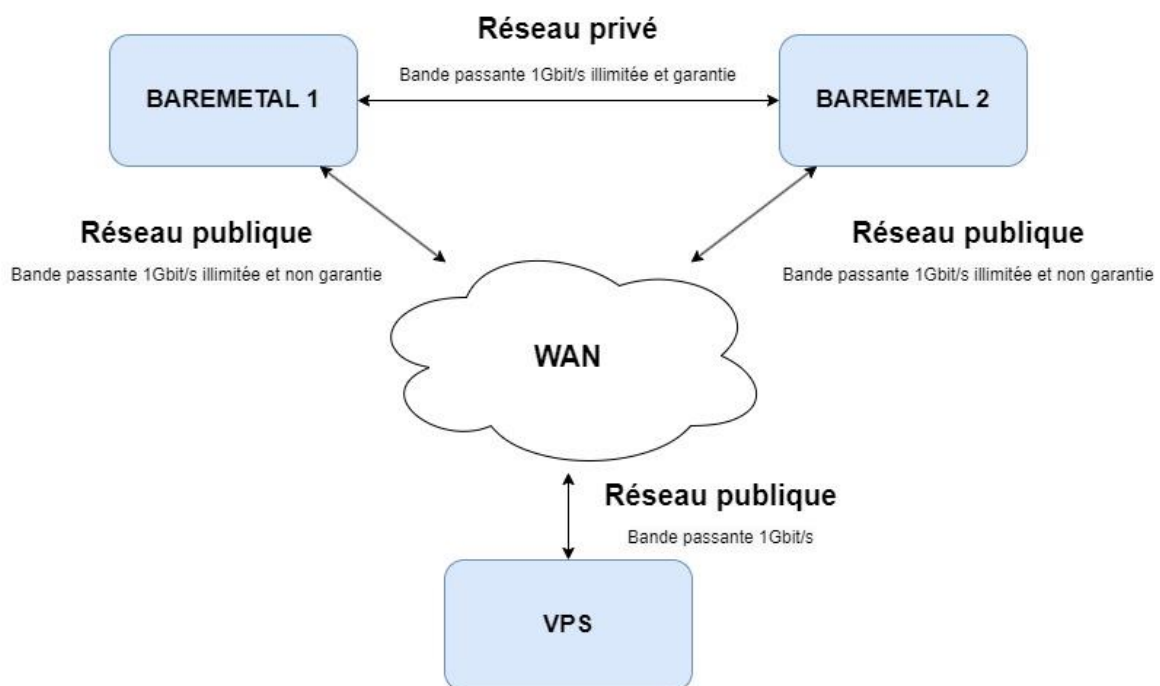


Figure 13 - Infrastructure réseau globale



L'infrastructure réseau du serveur Baremetal sera découpée de la façon suivante :

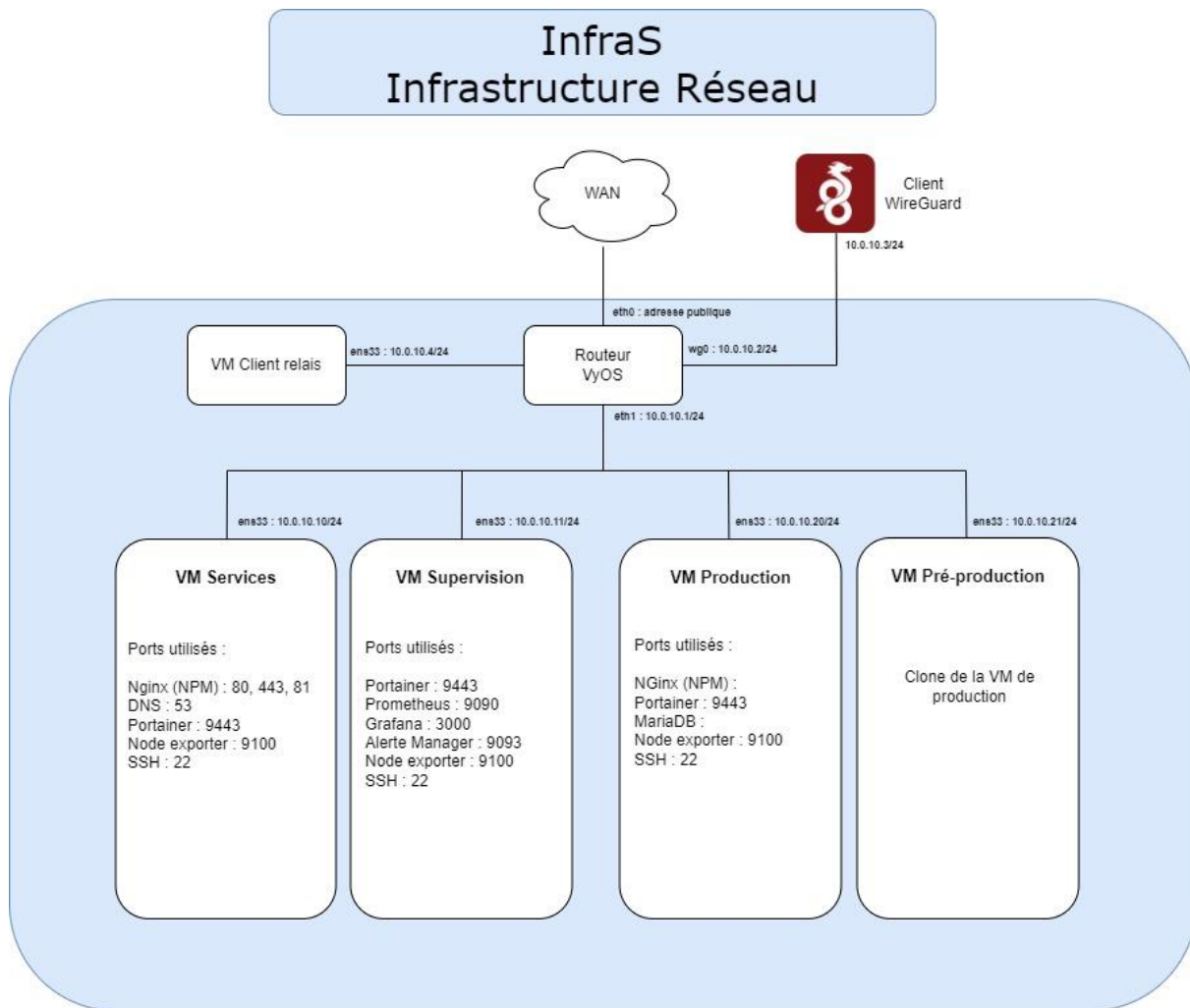


Figure 14 - Infrastructure réseau Baremetal

Le routeur VyOS sera le point d'entrée de toute l'infrastructure. Il gèrera le routage des paquets, sera garant de la sécurité des accès via son firewall, permettra de se connecter à la partie non-publique de l'infrastructure via un tunnel WireGuard.

Le Client relais sera le seul élément à avoir un accès SSH aux différentes VM du réseau pour établir un plus haut niveau de sécurité.



## 6.5.6. Mise en œuvre

### 6.5.6.1. Hyperviseur

L'hyperviseur est la base de toute plateforme de virtualisation. Il existe de nombreuses solutions, soit propriétaires, soit Open Sources. Les plus connues restent VMware avec sa solution ESXi (vSphere) pour la partie propriétaire et KVM pour les technologies Open Sources.

Proxmox VE est une solution à base de KVM. Avec son packaging gratuit, il est très bien fourni : base Debian, partitionnement des disques avec support LVM2, support des conteneurs LXC, outils de sauvegarde et de restauration via Proxmox Backup Server, fonction avancée de clustering ...

ESXi (vSphere Hypervisor) est une solution très aboutie, relativement facile à mettre en œuvre et qui offre des performances élevées. Il est toutefois nécessaire d'acquiescer une licence. Son tarif reste mesuré pour l'usage que l'on en aura, seulement 575€ pour la version vSphere Essentials (6 licences CPU pour 3 serveurs de 2 CPU maximum et 1 licence vCenter Server Essential).

Malgré les qualités flatteuses de Proxmox, le choix sera fait de partir sur un ESXi en version 7.03, nativement disponible chez l'hébergeur actuel OVH (Proxmox VE 7 l'est également).

Pour des raisons de coût et de facilité de développement, tout ce projet sera élaboré sur un serveur auto-hébergé en local car déjà disponible :



Produit	Dénomination
Serveur auto-hébergé	DELL R620
Processeur	2 CPU Intel Xeon-E 2650L – 10c/20t – 1.7GHz/2.1GHz
Stockage	2× 512GB SSD SATA + 3× 1TB SSD SATA Hard RAID5
Mémoire	160GB DDR3 ECC 1066MHz

Voici différentes captures du système ESXi en cours de déploiement :

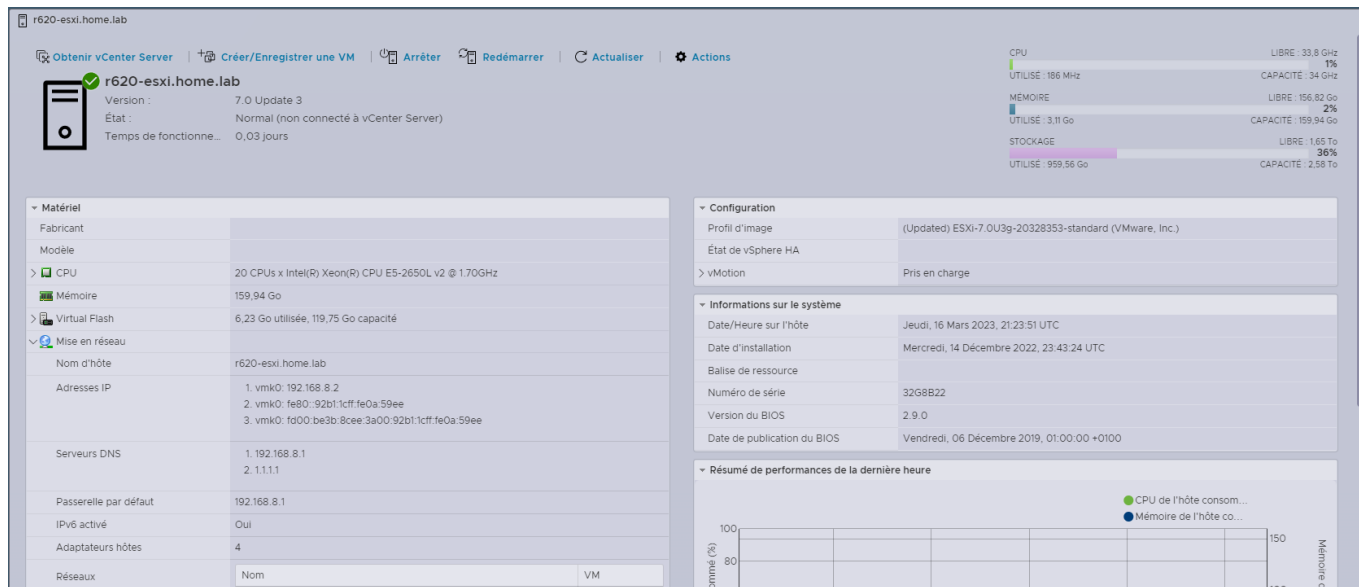


Figure 15- ESXi Hôte sur DELL R620



Figure 16 - Machines virtuelles

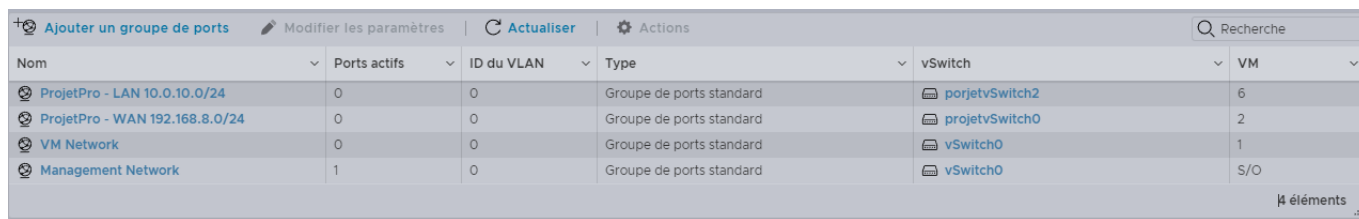


Figure 17 - Mise en réseau



Figure 18 – Datastore de stockage

### 6.5.6.2. Templating des VM

De base le système tournait sur Ubuntu. On restera donc sur le même système qui est familier à l'équipe qui l'utilisera, tout en l'upgradant simplement sur une version plus récente comme la 22.04 LTS. Toutes les VM sous Ubuntu seront sur la même base.

Par simplicité, nous utiliserons le système de templating disponible sur vSphere à défaut de trouver le temps de faire une vraie infrastructure via des outils comme Packer et Terraform de HiCorp. C'est une évolution future qui devra être mise en place, car c'est une vraie plus-value dans le but de raccourcir les délais de rétablissement en cas de crash total de l'infrastructure.

Le templating est très simple et tout aussi intuitif :

- Créer une simple VM comme souhaitée et mise à jour
- Installer les applications communes à toutes les VM
- Faire un « clean up » de la machine
- Arrêter la VM et la convertir en template

Il ne restera plus qu'à utiliser ce template plus tard pour un déploiement de nouvelle VM sous Ubuntu 22.04 LTS !

