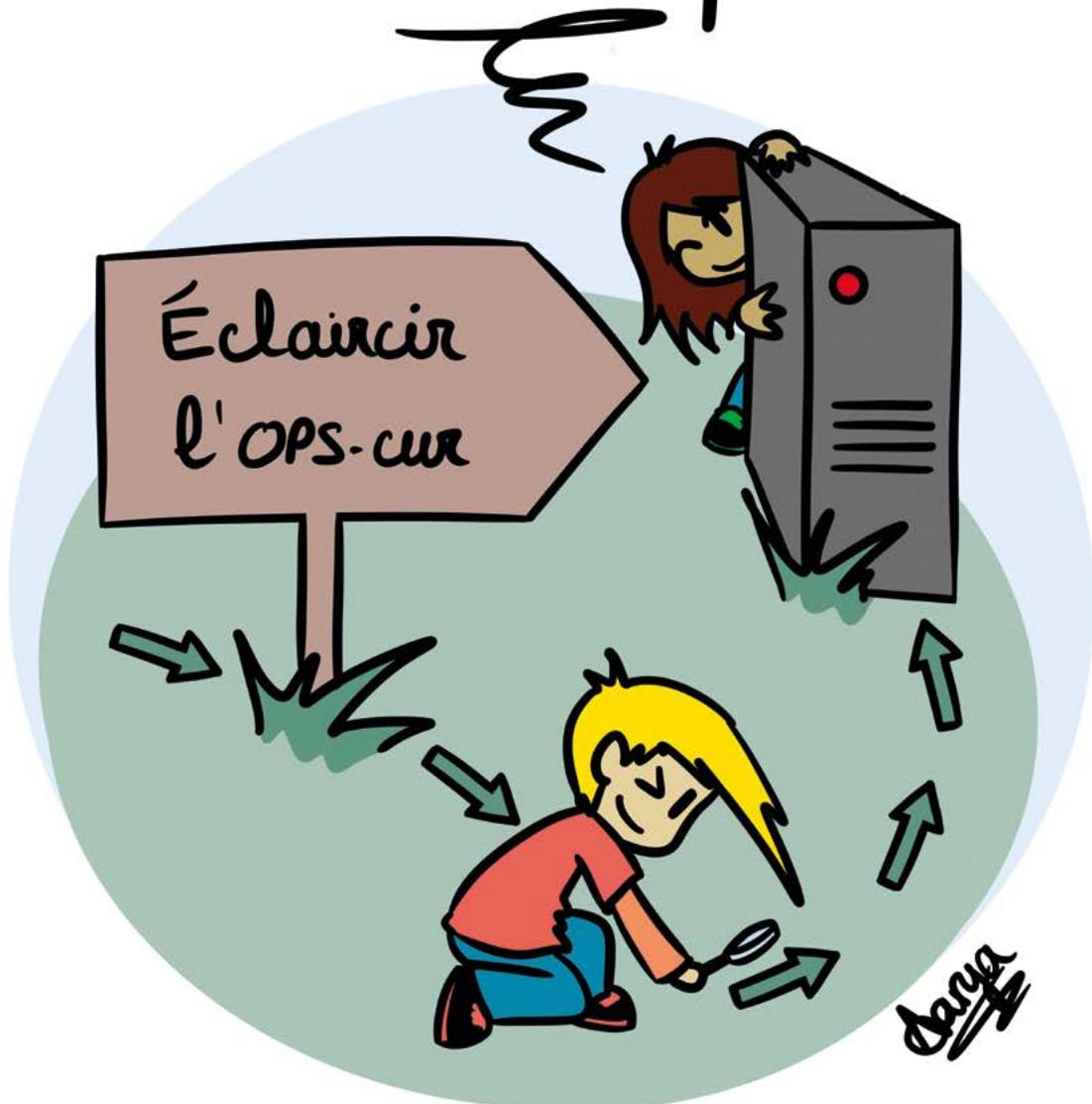




À la découverte du DevOps!



THERE IS A BETTER WAY

Projecteur sur Aryana Pezé



Derrière OCTO, il y a des centaines d'Octos, et autant de talents cachés...

À travers cette bande dessinée, la collection OCTO Presents tourne ses projecteurs sur la talentueuse Octo Aryana Pezé.

Après ses débuts dans le recrutement, Aryana trouve sa voie : devenir développeuse !

Elle intègre ainsi OCTO en tant que dev, puis rejoint la Tribu Ops.

Passionnée par le dessin, elle décide de mettre en scène le DevOps à travers ses BD.

À la découverte du DevOps

Partie 1

Je me lance dans le DevOps	6-25
Le DevOps ? C'est quoi ?	9
Quelques concepts clefs	10-11
La logique derrière mon infra	12
Les modèles Cloud	13
Cloud vs On-Premise	14
Introduction à Docker	15
Introduction au réseau : les adresses IP	16-17
L'Infrastructure as Code (IaC)	18
Pet vs Cattle	19
L'Intégration Continue (CI)	20
Le Déploiement Continu (CD)	21
Le Feature Flipping	22
Les Bases de Données	23
Les Logs	24
Le Monitoring	25

Partie 2

Pour aller plus loin	26 - 43
L'OS	29
Le Terminal	30
Le HTTPS	31
Haute Dispo à tout prix !	32
L'Authentification	33
Les Cookies	34
Une vie d'Ops !	35
Le Cache	36
Le SSH	37
Les Ports et les Protocoles	38
Le Firewall	39
L'informatique, ça creuse !	40
Le VPN	41
Le Proxy et le Reverse Proxy	42
Le Load Balancer	43

À la découverte
du DevOps!



Introduction

“Et toi? Tu fais quoi dans la vie?”

J'ai beau adorer mon métier, j'en suis arrivée à redouter la question. Comment expliquer le métier d'Ops en quelques mots à des personnes qui n'ont jamais fait d'informatique, alors que je peine déjà à l'expliquer à d'autres personnes du secteur?

Et quand je demandais à d'autres Ops...

Comment expliquez-vous notre métier?



Hmm...!
Bonne question!



On fait de la magie!



Moi je dis que je suis dev!
C'est plus simple...



Ou bien encore:

Vous me conseillez quoi pour débiter en Ops?



Hmm...
Bonne question



C'est vraiment vaste, l'Ops!



T'apprendras sur le tas!



... Un peu frustrant pour quelqu'un qui souhaite débiter dans le métier...!

Et voici comment cette BD a vu le jour!

L'objectif? Vulgariser certains sujets clés de l'informatique. Le monde de l'informatique étant vaste, il ne s'agit que d'un premier aperçu de ces sujets. A vous d'approfondir ceux qui vous intéressent!

Partie 1

Je me lance dans le DevOps 6-25

Le DevOps ? C'est quoi ?	9
Quelques concepts clefs	10-11
La logique derrière mon infra	12
Les modèles Cloud	13
Cloud vs On-Premise	14
Introduction à Docker	15
Introduction au réseau : les adresses IP	16-17
L'Infrastructure as Code (IaC)	18
Pet vs Cattle	19
L'Intégration Continue (CI)	20
Le Déploiement Continu (CD)	21
Le Feature Flipping	22
Les Bases de Données	23
Les Logs	24
Le Monitoring	25



Lorsque j'ai décidé de faire de l'Ops, je ne savais pas vraiment par où commencer. J'ai dû apprendre sur le tas, et heureusement, j'étais bien entourée!

Cette première partie vise à mettre en lumière les différents sujets que nous sommes amenés à aborder en tant qu'Ops. Chaque sujet est abordé différemment selon le contexte, mais cela donne une première idée de ce que peut être notre métier.

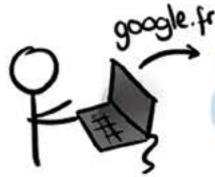
A' vous ensuite d'aller creuser les sujets qui vous intéressent!

Le DevOps ? C'est quoi ?

Salut! Je suis Aryana, et je suis Ops.



Je travaille avec les développeurs pour mettre leurs sites en ligne, et je m'assure que l'on puisse toujours accéder à notre site web.



INTERNET
(chercher l'adresse)

adresse

J'y dépose le code écrit par les développeurs



En ce moment, je bosse avec Max.

J'ai développé un nouveau site web! Tu peux me le mettre en ligne?



Paramétrer? Tu peux pas juste transférer mon code sur le serveur?



Il faut déjà définir le serveur, et ensuite il y a pas mal de configuration. Il faut gérer et installer tout ce qu'il lui faut pour exécuter le code.

Quelques jours plus tard...

Aryana, j'te cherchais! J'ai modifié mon code. Tu peux mettre le site à jour?



Encore? Bon, je te montre comment faire!

Moi?! C'est trop compliqué. Je te laisse faire!



J'pars en vacances demain. Tu vas faire comment sans moi, petit malin?

Héh! T'en fais pas j'ai tout prévu!



Tu auras juste 1 ligne de commande à taper et... TABAM!



C'est ça, être Ops! On facilite le travail des devs, pour assurer le meilleur service possible aux utilisateurs!



J'ai donc aussi configuré des outils pour gérer la performance de ton site. Regarde le nombre de connexions! On pourrait dupliquer les serveurs, installer un load-balancer, et pour la sécurité, as-tu-euh...



Quelques concepts clefs - 1

DEV



Personne en charge de la programmation d'une application ou d'un site web. Il les crée.

DEVOPS



Façon de travailler qui implique une grande collaboration entre les Devs et les Ops.

OPS



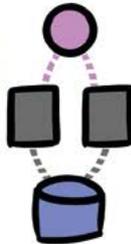
Personne qui facilite la livraison d'un site en production et s'assure de sa stabilité.

SERVEUR



Matériel informatique qui fournit des informations ou logiciels à d'autres ordinateurs reliés au même réseau (filaire, Intranet, Internet...).

INFRASTRUCTURE INFORMATIQUE



Ensemble des équipements et logiciels qui, connectés entre eux, vont former le squelette d'un système informatique. On parle aussi d'architecture informatique.

OS



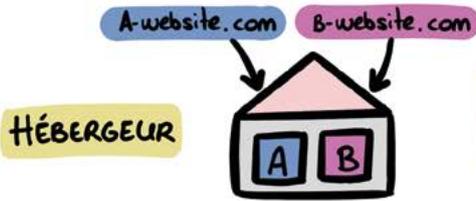
Système d'Exploitation : logiciel dans un appareil électronique qui pilote l'utilisation des ressources d'un outil informatique (mémoire, processeur, périphériques...).

TERMINAL



Point d'accès de communication entre l'homme et la machine. Il a souvent l'aspect d'un périphérique de saisie.

Quelques concepts clefs - 2



Un hébergeur met à disposition l'infrastructure nécessaire pour déployer un site web conçu par un tiers.

ENVIRONNEMENT DE DÉVELOPPEMENT

Ensemble des outils que les développeurs installent sur leur machine pour travailler (éditeur de texte, bibliothèques...).

ENVIRONNEMENT DE PRODUCTION

Environnement final sur lequel fonctionne la version finale du site, disponible aux utilisateurs.

IMAGE



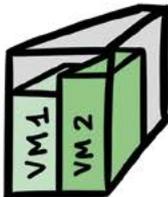
"Photo" de l'état souhaité du serveur de production: code et dépendances.

CONTENEUR

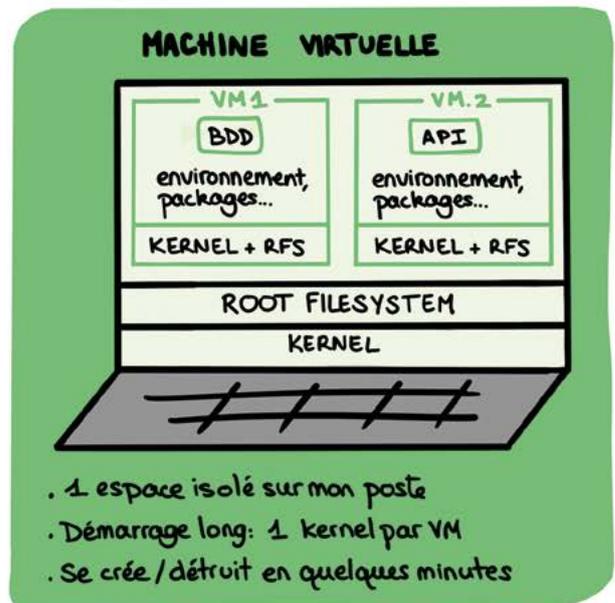
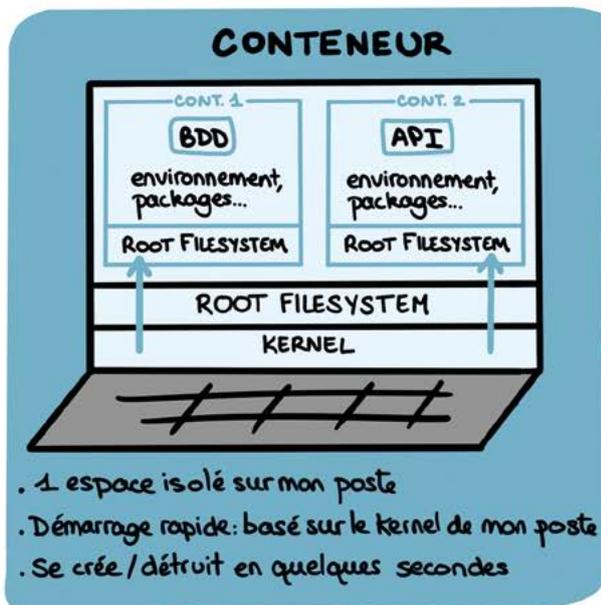


"Boîte" générée à partir d'une image. C'est un environnement isolé sur lequel tourne le code contenu dans l'image.

