Rapport d’alternance M2 ACDI

Modernisation d’un système d’information

Rapport rédigé par Brice Harismendy

étudiant en Master Informatique à l’université d’Angers

Responsable pédagogique :

Benoît Da Mota

Enseignant Chercheur

LERIA Université d’angers

benoit.damota@univ-angers.fr

Tuteur en entreprise :

Denis Pithon

Responsable de l’unité système de production

Département Maine et Loire

d.pithon@maine-et-loire.fr

Table des matières

1 Introduction

Ce document vise à rapporter ce que j'ai réalisé lors de mon alternance de seconde année de Master Informatique, mention ACDI, au sein du département de Maine et Loire, et plus particulièrement auprès de l'unité système de production.

J'ai choisi de terminer mon cursus de Master par une année d'alternance me permettant d'accumuler de l'expérience en entreprise. Cette année a été très formatrice tant au plan professionnel qu'au plan scolaire (à L'UFR de sciences)

Cette alternance avait pour objectif dans un premier temps de réaliser des tableaux de bord via l'outil Grafana pour faciliter la lecture de l'état de marche du parc informatique. La première partie portera donc sur la mise en place de Grafana.

Dans un second temps j'évoquerais la mise en place de la solution de surveillance Elastic qui m'a permis de faire un comparatif de solutions et d'apporter un visuel plus pertinent sur un ensemble de serveurs web.

Ensuite, j'aborderais les solutions que j'ai apportées aux problématiques d'automatisation de la gestion des machines virtuelles, ce qui a permis d'augmenter l'efficacité de mon équipe.

Finalement, je présenterais l'étude de mise en place d'un site web en haute disponibilité dont le but est d'assurer à ce dernier une disponibilité accrue.

1.1 L'organisme d'accueil

J'ai réalisé mon alternance au sein de la Direction Logistique et Système d'information Informatique (DLSI) du département de Maine et Loire et plus particulièrement dans l'unité Système de Production. Cette direction fait partie des composantes du pôle ressources du département de Maine-et-Loire.

Le département de Maine-et-Loire, est une institution publique au service du territoire et des habitants du Maine et Loire. Les décisions qu'il prend impacte la vie quotidienne et future des habitants de ce territoire. Un conseil départemental est une instance politique composée d'élus. Depuis les dernières élections en 2014 le président du département est M. Christian GILLET.

Le département est composé de plus de 2800 agents territoriaux qu'ils soient techniciens ou administratifs. Ils assurent la mise en place des politiques choisies par les conseillers élus du département.

Il y a deux grandes directions, la première est celle du développement social et de la solidarité qui, comme son nom l'indique gère ce qui est en lien au social qui s'occupe des problématiques liées aux handicapées, personnes âgées, à l'enfance à la famille et aux insertions...

La seconde direction est celle des territoires qui regroupe les services concernant les routes, l'éducation, la jeunesse, le sport, les transports et la mobilité, le patrimoine immobilier, l'environnement et le cadre de vie, l'accompagnement des territoires, la culture et le patrimoine, les archives départementales. Certains de ces domaines, le tourisme, la culture et le sport, sont travaillés en collaboration avec d'autres collectivités (la région et les intercommunalités).

Illustration 1: Organigramme du Conseil Départemental

1.1.1 Le fonctionnement de la DLSI

La DLSI, dirigée par M. Christian LECOMTE, veille au bon fonctionnement et à la cohérence des moyens logistiques permettant, au département de Maine-et-Loire, d'assurer ses missions.

Pour ce faire la DLSI se découpe en six services, le service usages numériques et internet, le service projets et intégration logiciels, le service réseau et sécurité, le service logistique accueil, le service relation citoyens usagers, le service postes informatiques et systèmes de production. C'est dans ce dernier service que j'ai réalisé mon année d'alternance.

L'illustration n°2 présente l'organigramme de la DLSI avec, en évidence, l'unité dans laquelle j'ai passé mon alternance.

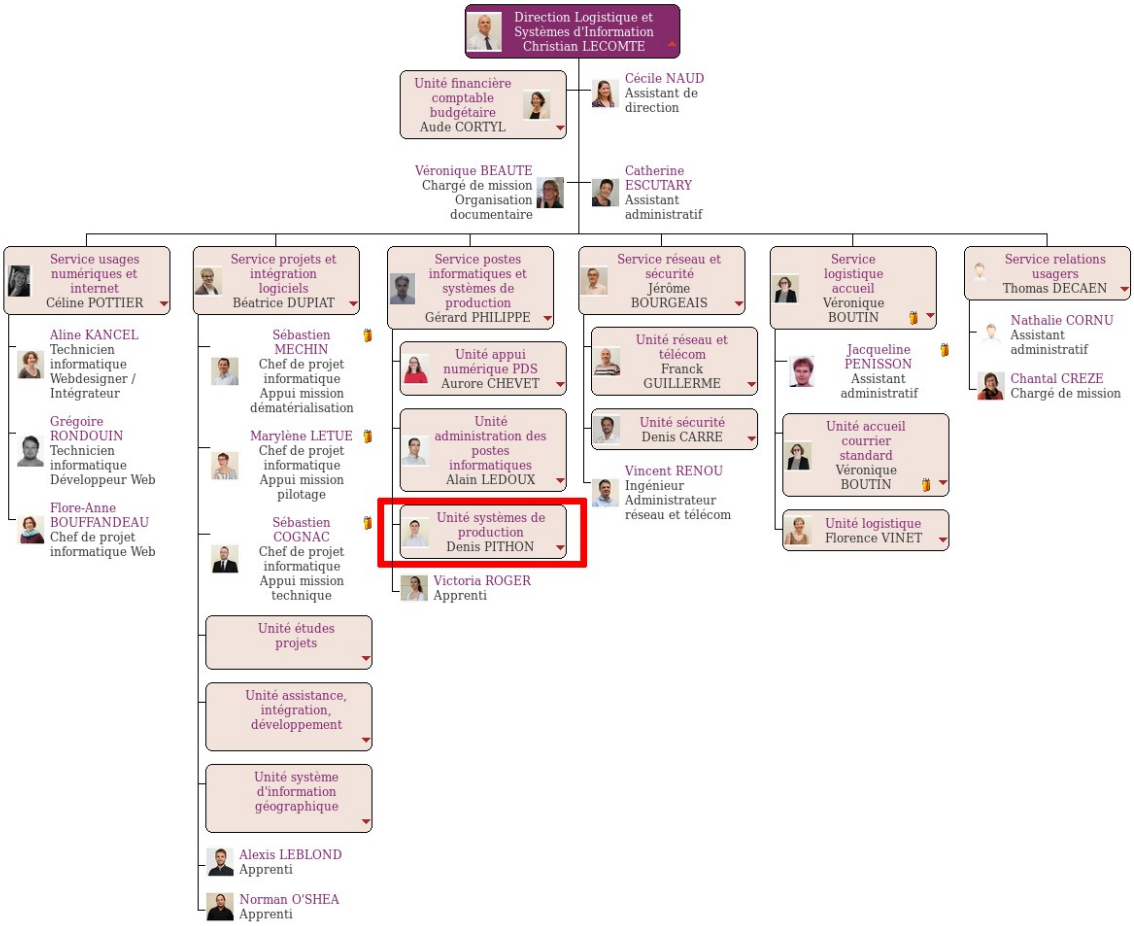


Illustration 2: Organigramme de la DLSI

On peut remarquer que chacun des pôles se découpe en services qui eux-même se découpent en unités, auxquelles sont assignées une ou plusieurs tâches spécifiques. Ainsi mon service (SPISP) a pour responsable M. Gérard PHILIPPE et est composé de trois unités distinctes l'unité administration des postes informatiques, l'unité appui numérique PDS et l'unité systèmes de production dans laquelle je travaille sous la responsabilité du responsable d'unité et tuteur de mon alternance M. Denis PITHON comme l'organigramme de l'unité le montre .

Illustration 3: Organigramme de l'unité système de production

L’unité système de production composée de quatre membres à temps plein gèrent 36 serveurs physiques 473 machines virtuelles dont 286 Linux et 139 Windows en fonctionnement sur trois sites différents, un site aux archives départementales, un site au siège du département et enfin un site boulevard Lavoisier.

Cette unité a plusieurs missions :

- La gestion des problématiques liées aux serveurs de stockage de sauvegarde et de restauration,
- La virtualisation et l’administration de systèmes Windows et Linux.
- L’administration de base de données, de la gestion des profils et des boîtes mails.
- La supervision des matériels et applications côtés serveurs.

Pour répondre dans les plus brefs délais aux problèmes des utilisateurs l’équipe a mis en place un outil de gestion des tickets. Cet outil à un double intérêt, dans un premier temps il permet de suivre la résolution de l’incident entre les différents membres de l’équipe mais il permet aussi de documenter les incidents s’ils se reproduisent.

1.2 La supervision du Système d’Information

Au cours de ce rapport je vais évoquer des technologies en rapport avec la supervision réseau. Le but de cette dernière est de remonter des informations permettant de connaître l’état de fonctionnement d’un matériel informatique, et en cas de panne acquérir des informations permettant d’en connaître l’origine éventuelle.

La supervision du SI recoupe un ensemble de procédures de surveillance, ayant pour but de répondre à 4 préoccupations principales :

- Préoccupation technique : surveillance du réseau, de l’infrastructure et des machines
- Préoccupation applicative : surveillance des application et des processus métiers
- Préoccupation de contrat service : surveillance du respect des indicateurs contractuels
- Préoccupation métier : surveillance des processus métiers de l’entreprise

La supervision permet en plus de la surveillance de déclencher des actions (script) en fonction d’alertes définies.

L’utilisation de ces technologies permet d’assurer un niveau de service important, de prévenir un certain nombre de problèmes avant qu’ils soient impactants (comme la saturation de l’espace disque…), enfin le temps d’intervention sur les pannes est grandement réduit. Tout ceci explique l’intérêt des directions informatiques pour ces techniques.