#### Retour sur le « clean up » de la VM :

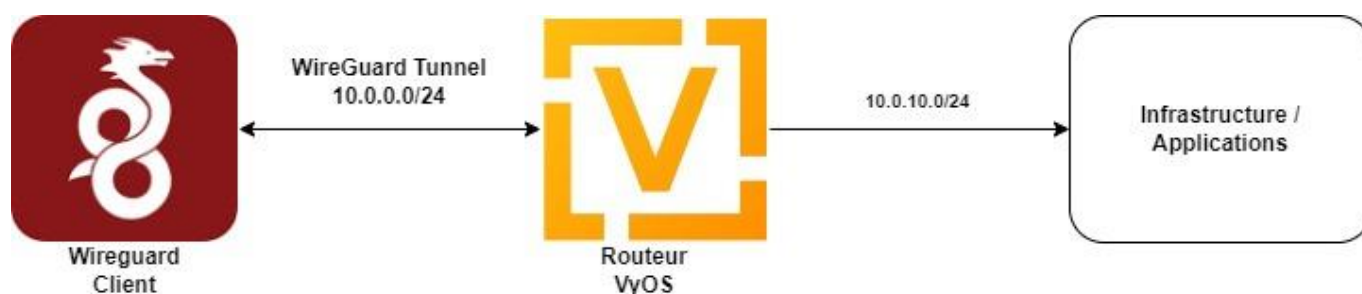
Comme le sysprep sous Windows, le « clean up » est nécessaire pour supprimer le machine-id et faire le ménage dans les fichiers comme le cache apt, les différents répertoires temporaires, l'historique des commandes ...

Le petit script de « clean-up » est en [annexe 2](#).

### 6.5.6.3. Routage avec VyOS

VyOS est un système d'exploitation réseau basé sur une distribution Debian et intègre tous les outils que l'on peut trouver sur des routeurs physiques comme ceux de CISCO. C'est un système très complet et qui permet de faire à la fois du routage mais aussi du firewall. Nous l'utiliserons donc dans ces objectifs, mais également avec le tunnel VPN WireGuard pour la maintenance.

VyOS n'a pas de WebUI, donc tout se passe en CLI. Comme pour un routeur physique, les manipulations sont assez intuitives : entrer en mode configuration, passer les commandes, « commit » les modifications et sauvegarder.



#### Configuration de VyOS

Comme mentionné sur le schéma réseau, nous allons attribuer trois interfaces à VyOS, une WAN et deux LAN. Il faudra faire le routage pour que la communication se fasse correctement dans les deux sens, ainsi que les règles de firewall.

La configuration comprendra donc le nom du système et du domaine, les interfaces réseau et le routage, les services à activer et les utilisateurs avec leurs droits.

*Le fichier de configuration complet se trouve en [annexe 3](#).*

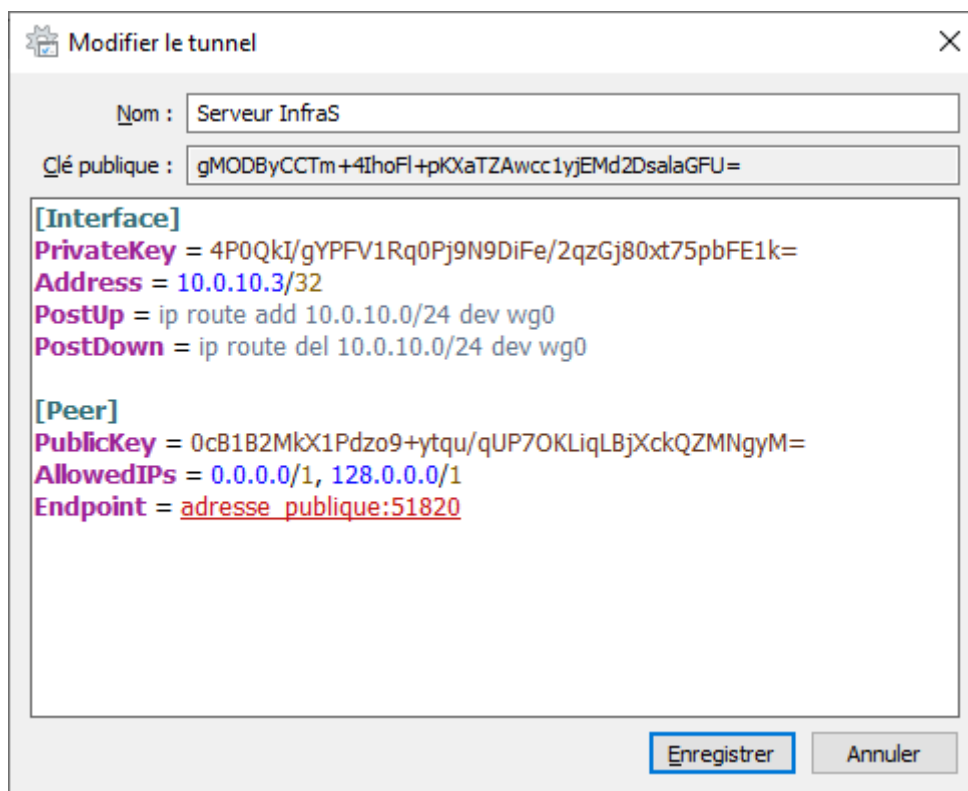
#### Configuration du VPN Wireguard

Malgré sa jeunesse, nous avons choisi comme VPN la solution de WireGuard au lieu de la plus connu OpenVPN. Ce dernier commence à accuser l'âge. Il est moins rapide, les latences sont plus faibles.

Nous viendrons donc configurer notre tunnel VPN via WireGuard pour accéder depuis l'extérieur aux services sensibles. Il suffit de créer les clés privées et publiques, configurer l'interface réseau et autoriser une connexion cliente sur le serveur.

*Le fichier de configuration se trouve en [annexe 4](#).*

La configuration de WireGuard sur le poste client est toute aussi simple. Attention toutefois à ne pas se mélanger avec les clés publiques/privées, chaque endpoint a besoin de sa propre paire de clé.



**Modifier le tunnel**

Nom : Serveur InfraS

Clé publique : gMODByCCTm+4IhoFl+pKXaTZAwwc1yjEMd2DsalaGFU=

**[Interface]**  
**PrivateKey** = 4P0QkI/gYPFV1Rq0Pj9N9DiFe/2qzGj80xt75pbFE1k=  
**Address** = 10.0.10.3/32  
**PostUp** = ip route add 10.0.10.0/24 dev wg0  
**PostDown** = ip route del 10.0.10.0/24 dev wg0

**[Peer]**  
**PublicKey** = 0cB1B2MkX1Pdzo9+ytqu/qUP7OKLiqLBjXckQZMNgyM=  
**AllowedIPs** = 0.0.0.0/1, 128.0.0.0/1  
**Endpoint** = adresse publique:51820

Enregistrer Annuler

Figure 19 - Configuration du client Windows

#### 6.5.6.4. Machine virtuelle « Services »

Sous cette dénomination seront regroupés les applications nécessaires au bon fonctionnement des différents services de l'infrastructure. Le serveur DNS permettra de faire les translations de nom de domaine vers IP et inversement, tandis que le reverse proxy s'occupera de l'accès aux différents services en WebUI.

##### Le DNS avec bind9

Le service DNS sera fourni par bind9 qui est le plus utilisé sur internet. Il servira à faire les résolutions entre les noms de machines et leurs adresses IP. Pour cela, nous devons configurer différentes zones de recherche : la zone directe et la zone indirecte. La zone directe permet de faire le lien entre un nom de machine et son adresse IP, alors que la zone indirecte fera l'inverse.

La configuration de bind9 se fait en trois étapes :

- La configuration du serveur maitre
- La configuration de la zone directe
- La configuration de la zone indirecte

Un nslookup sur une ip du réseau ou son hostname permettra de vérifier le bon fonctionnement du serveur DNS (drill ou dig donnera les informations similaires).

Le fichier de configuration complet se trouve en [annexe 4](#).

## Les conteneurs avec Docker

Nous utiliserons dès que possible des conteneurs Docker au lieu d'applications installées. Pourquoi ? Chaque application a ses propres besoins de dépendances. Une application Java par exemple nécessite une JDK installée, une application NodeJS nécessite la plateforme NodeJS installée et ainsi de suite.

Avec la conteneurisation des applications, tout est packagé directement dans cet « emballage ». Il se suffit à lui-même. Un conteneur est donc un ensemble d'application et de dépendances autonome. Il ne manque que le daemon Docker pour faire tourner l'ensemble.

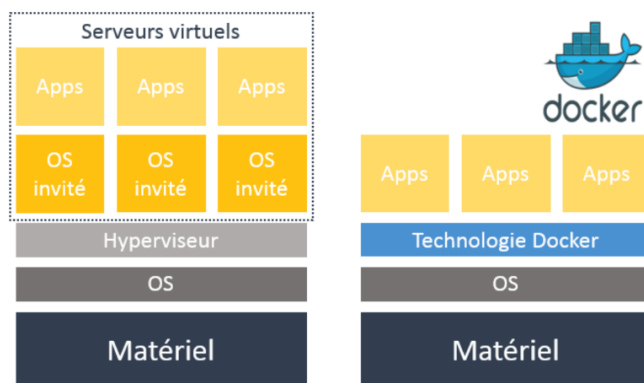


Figure 20 - Conteneur vs machine virtuelle

Grâce à ces conteneurs, il est rapide et facile de déployer des conteneurs spécifiques, soit en utilisant les ressources disponibles sur les registries publiques comme docker hub, soit en fabriquant nous-même nos propres images. L'installation de docker est bien documentée sur le site de docker ([ici](#)).

Pour déployer nos conteneurs docker, un simple docker run pourrait suffire. Mais comme nous allons personnaliser l'installation de nos stacks avec différents conteneurs, nous utiliserons « docker compose ».

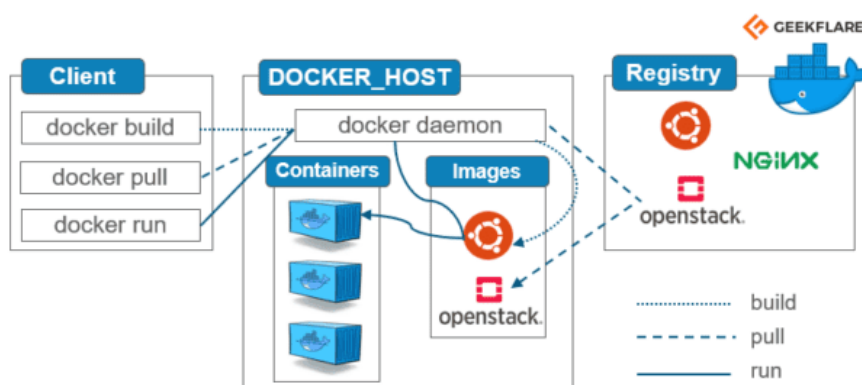


Figure 21 - Ecosystème docker

Il sera nécessaire de disposer également de volumes persistants (les données, les certificats SSL, la base de données), mais aussi de réseaux spécifiques pour isoler nos stacks. Chaque conteneur aura sa propre configuration.

Le fichier de configuration de la stack services est disponible en [annexe 6](#).

## Le reverse-proxy avec Nginx Proxy Manager

Nginx est un serveur web bien connu. Il est également un reverse proxy utilisé pour acheminer le trafic et le rediriger vers un autre serveur. La configuration de Nginx en tant que reverse proxy peut prendre du temps et être source d'erreurs et de mauvaises configurations.

Nginx Proxy Manager (NPM) est un système de gestion de reverse proxy qui fonctionne sur docker. Il est basé sur le serveur Nginx, et propose une WebUI pour une gestion et une configuration plus simple. Il prend en charge le SSL via Let's Encrypt, la gestion de compte utilisateur, des ACL ... et dispose d'une API, malheureusement moins bien accessible que NGINX. Pour cette raison, le choix n'est pas encore définitivement établi.

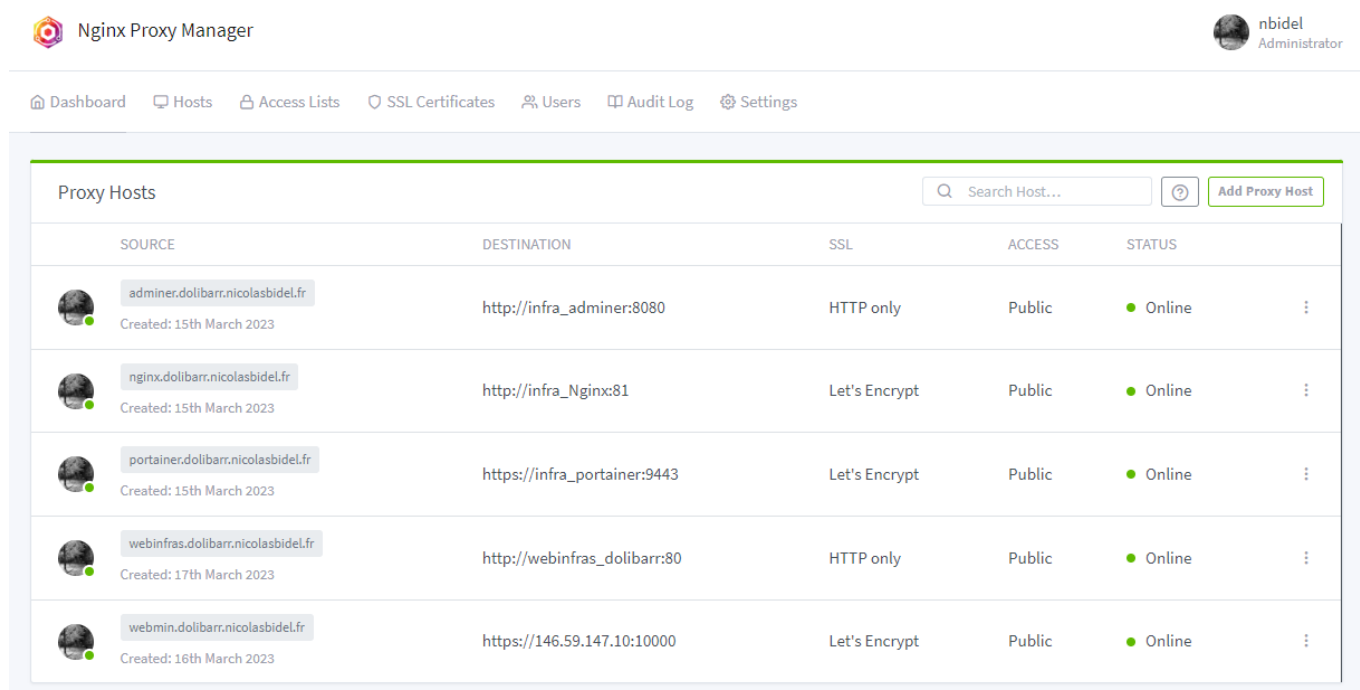


Figure 22 - Interface Nginx Proxy Manager

NPM permettra un accès simple et efficace aux divers WebUI des services accessibles sur l'infrastructure, comme portainer pour la gestion des conteneurs docker, prometheus, grafana pour la supervision ...