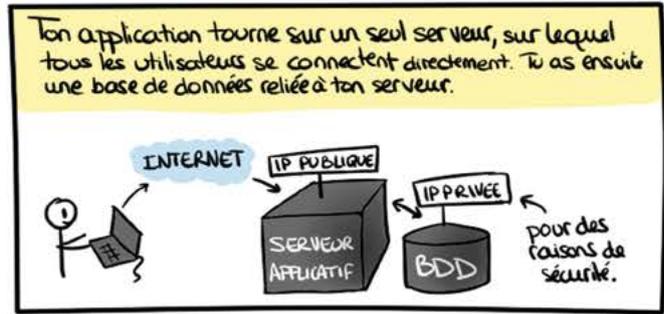
VM



Machine Virtuelle, qui simule un serveur. On peut donc avoir plusieurs VMs isolées les unes des autres sur un même serveur.



La logique derrière mon infra



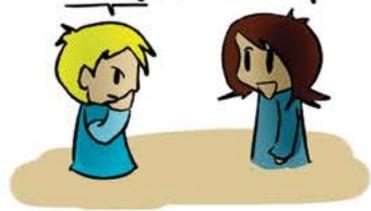
Cependant, tu dois penser à gérer la performance (que ton site soit rapide) et la scalabilité (croissance)



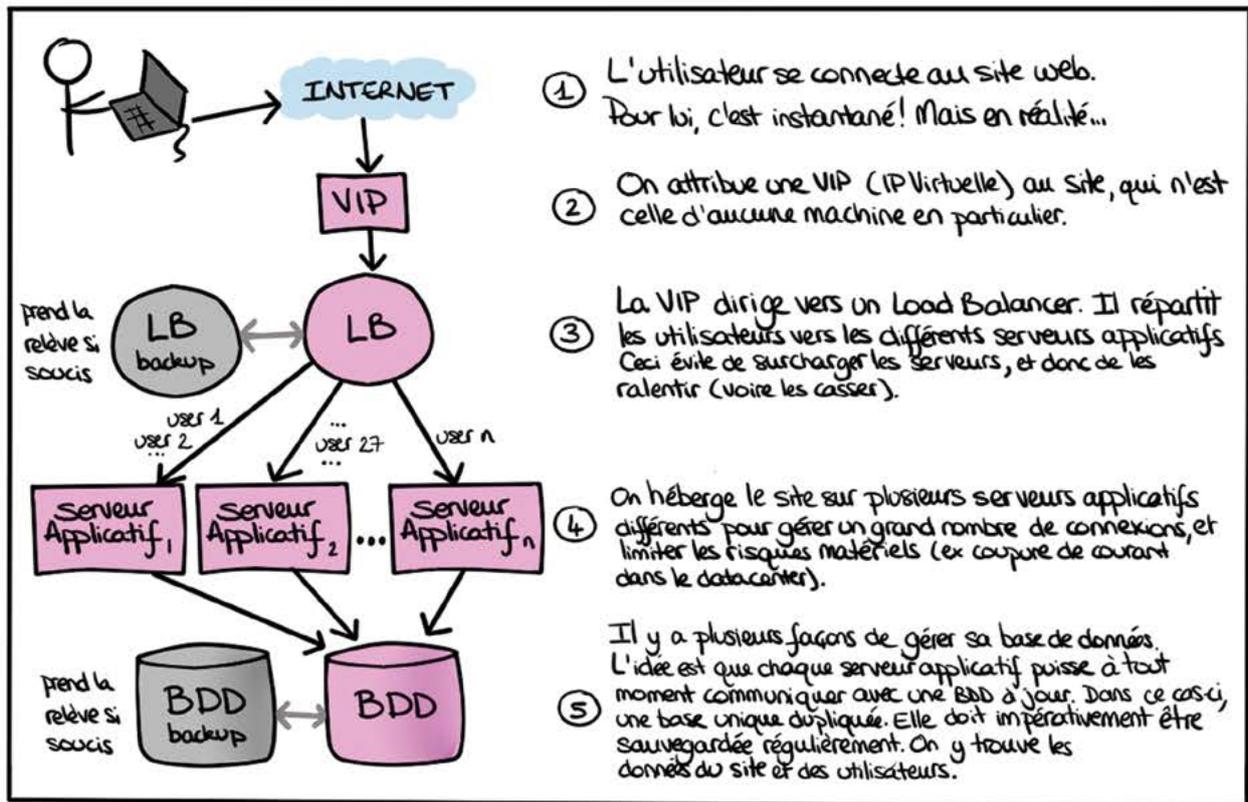
On veut de plus éviter les "SPOF" (Single Point Of Failure). Si un serveur tombe, le site doit continuer à fonctionner!



Hmm. Donc on doit tout dupliquer?



C'est presque ça!



Par contre ça peut être long et coûteux à mettre en place



Et tu vas faire ça comment?

Eh bien tout dépend du type d'hébergement souhaité.



Aucun soucis! Discutons-en!



J'ai compté sur toi. Mon site sera le plus performant du monde!

... c'est que j'ai la pression, moi, maintenant!



Les modèles Cloud

Hmm... Et pour chaque appli, on doit créer tout ça? Non

On pourrait tout monter nous-mêmes en interne, mais c'est long et coûteux.

Aujourd'hui, selon tes besoins, on va plutôt monter ton infra sur du PaaS, de l'IAAS, ou du CAAS.



c'est parti!

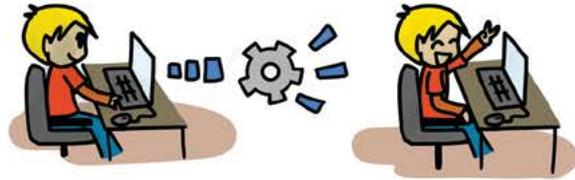
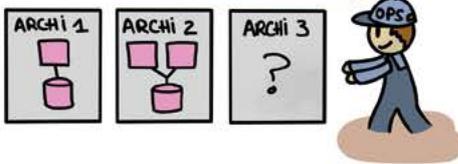
Je t'explique...

PAAS: Platform As A Service l'ami des petites applis! ☺☺☺

L'hébergeur (Heroku, OVH, Scalingo...) met des infrastructures à disposition. Ses Ops gèrent tout sauf ton serveur applicatif.

Tu choisies l'action qui te convient... et en quelques clics, tout est créé!

Tu déploies ton code quand tu veux - même pas besoin d'un Ops!



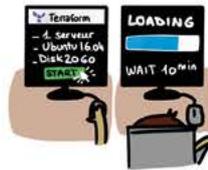
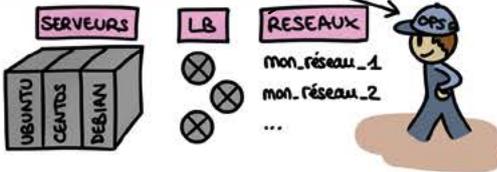
IAAS: Infrastructure As A Service mon infra sur mesure! € ☹☹

L'hébergeur (AWS, Azure, GCP) met des outils (serveurs, BDD) à disposition - que ses Ops vont gérer.

On discute pour définir tes besoins... DEV-OPS

J'automatise la création de l'infra, et le paramétrage des machines...

Je pousse mon code, et ton infra est prête!



CAAS: Containers As A Service Le service du futur? € ☹☹

Entre le PaaS et le IAAS (hébergé par AWS, Azure, GCP...). Mais au lieu de paramétrer ton serveur et d'y pousser ton code, on pousse l'image du conteneur souhaité!

On a ainsi exactement la même image sur tous nos environnements!

Je t'aide à configurer ton image initiale, ainsi que l'éventuel "orchestrateur" qui va gérer tes conteneurs, et toi tu fais évoluer l'image quand tu modifies ton code!



Donc tout dépend de la complexité de mon infra? Oui, et du budget.



Même si on paye chaque service en fonction de notre utilisation, certains sont plus onéreux que d'autres.



Ok pour le choix IAAS. PaaS. Mais pour le CAAS...?



Beaucoup d'Ops le conseillent. C'est simple et efficace, du moment que ton appli est 'stateless'. Sa popularité ne cesse de croître! Affaire à suivre!

Araya

Cloud vs On-Premise

Je veux faire du CAAS, mais mon chef veut tout garder en interne.



Le "on-premise" est parfois justifié, mais pas pour vous.

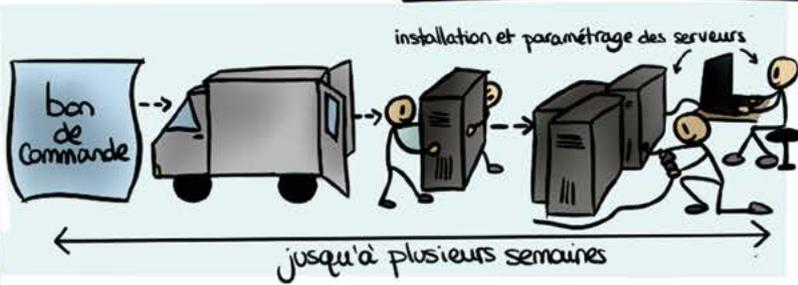


Je sais! Pourtant j'ai donné tous les arguments!

J'ai d'abord expliqué que le cloud gère la continuité de service, et la sauvegarde de données à échelle mondiale.



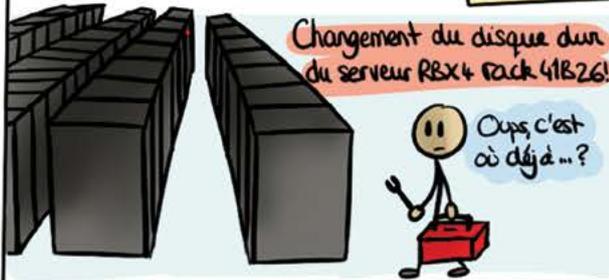
J'ai comparé les délais de livraison...



...l'allocation des ressources...



... et la maintenance



Mais en réalité il craint l'aspect mutualisation des données: le fait de partager ses serveurs avec d'autres entreprises.



Pas de soucis, tout est cloisonné. Mais si ça le gêne, il y a toujours l'option Cloud Privé:



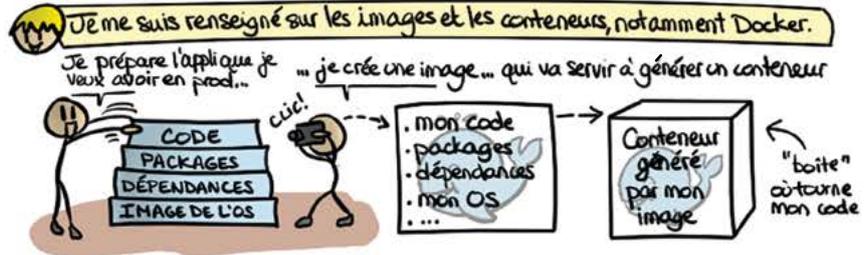
L'hébergeur réserve des serveurs pour ton entreprise. C'est plus cher, mais ça peut lui convenir...!



Intéressant... Je vais regarder ça et lui en parler! Merci!