2 La construction de tableaux de bord dynamiques avec Zabbix/Grafana

Nous allons aborder le projet d'interfaçage de Zabbix avec Grafana

2.1 Introduction

J'ai poursuivi le projet d'alternance de Thomas Calatayud qui dans son projet à choisit de mettre en place la solution de remonté d'incidents nommé Zabbix.

L'objectif de mon projet est d'interfacer Grafana avec Zabbix pour permettre de réaliser des tableaux de bord permettant aux utilisateurs, non formés à l'administration système et réseaux, de comprendre l'état de leur serveur.

Grafana permet ainsi de récupérer et d'agréger des données de 52 sources de données différentes comme Zabbix, ElasticSearch, ou Graphite. Ceci, couplé avec une lisibilité très importante et une documentation claire.

Ce projet fait suite à une lettre de mission émise par le département. Le but de cette mission est donc de mettre en place Grafana comme évoqué précédemment, pour réaliser les points suivants :

1. Un usage quotidien pour la surveillance du système d'information.
2. La génération de rapports.
3. La publication en temps réel du niveau de fonctionnement de certains éléments de système d'information l'intranet.

Le temps imparti pour réaliser ce projet était équivalent à celui de l'alternance mais j'ai réussi à le finir début février me permettant de réaliser d'autres projets.

2.2 Le fonctionnement de Zabbix

Zabbix est un outil de supervision qui permet de surveiller une grande diversité d'équipements physiques ou logiques. Outre sa grande polyvalence zabbix permet différentes approches du monitoring en fonction du type de service que l'on veut surveiller.

En effet un service ne se contrôle pas de la même façon si c'est un site web ou une application Java. J'ai donc installer différentes parties de Zabbix en fonction de la machine surveillée, notamment sur les machines ayant Java à surveiller car la JVM a un comportement spécial.

Zabbix se décompose en plusieurs modules :

- Zabbix-java-gateway : outils permettant la supervision de la JVM.
- Zabbix-client : le logiciel à installer sur les postes surveillés.
- Zabbix-proxy : outils permettant d'aggréger plusieurs clients.
- Zabbix-server : le serveur principal contenant la console d'administration.

Ainsi on peut imaginer cet agencement dans un réseau lambda :

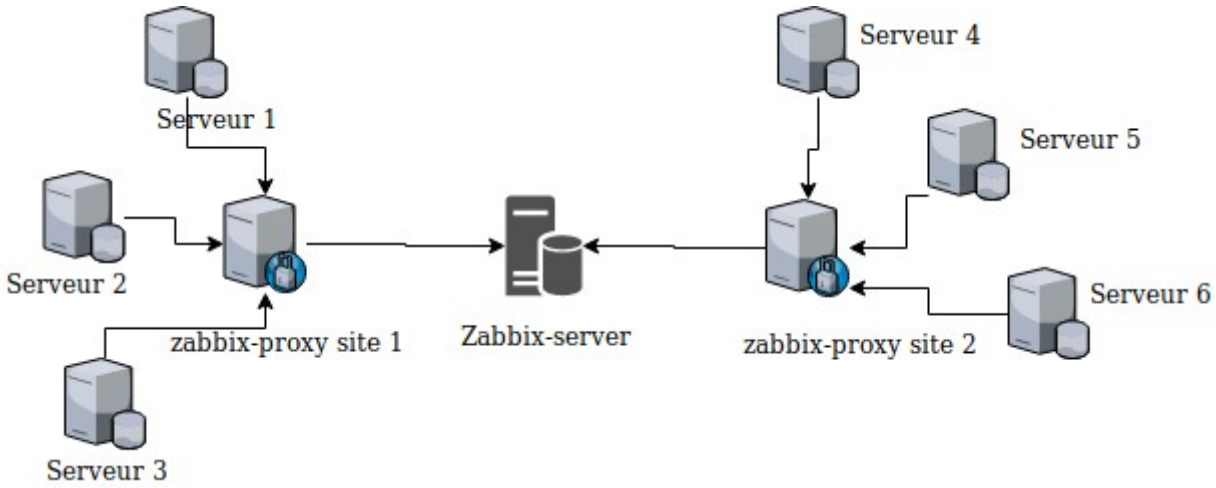


Illustration 4: Schéma d'une installation classique de Zabbix

L'illustration n°4 est un exemple assez théorique, en effet nous avons deux sites distincts (bâtiments différents …) nous pouvons voir que les sites distants agrègent leurs clients sur un proxy qui transmet les données au serveur central.

Si Serveur 1 est un serveur Tomcat par exemple, il utilise java, on y installe donc le logiciel zabbix-java-gateway et on déclare l'interface JMX sur le serveur Zabbix.

Le déploiement de Zabbix sur tout les clients du parc s'est déroulé via l'utilisation d'Ansible, (un outils de déploiement et de gestion des configurations sur plusieurs serveurs en simultané), en appliquant la même configuration à tout les clients. Lors de ce déploiement aucun client n'a été enregistré sur le serveur par un opérateur.

En effet il existe deux modes de fonctionnement pour les clients :

- Le mode Actif : le client va s'enregistrer de lui même sur le serveur, et il envoie de lui même, ses informations au serveur ou au proxy.
- Le mode Passif : le client doit être déclaré sur le serveur et ce dernier ou le proxy lui demanderont ses informations de monitoring.

Pour le stockage des données plusieurs solutions ont été envisagées. Dans un premier temps on a voulu innover par rapport aux bases de données du département et mettre en place le stockage des données dans ElasticSearch mais le support de ce dernier n'était pas officiel et était en cours de développement nous avons préféré la solution standard de base de données au département de Maine et Loire qui est PostGreSQL.

2.3 La remontée des données depuis les serveurs oVirt

Pour les serveurs de stockage de la marque NetApp, on ne pouvait pas utiliser Zabbix ces serveurs fonctionnent avec le système d'exploitation ONTAP, j'ai adopté une autre approche.

Pour récupérer les informations j'ai utilisé les solutions propriétaires de NetApp qui s'interfacent principalement avec une solution complémentaire à Zabbix nommé Graphite, on interfacera ensuite ce dernier avec Grafana.

Voici un schéma extrait du guide d'administration de NetAPP montrant comment mettre en place la remonté des données :

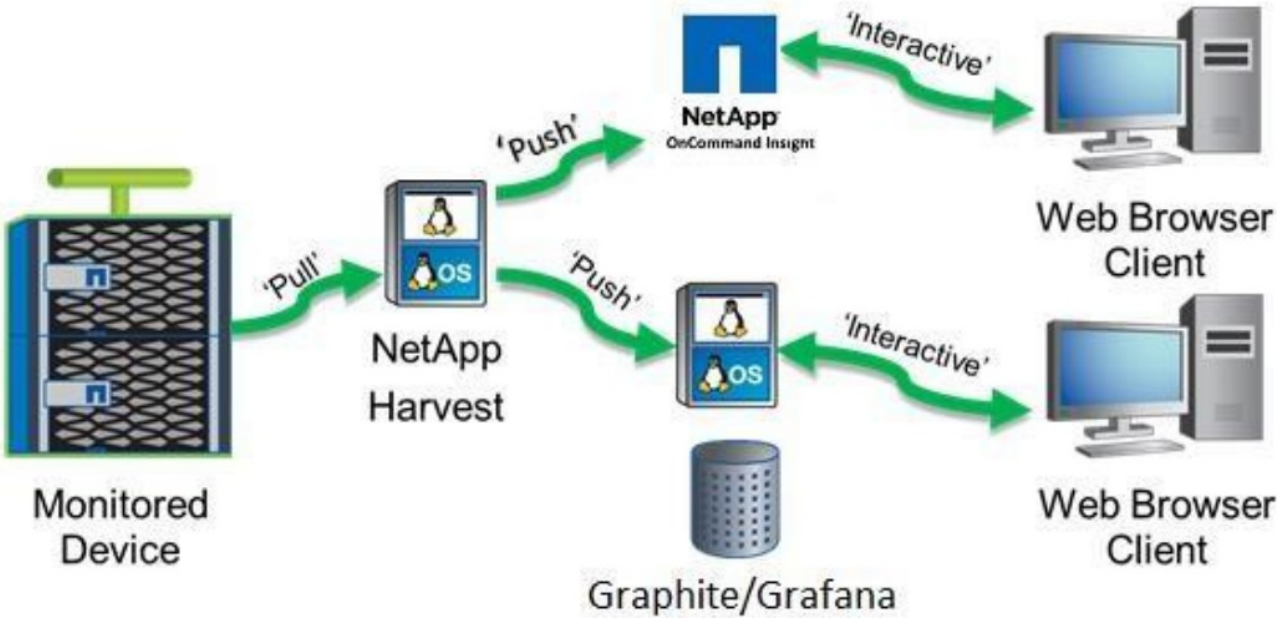


Illustration 5: Fonctionnement de NetApp Harvest

L'illustration n°5 nous montre que le logiciel NetApp Harvest, fonctionnant sous linux, va recevoir les données depuis le serveur oVirt, puis les transmets à Graphite ou Grafana.

Nous avons gardé Graphite en tampon entre NetApp Harvest et Grafana car c'est lui qui sera chargé de la rétention des données cantonnant Grafana à de l'affichage.

2.4 Le fonctionnement de Grafana

Grafana est un logiciel libre sous licence Apache 2.0 qui permet la visualisation de métriques, remontées par des logiciels de télésurveillances, sous la forme de tableaux de bord (« dashboard »).

En dehors de son but qui est de réaliser des affichages de données Grafana est un outil extrêmement complet qui permet d'interfacer de manière simple 52 sources de données sans compter les sources de données ajoutées par les plugins développés par la communauté. Pour finir Grafana permet également de faire une gestion des accès aux tableaux de bords ce qui permet de garder la confidentialité de données sensibles.

2.4.1 Les tableaux de bord

Les tableaux de bords sont des ensembles de « panels » (type de graphe) qui sont réalisables via l'interface web ou encore téléchargeable sur le site web. Il y en a 1600 de disponible.

Un tableau de bord peut avoir des graphes de différentes sources, on peut également affiner l'affichage grâce à des variables

Via l'illustration n°6 on peut voir un tableau de bord disponible sur le site de démonstration, en haut à droite on peut voir le champ « servers » qui est la variable dont je parlais précédemment, on peut ainsi choisir le serveur que l'on affiche.