La configuration est simple, il suffit de connaître le Domain Name du service, le type de connexion (http/https), l'IP de la destination (ou son hostname) et le port rattaché. Il est possible de faire une demande de certificat SSL directement via l'interface.



Figure 23 - NPM Configuration d'un host

### La gestion des conteneurs avec Portainer

Portainer est une solution en WebUI pour la gestion des conteneurs Docker. Elle regroupe tout l'écosystème docker avec les images, les conteneurs, les réseaux, les volumes, et la gestion centralisée d'autres instances portainer.

Il est possible de voir les logs, les stats et inspecter tous les éléments docker. Tout ce qu'il est possible de faire via la CLI Docker est ici pris en charge.

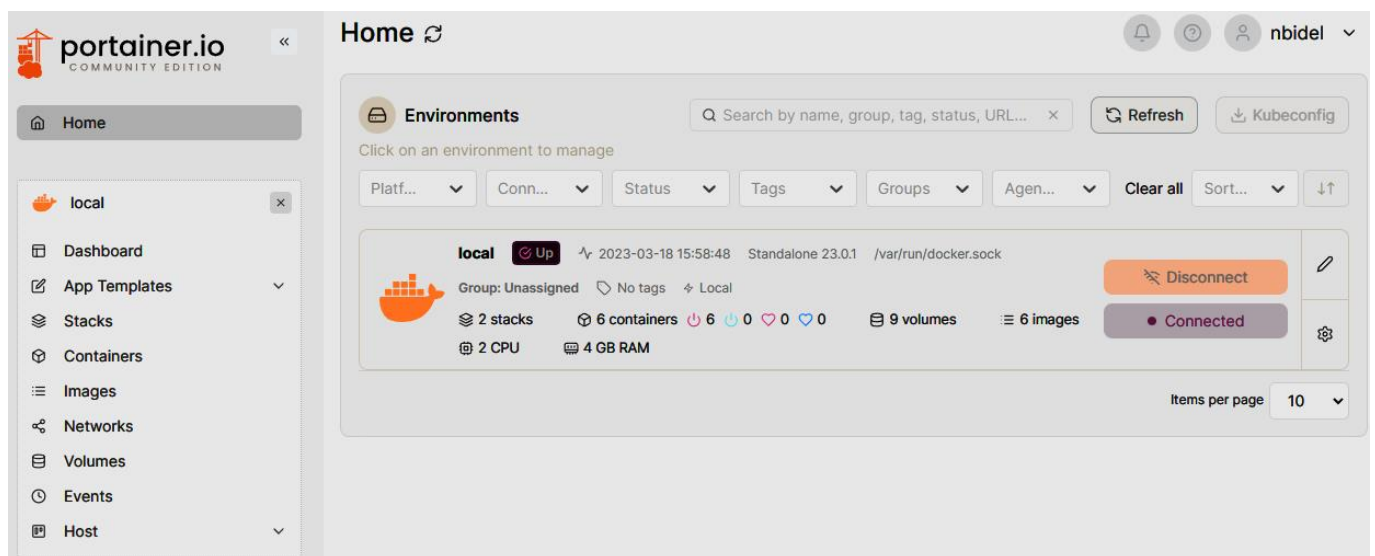


Figure 24 - Portainer et la gestion des conteneurs Docker



### 6.5.6.5. Supervision et métriques

Nous avons opté pour une simple exposition des métriques au lieu de configurer une solution complète de supervision style Nagios ou ZABBIX. Les métriques seront récupérées via les applications Node Exporter pour les machines virtuelles, et Container Exporter pour chaque conteneur. Comme pour la stack « services », nous opterons pour un déploiement via docker-compose.

La stack complète supervision contient prometheus / grafana / Node Exporter et Alerte Manager.

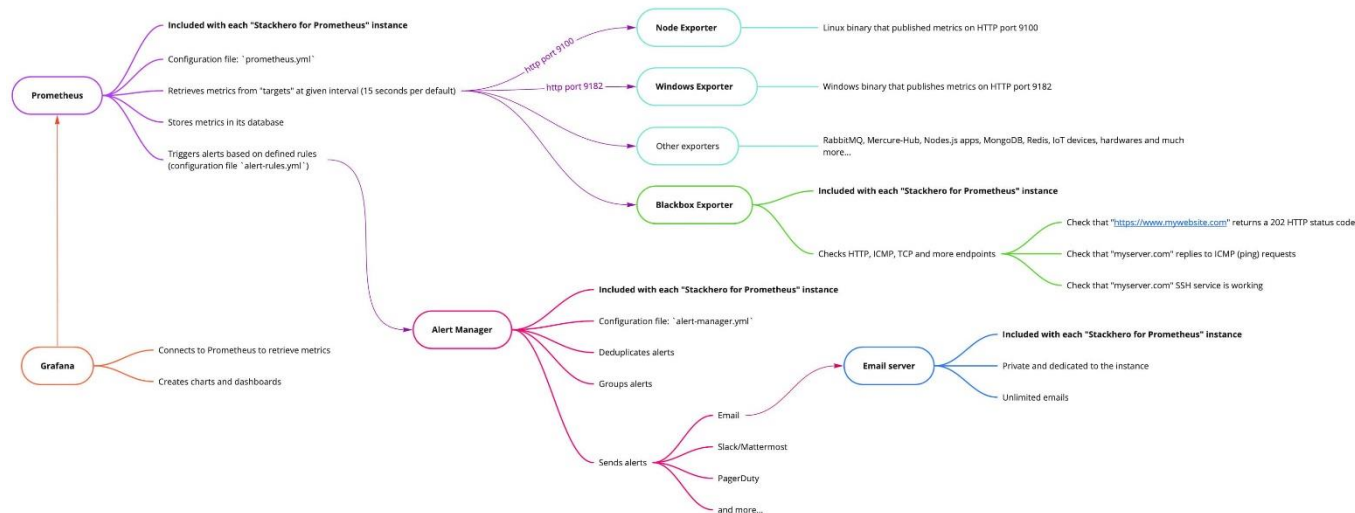


Figure 25 - Vue d'ensemble de la stack supervision

Le fichier de configuration de la stack supervision est disponible en [annexe 7](#).

### Prometheus : le scraping de données

Prometheus est donc un logiciel de surveillance informatique et de générateur d'alertes. Il enregistre les métriques en temps réel dans une base de données en se basant sur le contenu des points d'entrée exposé à l'aide du protocole HTTP.

Prometheus n'est pas avare de métriques. Il les collecte sous forme de séries temporelles. Il les récupère de façon active, en interrogeant divers Endpoint.

Le fichier de configuration de la stack supervision est disponible en [annexe 8](#).

Voici ce que nous donne Prometheus avec tous les endpoints de configurés, attendant les remontés de métriques.

Endpoint	State	Labels	Last Scrape	Scrape Duration	Error
<b>srv-preprod (0/2 up)</b>					
http://10.0.20.11:9100/metrics	DOWN	instance="10.0.20.11:9100" job="srv-preprod"	-41m 48s ago	10.0s	
http://10.0.20.11:9323/metrics	DOWN	instance="10.0.20.11:9323" job="srv-preprod"	-41m 56s ago	10.1s	
<b>srv-production (0/2 up)</b>					
http://10.0.20.10:9100/metrics	DOWN	instance="10.0.20.10:9100" job="srv-production"	-41m 54s ago	10.0s	
http://10.0.20.10:9323/metrics	DOWN	instance="10.0.20.10:9323" job="srv-production"	-41m 53s ago	10.0s	
<b>srv-services (0/2 up)</b>					
http://10.0.10.10:9100/metrics	DOWN	instance="10.0.10.10:9100" job="srv-services"	-41m 54s ago	10.1s	
http://10.0.10.10:9323/metrics	DOWN	instance="10.0.10.10:9323" job="srv-services"	-41m 52s ago	10.1s	
<b>srv-supervision (0/3 up)</b>					
http://10.0.10.11:9090/metrics	DOWN	instance="10.0.10.11:9090" job="srv-supervision"	-41m 51s ago	10.0s	
http://10.0.10.11:9100/metrics	DOWN	instance="10.0.10.11:9100" job="srv-supervision"	-42m 0s ago	10.0s	
http://10.0.10.11:9323/metrics	DOWN	instance="10.0.10.11:9323" job="srv-supervision"	-41m 56s ago	10.0s	

Figure 26 - Endpoint configurés dans Prometheus

Les métriques seront récupérées régulièrement comme déclarées dans le fichier de configuration de prometheus.

Le travail de Node Exporter sera d'exposer en http les informations du matériel (CPU, mémoire, disques, réseau) et du système d'exploitation via le Kernel. Les métriques sont pratiquement infinies.

```
# HELP go_gc_duration_seconds A summary of the pause duration of garbage collection cycles.
# TYPE go_gc_duration_seconds summary
go_gc_duration_seconds{quantile="0"} 0
go_gc_duration_seconds{quantile="0.25"} 0
go_gc_duration_seconds{quantile="0.5"} 0
go_gc_duration_seconds{quantile="0.75"} 0
go_gc_duration_seconds{quantile="1"} 0
go_gc_duration_seconds_sum 0
go_gc_duration_seconds_count 0
# HELP go_goroutines Number of goroutines that currently exist.
# TYPE go_goroutines gauge
go_goroutines 7
# HELP go_info Information about the Go environment.
# TYPE go_info gauge
go_info{version="go1.19.3"} 1
# HELP go_memstats_alloc_bytes Number of bytes allocated and still in use.
# TYPE go_memstats_alloc_bytes gauge
go_memstats_alloc_bytes 1111111111
```

Figure 27 - Métriques exposées par Node Exporter

## Grafana : le dashboard

L'ensemble sera intégré dans des dashboards sous grafana, permettant de visualiser les diverses données remontées. Il suffit de sélectionner un dashboard existant sur le site de grafana ou même de le construire soi-même.

On y choisi ensuite notre « data sources » et l'on obtient une vue d'ensemble des métriques récupérées par prometheus :

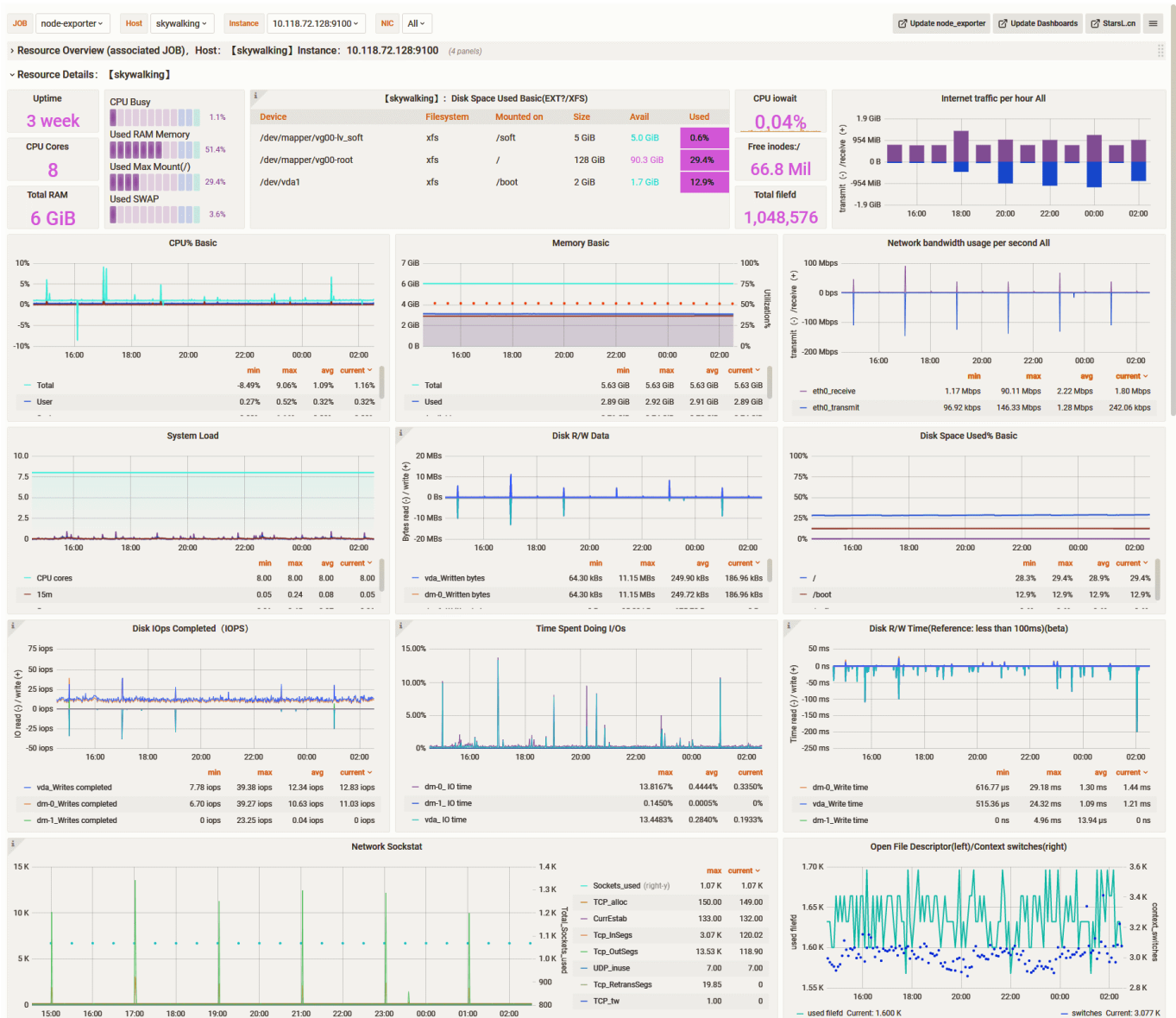


Figure 28 - Le dashboard de Node Exporter sur Grafana

## Alerte Manager : nous avertir

Alerte manager est la dernière pierre de cette stack de supervision. Il aura pour rôle de dédupliquer les alertes, les regrouper et nous les envoyer par le canal choisi (mail, discord, telegram ...). Mais avant de configurer Alerte Manager, il faut définir les règles de déclenchement.