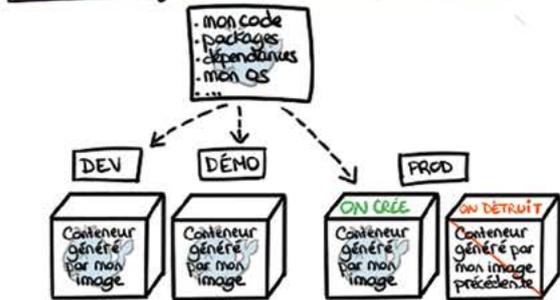
Chapeau

Introduction à Docker



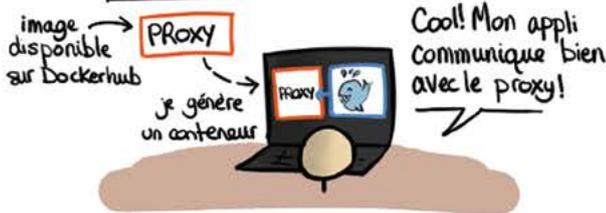
Toute l'équipe travaille à partir de la même image : facile, l'onboarding!

Et je suis sûr d'avoir exactement la même image sur tous les environnements! Seule la conf. varie.



De plus, la communauté et certaines entreprises mettent à disposition des images Docker utiles.

Ces ressources en ligne permettent de faire des tests rapidement.



En plus un conteneur se crée en un temps record!



Si tu as beaucoup de services ou conteneurs, il existe des orchestrateurs... Mais ce n'est pas ton cas!
Ok! D'autres conseils?



QUELQUES BONNES PRATIQUES!

Privilégie les images officielles, ou celles ayant beaucoup de téléchargements ou d'★! (attention aux malwares!) 

- * Le multistage build permet de déployer uniquement le nécessaire. Ça allège l'image et limite les failles de sécurité. (ex: pas de compilateur en prod!)
- * Respecte bien le 12FactorApp pour une appli facilement dockerisable.

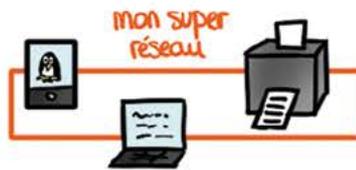
Introduction au réseau : les adresses IP - 1

yo! J'aimerais avoir une petite intro sur les IP et le réseau.

Ah, top! Commençons par les adresses IP.



Tout matériel connecté à un réseau a une carte réseau et est identifié par une adresse IP.



L'IPv4 (Internet Protocol Version 4) est la plus déployée.

format = $\overset{\text{1 octet}}{\text{xxx.xxx.xxx.xxx}}$
4 nombres de 0 à 255

Pour ce qui est du réseau, tu connais déjà le plus grand...!



Les fournisseurs d'accès à Internet allouent des adresses IP à chaque appareil connecté, via le protocole DHCP.



Si c'est une "adresse", ça signifie que je peux la contacter?

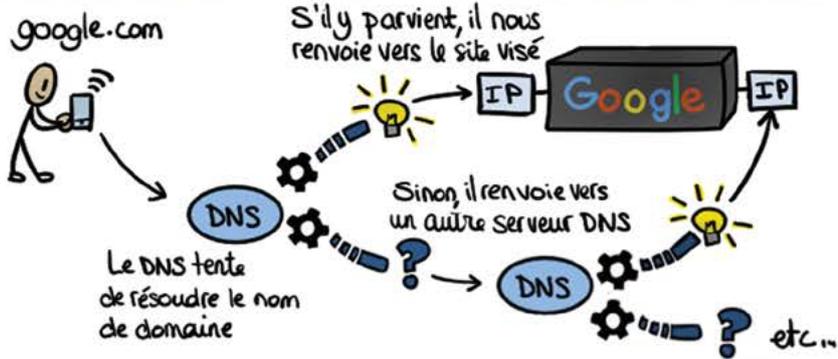
C'est ça! Pareil qu'une adresse postale!



Pour les sites web, pour plus de simplicité, on associe un nom de domaine à une adresse IP. Le service DNS (Domain Name System) est chargé de la traduction.

Mais c'est difficile à retenir...

Oui, d'où l'utilisation de noms de domaine!



S'il ne parvient pas à le résoudre (nom de domaine erroné, serveur indisponible...) tu auras un message d'erreur.

Et pour mon site? J'ai 2 serveurs en parallèle!

Aucun souci! Il suffit d'associer le nom de domaine au load-balancer.*



* Composant informatique, ici utilisé pour répartir le trafic entre les 2 serveurs.

Il gère la redirection vers tes serveurs.

Ah! Mais si on connaît l'IP de mes serveurs, on peut contourner mon load-balancer?

On pourrait, si tes serveurs avaient une IP publique - mais je leur ai donné des IP privées! Laisse-moi t'expliquer...



Araya

Introduction au réseau : les adresses IP-2

Il existe donc 4 milliards d'IPv4 sur Internet, de 0.0.0.0 à 255.255.255.255.

Sachant que j'ai moi-même plusieurs appareils, ça fait peu...

Sauf que tu n'as pas besoin de tout connecter à Internet.

Comme mon imprimante... Mais je veux la contacter depuis mon ordinateur.

On peut leur allouer des IP qui n'existent pas sur Internet. Trois plages d'adresses sont donc réservées pour les réseaux privés.

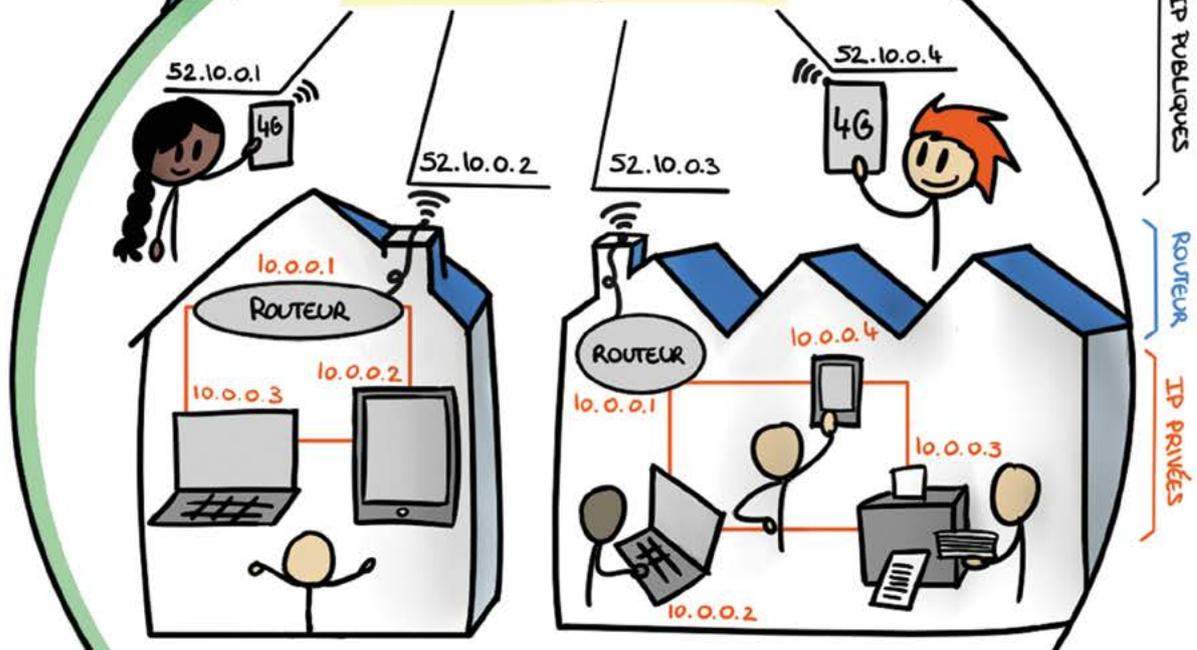
10.0.0.0 à 10.255.255.255
 172.16.0.0 à 172.31.255.255
 192.168.0.0 à 192.168.255.255



RÉSEAU MONDIAL INTERNET

Pour faire simple, on a donc une configuration qui ressemble à ça

Les IP publiques sont celles exposées sur Internet. Les IP privées sont uniquement accessibles sur un réseau privé.



On gagne ainsi beaucoup d'adresses IP, mais ce n'est pas suffisant, d'où l'apparition de l'IPv6.

16 octets
 format : X:X:X:X:X:X
 valeurs hexadécimales

Plus de 3×10^{38} adresses!

Un autre moyen d'économiser des IP est de passer par des IP dynamiques, plutôt que fixes.



Je vois. Et je peux la connaître?

Rien de plus facile! Il y a des tas de moyens!

TROUVER L'IP D'UN SITE :

- * dig mon-site.fr
- * ping mon-site.fr
- * etc...

TROUVER MON IP:

- * ipconfig (IP interne au réseau)

→ Plein de sites web permettent aussi de trouver ces informations!

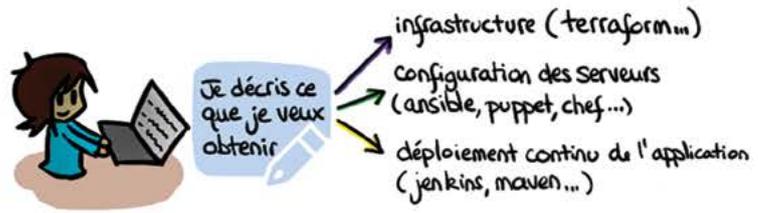


L'Infrastructure as Code (IaC)

Aaaah! Il me faut un nouvel environnement de dev! Je sais que c'est compliqué mais-

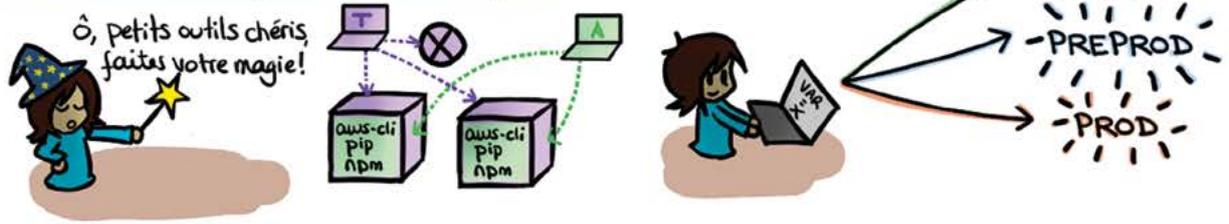


Je fais de l'Infrastructure as Code. C'est du code descriptif.



Lorsque je joue mon code, il crée ou modifie si besoin l'environnement cible pour qu'il corresponde à ma demande. On dit que le code est "idempotent".

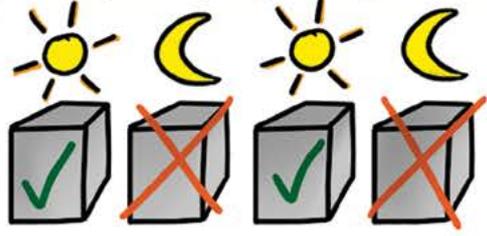
Il me suffit de changer certaines variables pour déployer mon code sur différents environnements.



De plus, si je pars, mon successeur sait exactement ce à quoi doivent ressembler les environnements. Comme en dev, la doc, c'est le code!

Idéalement, on fait aussi des nightly build: on détruit et on remonte l'infra toutes les nuits pour vérifier que tout fonctionne.

TU CASSES TOUT!? En dev, pas en prod!



Pour résumer:



...et je reconstruis tout quotidiennement pour détecter les bugs!



Ça me rassure! Et moi qui pensais que les Ops ne testaient rien...!



George

Pet vs Cattle

L'Infrastructure as Code (IaC) nous permet de facilement et rapidement détruire et remonter les machines. Cela a donc grandement impacté la façon de travailler des Ops! Plutôt que de travailler sur des serveurs spécifiques qu'on maintient et qu'on bichonne des mois durant, on préfère détruire et remonter les machines régulièrement. On parle de "Pet vs Cattle".

J'ai bien réfléchi. Les nightly builds, monter ou détruire rapidement un environnement... c'est grâce à l'infra as code!



L'IaC a donc changé la façon de travailler?



Yes. Les machines étaient rarement éteintes ou mises à jour. On corrigeait les bugs au cas par cas.

Hmm... Je vais te concocter un petit script!



Les 2 approches sont différentes. On parle de "Pet vs Cattle".



Avant: on choyait ses serveurs. Un serveur cassé était dur à remonter!



PET

VS

Maintenant; si ça plante, on casse tout et on recrée un clone tout propre!



CATTLE

"C'est violent..."