Illustration 6: Exemple de tableau de bord Grafana

2.4.2 Les panels

Il existe différents types de graphes, ces derniers sont affichés dans des panels. Il y a de base 8 types de panel que je vais détailler ici, cependant certains plugins en ajoutent d'autres.

Les principales sont :

1. Le *graph* : permet d'afficher une ou plusieurs courbes dans un repère de temps, on peut personnaliser les unités affichées en abscisse et en ordonné.
2. Le *Singlestat* : affiche qu'une données à la fois sous la forme de jauge ou de compteur.
3. Le *table* : affiche les données récoltées sous forme d'un tableau de 1 à plusieurs colonnes.
4. Le panel *Heatmap* : permet d'avoir une vue des histogrammes dans le temps.
5. Le panel *Text :* ajoute des informations dans le tableau de bord on peut vouloir afficher de simple texte.
6. Le panel *Row* : est utilisé pour de la présentation des tableaux de bords, il créé des sections dans le *dashboard.*
7. *Plugins List* : sert à la gestion des plugins panels important, il liste tout les plugins installés.
8. *Dashboard List* : Il liste tous les tableaux de bord existant et permet de naviguer vers ces derniers.

Dans l'illustration n°7 nous pouvons voir un tableau de bord affichant tout les types de panels tout en haut nous avons deux *Row* puis de gauche à droite nous avons un *Table*, un *Text*, un *Singlestat* en version compteur et un *Singlestat* en version texte et fond de couleur (ici rouge car il y a un problème). Puis en bas à gauche nous avons un *Graph*, ensuite au centre c'est un *Plugin List*

en-dessous du *Text* et au-dessus d'un *Dashboard List*. Enfin en bas à droite nous avons une *Heatmap*

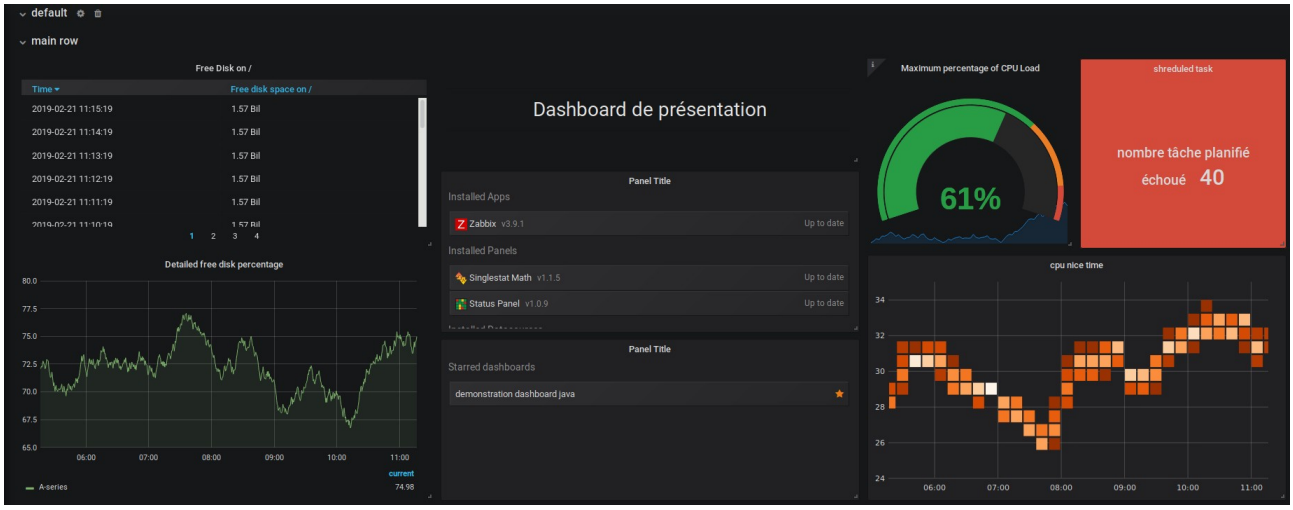


Illustration 7: Tableau de bord reprenant tous les panels de base

2.4.3 Les variables

Les variables sont un outil très intéressant pour obtenir des tableaux de bord vraiment dynamique. En effet pour reprendre un type de machine présent au département, à savoir les serveurs oVirt, une vue synthétique de l'ensemble des serveurs est intéressante mais, lors d'un problème, il peut être aussi intéressant de pouvoir visualiser uniquement la machine défectueuse.

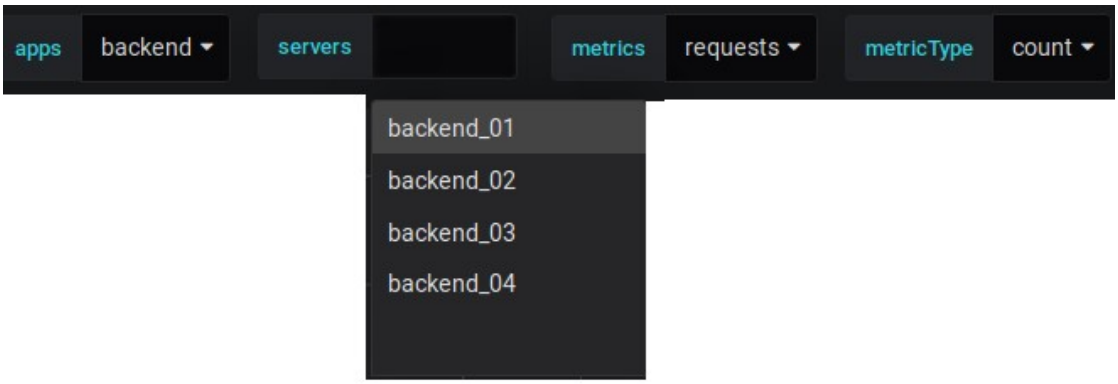


Illustration 8: Chaîne de 4 variables dynamique de tableau de bord

Dans l'illustration n°8 on peut voir que le tableau de bord a 4 variables. Ensuite il faut savoir que ces variables sont dynamiques entre-elles. Donc si j'avais choisi « frontend » dans « apps » il aurait affiché uniquement la liste des serveurs frontend dans « servers ». Pour finir on peut remarquer un champ texte à côté de « servers », c'est un champ de recherche.

2.4.4 L'interfaçage avec Graphite et Zabbix

Pour l'interfaçage entre Zabbix et Grafana j'ai utilisé un plugin développé par Alexander Zobnin, ce plugin est le seul moyen d'avoir Zabbix en source de données sur Grafana. Ensuite l'interfaçage avec Graphite étant supporté par défaut je n'ai eu aucune difficulté pour l'interfacer.

Pour sécuriser la remontée d'informations j'utilise deux techniques, tout d'abord l'isolation qui consiste à utiliser des comptes ne pouvant pas se connecter normalement via le site web et ne pouvant que lire les données. Ensuite j'ai utilisé le chiffrement via SSL(https).

2.5 Le déroulement de la mise en place

Pour réaliser ce projet j'ai été en total autonomie, j'ai commencé par réaliser une infrastructure de test, comportant :

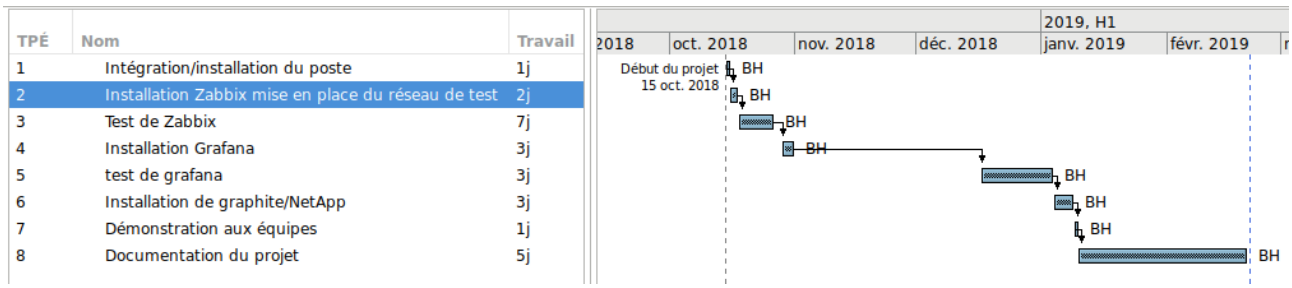
- un serveur de test contenant un zabbix-client et un zabbix-java-gateway,
- un zabbix-proxy,
- un zabbix-server,
- un Grafana
- un Graphite/NetApp Harvest.

Tout ceci m'a permit de tester la faisabilité du projet, de documenter l'installation mais également de valider les choix réalisés.

Une fois la phase de test terminée nous nous sommes réunis avec les personnels du service pour que je leur présente ce qui était possible de faire avec Grafana. Le fait d'ajouter des plugins a été évoqué et accepté qu'au strict nécessaire car il y a un trop grand risque de conflit avec Grafana ou que le développeur cesse le développement de sa solution.

Nous avons fini ce projet par le passage du serveur Grafana sur les serveurs de production et la réalisation des tableaux de bord nécessaire au service.

Pour me guider lors de la mise en place j'ai réalisé un diagramme de Gantt pour découper mes tâches et me forcer à atteindre des objectifs de résultat.