1- Il faut déclarer dans le fichier prometheus.yml le fichier des Rules et activer alert-manager :

```
#Déclaration dans prometheus
rule_files:
  - "rules-alerte.yml"

alerting:
  alertmanagers:
  - static_configs:
    - targets:
      - localhost:9093
```

2- Le principe est de définir quelques valeurs clés, comme le nom de l'alerte, la valeur de déclenchement, la durée avant alerte, la sévérité (warning, critical) et diverses annotations :

```
#Exemple de règle
- alert: "HostOutOfDiskSpace"
  expr: (node_filesystem_avail_bytes * 100) / node_filesystem_size_bytes < 10 and ON (instance, device, mountpoint) node_filesystem_readonly == 0
  for: 2m
  labels:
    severity: "warning"
  annotations:
    summary:
    description: "Le disque est presque plein (< 10% left)"
    value: "{{ $value }}"
```

3- Une fois les règles définies, il faut configurer les Receivers qui sont des groupes d'intégration de notification (email, discord, telegram, Slack ...) :

```
#Exemple de receivers

receivers:
- name: 'email'
  email_configs:
  - to: 'receiver_mail_id@gmail.com'
    from: 'mail_id@gmail.com'
    smarthost: smtp.gmail.com:587
    auth_username: 'mail_id@gmail.com'
    auth_identity: 'mail_id@gmail.com'
    auth_password: 'password'
  inhibit_rules:
  - source_match:
    severity: 'critical'
    target_match:
    severity: 'warning'
    equal: ['alertname', 'dev', 'instance']
```

4- Enfin, il ne reste plus qu'à définir les routes. Ce sont les liens entre les Receivers et les règles :

```
#Exemple de route

route:
  group_by: ['alertname']
  group_wait: 10s
  group_interval: 10s
  repeat_interval: 10s
  receiver: 'email'
```

## 6.5.6.6. ERP/CRM avec Dolibarr

*Cette partie a été faite en stage et est encore en cours de développement.*

Nous arrivons à la partie finale mais centrale de ce projet : construire autour d'une architecture conteneurisée une instance autonome de Dolibarr.

Pour rappel, Dolibarr est un ERP/CRM open source. Il est fourni sous forme d'application Web permettant d'offrir une solution de gestion d'entreprise, association ou institution depuis n'importe quel point du globe.

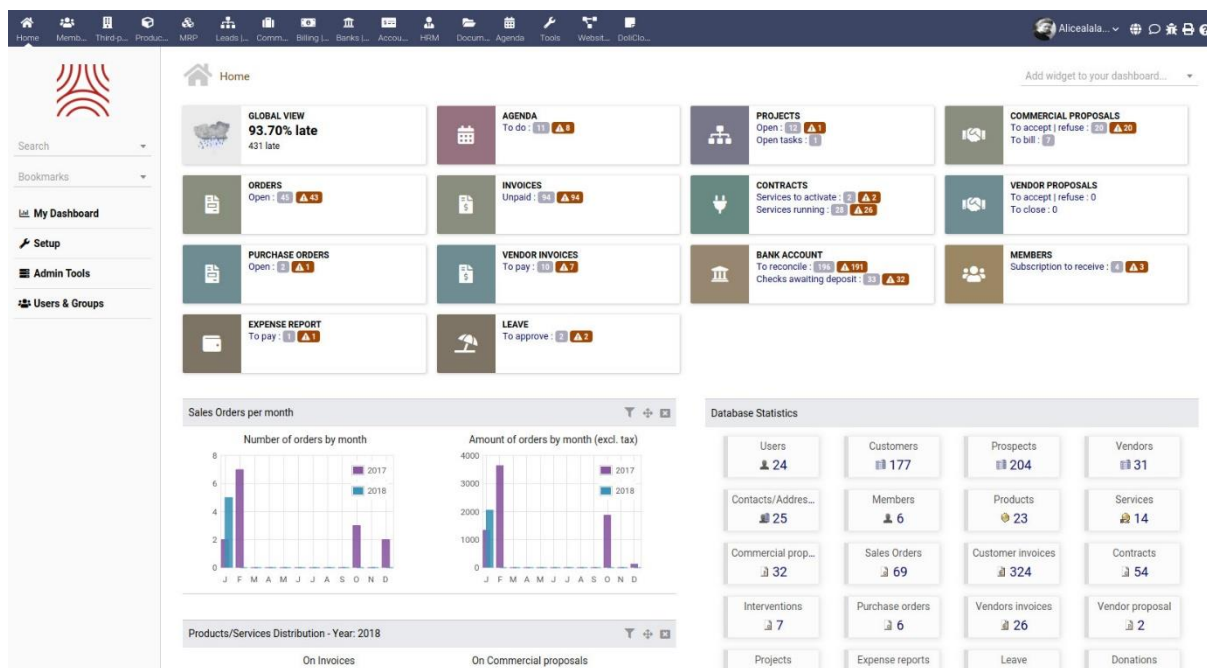


Figure 29 - ERP/CRM Dolibarr

Il nous faut donc un conteneur pour la base de données, un autre pour exposer l'application web Dolibarr, un réseau qui isolera ces deux conteneurs, et un script de déploiement.

Comme nous l'avons déjà fait plus haut avec les stacks « service » et « supervision », nous allons créer une stack « backend-Dolibarr » à base de conteneur officiel MariaDB pour la base de données et une image maison pour le moteur Apache/PHP.

### Image docker : à partir de rien, ou presque

Notre image personnalisée pour le backend de Dolibarr est basée sur une image de la fondation PHP qui est équipée d'une micro Debian Bullseye, d'un moteur Apache 2.4 et de PHP 7.4.33.

Nous y avons ajouté les divers modules PHP nécessaire au bon fonctionnement de Dolibarr comme Image Magick, gd, intl, imap, pecl, gmp, bcmath, xmlrpc, ldap, calendar, zip, mysql ...

Enfin nous faisons un nettoyage complet style « clean up » pour alléger au maximum l'image.

Le fichier docker-file de cette image est disponible en [annexe 10](#).

Il en reste plus qu'à construire l'image, se loguer sur le hub de Docker et la publier :

```
#Création de l'image à partir du docker-file
$ docker build -f dockerfile_dolibarr -t nicolas/infras-dolibarr

#Login sur le Hub de Docker
$ docker login --username $username --password $password

#Envois de l'image
$ docker push nbidel/infras-dolibarr:v1
```

### Bash : le script qui fait tout

Maintenant, tout va se passer via du scripting bash. Le premier script permet de créer l'infrastructure pour notre instance, c'est-à-dire le réseau Docker et les deux conteneurs qui y seront rattachés. On ira ajouter via l'API de NPM son « Proxy Host ».

Ce script prend en paramètres deux informations importantes que sont la version de Dolibarr à déployer et le nom de l'instance du client ou le paramètre --im :

```
$ ./creation_client.sh --v version-de-dolibarr -n nom-de-l-instance
$ ./creation_client.sh --im

#####
# Script de configuration pour création du backend dolibarr #
#####
START : [2023 03 18 20:26:37]
Quelle est la version de Dolibarr à installer (Exemple : '14' ou '15' ...) ? ____
Quel nom pour le client : ____
```

Le script est disponible en [annexe 11](#).

Le deuxième script qui est lancé pour le moment via un « docker exec » mais doit être remplacé par un [endpoint] à la fin de l'image Docker. Il permet de configurer l'instance complète (configuration de PHP, d'Apache, création de la base de données ...).

Le script est disponible en [annexe 12](#).

### 6.5.6.7. Sécurité

Le système d'information représente un patrimoine essentiel d'une organisation, il convient impérativement de le protéger. La sécurité informatique consiste à garantir que les ressources matérielles ou logicielles sont uniquement utilisées dans le cadre prévu. Pour cela, on doit prendre en compte les quatre objectifs de sécurités que sont la confidentialité, l'authenticité, l'intégrité et la disponibilité.

Certains de ces objectifs ont déjà été abordé plus haut, mais il est bon de faire une petite check-list de ce qui a déjà été mis en place.

## La confidentialité :

Le principe est que seules les personnes autorisées doivent avoir accès aux informations qui leur sont destinées (notions de droits ou permissions). Tout accès indésirable doit être empêché.

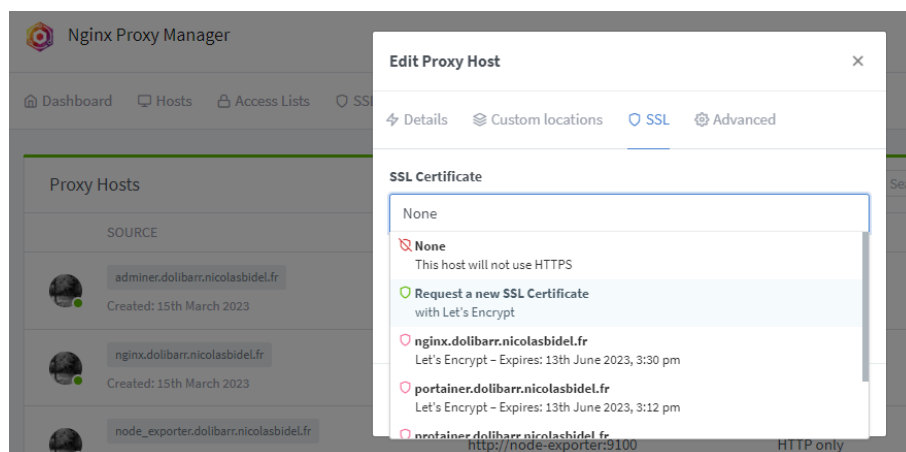
Pour cela, il a été mis en place différentes stratégies garantissant cette confidentialité :

1. Chiffrement par clé asymétrique : Les connexions **SSH** et **WireGuard** reposent sur un système de paire de clé dite asymétrique : une clé publique et une clé privée. Ces deux clés sont liées et non interchangeables, ce qui garantit la protection des échanges chiffrés. La clé publique code le message, la clé privée la décode.

```
#Générer la clé SSH :
ssh-keygen -t rsa -b 4096 -C "user@hote"

#Générer la clé WireGuard
$umask 077
$wg genkey | tee privatekey | wg pubkey > publickey
```

2. Les certificats : Lié au chiffrement, les certificats ont pour fonction de valider que la clé de chiffrement publique est authentique. Il détient les coordonnées du détenteur de la clé de chiffrement, sa date de validité ainsi que la valeur de la clé publique. Nous avons utilisé dans le cas de NPM des certificats Let's Encrypt bien connus.



3. Le protocole SSL : Ce protocole est un garant de la sécurité des échanges. C'est un protocole qui permet d'assurer la protection des transactions en ligne. Le HTTPS est une version sécurisée du http via le protocole SSL. Pour cette raison, tous les services WEB que nous avons déployé publiquement sont sécurisés via SSL (port 443 au lieu du 80 habituel).

Chaque échange de donnée doit donc être sécurisé de bout en bout. L'authenticité des utilisateurs est le second volet de la sécurité.

## L'authenticité :

Les utilisateurs doivent donc prouver leur identité par l'usage de code d'accès. Il ne faut pas mélanger identification et authentification : dans le premier cas, l'utilisateur n'est reconnu que par son identifiant public, tandis que dans le deuxième cas, il doit fournir un mot de passe ou une passe-phrase.



On appelle cela un secret. Mettre en correspondance un identifiant public avec un secret est le mécanisme permettant de garantir l'authenticité de l'identifiant.

- Les mots de passe : conformément aux recommandations de la CNIL, les mots de passe utilisés ont au minimum 12 caractères, avec majuscules, minuscules, chiffres et caractères spéciaux (exemple : « K4nG0uRou! »). Pour MariaDB, il a été choisi d'utiliser un mot de passe à 41 caractères (nombre maximum) avec les mêmes recommandations, auto-généré via la commande urandom :

```
MDP_CLIENT=$( < /dev/urandom tr -dc '[a-z][A-Z][0-9]_!@#%^&*()_+{}|:?= ' | fold -w 41 | head -n 1)
```

- Gestion des secrets : la gestion des mots de passe n'est pas à prendre à la légère. Dans notre cas, nous avons utilisé avec docker la gestion des fichiers d'environnement « .env » pour transmettre les informations entre conteneurs. Ce fichier caché est stocké en dehors des conteneurs avec des droits seulement pour l'utilisateur root. A la fin de la configuration de l'instance, le mot de passe passée en variable d'environnement est « unset ».