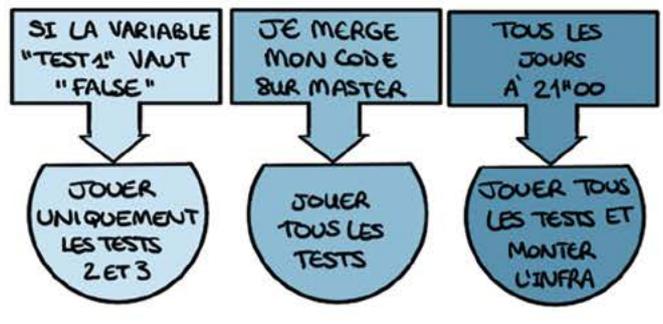
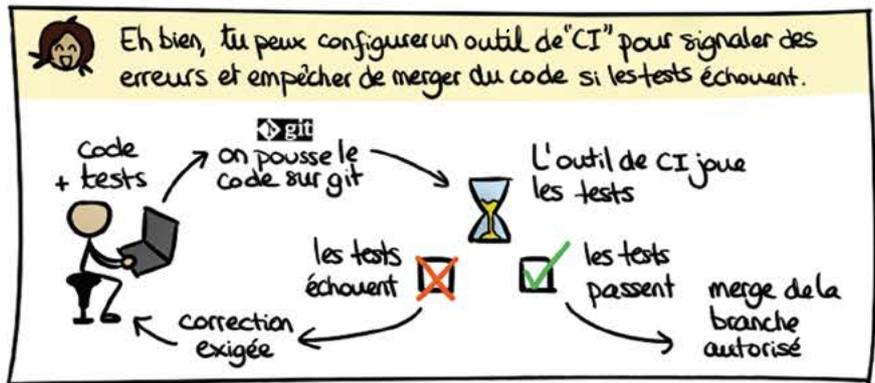
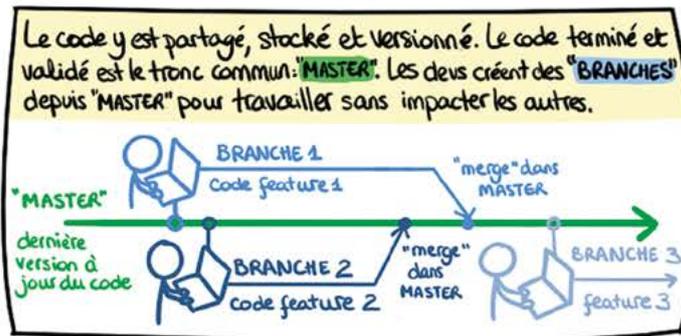


Mais c'est plus rapide, et efficace!



Abrya

L'Intégration Continue (CI)



Le Déploiement Continu

Tu me parlais de "CD" (Déploiement Continu)?



Dans l'idéal, un bug est détecté très tôt:



Mais en pratique, certains bugs sont bien cachés!



D'où l'intérêt de déployer en amont sur d'autres environnements.



C'est ce qu'on a fait! Mais on n'a décelé qu'une partie des bugs.



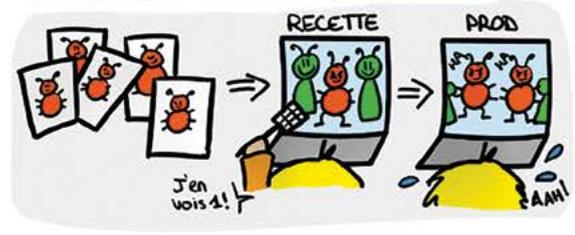
Quand avez-vous déployé en prod pour la dernière fois?



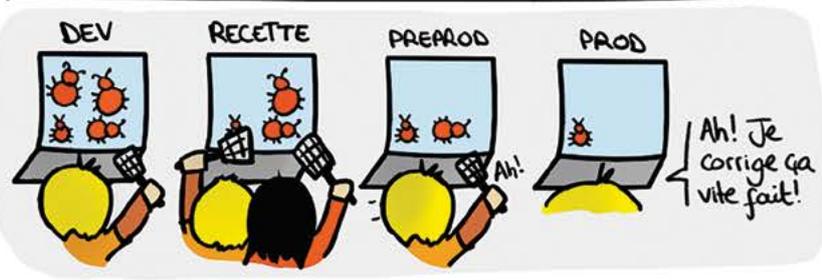
Donc 2 mois de code déployés en prod d'un seul coup?



Des centaines de lignes de code mises en prod en une seule fois...



Plus tu déploies souvent, plus les éventuels bugs déployés sont simples à déceler et à corriger. Et dans l'idéal la CD c'est déployer automatiquement à chaque modification du code!



Faites-le petit à petit: déployez déjà plus souvent. Puis automatisez progressivement le déploiement sur les divers environnements.



Une vraie CD ne s'obtient que dans 1 équipe bien rodée!



Mais déployer souvent signifie déployer des fonctionnalités incomplètes.



Eh bien, désactive-les!



Le Feature Flipping

Tu penses quoi du feature-flipping?



C'est GENIAL!

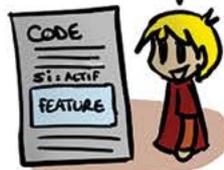
L'idée est de pouvoir à tout moment activer ou désactiver une fonctionnalité en production.



Je précise simplement à quelle condition l'activer.



Ceci implique cependant que le morceau de code est isolé du reste - ce qui est d'ailleurs une bonne pratique!



Si le site a un bug, on sait exactement quel fichier corriger*



* ⚠ L'abus de "if" peut avoir l'effet inverse!

Je vois. Et peut-on choisir sur quels serveurs activer la feature...?



Yes! Tu vois toutes les possibilités?

On peut faire du canary release! On active la fonctionnalité pour un nombre restreint d'utilisateurs.



Ok! On l'active 2 heures en Espagne!

Ainsi, si ça marche pour eux...



* Fantastique!

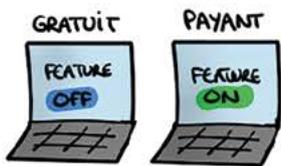
C'est un succès! On déploie partout!



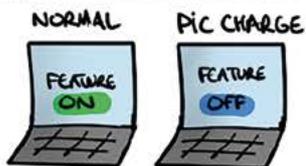
On peut aussi faire du feature-flipping pour une feature terminée.



Pour customiser des offres ...



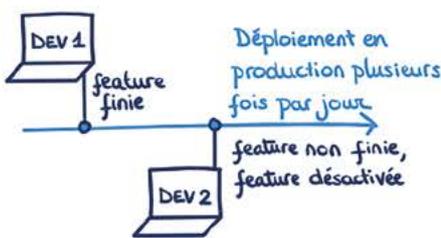
... ou gérer des pics de charge en désactivant des features non prioritaires.



Après, l'intérêt principal est que ça facilite le travail des devs. On ose davantage découper les features en petits morceaux rapides à développer...



... et ils peuvent sans crainte faire du vrai Déploiement Continu!



Je vois. Les développeurs mergent leur code très souvent, et ont toujours une version du code à jour!



Donc faites du feature-flipping! Et pour bien débiter:

- Les tests doivent couvrir la feature activée ET désactivée
- Bien supprimer les conditions si la feature n'est plus "flippée"
- Attention au flipping de features "cassantes" (ex schémas de bases de données)

Acary

Les Bases de Données

C'est indispensable un Système de Gestion de Base de Données (SGBD)?

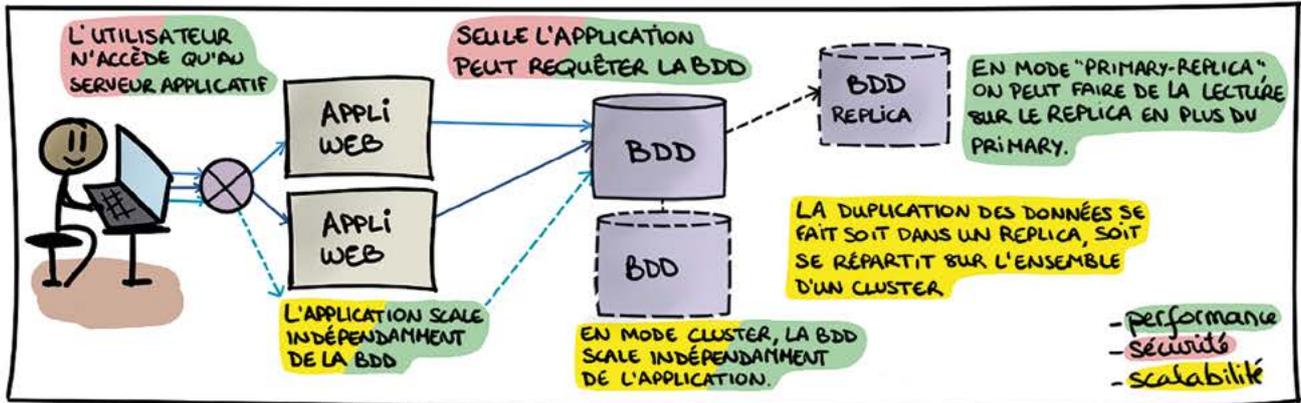


Ton serveur applicatif ne doit héberger aucune donnée: il est "stateless". C'est ton SGBD qui stocke les données.

SI TON SERVEUR ÉTAIT STATEFUL...



Donc on sépare le SGBD des serveurs applicatifs.



Il y a d'autres façons d'améliorer les performances de la base?

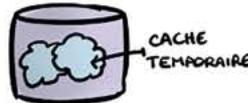
Oui! Des tas!



En augmentant le CPU ou la mémoire alloués...



...en faisant de la mise en cache...



...en optimisant le design de la base ou même, côté code, les requêtes à la base.



En réalité, il y a toute une réflexion à avoir sur les enjeux de ton projet et de ta base de données (BDD).

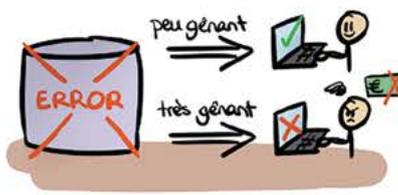


Eh bien, quel type de données vas-tu stocker?

Quel sera le volume de données à gérer?



A quel point est-ce gênant si ta BDD est indisponible?



Quel est le niveau désiré en termes de performance? Scalabilité? Cohérence des données?



Comment gérer la récupération des données en cas de perte de la BDD?



Veux-tu l'héberger chez un Cloud Provider, avec qui tu définirais un SLA (Accord de niveau de service):

- disponibilité
- performance
- scalabilité
- ...



Et tes données sont-elles sensibles? Il faut penser aux normes RGPD (de Protection de Données des utilisateurs)...!



Bref. Juste 2 ou 3 petites choses auxquelles penser!



Aarya

Les Logs



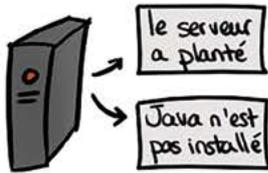
Ils devraient contenir l'historique de ce qui s'est passé sur ton système d'information :

Appli

- 5 dec 2019 à 9h30 utilisateur X connecté
- 5 dec 2019 à 9h31 Requête échoue
- 5 dec 2019 à 9h32 Appli crashe



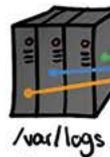
Oui! Certains logs techniques sont générés automatiquement...



... et d'autres logs techniques ou fonctionnels sont mis en place par les développeurs.



On peut les consulter: sur le serveur



ou dans un puits de logs vers lequel on les redirige*



* Privilégier le format json, géré par la majorité des puits de logs.
** Outils visant à réunir et stocker les logs au même endroit.

Tu vas souvent y recourir pour :



Comme il y a beaucoup de logs, on les catégorise par "niveaux" pour faciliter la lecture.

- DEBUG logs détaillés
- INFO normal
- WARNING alerte
- ERROR erreur
- FATAL erreur critique

On peut sélectionner les niveaux de logs à activer :

DEV	PROD
- tous les logs	- warning - error - fatal

Hmm... Et concrètement, je les implémente comment?



Tu définis ce que tu souhaites logger, et tu rédiges un message clair pour rapidement identifier l'origine du message. Par exemple :



Attention à ce que tu logges!

BIEN CHOISIR LES MOTS CLEFS ET NIVEAUX DE LOGS



- cohérence de la casse ex: error ≠ Error
- précision des mots
- choix du niveau de logs → facilite la recherche



MASQUER LES SECRETS

Info: user 214 a un nouveau mot de passe: <chiffré>

ÉCRIRE DES LOGS CLAIRS Éviter les logs du type:

Barry



Error: il y a un bug... bon courage!