TPÉ	Nom	Travail
1	Intégration/installation du poste	1j
2	Installation Zabbix mise en place du réseau de test	2j
3	Test de Zabbix	7j
4	Installation Grafana	3j
5	test de grafana	3j
6	Installation de graphite/NetApp	3j
7	Démonstration aux équipes	1j
8	Documentation du projet	5j

Illustration 9: Gantt du projet Grafana

2.6 Le résultat

Via ce projet, le service Postes Informatiques et Systèmes de Production s'est doté d'une solution puissante de visualisation des données récupérées sur les systèmes de supervision nommés Zabbix et Graphite. Ce service peut dorénavant proposer des tableaux de bord dynamiques à des utilisateurs non techniciens comme les chefs de projet et responsables d'applications, tout en ayant

pour leurs propres besoins des tableaux de bord de supervision. J’ai donc créé les deux tableaux de bord visibles dans les illustrations 10 et 11

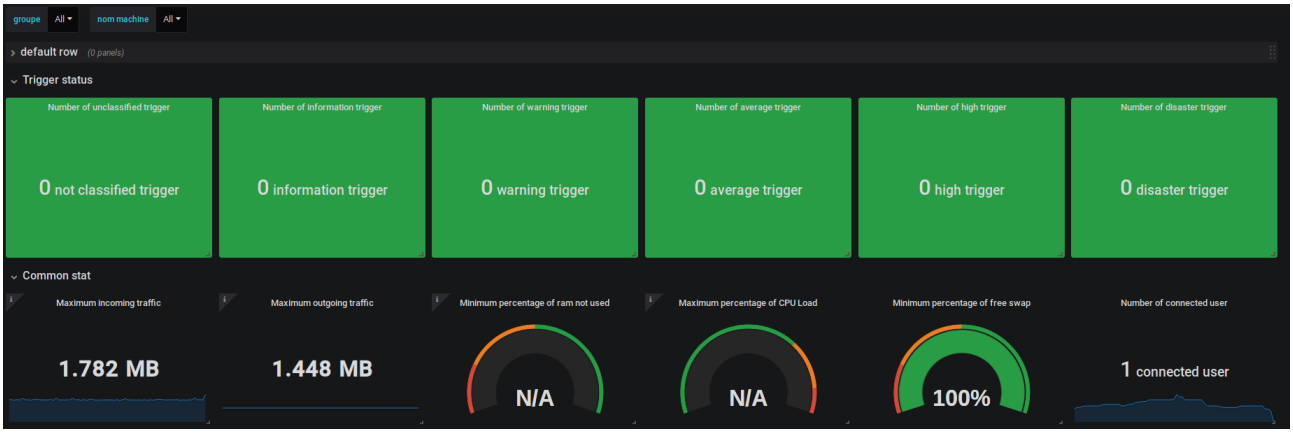


Illustration 10: Tableaux de bord pour Zabbix

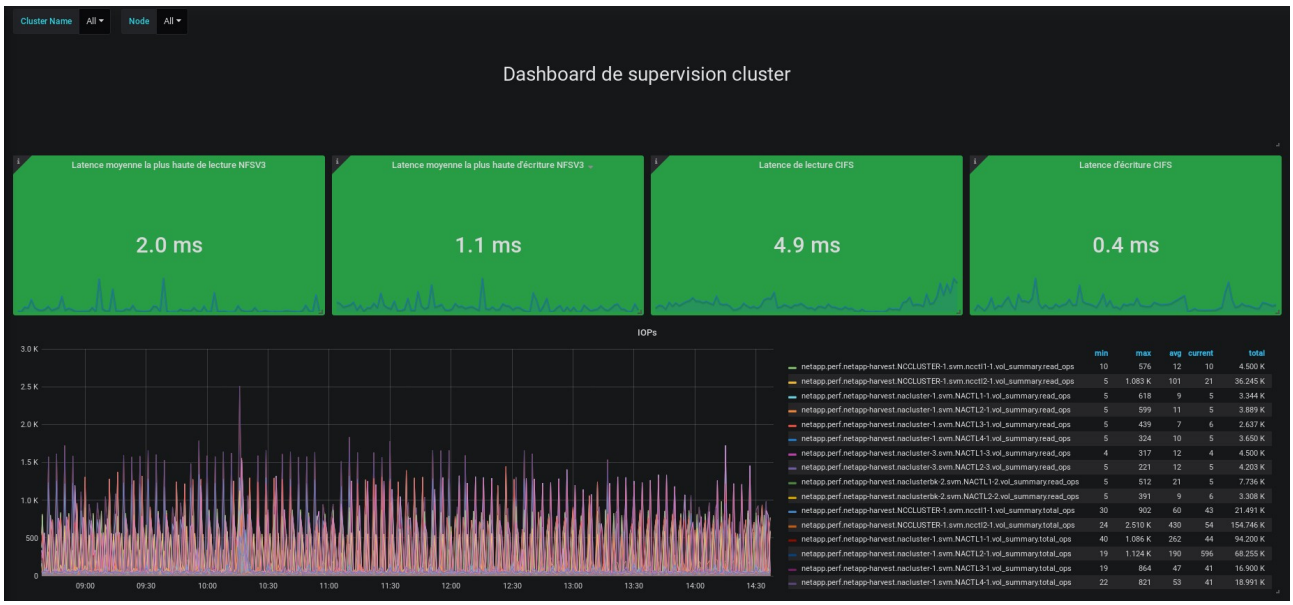


Illustration 11: Tableaux de bord pour NetApp

3 La supervision avec la pile logicielle Elastic

Pour la supervision des sites web Zabbix et Grafana n'apportaient pas assez d'éléments de base pour avoir un suivi détaillé des événements qui se déroulaient sur le serveur. C'est pour cela que j'ai mis en place la pile logicielle ELK (ElasticSearch, Logstach, Kibana).

Les serveurs web du conseil départemental produisent des logs qui sont des retranscriptions des événements se déroulant sur les serveurs. Jusqu'à présent ces logs n'étaient pas exploités en continu mais uniquement en cas de panne.

Cette suite logicielle a pour but d'exploiter ces journaux d'événements pour en connaître un peu plus sur les utilisateurs de ce serveur, comment il est utilisé mais également pour visualiser des problèmes non visibles au premier abord.

Cette solution est potentiellement utilisable pour les quinze sites web publiques du département.

Mon rôle a été de voir ce qu'il était possible de faire via cette suite logicielle et le présenter aux différents responsables d'unités concernées.

3.1 Le fonctionnement d'ElasticSearch

ElasticSearch est une base de données orientée documents (en JSON) qui a le même principe de fonctionnement que les bases de données NoSQL.

Le moteur de recherche d'ElasticSearch se nomme Lucene et est, dans son fonctionement, similaire aux moteurs de recherche Google, Yahoo, Qwant ..., car il permet de réaliser des requêtes complexes et est capable donner un poids à chacun des résultats. La capacité maximale d'un serveur Lucene est de 2 milliards de documents. Au-delà, un nettoyage est effectué pour libérer de la place.

ElasticSearch fonctionne avec un ou plusieurs nœud Lucene qui peuvent fonctionner en cluster grâce à ElasticSearch ce qui nous permet de pouvoir augmenter la capacité de stockage facilement en rajoutant des serveurs Lucene.

Ce système de base de données NoSQL de part sa grande capacité de stockage est intéressant dans la mesure où les serveurs produisent en continu une grande quantité de logs.

Illustration 12: Fonctionnement en cluster

3.2 Le fonctionnement de Logstash

Le but de *Logstash* est de faire la liaison entre les différentes sources de données et la base ElasticSearch. Pour ce faire on lui défini une interface d'entrée et une interface de sortie, ensuite on définit un ensemble de filtres entre les deux interfaces.

Avec Logstash on centralise les fichiers de journaux (*logs*), on le trouve du côté du serveur. Il permet d'extraire les informations utiles via les filtres placés entre l'entrée et la sortie. Cet outil comme la majorité d'Elastic est distribué sous licence logicielle Apache.

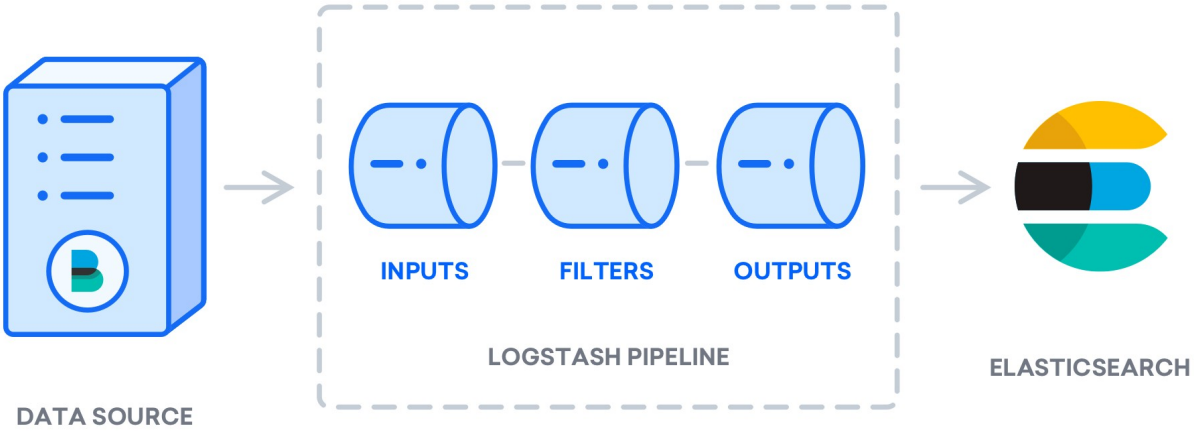


Illustration 13: Schéma de fonctionnement de Logstash

3.3 Le fonctionnement de Kibana

Kibana est un système de visualisation des données pour ElasticSearch il permet de disposer de tableaux de bord et de graphiques synthétiques et est assez intuitif. Ce greffon est publié lui aussi sous licence Apache.

Comme dans Grafana, Kibana dispose d'un ensemble de graphes assez classiques : histogrammes, graphes linéaires, camemberts, graphiques en rayons de soleil entre autres. On peut également développer nos propres visualisations via le langage Vega.

Kibana permet en plus de la visualisation des *Logs*, sur une période de temps allant de quinze minutes à cinq ans. Le logiciel embarque du machine learning non supervisé pour la détection d'anomalies.