### L'intégrité :

Les données doivent être celles que l'on attend, et ne doivent pas être altérées de façon fortuite, illicite ou malveillante. En clair, les éléments considérés doivent être exacts et complets. Dans cette optique, une politique de sauvegarde doit être mise en place. Comme mentionné plus haut, cette partie sera produite dans un deuxième temps, les échéances étant trop courte pour poursuivre dans le projet initial.

L'accent sera mis sur la sauvegarde des données. La règle 3-2-1 sera appliquée. Elle stipule qu'il doit y avoir au moins trois copies de données ; deux des sauvegardes doivent être stockées sur des types de support différents, et au moins une sauvegarde doit être stockée hors site.

Les données peuvent être des fichiers de configuration système ou applicatif, des données clientes, mais aussi les fichiers journaux (LOG) des activités des utilisateurs, des anomalies et des événements liés à la sécurité. Une grande attention y sera portée pour être en règle avec la loi RGPD.

### La disponibilité :

Enfin, l'accès aux ressources du système d'information doit être permanent et sans faille durant les plages d'utilisation prévues. La disponibilité garantit que les utilisateurs ont un accès rapide et ininterrompu aux informations contenues dans le système. Les méthodes qui permettent cette disponibilité sont :

- La répartition équitable des ressources
- La haute disponibilité pour maintenir de façon opérationnelle les services et systèmes d'information (disponibilité « cinq neuf »)
- La redondance qui fait référence à la duplication ou basculement du système (reconstruction) via un procédé de cluster par exemple.





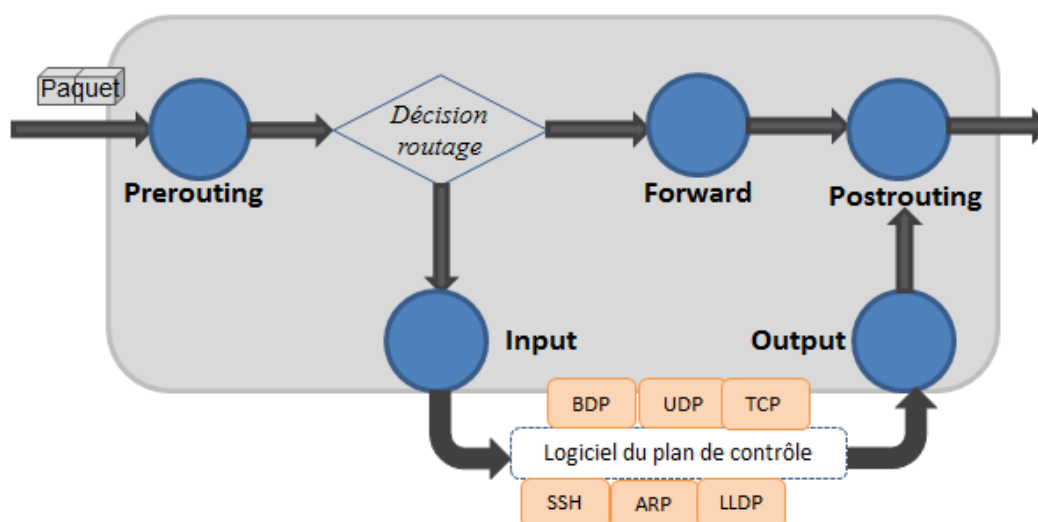
Un PRA ou plan de reprise d'activité devra être défini pour permettre une réaction efficace en cas de défaillance grave.

### Firewall :

Enfin, chaque machine virtuelle c'est vue protégé l'accès via des règles de firewall avec IPTables. En effet, toutes n'ont pas les mêmes nécessités réseau.

IPTable est un logiciel qui travaille sur les tables de sécurité du firewall, fournies par le noyau linux. Cette fonction manipule également les chaînes qu'un utilisateur peut ajouter ou supprimer, au niveau des règles du firewall, afin d'ajuster la sécurité de ses équipements. En somme, IPTable s'appuie sur différents modules du noyau linux, ainsi que différents protocoles permettant aux administrateurs réseau, ou aux administrateurs système d'en tirer profit.

Les règles IPTable ne peuvent être exécutées que si l'on possède le privilège du super-utilisateur root et ces dernières s'appuient sur les notions de chaînes (aussi appelées points d'inspection du trafic) et de tables pilotées par des règles de sécurité. On peut schématiser cela par le graphique ci-dessous :



Chaque paquet réseau sera inspecté et en fonction des règles établies en Prerouting, Input, Output, Postrouting ou même en Forward, un accès lui sera donné ou refusé.

```
# Exemple de règle pour le protocole SSH
iptables -A INPUT -i ens192 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ens192 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
iptables -A OUTPUT -o ens192 -p tcp --dport 22 -m state --state NEW,ESTABLISHED -j ACCEPT
iptables -A INPUT -i ens192 -p tcp --sport 22 -m state --state ESTABLISHED -j ACCEPT
```

Il faut donc établir pour chaque ports et protocoles utilisés des règles de firewall, en entrée, en sortie, sur les connexions initiales, sur les connexions déjà établies ... et ce sur chaque machine virtuelle.

Pour nous aider à établir ces règles, le schéma d'infrastructure réseau visible dans le [chapitre 6.5.5](#) nous aide grandement.

## Les connexions SSH :

L'utilisation comme seul facteur d'authentification le mot de passe n'est pas la technique la plus sécurisée. En effet, il est possible de se connecter via un autre facteur, la clé. Cette clé permet tout d'abord de ne pas avoir à retenir de mot de passe différents en fonction des utilisations, mais cela permet aussi de renforcer la sécurité.

L'authentification par pair de clé est un des moyens les plus courant avec SSH. Ce type de chiffrement a pour avantage de pouvoir distribuer la clé publique sans risquer que les messages soient déchiffrés avec, étant donné que seule la clé privée permet de déchiffrer les messages. L'algorithme utilisé est le RSA avec un niveau de 4096 bits.

De plus, une restriction sera faite au niveau des serveurs SSH, IP source prédéfinie, port changé, utilisateur root exclu, nombre de tentatives limités...

Voici un exemple type de fichier de configuration. Il est adapté en fonction des ressources (utilisateurs et machine distantes) ayant accès au serveurs SSH :

```
Port 202
ListenAddress 10.0.10.xxx

HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_ecdsa_key
HostKey /etc/ssh/ssh_host_ed25519_key
LogLevel VERBOSE

PermitRootLogin yes
MaxAuthTries 2
MaxSessions 10
ClientAliveInterval 30
PubkeyAuthentication yes

AuthorizedKeysFile .ssh/authorized_keys .ssh/authorized_keys2
```

## 7. Conclusion

Ce projet fut pour moi une expérience fort enrichissante. Il m'a permis d'asseoir les connaissances acquises durant ces 5 mois de formation. En effet, malgré beaucoup de mises en situation via des travaux pratiques, certains points n'étaient pas maîtrisés.

J'ai pu développer mes compétences sur les deux hyperviseur ESXi et Proxmox VE, avec la gestion des réseaux, des datastores, le templating. J'ai même pu m'initier au clustering sans toutefois avoir eu le temps de le déployer pour ce projet.

Docker fût pour moi une véritable révélation. Travailler sur les conteneurs fut un réel plaisir, malgré des défis de compréhension au début assez troublants (gestion des volumes persistants). J'ai pu faire ma première image Docker et l'utiliser tout au long de ce projet. Malgré la mauvaise réputation de bash, j'ai bien aimé « développer » mes scripts de déploiement d'instance web Dolibarr.

Toutefois, j'ai beaucoup de regrets sur le manque de temps disponible dans ce genre de formation. J'ai beaucoup appris, mais pour pouvoir finaliser le projet que j'avais en tête, il m'aurait fallu bien 6 mois de plus. Je regrette par exemple de ne pas avoir eu le temps de mettre en pratique durant ce projet les quelques compétences acquises avec Ansible. J'aurais aimé passer plus de temps pour automatiser les déploiements avec cet outil vraiment incroyable. Je n'ai pas eu le temps non plus de déployer une Infrastructure As Code avec les produits de HiCorps comme Packer et Terraform.

Le projet est fonctionnel à ce jour, mais mérite de grandement être amélioré avec des outils comme Ansible et Terraform et aussi documenté. Une étape importante sera la mise en place de solution de sauvegarde et de clustering ou HA.

Ce projet fût lors de mon stage une bonne introduction et préparation aux problématiques terrain comme la méthodologie, l'organisation, la tenue des échéances et même les impondérables comme les pannes internet ou matériel comme sur mon serveur DELL R620.

Finalement, je sais maintenant vers quoi m'orienter pour ma future carrière professionnelle : le DevOps et l'automatisation. Merci d'avoir pris le temps de lire jusqu'au bout.

## 8. Glossaire

**AGILE** : méthodologie de gestion de projet basé sur le Manifeste Agile édité en 2001. Elle prône l'adaptation des procédés de création au fil de l'évolution du projet.

**AMDEC** : Méthode d'Analyse des Modes de Défaillance, de leur Effets et de leur Criticité. C'est un outil utilisé dans la démarche qualité et dans le cadre de la sûreté de fonctionnement.

**BAREMETAL** : Un serveur baremetal est un serveur informatique physique qui est utilisé par un seul consommateur, ou locataire, uniquement. Chaque serveur proposé à la location est un matériel physique distinct qui est un serveur fonctionnel à part entière.

**CLUSTER** : En informatique il s'agit d'un groupe de ressources, telles que des serveurs. Ce groupe agit comme un seul et même système. Il affiche ainsi une disponibilité élevée, voire, dans certains cas, des fonctions de traitement en parallèle et d'équilibrage de la charge.

**CRM** : Le CRM (**C**ustomer **R**elationship **M**anagement), est l'équivalent de la Gestion de la Relation Client (GRC) en français. Il vise à optimiser les relations et interactions d'une entreprise avec ses prospects et clients dans une démarche de fidélisation.

**DASHBOARD** : Il s'agit d'un ensemble d'indicateurs de performance (KPIs) qui permettent de suivre une activité, si possible en temps réel, afin de réagir en conséquence le plus rapidement possible.

**DEVOPS** : DevOps est un ensemble de pratiques et d'outils, ainsi qu'une philosophie culturelle. Son but est d'automatiser et d'intégrer les processus entre les équipes de développement et informatiques.

**DNS** : Le **D**omain **N**ame **S**ystem est un service informatique distribué qui associe les noms de domaine Internet avec leurs adresses IP ou d'autres types d'enregistrements.

**ERP** : Un ERP est un système de gestion des ressources pour les entreprises (Enterprise Resource Planning).

**ENDPOINT** : Un point de terminaison de communication est un type de nœud de réseau de communication. C'est une interface exposée par un interlocuteur ou par un canal de communication.

**PHP-FPM** : PHP-FPM (**F**astCGI **P**rocess **M**anager) est une interface permettant la communication entre un serveur Web et PHP, basée sur le protocole FastCGI.

**GANTT** : Le diagramme de Gantt est un outil utilisé en ordonnancement et en gestion de projet et permettant de visualiser dans le temps les diverses tâches composant un projet. Il s'agit d'une représentation d'un graphe connexe, valué et orienté, qui permet de représenter graphiquement l'avancement du projet.

**HYPERVISEUR** : Un hyperviseur est une plate-forme de virtualisation qui permet à plusieurs systèmes d'exploitation de fonctionner en parallèle dans une seule machine physique.



**LTS** : En informatique, une version **Long-Term Support** désigne une version spécifique d'un logiciel dont le support est assuré pour une période de temps plus longue que la normale.

**OCEN** : Les **O**érateurs **C**ommerciaux d'**E**nvergure **N**ationale désignent les 4 grands opérateurs nationaux présents commercialement sur les réseaux de fibre à l'abonné grand public (Bouygues Telecom, Free, Orange et SFR).

**RACI** : L'acronyme RACI ou RAM désigne dans le domaine du management une matrice des responsabilités. Elle indique les rôles et les responsabilités des intervenants au sein de chaque processus et activité.

**REGISTRY** : Une registry (registre en français) est une base de données qui sert au stockage des structures de données utilisées pour la communication entre applications. C'est un point central où les développeurs peuvent enregistrer et trouver des schémas utiles pour créer des applications spécifiques.

**REVERSE PROXY** : Un proxy inverse ou serveur mandataire inverse est un type de serveur, habituellement placé en frontal de serveurs web. Contrairement au serveur proxy qui permet à un utilisateur d'accéder au réseau Internet, le proxy inverse permet à un utilisateur d'Internet d'accéder à des serveurs internes.

**SCRAPING** : Le web scraping est une technique d'extraction du contenu de sites Web, via un script ou un programme, dans le but de le transformer pour permettre son utilisation dans un autre contexte comme l'enrichissement de bases de données, le référencement ou l'exploration de données.

**SCRUM** : Scrum est un Framework de gestion de projet flexible qui aide les équipes à structurer et à gérer leur travail selon un ensemble de valeurs, de principes et de pratiques.

**STACK** : Une stack (pile en français) est une structure de données fondée sur le principe « dernier arrivé, premier sorti », ce qui veut dire qu'en général, le dernier élément ajouté à la pile est le premier à en sortir.

**TERRAFORM** : Terraform est un environnement logiciel d'« infrastructure as code » publié en open-source par la société HashiCorp. Cet outil permet d'automatiser la construction des ressources d'une infrastructure de centre de données comme un réseau, des machines virtuelles, un groupe de sécurité ou une base de données.