PRÉCISER L'APPLI CONCERNÉE

Info: [App1]: fichier x241y9 sauvegardé dans le dossier "Fini"

Pour finir, quelques pistes de réflexion concernant le stockage des logs :



- durée mini / maxi légale de conservation
- contraintes RGPD
- coût du stockage et des flux des logs
- variations de performance liées aux logs
- gestion de la perte de données

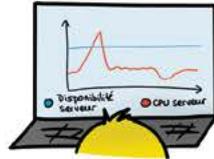
Le Monitoring

Dis! Le monitoring, c'est pour vérifier l'état de mon application?

Entre autres!



Le monitoring consiste à superviser l'activité ou l'état d'un système informatique. Il permet de :



Vérifier l'état de santé de l'infrastructure et des logiciels



surveiller la performance du site



superviser le business

Les objectifs sont multiples:

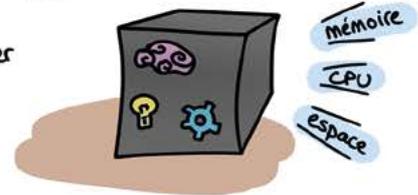
- récupérer les données clés concernant l'activité du site (parcours utilisateur, état d'une transaction...)
- vérifier la disponibilité du service
- anticiper (pics d'activité...)
- mesurer pour améliorer
- avoir de la visibilité sur l'état de ton infrastructure (nombre de serveurs actifs, performance...)

Le provider remonte déjà certaines métriques, mais c'est limité...

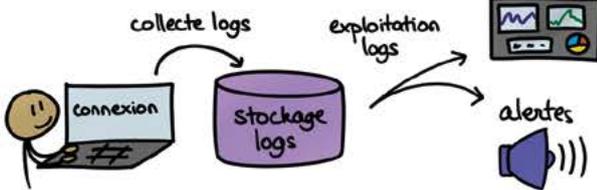


À toi d'aller chercher les autres!

Il existe des outils pour récupérer des métriques système...



...tandis que d'autres se basent sur les logs.



Cela implique d'avoir des logs bien précis et clairs pour les exploiter



De plus, la plupart des outils proposent un système d'alerte.



Donc là, si je veux commencer?

Commence par créer des alertes...

Ah! Une anomalie!

biip!

Centralise tes graphes sur un unique dashboard par profil (ops, dev, métier...)



...tu verras ainsi en un coup d'œil de ton site, et de potentiels patterns ou corrélations.

Hmmm... Les performances baissent tous les samedis soir suite à un pic de connexions.

⚠ Limitez les alertes ou vous serez vite noyés!

Définissez bien ce que vous souhaitez monitorer...



... choisissez le ou les outils dont vous aurez besoin...



... et y a plus qu'à tout mettre en place!

Génial! Et si mon appli de monitoring crashe?



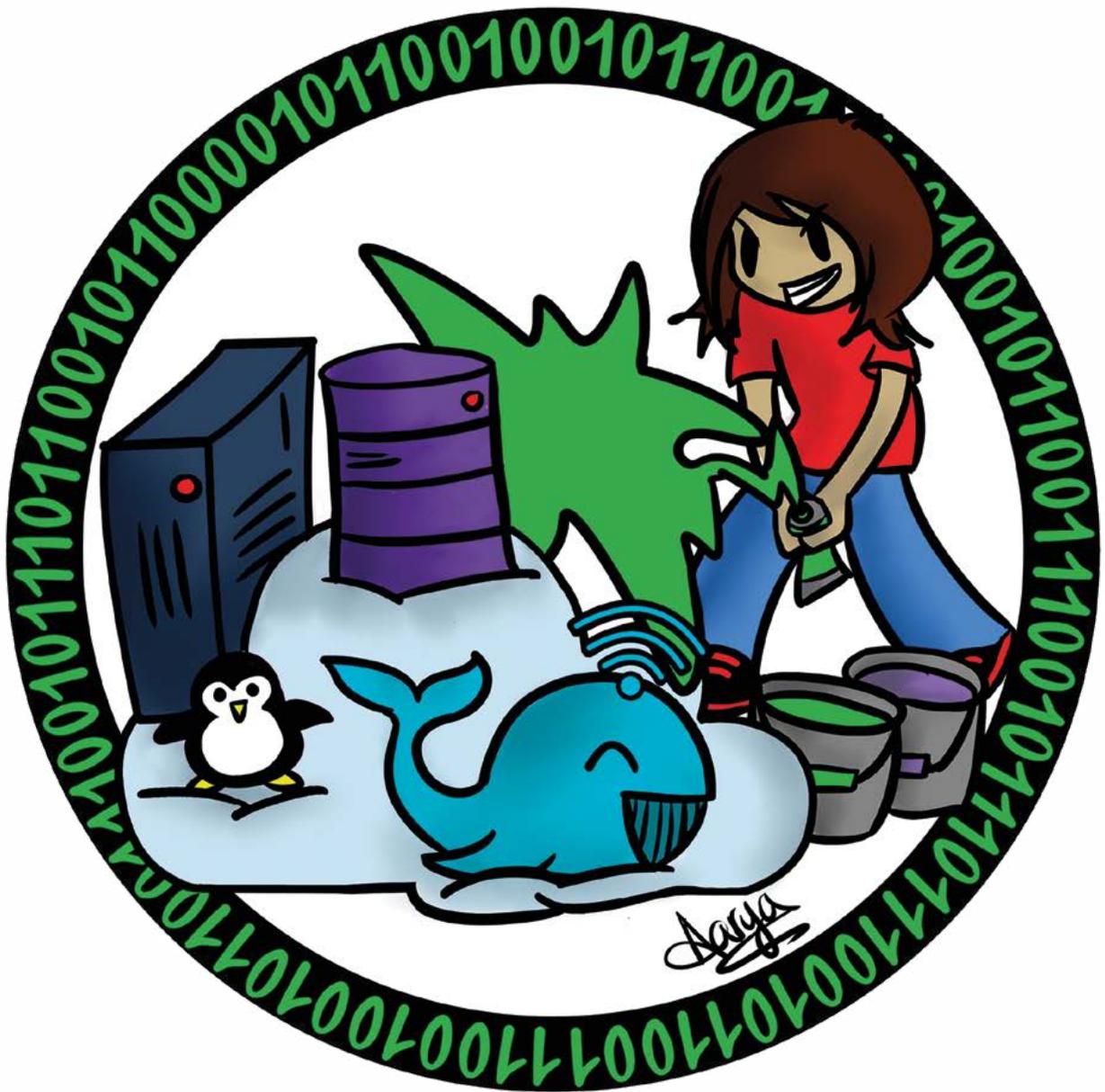
Heh, bien vu! Tu peux bien sûr la monitorer: on parle de Deadman Stack (ou Watchdog).

Aarya

Partie 2

Pour aller plus loin 26 - 43

L'OS	29
Le Terminal	30
Le HTTPS	31
Haute Dispo à tout prix !	32
L'Authentification	33
Les Cookies	34
Une vie d'Ops !	35
Le Cache	36
Le SSH	37
Les Ports et les Protocoles	38
Le Firewall	39
L'informatique, ça creuse !	40
Le VPN	41
Le Proxy et le Reverse Proxy	42
Le Load Balancer	43



L'informatique fourmille d'outils et de termes auxquels nous sommes régulièrement confrontés, aussi bien en tant que professionnels qu'en tant que particuliers, sans pour autant que l'on sache exactement comment ils fonctionnent.

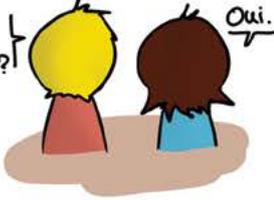
Le moment est enfin venu de creuser ces concepts !

L'OS

SYSTEME D'EXPLOITATION (OS)



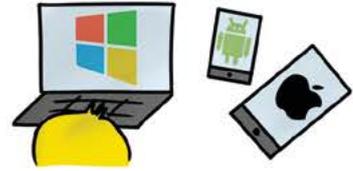
Je suis obligé d'en installer un sur mon serveur?



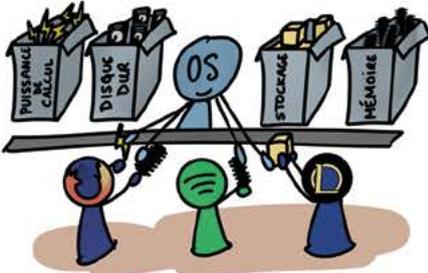
Les OS sont indispensables sur tout dispositif amené à faire du multitâches ou à gérer plusieurs utilisateurs.



L'OS est d'ailleurs souvent fourni avec ton ordinateur ou ton smartphone. Il est l'un des premiers programmes exécutés au démarrage de l'appareil.



L'OS orchestre l'allocation de ressources aux diverses applications.



Les appels système sont initiés par l'utilisateur ou par les logiciels eux-mêmes



Ils sont tous constitués de :

- Un KERNEL (noyau) : gère les fonctions clés de l'OS
- Un SHELL (interpréteur de commandes) pour communiquer avec l'OS
- Un FILESYSTEM (système de fichiers) : gère l'organisation et les droits des fichiers
- Des DRIVERS (pilotes) : fichiers qui gèrent le fonctionnement des périphériques

Donc comment choisir son OS?

Ça dépend des outils dont tu as besoin.



Certains optent pour Windows :



On lui reproche en général :

- d'être la principale cible de cyberattaques ...
- d'être un produit d'appel Microsoft pour s'engouffrer dans les grandes entreprises ...
- de collecter des données utilisateur ...
- d'être moins adapté au dev que d'autres OS (droits, terminal moins accessible...), mais cela change, notamment avec WSL: Windows Subsystem for Linux.

D'autres penchent pour MacOS :



Mais le côté 'vendor-locking' est dissuasif :



Enfin, le dernier OS connu est GNU/Linux :



Néanmoins, la prise en main est un peu plus complexe



et de nombreux outils y sont indisponibles, crashent, ou sont compliqués à installer :



mais Wine et les outils en SaaS facilitent bien les choses!

Je vois. Et côté serveur?

Sauf contexte particulier, on privilégie Linux :



Ok! Partons sur du Linux pour le serveur! Mais côté perso...

Teste par toi-même! Le "Dual Boot" permet d'avoir plusieurs OS sur ton ordinateur. L'idéal pour en tester un sans rien désinstaller!



Chanya

Le Terminal

Le Terminal (aussi dit "Console" ou "Interface de ligne de commande") permet de donner des instructions à l'ordinateur.

Comme ce n'était pas simple, les Interfaces Utilisateur sont apparues fin 1970's pour rendre les ordinateurs plus accessibles...

«Donc pourquoi t'obstiner à tout faire en ligne de commande...?!»

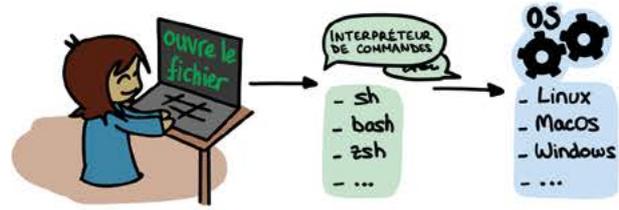
C'est plus pratique!

...ben voyons...



Déjà, je n'ai souvent pas le choix lorsque je travaille sur un serveur distant.

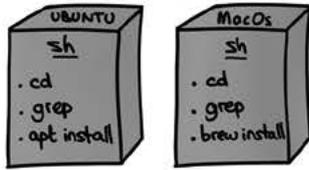
J'envoie directement mes commandes au système d'exploitation (OS) via un interpréteur de commandes.



Contrairement à l'interface, les commandes varient très peu d'un OS ou d'un interpréteur à l'autre.

Cela facilite donc le partage d'informations: c'est plus pratique de recopier une commande que d'expliquer où cliquer.

Donc si tu cherches une commande, Internet est ton ami! Quelqu'un se sera toujours posé la question avant toi...!



Et enfin, en cas d'erreur, j'ai un message d'erreur détaillé.

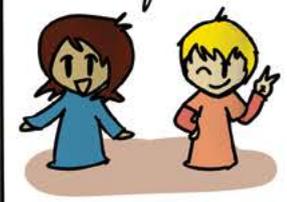
Après, chacun son terminal et chacun ses habitudes: shell, couleurs, raccourcis...

Et toi? Tu utilises quelles commandes? Hehe! Il existe même une commande pour répondre à cette question! Tadam!