Enfin, pour créer des visuels parlant à des non informaticiens, Kibana est doté d'un système de *Canvas* qui permet de créer des visuels avec des logos comme celui qu'on peut voir en illustration 14

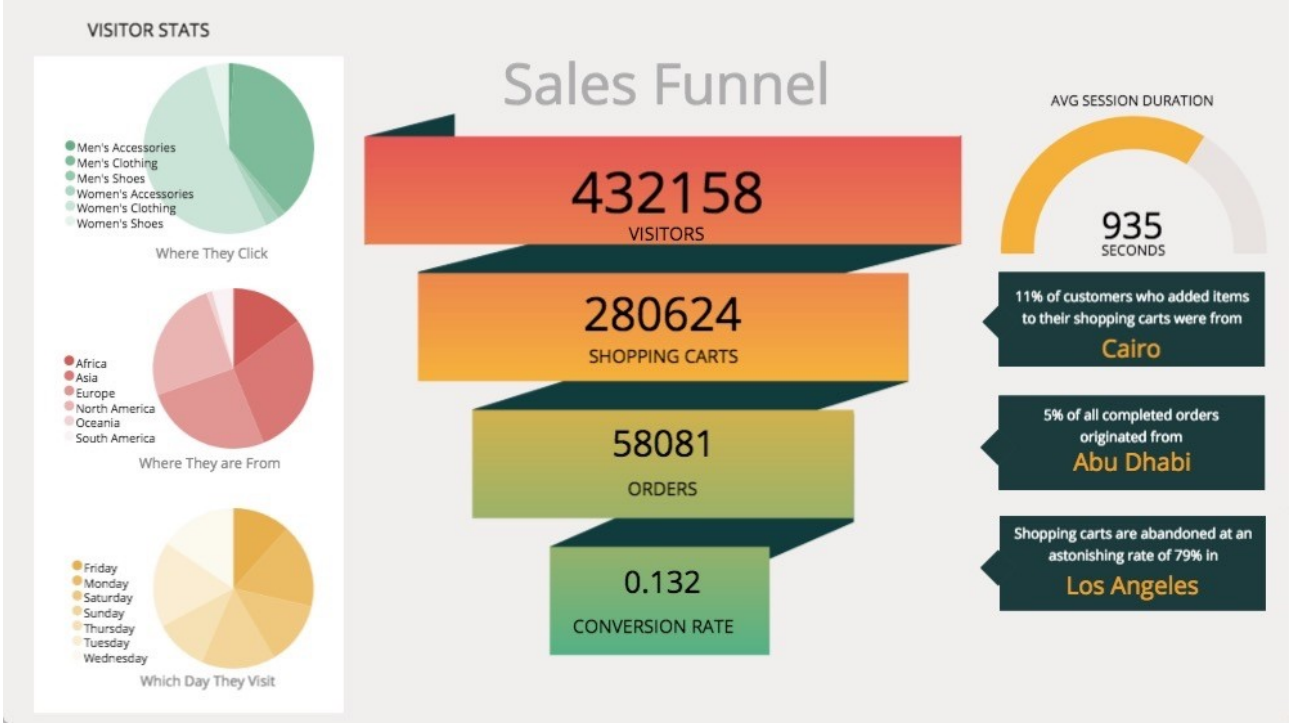


Illustration 14: Visuel exemple proposé dans la documentation

3.4 Les solutions pour remonter les données depuis les clients

La suite de logicielle Elastic dispose de trois principales solutions pour remonter les informations depuis un serveur :

1. Metricbeat remonte des données sur l'état du serveur en temps réel
2. Packetbeat remonte des analyses des protocoles utilisés sur le serveur (port,…)
3. Filebeat remonte directement les journaux d'applications.

La mise en place de ces systèmes de remontée d'événements et de la pile logicielle sont assez simples et connus.

Il existe trois autres solutions de remontées d'informations que j'ai écartées comme Winlogbeat qui remonte les fichiers de journaux de serveurs fonctionnant sous Windows. Hors tous les serveurs web concernés sont sous Linux ; Heartbeat développé pour la haute disponibilité. Finalement Auditbeat, qui lui est destiné a visualiser l'activité des utilisateurs d'un serveur en se servant d'Auditd, l'outil d'audit de linux.

3.5 La mise en place de la solution

La mise en place de la solution a été très rapide car j'ai su trouver les ressources en lignes pour pouvoir installer cette pile logicielle. J'ai réalisé un serveur qui héberge les données et la pile logicielle. Pour sécuriser l'accès j'ai mis en place un reverse proxy permettant :

- La redirection du port 80 vers le port 5601.
- La sécurisation de l'accès via le système d'authentification d'apache.

Ensuite j'ai commencé à déployer ma solution sur le serveur hébergeant wikianjou car c'est un des sites les plus consultés du conseil départemental et le plus à même d'avoir des données intéressantes. Un problème qui est très vite apparu, une fois le client configuré, c'est qu'au niveau des logs (journaux d'évènement) on avait toujours la même IP pour le client. Cela était lié au reverse proxy, j'ai donc dû changer la configuration pour régler ce problème.

3.6 Conclusion du projet

Ce projet a permis de compléter les besoins actuels en supervision système et réseau de la DLSI. Bien que rapide à mettre en place et assez documenté j'ai pu découvrir un outil polyvalent et moderne de gestion des événements sur un ensemble de serveurs et qui permet également de réaliser du Data Mining non supervisé et de la communication via des graphiques intuitifs.

L'illustration 15 est un exemple d'un ensemble de graphique

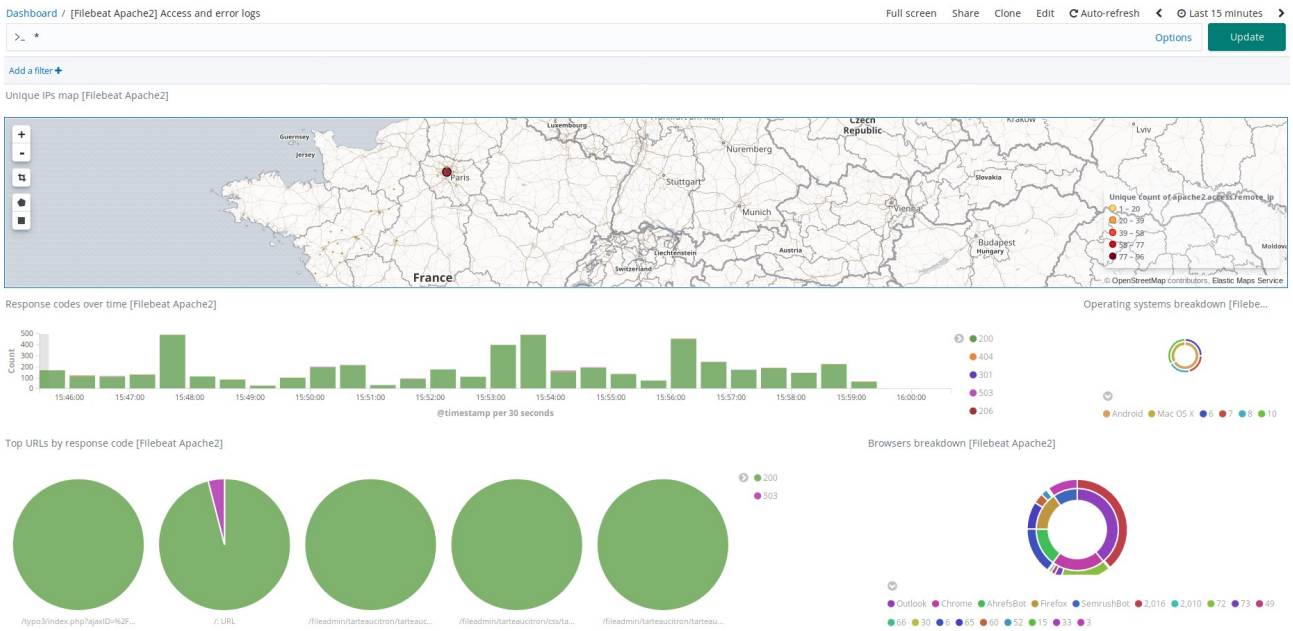


Illustration 15: Graphique concernant apache sur wikianjou

Dans cette image on peut voir en haut une carte qui nous permet de facilement visualiser la provennance les connexions au site web. Juste en-dessous à gauche nous avons un graphique affichant les codes réponses dans le temps. Ici on voit majoritairement le code 200 qui veut dire que tout s'est bien passé. A droite, nous avons un autre graphique montrant les systèmes d'exploitations mobiles utilisés pour accéder au site. Sur la dernière ligne nous avons à gauche les codes de réponse par URL ; on remarque d'ailleurs que le second remonte des erreurs 503 qui indique un service indisponible. Pour finir, sont listés les navigateurs employé et leurs versions.

Cette suite de logiciel nous a permit d'identifier les causes de problèmes récurrent sur le serveur hébergeant le site www.wiki-anjou.fr. En effet le serveur nécessitait un redémarrage de temps en temps sans que la cause soit clairement identifié car le problème semblait aléatoire. Via cette solution nous avons pus très clairement visualiser les actions réalisées par les utilisateurs sur le serveur. La cause du problème : deux robots (Ahrefbot et Semrushbot) indexaient le site web de temps en temps et créaient une surcharge temporaire sur le serveur, menait à l'indisponibilité du service.

4 Automatisation des tâches liées aux machines virtuelles

4.1 Mission à réaliser

Dans ce projet mon but est de réaliser des scripts en python 3.4 qui ont pour objectif d'interagir avec le parc de machines virtuelles du conseil départemental et ainsi d'automatiser et simplifier les tâches d'administration redondantes.

Dans ce projet j'ai eu deux scripts à réaliser :

- Le premier un script de suppression d'une machine virtuelle, ce qui comprend sa suppression dans oVirt, mais également dans le DNS, dans Zabbix et dans IpAdmin qui est le système de réservation des adresses IP du conseil départemental. Cette tâche est redondante et fastidieuse car elle demande de se connecter à trois interfaces web et de rechercher dans chacune les occurrences de la machine virtuelle.
- Le second script servira au clonage de machine virtuelle. En effet cette tâche jusqu'à lors non automatisée était extrêmement chronophage. En plus de la configuration de la nouvelle machine il fallait également faire les enregistrements DNS et IP dans différents logiciels. Le but de ce dernier est d'automatiser tout ce qui n'ai pas propre à la configuration de la machine

4.2 Reprise de projet

Un projet existant est déjà utilisé en production mais ne concerne uniquement que la création, le redémarrage, ou encore l'audit du parc (lister les différents système d'exploitation utilisé). Je suis donc reparti de ce projet qui m'a permis d'avoir une première approche du développement de script de gestion de machines virtuelles.