**TLS** : La **T**ransport **L**ayer **S**ecurity ou « Sécurité de la couche de transport », et son prédécesseur la Secure Sockets Layer ou « Couche de sockets sécurisée », sont des protocoles de sécurisation des échanges par réseau informatique, notamment par Internet.

**VPN** : Un **V**irtual **P**rivate **N**etwork est un « réseau privé virtuel », à savoir un service qui établit une connexion chiffrée et sécurisée entre deux points distincts.

**VPS** : Un **V**irtual **P**rivate **S**erver (serveur dédié virtuel en français) est une méthode de partitionnement d'un serveur en plusieurs serveurs virtuels indépendants qui ont chacun les caractéristiques d'un serveur dédié, en utilisant des techniques de virtualisation.



## 9. Bibliographie – Webographie

---

Cours Vidéos :

- <https://www.linkedin.com/learning/l-essentiel-de-la-gestion-du-dns> par Rudi Bruchez
- <https://www.linkedin.com/learning/decouvrir-docker> par Samir Lakhdari
- <https://www.linkedin.com/learning/decouvrir-vsphere-6> par Samir Lakhdari

Livres :

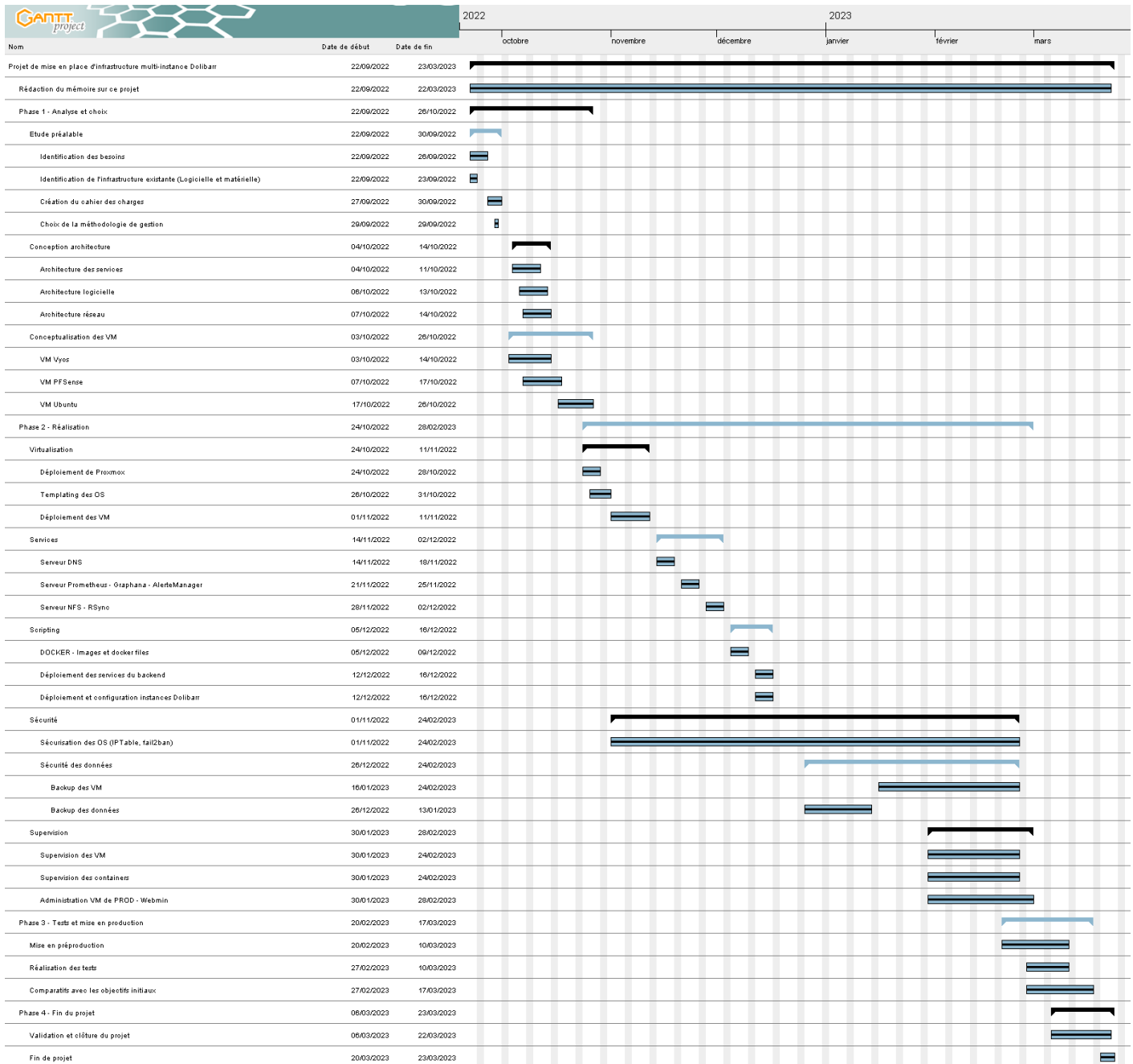
Ansible for DevOps de **Jeff Geerling**

Sites Web :

- <https://hub.docker.com/>
- <https://docs.docker.com/>
- <https://www.ubuntu-fr.org/>
- <https://github.com/Dolibarr/dolibarr>
- <https://www.it-connect.fr/>

## 10. Annexes

### 10.1. Annexe 1 : Diagramme de GANTT



## 10.2. Annexe 2 : Script « clean-up »

```
#####
##                                CLEAN-UP SCRIPT                                ##
#####
#!/bin/bash

# #Supprimer les fichiers de machine-id :
rm /etc/machine-id
rm /var/lib/dbus/machine-id

#Récréer un nouveau fichier machine-id vide :
touch /etc/machine-id

#Créer un lien symbolique du fichier /var/lib/dbus/machine-id :
ln -s /etc/machine-id /var/lib/dbus/machine-id

# Clean up de Apt :
apt-get autoremove
apt-get autoclean
apt-get clean

# Clean up de /tmp :
rm -rvf /tmp/*

# Clean up des logs :
rm -vf /var/log/*.log.*
rm -vf /var/log/*/*

# Clean up de l'historique :
cat /dev/null > ~/.bash_history && history -c
```

## 10.3. Annexe 3 : Configuration de VyOS

```
#####
##                                Config VyOS                                ##
#####

Configure
#Configuration du nom et du domaine
set system domain-name 'projet-pro.lab'
set system host-name 'vRouter'

#Configuration des interfaces
set interfaces ethernet eth0 description 'WAN'
set interfaces ethernet eth0 address '192.168.8.30/24'
set interfaces ethernet eth0 hw-id '00:0c:29:4d:ef:e0'
set interfaces ethernet eth1 description 'LAN'
set interfaces ethernet eth1 address '10.0.10.1/24'
set interfaces ethernet eth1 hw-id '00:0c:29:4d:ef:ea'

#Configuration du routage
set nat source rule 100 description 'NAT-LAN-TO-WAN'
set nat source rule 100 outbound-interface 'eth0'
set nat source rule 100 translation address 'masquerade'
set protocols static route 0.0.0.0/0 next-hop 192.168.8.1
set protocols static route 10.0.10.0/24 description 'LAN'
set protocols static route 10.0.10.0/24 next-hop 10.0.10.1

#Configuration du user et de ses droits
set system config-management commit-revisions '100'
set system login user vyos level 'admin'
set system login user vyos authentication encrypted-password
'$6$1qKCY/xVhtMBT7Me$b7sNKEyXOb4k13G9tRRZ2EcVSG6efelx3y25m5LFV578sD3q0gyN0L71UK5T829y/U2FffHmfOT17c4x6U
S/W0'

set system syslog global facility all level 'info'
set system syslog global facility protocols level 'debug'

set service ssh port '22'

# Configuration du serveur de temps
set system ntp server '0.pool.ntp.org'
```



```

set system ntp server '1.pool.ntp.org'
set system ntp server '2.pool.ntp.org'
set system time-zone 'Europe/Paris'

#Activation et configuration firewall
set firewall all-ping 'enable'
set firewall broadcast-ping 'disable'
set firewall receive-redirects 'disable'
set firewall send-redirects 'enable'
set firewall source-validation 'disable'
set firewall syn-cookies 'enable'
set firewall twa-hazards-protection 'disable'

# Règles pour autoriser les connexion http, HTTPS et SSH depuis le WAN
set firewall name OUTSIDE-IN default-action 'drop'
set firewall name OUTSIDE-IN rule 10 action 'accept'
set firewall name OUTSIDE-IN rule 10 state established 'enable'
set firewall name OUTSIDE-IN rule 10 state related 'enable'
set firewall name OUTSIDE-IN rule 20 action 'accept'
set firewall name OUTSIDE-IN rule 20 destination port '80'
set firewall name OUTSIDE-IN rule 20 protocol 'tcp'
set firewall name OUTSIDE-IN rule 20 state new 'enable'
set firewall name OUTSIDE-IN rule 21 action 'accept'
set firewall name OUTSIDE-IN rule 21 destination port '443'
set firewall name OUTSIDE-IN rule 21 protocol 'tcp'
set firewall name OUTSIDE-IN rule 21 state new 'enable'
set firewall name OUTSIDE-IN rule 22 action 'accept'
set firewall name OUTSIDE-IN rule 22 destination port '22'
set firewall name OUTSIDE-IN rule 22 protocol 'tcp'
set firewall name OUTSIDE-IN rule 22 state new 'enable'

#Règle pour autoriser les connexions déjà établies ainsi et ping/ntp/ssh
set firewall name OUTSIDE-LOCAL default-action 'drop'
set firewall name OUTSIDE-LOCAL rule 10 action 'accept'
set firewall name OUTSIDE-LOCAL rule 10 state established 'enable'
set firewall name OUTSIDE-LOCAL rule 10 state related 'enable'
set firewall name OUTSIDE-LOCAL rule 20 action 'accept'
set firewall name OUTSIDE-LOCAL rule 20 icmp type-name 'echo-request'
set firewall name OUTSIDE-LOCAL rule 20 protocol 'icmp'
set firewall name OUTSIDE-LOCAL rule 20 state new 'enable'
set firewall name OUTSIDE-LOCAL rule 30 action 'drop'
set firewall name OUTSIDE-LOCAL rule 30 destination port '22'
set firewall name OUTSIDE-LOCAL rule 30 protocol 'tcp'
set firewall name OUTSIDE-LOCAL rule 30 recent count '4'
set firewall name OUTSIDE-LOCAL rule 30 recent time '60'
set firewall name OUTSIDE-LOCAL rule 30 state new 'enable'
set firewall name OUTSIDE-LOCAL rule 31 action 'accept'
set firewall name OUTSIDE-LOCAL rule 31 destination port '22'
set firewall name OUTSIDE-LOCAL rule 31 protocol 'tcp'
set firewall name OUTSIDE-LOCAL rule 31 state new 'enable'
set firewall name OUTSIDE-LOCAL rule 40 action 'accept'
set firewall name OUTSIDE-LOCAL rule 40 destination port '161'
set firewall name OUTSIDE-LOCAL rule 40 protocol 'udp'
set firewall name OUTSIDE-LOCAL rule 40 state new 'enable'

set interfaces ethernet eth0 firewall in name 'OUTSIDE-IN'
set interfaces ethernet eth1 firewall local name 'OUTSIDE-LOCAL'

commit
save

```



## 10.4. Annexe 4 : Configuration de WireGuard sur VyOS

```
#####
##                               VyOS WireGuard CONFIG                               ##
#####

#On génère la paire de clé
wg genkey | tee /etc/wireguard/wg-private.key | wg pubkey | sudo tee /etc/wireguard/wg-public.key

configure
#Configuration de Wireguard
set interfaces wireguard wg0 address 10.0.10.2/24
set interfaces wireguard wg0 port 51820

#Création du client WireGuard
generate wireguard client-config nbidel_ID_01 interface wg0 server IP_PUBLIC address 10.0.0.4/24

# Création de l'interface du client
set interfaces wireguard wg0 peer nbidel_ID_01 allowed-ips '10.0.10.3/32'
set interfaces wireguard wg0 peer nbidel_ID_01 pubkey 'Public key'
commit; save

# Configuration du client WireGuard sous Windows par exemple
[Interface]
PrivateKey = 'Private key'
Address = 10.0.0.4/32
DNS = 1.1.1.1
[Peer]
PublicKey = 'Public key'
Endpoint = IP_PUBLIC:51820
AllowedIPs = 0.0.0.0/0, :::/0
```

## 10.5. Annexe 5 : Configuration de bind9

```
#####
##                               VyOS WireGuard CONFIG                               ##
#####

#Edition du fichier de configuration du serveur maitre
#/etc/bind/named.conf.local

#Configuration de la zone directe
zone "projetpro.lab" IN {
    type master;
    file "/etc/bind/direct-projetpro.lab";
};

#Configuration de la zone indirecte
zone "10.0.10.in-addr.arpa" IN {
    type master;
    file "/etc/bind/invers-projetpro.lab";
};

#Edition du fichier de configuration de la zone directe
#/etc/bind/direct-projetpro.lab
$TTL      604800
@         IN      SOA      projetpro.lab. root.projetpro.lab. (
                        1          ; Serial
                        604800     ; Refresh
                        86400      ; Retry
                        2419200    ; Expire
                        604800 )   ; Negative Cache TTL
;
@         IN      NS       srv-dhcp-dns.projetpro.lab.
vrouter   IN      A        10.0.10.1
client-relay IN    A        10.0.10.4
srv-dhcp-dns IN    A        10.0.10.10
srv-supervision IN A        10.0.10.11
srv-prod   IN      A        10.0.10.20
srv-pre-prod IN    A        10.0.10.21

#Edition du fichier de configuration de la zone indirecte
```



```

#/etc/bind/indirect-projetpro.lab
$TTL      604800
@          IN      SOA      projetpro.lab. root.projetpro.lab. (
                        1          ; Serial
                        604800     ; Refresh
                        86400      ; Retry
                        2419200    ; Expire
                        604800 )   ; Negative Cache TTL
;
@          IN      NS       srv-dhcp-dns.projetpro.lab.

; zone serveurs systeme
$ORIGIN    10.0.10.in-addr.arpa.
1          IN      PTR      vrouter.
4          IN      PTR      client-relay
10         IN      PTR      srv-dhcp-dns.projetpro.lab.
11         IN      PTR      srv-supervision.projetpro.lab.
20         IN      PTR      srv-prod.projetpro.lab.
21         IN      PTR      srv-pre-prod.projetpro.lab.