GESTION DE FICHIERS	RÉSEAU	DONNÉES SYSTÈME
<ul style="list-style-type: none"> cd : changer de dossier chmod : changer les droits d'accès à un fichier ou dossier cp : copier un fichier ou dossier grep : rechercher une expression dans un fichier ou dossier gzip : compresser un fichier en .gz ls : lister les fichiers ou dossiers présents open : ouvrir un fichier rm : supprimer un fichier ou dossier 	<ul style="list-style-type: none"> dig : obtenir l'IP d'un domaine ping : vérifier la disponibilité d'un serveur ssh : se connecter à un serveur distant topdump : écouter le trafic sur une interface réseau wget : télécharger un fichier 	<ul style="list-style-type: none"> date : afficher la date et l'heure du : afficher l'espace dossier utilisé which : localiser une commande
	<p>COMMANDES OUTIL</p> <ul style="list-style-type: none"> { brew, apt, yum, ... } install <tool> <tool> : lancer une commande liée à l'outil 	
	<p>GESTION DES PROCESSUS</p> <ul style="list-style-type: none"> kill : envoyer un signal à un processus, souvent pour l'arrêter. ps : afficher les processus actifs 	

Et pour plus d'infos sur une commande, tapez "man" ou "tldr" + <commande> dans le terminal!



Araya

Le HTTPS

QUOI?! Ton site web est en "http"?!
Ouais... Et alors...?

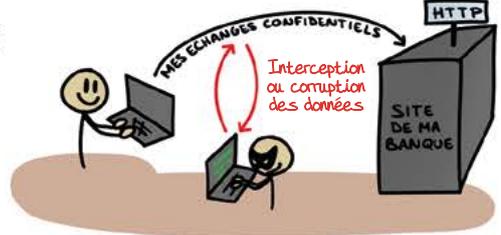


Passer en https (http "secure") est le minimum requis en sécurité. C'est sécurisé via du "TLS".*



* Viser une version récente pour plus de sécurité!

En http, tu n'as aucune garantie que ton interlocuteur est bien celui que tu penses...



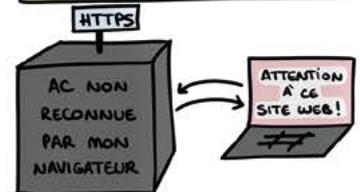
Le protocole TLS (Transport Layer Security) sécurise les échanges Internet. Une Autorité de Certification (AC) délivre un certificat qui authentifie le serveur TLS souhaité.



Ce certificat garantit l'identité du serveur et le fait qu'il sait chiffrer ses flux.



Attention: toutes les AC n'ont pas la même légitimité. Les navigateurs ont chacun leurs "AC de confiance".



Le https t'assure de l'identité de ton interlocuteur puis établit une connexion chiffrée et sécurisée.



L'URL d'un site fiable est précédée d'un cadenas ou de "https".

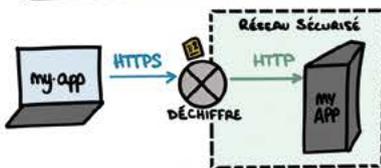


Et je mets ce certificat sur le serveur de mon application?

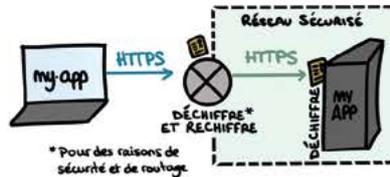
Il y a 3 façons de gérer le https.



1 Mon certificat est à l'entrée de mon infra. Les flux circulent en clair à l'intérieur.

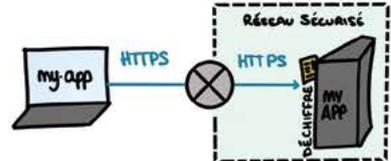


2 Il y a un certificat à l'entrée, et un autre sur le serveur applicatif.



* Pour des raisons de sécurité et de routage.

3 Le certificat est sur le serveur applicatif.



L'option 4 facilite le debug et l'installation de sondes...



... mais la sécurité préfère les autres :

- la 2 permet de contrôler les flux entrants
- la 3 garantit la confidentialité des données



Autre avantage du https: ton site générera plus de trafic.

- ↑ il sera mieux référencé dans google
- 🔒 les navigateurs ne mettront plus d'alerte dessus
- 💰 les clients seront plus susceptibles d'acheter

Donc passe en https!



Daria

Haute Dispo à tout prix !

Jamais on ne perdra tous nos serveurs simultanément!

Faut être prêt à tout! Admettons que ça arrive! Comment on répare le site?



Bon. Admettons que tous nos datacenters prennent feu en Irlande.



... ben crois-moi que ce jour-là ton site sera la moindre de tes priorités...



... parce que la moitié de la planète qui prend feu simultanément, t'es plus vraiment face à un problème de datacenters...!



L'Authentification

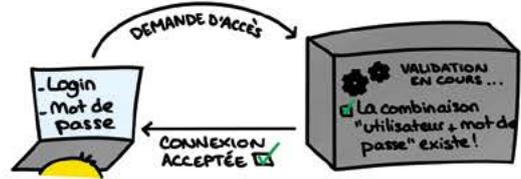
A l'aide!
On a hacké mon compte!

Zut! Malgré le MFA
(Multi Factor Authentication)?

... Le quoi...?

Ah... Bon, tu connais le principe de l'authentification simple?

A la connexion, le système vérifie ton identité en comparant tes identifiants avec ceux stockés sur le serveur d'authentification.



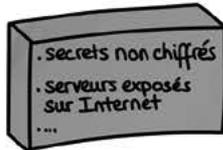
Mais ceci est loin d'être suffisant niveau sécurité:

les identifiants sont souvent trop simples à deviner...

...le serveur d'authentification est souvent mal sécurisé...

...les gens utilisent souvent le même mot de passe partout...

C'est très facile pour une personne mal intentionnée de récupérer les accès!



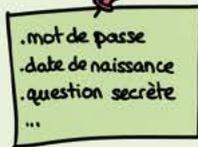
Le MFA vérifie ton identité d'au moins 2 façons différentes et indépendantes. Si l'une est non réutilisable (temporaire), on parle d'authentification "forte".

Les 3 principaux types d'authentification sont l'authentification:

Ah! Comme les confirmations par sms!



MÉMORIELLE: 1 chose que l'utilisateur connaît



MATÉRIELLE: 1 chose que l'utilisateur possède



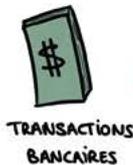
BIOMÉTRIQUE: 1 chose qui caractérise l'utilisateur



Le télétravail, le BYOD (Bring Your Own Device (to work)), l'IoT et la présence de données sensibles sur Internet font de l'authentification un enjeu de société fort!

Pour limiter tous ces risques on rajoute des autorisations en plus de l'authentification.

Ok! Bon, je vais me mettre au MFA...



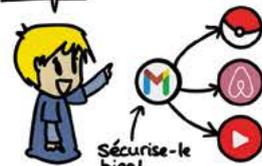
C'est activable sur la plupart des sites connus. Privilégie les outils de génération de mots de passe temporaires.

Et surtout, sécurise tes mots de passe! Ils seront durs à retenir, mais tu peux t'aider d'un gestionnaire de mots de passe.



Sinon le sso (Single Sign-On) permet d'accéder à plusieurs applications via une seule authentification.

... comme quand je me connecte à Pokemon Go via Gmail!



Enfin, si tu veux vérifier, des sites comme 'haveibeenpwned.com' permettent de vérifier si ton email est compromis.

Bon... Je lance l'opération "MFA"!



Yes, le risque 0 n'existe pas, mais ça le réduit grandement!

Les Cookies

Ben alors! Pourquoi tu râles?

J'en ai marre des cookies! On peut pas juste les interdire?



Mais certains cookies sont indispensables à la navigation!

Comment ça?



Une fois authentifié sur un site, le serveur doit pouvoir se souvenir de toi pendant toute ta navigation, sinon...



À la base, un cookie sert donc à améliorer la navigation. C'est un ensemble de données générées par le serveur web, et envoyé au navigateur. Le navigateur renvoie ce cookie avec chaque requête pour valider son identité. À la déconnexion, le cookie est détruit.

Les cookies "de session", valables la durée d'une connexion, vérifient l'identité de la personne qui navigue. Ils sont très utiles sur les sites de e-commerce par exemple.



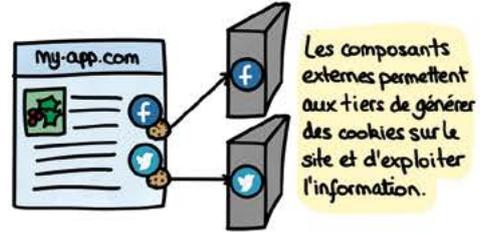
Les cookies "permanents" sont stockés (potentiellement indéfiniment) sur ton appareil, pour faciliter tes futures connexions.



Normalement, seul le site où tu navigues peut déposer des cookies. Ces cookies sont "propriétaires".



Les cookies controversés sont les cookies "tiers".



Ces tiers récoltent ainsi un tas de données, et les exploitent à des fins marketing par exemple.



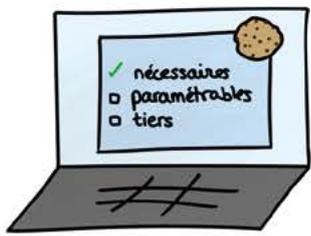
En Europe, les sites sont soumis au RGPD (Règlement Général sur la Protection des Données). Ils doivent lister les cookies utilisés, et obtenir le consentement des utilisateurs.



Que se passe-t-il si je refuse les cookies? En général, rien. La navigation est parfois moins fluide.



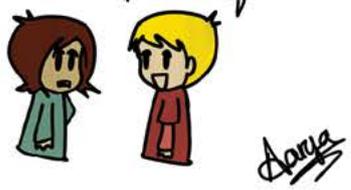
Les cookies "nécessaires" à la navigation sont forcément autorisés, mais tu as le choix pour les autres.



En plus de pouvoir refuser les cookies tiers, il existe plusieurs moyens de se protéger des cookies intempestifs:

- Navigateur qui respecte mieux la vie privée... (with icons for Brave, Firefox, and Safari)
- plugins dans le navigateur... (with a cookie icon)
- Supprimer les cookies du navigateur (with a trash can icon)

Au final, je n'échapperai pas aux bannières sur les cookies...
... Firefox et Safari bloquent déjà certains cookies tiers, et Google s'y met aussi...
Affaire à suivre!



Une vie d'Ops

Quand je dis que j'installe un nouveau serveur:

Ce que mes amis s'imaginent:



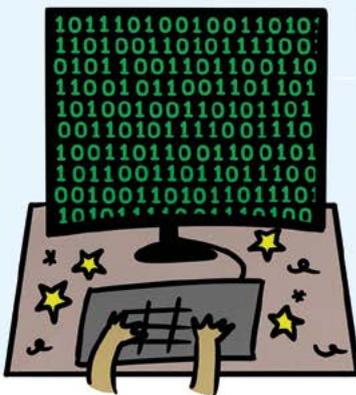
Ce que les devs s'imaginent:



Ce que mes sponsors s'imaginent:

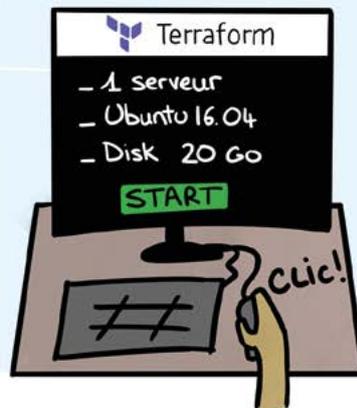


Ce que je m'imagine:



Aarya

Ce que je fais réellement:

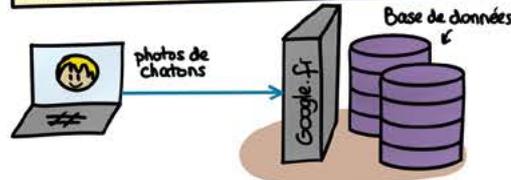


Le Cache

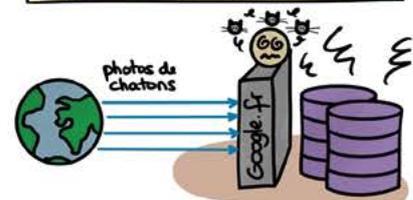
Hmm... C'est un soucis de cache... Tu t'y connais en gestion de cache?



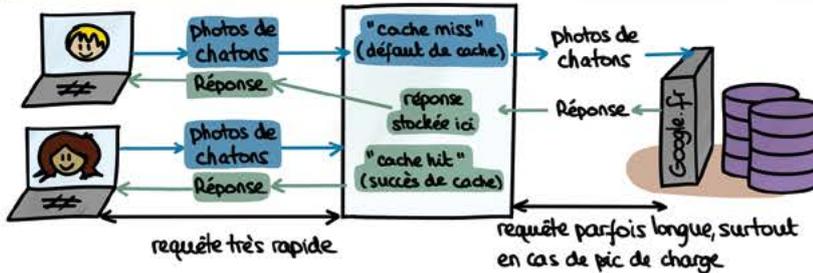
À chaque clic ou recherche, une requête est envoyée à un serveur pour afficher la page attendue.