J'ai rencontré des problèmes avec les scripts existants car ils n'étaient pas commentés, j'ai donc commencé par entreprendre d'adapter le plus de code possible aux normes pep20 et pep8 en m'aidant d'outils comme Pycharm un IDE assez réputé en Python.

J'en ai profité également pour restructurer le projet de manière à ce qu'il soit plus lisible en cas de futures reprises.

4.3 État de l'art

Pour interagir avec les différents organes de gestion du réseau que sont le serveur DNS, L'IpAdmin, le serveur oVirt et le serveur Zabbix il faut utiliser leur API respective. Pour ce qui est du DNS, de l'IPAdmin et d'oVirt les choix on déjà été réalisés pour des scripts précédents. Pour ce qui est de Zabbix il en est autrement. Je suis le premier dans l'unité à faire un script qui interagit avec ce dernier je devais donc faire un état de l'art des différentes solution pour interagir avec l'API de Zabbix.

j'ai basé mon choix sur différents critères :

- en cherchant des exemples d'implémentation,
- en vérifiant qu'il y ai une activité récente concernant la librairie.
- une documentation assez détaillée est nécessaire également.
- la possibilité de pouvoir l'installer par Pypi est également requis pour faciliter les scripts d'installation d'environnement de développement ou pour passer en production.
- le support de python 3.
- Pour finir un plus serait qu'il y ait plusieurs personnes pour la maintenir.

J'ai ensuite mis une note à chaque API, mon barème était de 1 point par attribut binaire égale à oui auquel j'ajoute le total des attributs numériques. Sachant que je prend cette note comme indicateur et pas comme une vérité absolue et que j'écartais à posteriori les APIs ne répondant pas aux critères de base.

J'ai ainsi obtenu le tableau suivant :

Nom	Auteur	Maintenu récemment	Nombre de mainteneurs	Distribué par Pypi	Support de python 3	Documentation présente	Support Zabbix 4	***Note***
py-zabbix	*Alexey Dubkov*	Oui	1	Oui	Oui	Oui	Oui	5
pyzabbix	*Luke Cyca*	Oui	2	Oui	Oui	Oui	Oui	6
ZabbixPython API	*Frank Yao*	Non	1	Non	Non	Non	Non	1
zabbix	*gescheit*	Non	1	Oui	Oui	Non	Non	3
zabbix_API	*Grigoriy Netsman*	N/A	N/A	N/A	N/A	N/A	N/A	0
zabbix_client	*Jesús Losada*	Non	1	Oui	Oui	Non	Non	3
zabbix-API-erigones	*Erigones*	Non	1	Oui	Oui	Non	Non	3
pyZabbixSender	*Kurt Momberg*	Non	1	Non	Non	Non	Non	1
ZabbixAPI_py	*Diego Rodrigues*	Oui	1	Non	Oui	Non	Non	3

On voit ici que *pyzabbix* l'emporte j'ai donc tenté une première implémentation sur un serveur de test avec cette API ce qui fût un succès et j'ai donc décidé de rester sur cette solution.

4.4 Réalisation et mise en place

Pour le développement de mes scripts j'ai réalisé mes tests de manière méthodique en effet sachant qu'ils allaient impacter le système de virtualisation j'ai voulu en assurer la sécurité et donc éviter tous les effets de bords possibles.

Pour ce faire au niveau du DNS et de IPAdmin j'ai suivi en détail les actions côté serveur pour voir tout ce qu'il se passait sur ces derniers durant l'exécution de mon script. En ce qui concerne le serveur oVirt j'ai restreint mon droit d'accès à un datacenter de test complètement déporté du datacenter de production. En ce qui concerne Zabbix j'ai réalisé mes test sur le serveur que j'avais réalisé lors de mon premier projet, sachant qu'en plus je devais supprimé un hôte ce qui est une tâche risqué. Via ces tests j'ai pu définir les droits minimaux pour l'utilisation de l'API mais également tester sans risque d'impacter le vrai serveur Zabbix de production.

4.5 Conclusion du projet

Au final, ce projet m'a permis de gérer la reprise d'un projet n'ayant pas été développé en suivant des règles de développement normé et ayant été peu ou pas documenté. Ce projet est actuellement activement utilisé dans les tâches d'administration de l'ensemble des machines virtuelles et il permet ainsi de diminuer drastiquement le temps passé sur ces tâches assez basiques.

5 Ajout de fonctionnalités à Grafana

Lors de la mise en place des écrans dans les bureaux nous avons rencontré un problème, en effet les personnes concernées regardait irrégulièrement l'écran, il pouvait donc se passer 15 minutes avec une alerte sans que personne ne la remarque.

5.1 Les solutions envisagées

Dans un premier temps au lieu de me lancer dans la modification de Grafana j'ai choisi de faire un état de l'art des solutions possibles, il est apparu qu'une alerte sonore était possible sur certains graphiques de Grafana le problème étant que nous n'utilisions pas ces derniers.

En regardant les améliorations proposées à Grafana, l'idée d'une alerte sonore avait été évoquée mais refusée par l'équipe de développement et personne n'avait rien proposé de semblable. Ainsi j'en suis arrivé à la conclusion qu'il fallait que je développe cette fonctionnalité.

5.2 La mise en place de la solution

Au premier abord j'ai voulu ajouter la fonctionnalité en modifiant le code source de Grafana, cependant le problème de la maintenance lors des mises à jour serait beaucoup plus complexe.

Pour simplifier l'ensemble du processus de maintenance et éviter que la nouvelle fonctionnalité soit problématique dans le futur j'ai décidé de l'intégrer via un addon au navigateur Firefox, ces addons sont développés en Javascript.

Pour tester le fonctionnement du plugin j'ai utilisé mon infrastructure de test mise en place au début de mon alternance, j'ai ainsi pu validé le fonctionnement de bout en bout du plugin que j'ai ensuite déployé.

5.3 Le déploiement en production

Une fois toute les spécificité du plugin validé avec l'équipe technique je l'ai déployé sur les postes affichant les écrans. Ce plugin se montre très intéressant car il alerte toute les personnes concernées très rapidement. Nous sommes ainsi passé à une prise en compte instantanée des erreurs ayant la criticité la plus élevée.

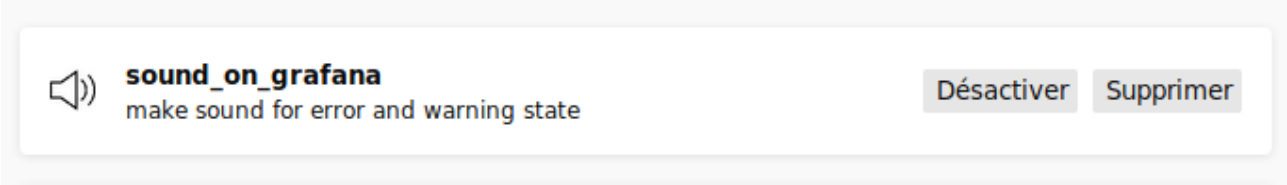


Illustration 16: L'addon installé sur le poste client

6 Système de gestion des logs collège

Lors d'une ouverture de session dans un collège une entrée est créée dans un fichier de log, ces fichiers sont archivés toutes les 24 heures dans un dossier compressé. Le but est ainsi de réduire le temps nécessaire pour connaître pour savoir quel utilisateur était connecté sur le poste informatique lors de l'apparition d'un problème.

6.1 Étude du besoin

Dans un premier temps j'ai commencé par lister les caractéristique fonctionnelles souhaitées par les clients pour pouvoir aboutir à un cahier des charges fonctionnelles. j'ai été libre de choisir les technologies utilisées pour ce projet.

Pour pouvoir rechercher des données dans l'ensemble des fichiers de log il a été décidé de réaliser un outil en la ligne de commande et une interface web plus ergonomique les technologies utilisées seront :

- Pour la commande : Développement en python3, respect des normes PEP 8 et 20
- Pour l'interface web : requête asynchrone à une API Flask, Javascript en front-end, communication des données via JSON.

6.2 Principe de fonctionnement de l'interface web

Pour valider le fonctionnement auprès de mes responsables j'ai réalisé un diagramme de séquence montrant le fonctionnement par étape de l'interface web.

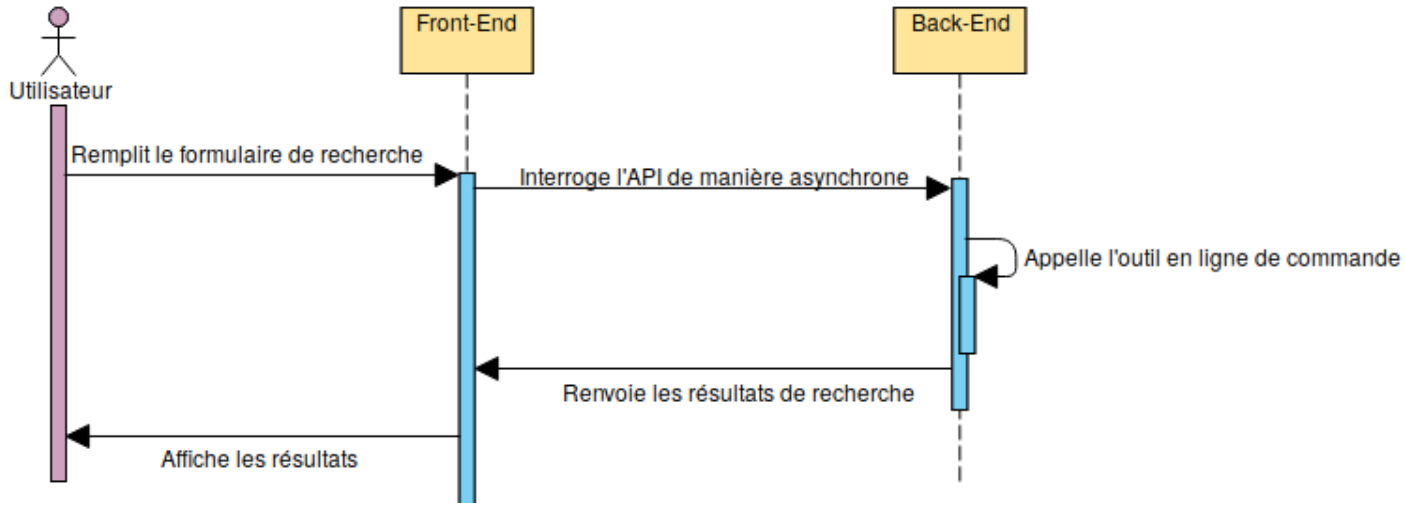


Illustration 17: Diagramme de séquence de l'interface web

Le diagramme que l'on voit dans l'illustration n°17 nous permet ainsi d'illustrer le fait que l'interface vient s'implémenter par dessus l'outil en ligne de commande, en effet le fait que ce dernier soit développé en python, comme l'API, j'ai pus importer l'outil comme une bibliothèque.