```

## 10.6. Annexe 6 : Fichier docker-compose stack « service »

```

#####
##                                docker-compose stack « service »                                ##
#####
version: "3"
services:
  portainer:
    container_name: infra_portainer
    image: portainer/portainer-ce:latest
    restart: unless-stopped
    volumes:
      - /_infra/portainer/data:/data
      - /var/run/docker.sock:/var/run/docker.sock
    ports:
      - '127.0.0.1:9443:9443'
  nginx:
    image: 'jc21/nginx-proxy-manager:latest'
    container_name: infra_Nginx
    restart: unless-stopped
    ports:
      # These ports are in format <host-port>:<container-port>
      - '80:80' # Public HTTP Port
      - '443:443' # Public HTTPS Port
      - '81:81' # Admin Web Port
      # Add any other Stream port you want to expose
      # - '21:21' # FTP
    environment:
      DB_MYSQL_HOST: "infra_mariadb"
      DB_MYSQL_PORT: 3306
      DB_MYSQL_USER: "npm"
      DB_MYSQL_PASSWORD: "npm"
      DB_MYSQL_NAME: "npm"
    volumes:
      - /_infra/nginx/data:/data
      - /_infra/nginx/letsencrypt:/etc/letsencrypt
    depends_on:
      - db
  db:
    image: 'jc21/mariadb-aria:latest'
    container_name: infra_mariadb
    restart: unless-stopped
    environment:
      MYSQL_ROOT_PASSWORD: 'npm'
      MYSQL_DATABASE: 'npm'
      MYSQL_USER: 'npm'
      MYSQL_PASSWORD: 'npm'
    volumes:
      - /_infra/mariadb/data/mysql:/var/lib/mysql

```

## 10.7. Annexe 7 : Fichier docker-compose stack « supervision »

```
#####
##                                docker-compose stack « supervision »                                ##
#####
version: 3

services:
  portainer-agent:
    container_name: portainer_agent
    image: portainer/agent
    volumes:
      - /var/run/docker.sock:/var/run/docker.sock
      - /var/lib/docker/volumes:/var/lib/docker/volumes
    ports:
      - 127.0.0.1:9001:9001
    restart: always

  prometheus:
    image: prom/prometheus
    container_name: prometheus
    ports:
      - 127.0.0.1:9090:9090
    restart: unless-stopped
    volumes:
      - /data/compose/prometheus-grafana/prometheus/prometheus.yml:/etc/prometheus/prometheus.yml

  grafana:
    image: grafana/grafana
    container_name: grafana
    ports:
      - 127.0.0.1:3000:3000
    restart: unless-stopped
    environment:
      - GF_SECURITY_ADMIN_USER=admin
      - GF_SECURITY_ADMIN_PASSWORD=Admin&n2b34i
    volumes:
      - /data/compose/prometheus-grafana/grafana:/etc/grafana/provisioning/datasources
      - /data/grafana:/var/lib/grafana

  alertmanager:
    container_name: alertmanager
    image: prom/alertmanager
    restart: unless-stopped
    ports:
      - 9093:9093
    volumes:
      - /data/alertmanager/config:/config
      - /data/alertmanager/data:/data
    command: --config.file=/config/alertmanager.yml --log.level=debug
```

## 10.8. Annexe 8 : Fichier prometheus.yml

```
#####
##                                fichier de configuration prometheus                                ##
#####
global:
  scrape_interval: 15s
  scrape_timeout: 10s
  evaluation_interval: 15s
alerting:
  alertmanagers:
    - static_configs:
        - targets:
            - localhost:9093
      scheme: http
      timeout: 10s

scrape_configs:
# Serveur de services
- job_name: srv-services
  honor_timestamps: true
  metrics_path: /metrics
```

```

scheme: http
static_configs:
- targets:
  - 10.0.10.10:9100
  - 10.0.10.10:9323

# Serveur de supervision
- job_name: srv-supervision
  honor_timestamps: true
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - 10.0.10.11:9090
    - 10.0.10.11:9100
    - 10.0.10.11:9323

#Serveur de production
- job_name: srv-production
  honor_timestamps: true
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - 10.0.10.20:9100
    - 10.0.10.20:9323

# Serveur de pré-production / developpement
- job_name: srv-preprod
  honor_timestamps: true
  metrics_path: /metrics
  scheme: http
  static_configs:
  - targets:
    - 10.0.10.21:9100
    - 10.0.10.21:9323

```

## 10.9. Annexe 9 : Fichier configuration alertmanager.yml

```

#####
##                                fichier de configuration Alerte Manager                                ##
#####
route:
  receiver: 'mail'
  repeat_interval: 4h
  group_by: [ alertname ]

receivers:
- name: 'mail'
  email_configs:
  - smarthost: 'smtp.gmail.com:465'
    auth_username: 'nicolas.bidel74@gmail.com'
    auth_password: "Gmailn2B34i"
    from: 'nicolas.bidel74@gmail.com'
    to: 'nicolas.bidel74@gmail.com'

```

## 10.10. Annexe 10 : Docker-file image Backend-Dolibarr

```
#####
##                                docker-compose stack « service »                                ##
#####
FROM php:7.4.33-apache-bullseye
LABEL description="Custom backend Dolibarr for infraS"
LABEL maintainer="Nicolas Bidel <nicolas.bidel@wanadoo.fr>"

#installation des dependances et app
RUN apt-get update -y \
    && apt-get install -y --no-install-recommends sudo tree nano libjpeg62-turbo-dev libldap2-dev \
    libpq-dev libxml2-dev libzip-dev default-mysql-client cron

# Configuration des app
RUN wget https://raw.githubusercontent.com/NicolasBernaerts/ubuntu-scripts/master/image/imagemagick-
enable-pdf-install.sh \
    && chmod +x ./imagemagick-enable-pdf-install.sh \
    && bash ./imagemagick-enable-pdf-install.sh \
    && rm ./imagemagick-enable-pdf-install.sh

# extension: gd
RUN apt-get install -y libfreetype-dev libpng-dev libjpeg-dev \
    && docker-php-ext-configure gd --with-freetype --with-jpeg \
    && docker-php-ext-install -j$(nproc) gd \
    && apt-get autoremove -y libfreetype-dev libpng-dev libjpeg-dev

# extension: intl
RUN apt-get install -y libicu-dev \
    && docker-php-ext-install -j$(nproc) intl \
    && apt-get autoremove -y libicu-dev

# extension: imap
RUN apt-get install -y libc-client-dev libkrb5-dev \
    && docker-php-ext-configure imap --with-kerberos --with-imap-ssl \
    && docker-php-ext-install -j$(nproc) imap \
    && apt-get autoremove -y libc-client-dev libkrb5-dev

# extension: imagick (pecl)
RUN apt-get install -y libmagickwand-dev --no-install-recommends ghostscript --no-install-recommends \
    && printf "\n" | pecl install imagick \
    && docker-php-ext-enable imagick

# extension: gmp
RUN apt-get install -y libgmp-dev \
    && docker-php-ext-install -j$(nproc) gmp \
    && apt-get autoremove -y libgmp-dev

RUN apt-get install libxml2-dev -y \
    && docker-php-ext-install -j$(nproc) bcmath xmlrpc ldap calendar zip mysqli \
    && apt-get autoremove -y libxml2-dev

# configurations diverses
RUN a2enmod actions proxy_fcgi setenvif alias cgi expires headers http2 proxy_http rewrite \
    socache_shmcb ssl

RUN apt-get autoremove -y && apt-get purge -y && rm -rf /var/lib/apt/lists/* && rm -rf /tmp/*

# Finallisation
EXPOSE 80/TCP 443/TCP
ENTRYPOINT ["/usr/sbin/apache2ctl", "-D", "FOREGROUND"]
```



## 10.11. Annexe 11 : Script création infrastructure Dolibarr

```
#####
##                                docker-compose stack « service »                                ##
#####
#!/bin/bash
clear
echo "#####"
echo "# Script de configuration pour création du backend dolibarr #"
echo "#####"

echo "START : [$(date +"%Y %m %d %H:%M:%S")]"
filename='creation_client.sh'

# Dans un premier temps on vérifie que l'utilisateur a bien rentré des paramètres ou le paramètre --im
echo $1
if [[ -n $1 ]] && [[ $1 = "--im" ]]; then
    read -rp "Quelle est la version de Dolibarr à installer (Exemple : '14' ou '15' ...) ? " DOLI_VERSION
    read -rp "Quel nom pour le client : " NOM_CLIENT
elif [[ $# -ge 2 ]]; then
    while [ $# -gt 0 ]; do
        if [[ $1 == "--" ]]; then
            z="{1/--/}"
            declare $z="$2"
            fi
        case "$z" in
            v) DOLI_VERSION=${v};;
            n) NOM_CLIENT=${n};;
        esac
        shift
    done

elif [[ $# -lt 2 ]]; then
    echo "-----"
    echo "ERREUR - Il manque $(2-$#) paramètre(s) sur 2"
    echo "Veuillez recommencer en utilisant la commande"
    echo "sudo bash $filename --v version_de_dolibarr --n nom_du_client"
    echo "ou"
    echo "sudo bash $filename --im"
    echo "-----"
    echo "[$(date +"%Y %m %d %H:%M:%S")] END"
else
    echo "-----"
    echo "ERREUR - Inconnu"
    echo "Veuillez recommencer"
    echo "-----"
    echo "[$(date +"%Y %m %d %H:%M:%S")] END"
fi

# Création des fichiers de configuration client
if [[ ! -d /working_dir/clients/"${NOM_CLIENT}" ]]; then
    echo "[INIT] => Création du répertoire de configuration des fichiers Docker pour le client"
    mkdir -p /working_dir/clients/"${NOM_CLIENT}"
    ln -sf /working_dir/clients/000_default/dolibarr.yml /working_dir/clients/"${NOM_CLIENT}"/dolibarr.yml
    echo "[DONE] => ..."
else
    echo "Répertoire déjà existant"
fi

# Génération du mot de passe et stockage des variables dans le .env (variable d'environnement lors de la création du conteneur)
echo "[INIT] => Création du fichier des secrets"
MDP_CLIENT=$(< /dev/urandom tr -dc '[a-z][A-Z][0-9]_!@#%^&*()_+{}|:;?=' | fold -w 41 | head -n 1)
cat > /working_dir/clients/"${NOM_CLIENT}"/.env << EOF
NOM_CLIENT=${NOM_CLIENT}
MDP_CLIENT=${MDP_CLIENT}
DOLI_VERSION=${DOLI_VERSION}
DOLI_DB_HOST=${NOM_CLIENT}_mariadb
DOLI_DB_HOST_PORT=3306
DOLI_DB_USER=${NOM_CLIENT}
DOLI_DB_PASSWORD=${MDP_CLIENT}
DOLI_DB_NAME=${NOM_CLIENT}
DOLI_URL_ROOT=http://"${NOM_CLIENT}_dolibarr
MYSQL_ROOT_PASSWORD=root
MYSQL_DATABASE=${NOM_CLIENT}
DOLI_ADMIN_LOGIN=InfraS
```



```

DOLI_ADMIN_PASSWORD='${MDP_CLIENT}'
DOLI_INSTALL_AUTO=1
DOLI_CRON=1
NGINX_IDENTITY='nicolas.bidel@wanadoo.fr'
NGINX_IDENTITY_PWD='Admin&n2b34i'
PHP_INI_DIR=/usr/local/etc/php
PHP_INI_DATE_TIMEZONE="Europe/Paris"
PHP_INI_MEMORY_LIMIT=-1
EOF
    echo "[DONE] => ..."

# Création du docker_network pour le client
function create_client_network()
{
    if [ ! "$(docker network ls | grep ${NOM_CLIENT}_network)" ]; then
        echo "[INIT] => Création du docker_network pour ${NOM_CLIENT} ..."
        docker network create "${NOM_CLIENT}_network" --attachable
        echo "[DONE] => ${NOM_CLIENT}_network créé"
        echo "[INIT] => Connexion de Nginx sur ${NOM_CLIENT}_network ..."
        docker network connect ${NOM_CLIENT}_network infra_Nginx
        echo "[INIT] => Connexion de adminer sur ${NOM_CLIENT}_network ..."
        docker network connect ${NOM_CLIENT}_network infra_adminer
    else
        echo "Le docker_network ${NOM_CLIENT}_network existe déjà."
    fi
}

# Dolibarr container creation
function create_client_container()
{
    if [ ! "$(docker ps -a | grep ${NOM_CLIENT})" ]; then
        echo "[INIT] => Création de dolibarr pour le client \"${NOM_CLIENT}\" ..."
        docker compose -f /working_dir/clients/"${NOM_CLIENT}"/dolibarr.yml up -d
        echo "[DONE] => Conteneur ${NOM_CLIENT}_dolibarr et ${NOM_CLIENT}_mariadb créés"
    else
        echo "Conteneurs dolibarr et mariadb client déjà créé"
    fi
}

#Récupération du TOKEN via API NPM
function nginx_token()
{
    TOKEN=$(curl -s --request POST --url 'https://nginx.dolibarr.nicolasbidel.fr/api/tokens' --header
'Content-Type: application/json' --data '{"identity": "nicolas.bidel@wanadoo.fr",
"secret": "Admin&n2b34i"}' | jq .token -r)
}

#Insertion du Proxy-Host via API NPM
function nginx_insert_proxyhost()
{
    curl -s --request POST --url 'https://nginx.dolibarr.nicolasbidel.fr/api/nginx/proxy-hosts?=' --
header "Authorization: Bearer ${TOKEN}" --header 'Content-Type: application/json' --data
'{"domain names": ["${NOM_CLIENT}.dolibarr.nicolasbidel.fr"], "forward scheme": "http", "forward host":
"${NOM_CLIENT}_dolibarr", "forward_port": "80", "advanced_config": ""}'
}

function run()
{
    create_client_network
    create_client_container
    nginx_token
    nginx_insert_proxyhost

#Configuration du client Dolibarr // Passer en ENDPOINT pour la prochaine MAJ
    docker exec -ti ${NOM_CLIENT}_dolibarr bash -c "bash /build/configuration_dolibarr.sh"
}

run
set -e