Sauf que ce sont souvent les mêmes requêtes qui reviennent...



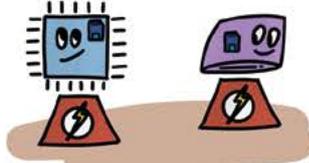
Le cache est un système de mémoire intermédiaire, qui conserve les données dans le but d'une utilisation ultérieure. L'objectif est d'accéder plus rapidement aux données souvent utilisées ou générées par des processus complexes. Il y a des caches à divers stades de la requête.



Un cache a une taille limitée. Une fois plein, il écrase les anciennes données par des données plus pertinentes.



Il existe des caches hardware, pour le processeur et les disques durs, qui ont besoin d'être extrêmement rapides...



... et une flopée de caches software :



Enfin, les CDN (Content Delivery Network) peuvent stocker en cache une version de ton site, à divers endroits du monde.



Ok c'est pas mal! Mais je ne sais pas comment le configurer...



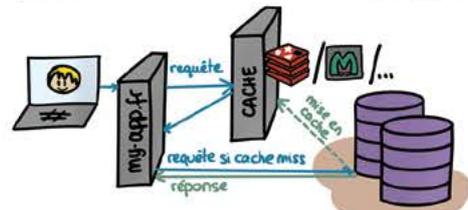
Le navigateur gère lui-même le stockage en cache sur le disque dur des utilisateurs. L'utilisateur a la main seulement pour le vider.



Côté base de données, un cache est souvent déjà mis en place et on configure surtout sa taille.



Côté serveur web, on peut configurer un CDN ou des outils comme Redis ou Memcached.



On peut donc choisir la taille du cache, et ce qu'on y met!



C'est aussi un indicateur à surveiller : s'il se remplit trop vite, ta requête est peut-être trop complexe...



Dans ton code j'ai trouvé une requête de plusieurs lignes... Je pense que c'est ça qui a cramé le cache de la base de données.



Le SSH

Dis, tu peux te connecter à mon serveur applicatif?



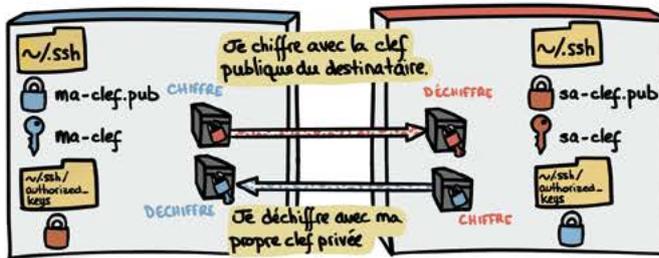
On utilise le protocole SSH (Secure Shell) pour se connecter sur un serveur distant.



On indique l'utilisateur et l'adresse IP du serveur à contacter:



La sécurité des flux est assurée par un système de cryptographie asymétrique. Chaque machine a une paire de clés privée ("sa-cléf") et publique ("sa-cléf.pub"). Je déchiffre le message chiffré par ma clé publique avec ma clé privée.



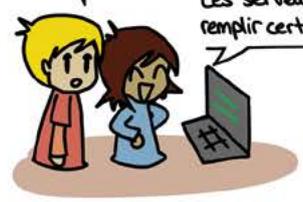
Donc quand tu te connectes, il chiffre tous les flux entre un client et un serveur. Le processus est similaire au https.



Attention! Ne communique jamais ta clé privée. Si elle est compromise, re-génère une nouvelle paire de clés.

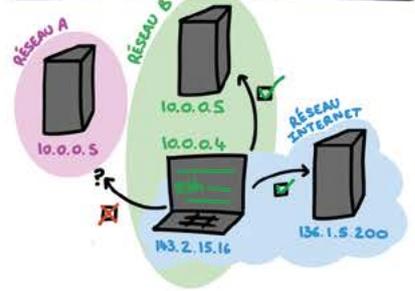


Avec ça, tu peux te connecter à n'importe quel serveur?

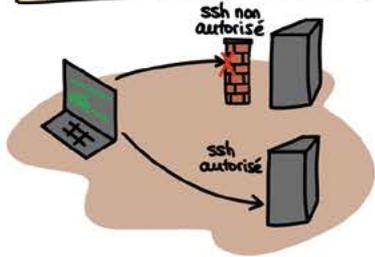


Non, heureusement! Les serveurs doivent remplir certains critères.

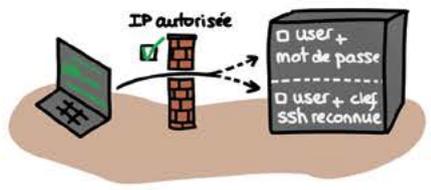
Premièrement, le serveur appelé doit être accessible depuis ton réseau



Ensuite, le serveur doit accepter les flux SSH entrants (par défaut sur le port 22).



Enfin, il faut avoir le droit d'accéder au serveur: IP autorisée + User du serveur + mot de passe ou clé ssh reconnue



Mais mon serveur n'est pas exposé sur Internet, et tu n'es pas dans son réseau...



Dans quels cas fais-tu du SSH?



- Les principales raisons sont:
- pour configurer des serveurs (via ansible par exemple)
 - pour déboguer des serveurs
 - pour transférer des fichiers sur un serveur

Et si je veux faire du SSH?



Voici quelques petites choses à savoir:

- ssh-keygen: génère une paire de clés privée-publique
- ~/.ssh/config = fichier pour configurer sa commande ssh
- bien restreindre les droits de la clé privée, car sinon la clé publique est inutilisable.
- il existe des outils pour empêcher le brute force ssh (ssh-guard, fail2ban...)
- Windows: on utilise "putty" pour le ssh.

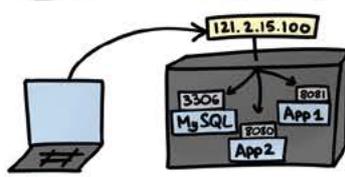
Les Ports et les Protocoles

C'est quoi un port?

C'est un point d'entrée sur ton serveur. Ça lui permet de distinguer les applications sur le serveur.



Le port est nécessaire pour établir une connexion:



⚠️ 1 seule application par port!

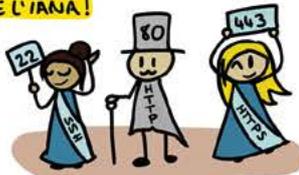
Comment savoir quel port contacter?

Les ports vont de 0 à 65535.



Les ports 0 à 1023 sont "reconnus" ou "réservés", et enregistrés auprès de l'IANA (Internet Assigned Numbers Authority). Ce sont surtout des processus système très utilisés.

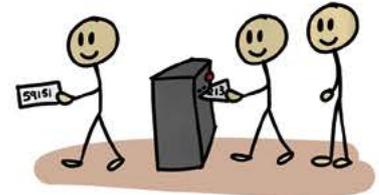
STARS DE L'IANA!



Les ports de 1024 à 49151 sont "enregistrés" ou "semi-réservés".

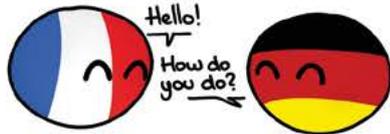


Les ports 49152 à 65535 sont "dynamiques" ou "éphémères", valables juste sur la durée de la connexion.



On se connecte à un port via un protocole. C'est une convention - un ensemble de règles - qui permet à plusieurs appareils ou logiciels de communiquer.

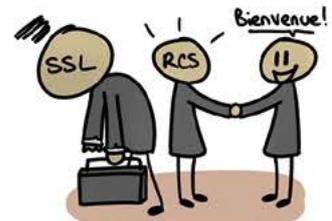
Protocole: English (UK)



Les modèles "OSI" et "TCP/IP" séparent ces protocoles par couches.

TCP/IP	OSI
APPLICATION	
	PRÉSENTATION
	SESSION
TRANSPORT	
INTERNET	RÉSEAU
HÔTE - RÉSEAU	LIAISON
	PHYSIQUE

Il existe un tas de protocoles, certains désuets, et d'autres tout récents!



Les protocoles les plus connus sont:

HTTP: HyperText Transfer Protocol
HTTPS: HTTP + couche de chiffrement SSL ou TLS



Protocole de communication client-serveur développé par le World Wide Web.

SSH: Secure Shell



protocole qui impose un échange de clés de chiffrement en début de connexion.

SMS: Short Message Service

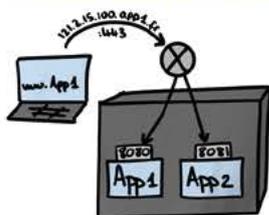
Hmm... J'ai compris que certains processus ont des ports définis... Mais comment connaît-on les autres?



Certaines commandes permettent de voir les ports actifs sur ton serveur.



Dans notre réseau, c'est à nous de configurer les ports et rediriger les flux.



On peut aussi changer les ports par défaut pour brouiller les pistes...



... au risque d'embrouiller l'équipe technique aussi...

Donc si quelqu'un connaît l'adresse du serveur et le port d'un service, il est vulnérable!



D'où les systèmes d'authentification (comme on l'a vu pour le ssh), et... l'utilisation de firewalls!

change

Le Firewall

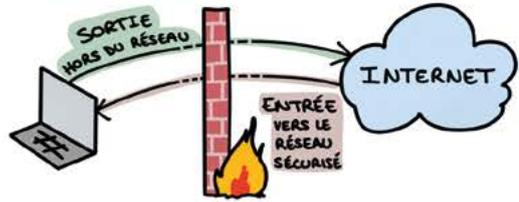
Tu vois? J'ai la bonne IP, le bon port, mais la connexion au serveur échoue...



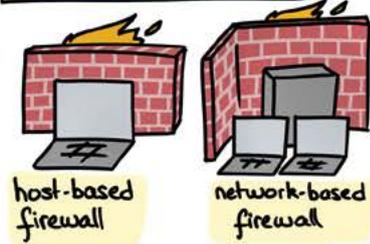
Ah! C'est à cause du firewall (pare-feu) qui sécurise ton serveur!



Un firewall contrôle, analyse et autorise ou non la communication entre plusieurs réseaux, en fonction de règles prédéfinies. Il peut y en avoir plusieurs entre 2 réseaux.



Il s'applique soit à un seul ordinateur, soit à un réseau.



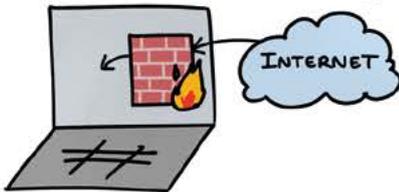
Il peut être physique ou virtuel.



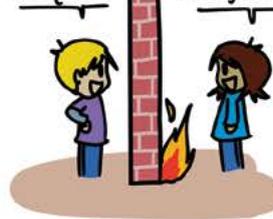
Les firewalls matériels sont des machines dédiées à ce filtrage. Certains routeurs embarquent déjà un firewall.



Les firewalls virtuels peuvent être plus ou moins complexes. Ils sont souvent utilisés pour protéger une infrastructure cloud ou des postes individuels.



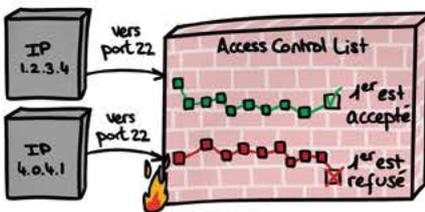
Et il filtre sur quoi? Ça dépend des firewalls



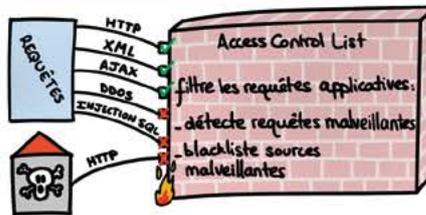
Le firewall sans état (stateless packet inspection firewall) analyse chaque paquet entrant en fonction d'une checklist configurée.



Le firewall à état (stateful packet inspection firewall) analyse le 1er paquet entrant et vérifie que chaque paquet est bien la suite du précédent.