6.3 Mise en place

Une fois que le projet a correctement été défini je l'ai réalisé en commençant par l'outil de recherche en ligne de commande puis en réalisant un prototype de page web fonctionnant avec des données fictives. Finalement j'ai ajouté l'- qui permit la communication de l'interface vers le back-end.

Une fois la phase de développement terminée je l'ai déployé sur un serveur sécurisé.

Ce projet fût très enrichissant car il m'a permis d'approfondir mes connaissances en développement informatique. Dans l'illustration n°18 on peut voir l'interface web en fonctionnement et illustration n°19 on a la même recherche mais dans un terminal.

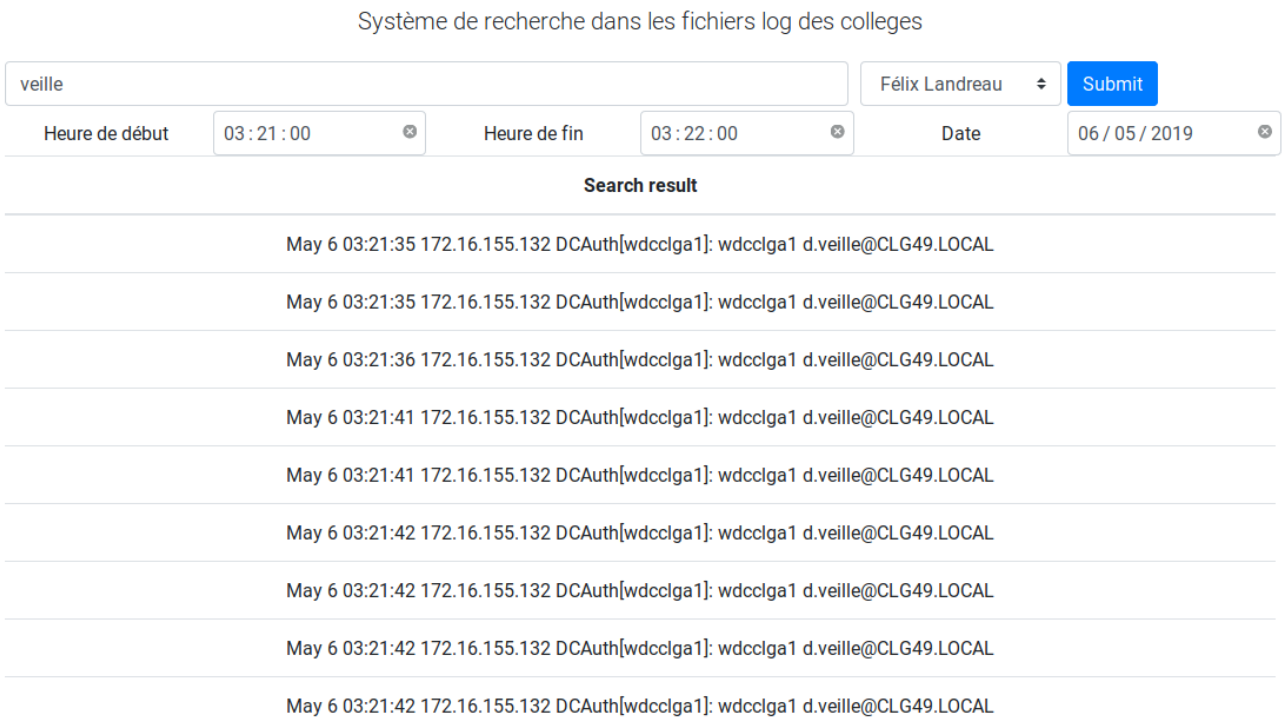


Illustration 18: exemple de recherche depuis l'interface web (résultat partiel)

```
(venv) bharismendy@brice:~/Documents/log_finder/api/bin$ ./find_line.py -c 0490953V -s veille -t 03:00:00 -T 04:00:00 -d 06/05/2019
May  6 03:21:35 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
May  6 03:21:35 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
May  6 03:21:36 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
May  6 03:21:41 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
May  6 03:21:41 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
May  6 03:21:42 172.16.155.132 DCAuth[wdcclga1]: wdcclga1 d.veille@CLG49.LOCAL
```

Illustration 19: exemple de recherche depuis un terminal (résultat partiel)

7 Étude de la mise en place de la haute disponibilité

7.1 Introduction

La haute disponibilité est un enjeux important pour des sites web utilisés par énormément d'utilisateurs. Elle permet une plus grande tolérance aux pannes et par conséquent octroie une meilleure qualité de service. La haute disponibilité est également très importante pour les sites de vente en ligne car elle permet d'éviter une coupure du service qui impliquerait un manque à gagner, c'est ce qui est arrivé à Amazon le 19 août 2013 pendant 25 à 40 minutes qui a entraîné 4,72 millions de dollars de manque à gagner.

7.2 Mesurer la haute disponibilité

La haute disponibilité est mesurable par le taux de disponibilité exprimé en pourcentage ou en nombre de neuf.

Voici un tableau récapitulatif :

Disponibilité en %	Indisponibilité par année	Indisponibilité par mois	Indisponibilité par semaine
90 % (un neuf)	36,5 jours	72 heures	16,8 heures
95 %	18,25 jours	36 heures	8,4 heures
98 %	7,30 jours	14,4 heures	3,36 heures
99 % (deux neuf)	3,65 jours	7,20 heures	1,68 heures
99,5 %	1,83 jours	3,60 heures	50,4 minutes
99,8 %	17,52 heures	86,23 minutes	20,16 minutes
99,9 % (trois neuf)	8,76 heures	43,2 minutes	10,1 minutes
99,95 %	4,38 heures	21,56 minutes	5,04 minutes
99,99 % (quatre neuf)	52,56 minutes	4,32 minutes	1,01 minutes
99,999 % (cinq neuf)	5,26 minutes	25,9 secondes	6,05 secondes
99,9999 % (six neuf)	31,5 secondes	2,59 secondes	0,605 secondes

7.3 Les techniques pour mettre en place la haute disponibilité

Il existe de nombreuses manières pour mettre en place la haute disponibilité tout dépendra des moyens mis en œuvre par la structure pour cette dernière. L'infrastructure classique est de réaliser un cluster avec une redondance des matériels, cette dernière doit également pouvoir permettre une sécurisation des données. Une fonctionnalité qui n'est pas disponible sur tout les serveurs mais qui aide la mise en place d'une haute disponibilité est la possibilité de reconfigurer le

serveur à chaud c'est à dire de changer une pièce (un disque dur par exemple) sans couper le serveur. En plus de ces dispositions matériel, différents modes de fonctionnement peuvent être mis en place (mode dégradé et mode panique) ainsi qu'un plan de secours et un système de sauvegarde poussé.

La haute disponibilité désigne donc un ensemble de techniques et de moyens mis en œuvre pour assurer l'accès à un serveur ou un service. Le problème est, que ces techniques peuvent devenir rapidements onéreuse il faut donc définir les besoins et ajuster la structure en conséquence.

Dans notre cas le département veut se limiter à une haute disponibilité logicielle avec une réplication des bases de données et des serveurs maîtres en mode maître - maître. Pour ce qui est de l'accès un haproxy fera la balance de charge avec maintien de connexion entre les deux groupes de serveur et pour chaque groupe de serveurs un Heartbeat assurera la le changement de rôle.

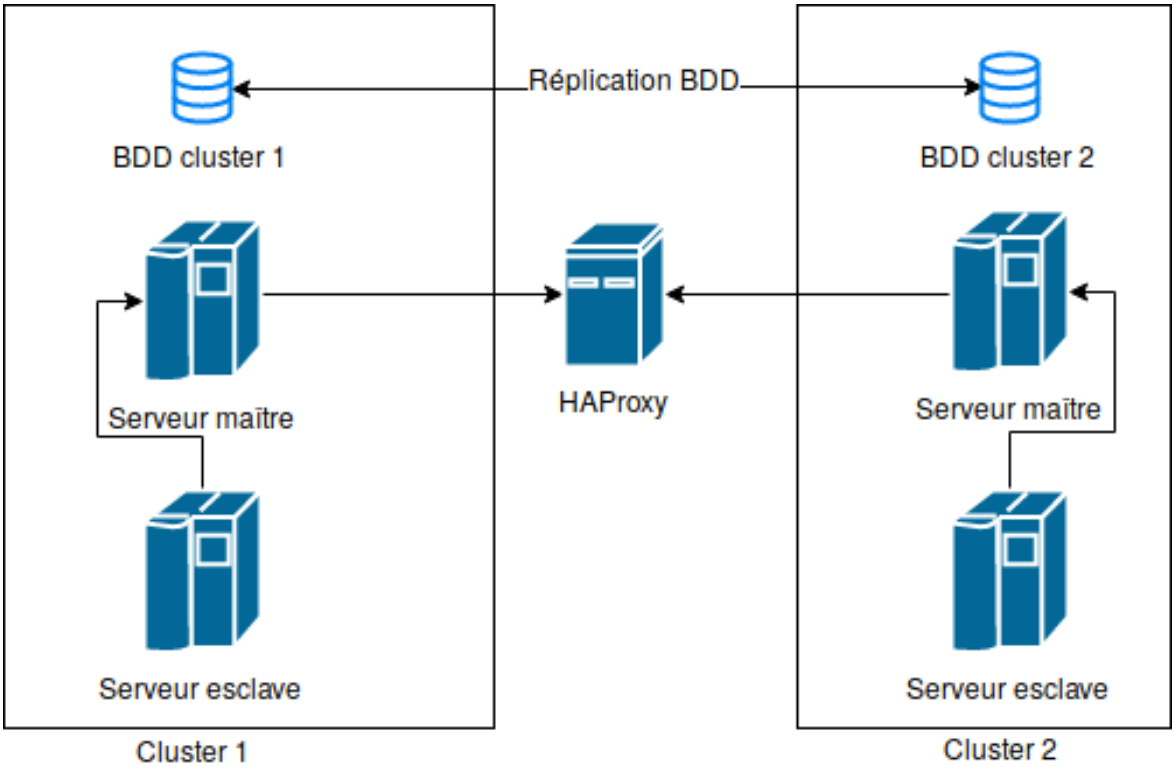


Illustration 20: schéma théorique d'une infrastructure pour un site en haute disponibilité