```





## 10.12. Annexe 12 : Script configuration instance Dolibarr cliente

```
#!/bin/bash
#####
# Date : Mars/2023 #
# Version : alpha #
# Auteur : Nicolas Bidel (nicolas.bidel@wanadoo.fr) #
# Modifié le : 18/03/2023 #
# ##### #
# Commande de lancement : sudo bash creation_instance.sh ou ./creation_instance #
# Paramètres requis : --v version de dolibarr désirée (10, 11, etc...) #
# --n nom_du_client #
# OU --im (saisie à la demande) #
#####

# Variables
HTDOCS=/var/www/${NOM_CLIENT}/htdocs
WWW_USER_ID=33
WWW_GROUP_ID=33

function initDolibarr()
{
    local CURRENT_UID
    CURRENT_UID=$(id -u www-data)
    local CURRENT_GID
    CURRENT_GID=$(id -g www-data)
    usermod -u ${WWW_USER_ID} www-data
    groupmod -g ${WWW_GROUP_ID} www-data

    mkdir -p /mnt/web
    ln -s /mnt/web /var/www/${NOM_CLIENT}
    tar -C /mnt/web -xf /build/${DOLI_VERSION}.tar.gz

    echo "[INIT] => Creation du repertoire /mnt/data/..."
    mkdir -p /mnt/data/
    chown -R www-data:www-data /mnt
    chmod 755 -R /mnt

    echo "[INIT] => mise a jour de PHP Config ..."
    cat > ${PHP_INI_DIR}/conf.d/dolibarr-php.ini << EOF
    date.timezone = ${PHP_INI_DATE_TIMEZONE}
    memory_limit = ${PHP_INI_MEMORY_LIMIT}
EOF

    if [[ ! -f ${HTDOCS}/conf/conf.php ]]; then
        echo "[INIT] => mise a jour de Dolibarr Config ..."
        cat > ${HTDOCS}/conf/conf.php << EOF
<?php
\Dolibarr_main_url_root='${DOLI_URL_ROOT}';
\Dolibarr_main_document_root='${HTDOCS}';
\Dolibarr_main_url_root_alt='/custom';
\Dolibarr_main_document_root_alt='${HTDOCS}/custom';
\Dolibarr_main_data_root='/mnt/data';
\Dolibarr_main_db_host='${DOLI_DB_HOST}';
\Dolibarr_main_db_port='${DOLI_DB_HOST_PORT}';
\Dolibarr_main_db_name='${NOM_CLIENT}';
\Dolibarr_main_db_prefix='llx_';
\Dolibarr_main_db_user='${NOM_CLIENT}';
\Dolibarr_main_db_pass='${MDP_CLIENT}';
\Dolibarr_main_db_type='mysqli';
\Dolibarr_main_db_character_set='utf8mb3';
\Dolibarr_main_db_collation='utf8mb3_unicode_ci';

// Authentication settings
\Dolibarr_main_authentication='dolibarr';

// Security settings
\Dolibarr_main_prod='1';
\Dolibarr_main_force_https='0';
\Dolibarr_main_restrict_os_commands='mysqldump, mysql, pg_dump, pgrestore';
\Dolibarr_nocsrftoken='0';
\Dolibarr_main_distrib='standard';
EOF
fi
}
```



```

# Configuration du site apache par défaut
cat > /etc/apache2/sites-available/000-default.conf << EOF
<VirtualHost *:80>
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/${NOM_CLIENT}/htdocs
    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
EOF

service apache2 reload

echo "[INIT] => mise a jour du proprietaire sur Dolibarr Config ..."
chown www-data:www-data "${HTDOCS}"/conf/conf.php
chmod 444 ${HTDOCS}/conf/conf.php

# Ecriture du fichier conf Apache
cp -a /build/apache.conf /etc/apache2/sites-available/${NOM_CLIENT}.conf
sed -i -e 's;DNS;${DNS};g' -e 's;NAME;${NOM_CLIENT};g' /etc/apache2/sites-
available/${NOM_CLIENT}.conf
echo "[PASSE] ... Ecriture de la configuration du fichier '/etc/apache2/sites-
available/${NOM_CLIENT}.conf'"
chown root:root -v /etc/apache2/sites-available/${NOM_CLIENT}.conf
chmod 0644 -v /etc/apache2/sites-available/${NOM_CLIENT}.conf
echo "[PASSE] ... Application des droits requis au fichier '/etc/apache2/sites-
available/${NOM_CLIENT}.conf'"

# Configuration fichier de base apache
# sed -ri -e 's!/var/www/html!${APACHE_DOCUMENT_ROOT}!g' /etc/apache2/sites-available/*.conf
}

function testExistingDatabase()
{
    r=1
    while [[ ${r} -ne 0 ]]; do
        mysql -u ${NOM_CLIENT} --protocol tcp -p${MDP_CLIENT} -h ${DOLI_DB_HOST} -P 3306 --connect-
timeout=5 -e "status" > /dev/null 2>&1
        r=$?
        if [[ ${r} -ne 0 ]]; then
            echo "[WAIT] Attente que la bdd soit up ..."
            sleep 2
        fi
    done
}

function lockInstallation()
{
    touch /mnt/data/install.lock
    chown www-data:www-data /mnt/data/install.lock
    chmod 400 /mnt/data/install.lock
}

function initializeDatabase()
{
    for fileSQL in ${HTDOCS}/install/mysql/tables/*.sql; do
        if [[ ${fileSQL} != *.key.sql ]]; then
            echo "Importing table from $(basename "${fileSQL}") ..."
            sed -i 's/--.*//g;' "${fileSQL}" # remove all comment
            mysql -u ${NOM_CLIENT} -p${MDP_CLIENT} -h ${DOLI_DB_HOST} -P 3306 ${NOM_CLIENT} < ${fileSQL}
        fi
    done

    for fileSQL in ${HTDOCS}/install/mysql/tables/*.key.sql; do
        echo "Importing table key from $(basename ${fileSQL}) ..."
        sed -i 's/--.*//g;' "${fileSQL}"
        mysql -u ${NOM_CLIENT} -p${MDP_CLIENT} -h ${DOLI_DB_HOST} -P 3306 ${NOM_CLIENT} < ${fileSQL} >
/dev/null 2>&1
    done

    for fileSQL in ${HTDOCS}/install/mysql/functions/*.sql; do
        echo "Importing $(basename "${fileSQL}") ..."
        sed -i 's/--.*//g;' "${fileSQL}"
        mysql -u ${NOM_CLIENT} -p${MDP_CLIENT} -h ${DOLI_DB_HOST} -P 3306 ${NOM_CLIENT} < ${fileSQL} >
/dev/null 2>&1
    done
}

```



```

for fileSQL in "${HTDOCS}"/install/mysql/data/*.sql; do
    echo "Importing data from $(basename "${fileSQL}") ..."
    sed -i 's/--.*//g;' "${fileSQL}"
    mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" <
"${fileSQL}" > /dev/null 2>&1
done

echo "Create SuperAdmin account ..."
pass_crypted=$(echo -n "${DOLI_ADMIN_PASSWORD}" | md5sum | awk '{print $1}')
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "INSERT
INTO llx_user (entity, login, pass_crypted, lastname, admin, statut) VALUES (0, '${DOLI_ADMIN_LOGIN}',
'${pass_crypted}', 'SuperAdmin', 1, 1);" > /dev/null 2>&1

echo "Set some default const ..."
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "DELETE
FROM llx_const WHERE name='MAIN_VERSION_LAST_INSTALL';" > /dev/null 2>&1
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "DELETE
FROM llx_const WHERE name='MAIN_NOT_INSTALLED';" > /dev/null 2>&1
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "DELETE
FROM llx_const WHERE name='MAIN_LANG_DEFAULT';" > /dev/null 2>&1
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "INSERT
INTO llx_const(name,value,type,visible,note,entity) values('MAIN_VERSION_LAST_INSTALL',
'${DOLI_VERSION}', 'chaine', 0, 'Dolibarr version when install', 0);" > /dev/null 2>&1
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "INSERT
INTO llx_const(name,value,type,visible,note,entity) VALUES ('MAIN_LANG_DEFAULT', 'auto', 'chaine', 0,
'Default language', 1);" > /dev/null 2>&1
mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" -e "INSERT
INTO llx_const(name,value,type,visible,note,entity) VALUES ('SYSTEMTOOLS_MYSQLDUMP',
'/usr/bin/mysqldump', 'chaine', 0, '', 0);" > /dev/null 2>&1
}

function migrateDatabase()
{
    TARGET_VERSION=$(echo "${DOLI_VERSION}" | cut -d. -f1)$(echo "${DOLI_VERSION}" | cut -d. -
f2).0"
    echo "Schema update is required ..."
    echo "Dumping Database into /var/www/documents/dump.sql ..."

    mysqldump -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" >
/mnt/backups/1st.sql
    r=${?}
    if [[ ${r} -ne 0 ]]; then
        echo "Dump failed ... Aborting migration ..."
        return ${r}
    fi
    echo "Dump done ... Starting Migration ..."

    echo "" > /mnt/backups/migration_error.html
    pushd "${HTDOCS}"/install > /dev/null
    php upgrade.php "${INSTALLED_VERSION}" "${TARGET_VERSION}" >> /mnt/backups/migration_error.html 2>&1
    && \
    php upgrade2.php "${INSTALLED_VERSION}" "${TARGET_VERSION}" >> /mnt/backups/migration_error.html 2>&1
    && \
    php step5.php "${INSTALLED_VERSION}" "${TARGET_VERSION}" >> /mnt/backups/migration_error.html 2>&1
    r=${?}
    popd > /dev/null

    if [[ ${r} -ne 0 ]]; then
        echo "Migration failed ... Restoring DB ... check file /var/www/documents/migration_error.html for
more info on error ..."
        mysql -u "${NOM_CLIENT}" -p"${MDP_CLIENT}" -h "${DOLI_DB_HOST}" -P 3306 "${NOM_CLIENT}" <
/mnt/backups/1st.sql
        echo "DB Restored ..."
        return ${r}
    else
        echo "Migration successful ... Enjoy !!"
    fi

    return 0
}

function clean_container()
{
    apt-get autoremove -y
    apt-get purge -y
    rm -rf /var/lib/apt/lists/*
    rm -rf /tmp/*
}

```



```

}
function run()
{
    initDolibarr
    echo "Current Version is : ${DOLI_VERSION}"

    if [[ ${DOLI_INSTALL_AUTO} -eq 1 && ! -f /mnt/data/install.lock ]]; then
        testExistingDatabase

        mysql -u ${NOM_CLIENT} -p${MDP_CLIENT} -h ${DOLI_DB_HOST} -P ${DOLI_DB_HOST_PORT} ${NOM_CLIENT} -e
        "SELECT Q.LAST_INSTALLED_VERSION FROM (SELECT INET_ATON(CONCAT(value, REPEAT('.', 3 -
        CHAR_LENGTH(value) + CHAR_LENGTH(REPLACE(value, '.', '')))) as VERSION_ATON, value as
        LAST_INSTALLED_VERSION FROM llx_const WHERE name IN ('MAIN_VERSION_LAST_INSTALL',
        'MAIN_VERSION_LAST_UPGRADE') and entity=0) Q ORDER BY VERSION_ATON DESC LIMIT 1" >
        /tmp/lastinstall.result 2>&1
        r=$?
        if [[ ${r} -ne 0 ]]; then
            initializeDatabase
        else
            INSTALLED_VERSION=$(grep -v LAST_INSTALLED_VERSION /tmp/lastinstall.result)
            echo "Last installed Version is : ${INSTALLED_VERSION}"
            if [[ "$(echo ${INSTALLED_VERSION} | cut -d. -f1)" -lt "$(echo ${DOLI_VERSION} | cut -d. -f1)"
            ]]; then
                migrateDatabase
            else
                echo "Schema update is not required ... Enjoy !!"
            fi
        fi

        if [[ ${DOLI_VERSION} != "develop" ]]; then
            lockInstallation
        fi
        fi
        clean_container
        echo "Mot de passe dolibarr : ${MDP_CLIENT}"
        unset MDP_CLIENT, DOLI_DB_PASSWORD, DOLI_ADMIN_PASSWORD
    }

    run
    set -e

    if [ "${1#-}" != "$1" ]; then
        set -- apache2-foreground "$@"
    fi

    exec "$@"
}

```