À noter qu'ici on utilise deux modes, maître et esclave. Maître est le mode ou le serveur est actif et répond aux utilisateurs, dans le mode esclave le serveur est passif, en cas de réplication le serveur maître est le seul qui peut mettre à jour le serveur esclave. Quand on a une réplication maître-maître cela implique que n'importe quel serveur peut mettre à jour l'ensemble des serveurs et ils répondent tous aux requêtes.

Dans l'illustration n°20 le serveur de balance de charge n'est pas en haute disponibilité car bien que central il risque peu d'avarie, cependant il est aisé de le redonder.

8 Outils d’assistance mis en place

Pour la mise en place de mes différents projets j’ai dû créer des documentations en markdown qui n’étaient pas très lisible en l’état. Par conséquent pour améliorer sa lisibilité il a été décidé de les transformer en PDF.

Pour la mise en place de cette décision j’ai choisi de créer un script en python 3.6 qui recherche tout les fichiers ayant l’extension « .md » dans une arborescence donnée et qui transforme ces fichiers en PDF.

Dans ce script j’utilise deux logiciels « markdown » et « htmldoc » pour la conversion des fichiers. Pour optimiser le code j’ai également utilisé la librairie « subprocess » et sa fonction « Popen » qui permet de lancer plusieurs processus en parallèle.

Le problème avec ce script était le support partiel des syntaxes Markdown, j’ai donc approfondi mon état de l’art et en comparant les différentes solutions j’ai choisi pandoc qui est beaucoup plus polyvalent et complet, et répondait aux différents besoins.

8.1 Suivi de l’avancement des projets

Pour me permettre de suivre l’avancement de mes projets dans le temps et ainsi communiquer sur ces derniers j’ai utilisé un diagramme de Gantt, bien que ceux-ci soit plus généralement utilisé pour la gestion d’équipe, il m’ont permis de faciliter la communication avec mes supérieurs.

Dans l’illustration n° 21 vous trouverez une version du diagramme de Gantt, que vous pourrez voir en annexe en version plus étendue

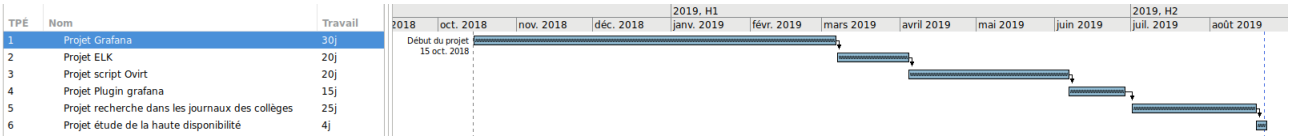


Illustration 21: Diagramme de Gantt

9 Conclusion

Cette alternance m’a permis d’enrichir mes connaissances en informatique en étant confronter à des problématiques réelles liées aux systèmes d’information. J’ai également pu accroître mes connaissances dans différents langages comme le Python ou le JavaScript, ce qui m’a permis de devenir très polyvalent et plus efficace dans la réalisation de mes missions en informatique.

Lors de cette année j’ai pu réaliser différents projets sur le thème de la gestion de parc de serveurs en informatiques ces projets m’ont demandé un panel de compétences larges allant du domaine de l’administration d’un système d’information au développement informatique en passant par la gestion de projet.

Au terme de cette expérience, l’ensemble des projets auxquels j’ai participé sont arrivés à leurs termes et ont été mis en production, pour ce qui concerne l’état de l’art sur la haute disponibilité il sera utilisé pour le prochain apprenti qui arrivera en septembre.

10 Remerciements

Je tiens dans un premier temps à remercier Christian Lecomte Directeur des services informatique, Gérard Philippe, chef du service informatique ainsi que Denis Pithon mon responsable d’alternance qui a su m’aider à progresser et à mener mes projets à bien. Je remercie également les alternants Jonathan Deramaix, Victoria Roger, et Norman Oshea ainsi que le personnel de la DLSI pour leur accueil et le partage de connaissance.

11 Webographie

Interfaçage zabbix/grafana/netapp :

- http://docs.grafana.org/
- https://www.zabbix.com/documentation/
- https://www.digitalocean.com/community/tutorials/how-to-install-apache-tomcat-8-on-centos-7
- https://geekflare.com/enable-jmx-tomcat-to-monitor-administer/
- https://community.netapp.com/t5/OnCommand-Storage-Management-Software-Articles-and-Resources/How-to-install-Graphite-and-Grafana/ta-p/109456/page/4
- http://blog.asquareb.com/blog/2014/11/19/adding-users-to-graphite/
- https://github.com/graphite-project/carbon/issues/327
- https://www.digitalocean.com/community/tutorials/how-to-install-and-configure-grafana-to-plot-beautiful-graphs-from-zabbix-on-centos-7
- http://www.thesysadminhimself.com/2013/08/configuring-web-proxy-on-centos.html
- https://computingforgeeks.com/how-to-install-and-configure-zabbix-agent-on-ubuntu-18-04/

Plugin Firefox :

- https://developer.mozilla.org/fr/docs/Mozilla/Add-ons/WebExtensions
- https://developer.mozilla.org/fr/docs/Web/JavaScript/Reference/Objets_globaux/Generator
- https://blog.mozilla.org/addons/2009/01/28/how-to-develop-a-firefox-extension/

Développement python :

- https://www.python.org/dev/peps/pep-0008/
- https://openclassrooms.com/fr/courses/4425111-perfectionnez-vous-en-python/4464292-devenez-zen-avec-la-pep-20
- https://docs.python.org/fr/3/
- https://palletsprojects.com/p/flask/

Développement Javascript :

- https://developer.mozilla.org/fr/docs/Web/JavaScript

- https://developer.mozilla.org/fr/docs/Web/API/Document_Object_Model

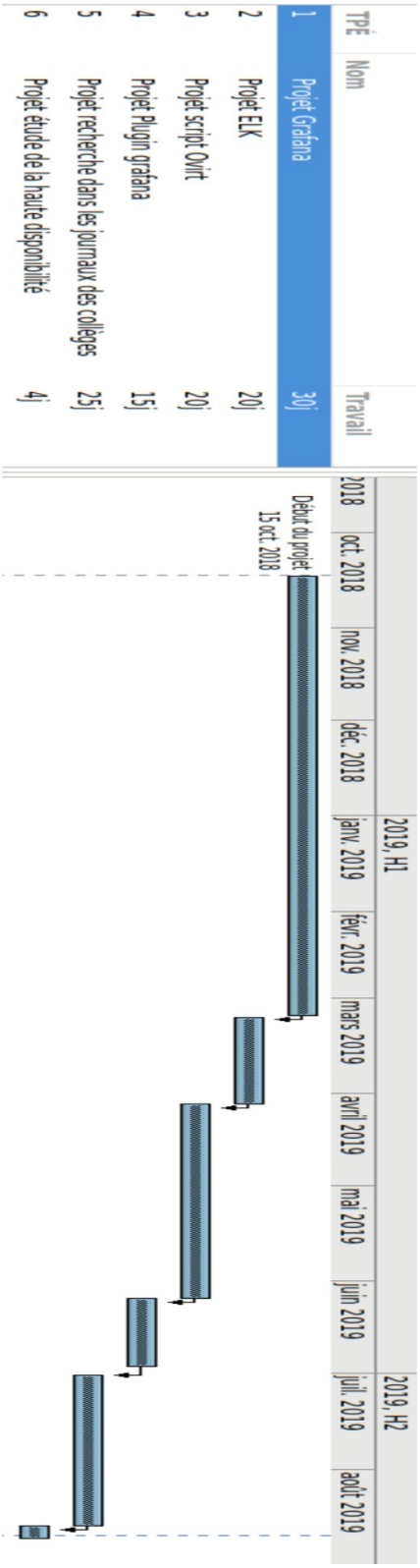
Mise en place ElasticSearch Kibana Logstash :

- https://www.elastic.co/what-is/elk-stack
- https://www.elastic.co/guide/index.html

Haute disponibilité :

- http://www.linux-ha.org/wiki/Heartbeat
- http://www.haproxy.org/
- http://www.responsive-mind.fr/replication-mysql-master-master/
- https://www.celeste.fr/fondamentaux-haute-disponibilite-entreprise

12 Annexe


TPÉ
Nom
Travail
1
Projet Grafana
30j
2
Projet ELK
20j
3
Projet script Ovirt
20j
4
Projet Plugin grafana
15j
5
Projet recherche dans les journaux des collèges
25j
6
Projet étude de la haute disponibilité
4j
2018
oct. 2018
nov. 2018
déc. 2018
janv. 2019
févr. 2019
mars 2019
avril 2019
mai 2019
juin 2019
juil. 2019
août 2019
2019, H1
2019, H2
Début du projet
15 oct. 2018

Index des illustrations