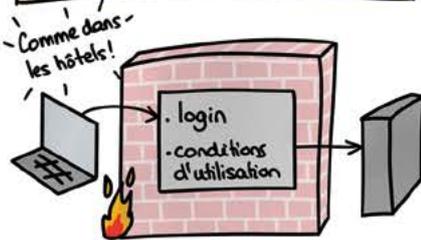
Le firewall applicatif (Web Application Firewall ou WAF) vérifie que les requêtes sont conformes au protocole attendu (HTTP, HTTPS), et protège des principales vulnérabilités applicatives.



Le firewall identifiant, filtre les connexions par utilisateur.



Le firewall captif intercepte les usagers pour leur présenter une page spéciale.



Du coup, un firewall ça suffit niveau sécurité?

C'est un début. Il y a beaucoup à dire sur le sujet!

...Ok... Bon, et en attendant je fais comment?

Heh! Viens, on va whitelister ton IP dans le firewall!



L'informatique, ça creuse !

... et c'est ainsi que Gherkin a clairement amélioré Cucumber!



... donc j'ai développé une nouvelle recette sur Chef pour déployer mon serveur!



Et tu les gères comment tes cookies?



Chai et Mocha restent très complets.

Ah! Et sinon t'as entendu parler de Mockito?



... en fait fallait le mettre entre quotes...



Eh ben! Ça te donne faim le dev!

... T'as pas idée...



Le VPN

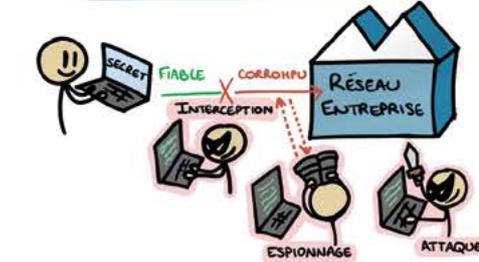
Dis! On n'a pas accès à l'Intranet depuis chez nous?

Si! Installe le VPN!

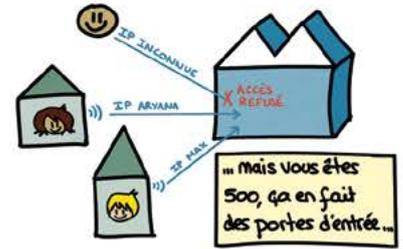
...chu obligé...?



Eh bien, on ne peut pas tout ouvrir sur Internet, car c'est open bar pour les pirates...!



On pourrait créer un filtre sur les IPs des salariés...



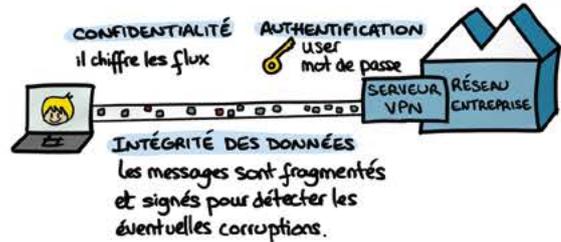
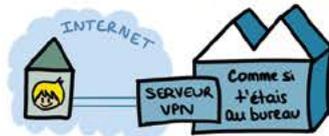
... mais vous êtes 500, ça en fait des portes d'entrée...

... et comme on l'a vu, une IP peut changer. C'est trop compliqué!



Ça nous laisse le VPN (Virtual Private Network) un réseau privé virtuel. Il permet à une machine de rejoindre un réseau et ses ressources.

Il crée un tunnel sécurisé, via Internet, entre 2 machines. Internet a 3 vulnérabilités que le VPN corrige:



Bon... Que dois-je installer?

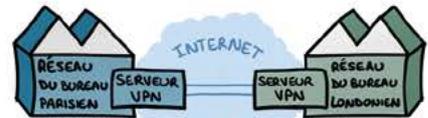
Installe l'application VPN paramétrée par ton entreprise sur ton ordinateur.



Chouette! C'est grâce à un VPN que nos bureaux de Paris et Londres partagent le même réseau?



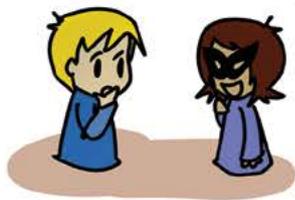
La différence est que c'est un serveur du réseau qui porte le client VPN:



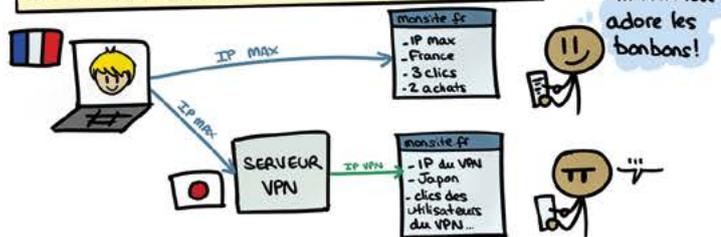
... donc toi, tu n'as rien à faire!

Il y a d'autres raisons d'utiliser un VPN?

Oui! Pour les particuliers, c'est l'anonymat!



Quand tu te connectes à un site, tu envoies des informations exploitables et revendables, dont ta localisation.



Mais alors, le serveur VPN a tes données!

Beaucoup ont une politique de non conservation des données.



Hmm... Mais le VPN préserve l'anonymat de personnes mal intentionnées...

Oui... Mais il permet aussi de naviguer incognito dans des pays répressifs.



JÉ vois... C'est pratique un réseau si fiable...!

Pratique, mais pas magique! C'est sécurisé mais ça ne te dispense pas de faire du TLS...!

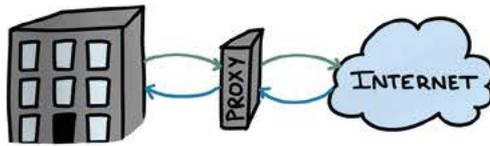


Le Proxy et le Reverse Proxy

Pourquoi n'ai-je pas accès à ce site?
C'est dû au serveur proxy!



Le proxy est un serveur intermédiaire entre un réseau local et un autre réseau (comme Internet). Il est très utilisé en entreprise.



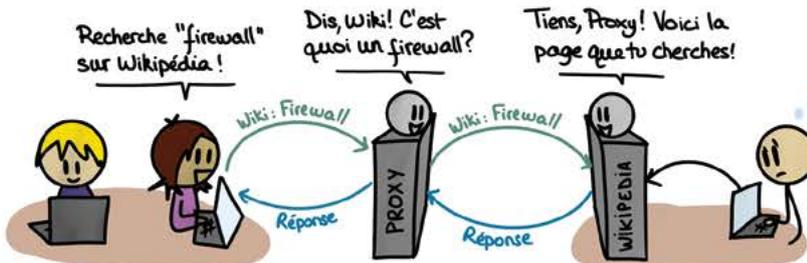
Ah! Comme un firewall?

Non! Un firewall ne fait qu'autoriser ou bloquer les connexions.



Le proxy sert vraiment d'intermédiaire

Il masque ton IP (mais ne garantit pas l'anonymat)...



Qui est ce 'proxy' qui visite mon site si souvent?

Il accélère la navigation...



Et je compresse les données et filtre les contenus lourds!



Il gère la journalisation des requêtes...



Je logge toutes les requêtes (et IP) qui passent par moi!



Enfin, il peut filtrer l'accès à certains sites...



... ou à l'inverse, permet de contourner la censure.



ÇA c'est un pote!



Comme un VPN?

Le proxy gère juste le http/https. Le VPN couvre tout ton appareil et chiffre les données.



Et le reverse-proxy fait pareil pour le trafic entrant dans le réseau?

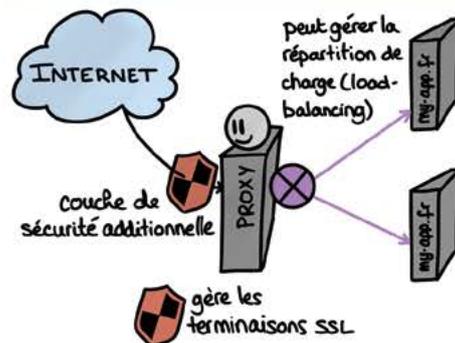
Exactement!



Il a les mêmes caractéristiques...



...avec quelques avantages supplémentaires.



Du coup je fais comment pour ce site...?

... Passe par la 4G...?



Araya

Le Load Balancer

Tu parles souvent de load balancing et de répartition de charge...



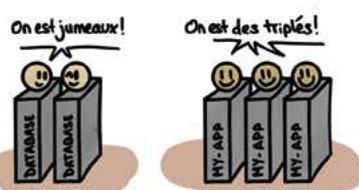
C'est utile lorsque l'on a plusieurs instances - ou serveurs - de la même application?



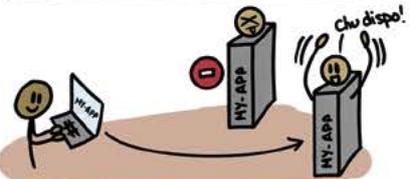
En réalité, un serveur ne peut gérer qu'une quantité limitée de requêtes simultanées. Au-delà, il risque de ralentir, voire de s'arrêter, ce qui impactera l'utilisateur.



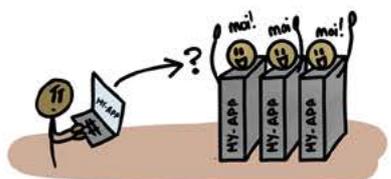
Donc on met en place des groupes de serveurs identiques, pour des questions de performance et de disponibilité.



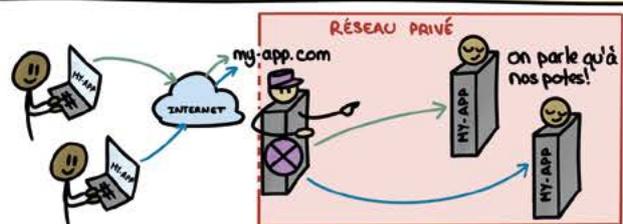
Ainsi on répartit les flux entre ces serveurs, et si l'un tombe en panne, on redirige les utilisateurs vers les serveurs fonctionnels, sans qu'ils ne s'en aperçoivent.



Sauf que l'on ne peut pas juste instancier plusieurs serveurs à la même adresse...



C'est là qu'intervient le load balancer. C'est lui, et non les serveurs applicatifs, qui est exposé sur Internet. Il va ensuite se charger de rediriger et répartir les flux entrants sur les serveurs concernés.

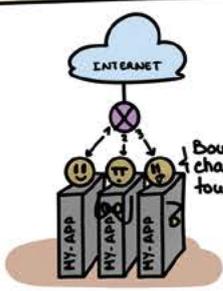


Ok! Et pour "scaler", j'ai juste à rajouter un serveur...!

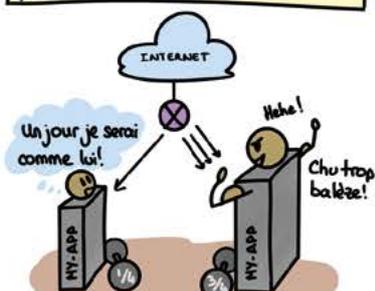
Et pour gérer la répartition, il y a diverses méthodes, certaines très simples, d'autres basées sur des algorithmes plus complexes.



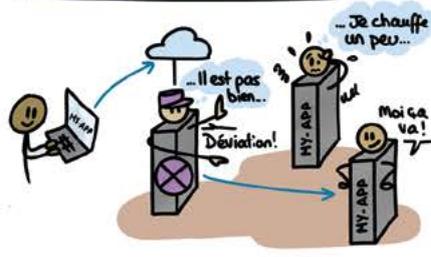
Le "Round Robin" est la plus simple. Il alterne bêtement entre chaque serveur.



Le "Round Robin Pondéré" gère les flux en fonction de la pondération donnée aux serveurs.



D'autres méthodes gèrent la répartition en fonction de l'état des serveurs: le nombre de connexions, l'utilisation du CPU, de la mémoire...



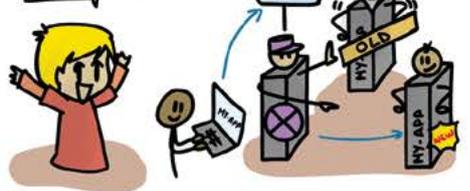
Mais ça implique que le load balancer ait accès aux métriques des serveurs!



Le load balancer effectue un "health-check" régulier pour connaître l'état de chaque serveur.



Top! Eh, on peut donc déployer une nouvelle version de l'application en parallèle de l'ancienne, puis rediriger les utilisateurs. Ils ne seront donc pas impactés!



Effectivement! On parle alors de "Zero Downtime Deployment!"



J'adore mon métier !

Ah! Ça fait du bien de se changer les idées! C'est sûr!



C'était intense ces derniers temps!

C'est parce que tu aimes sortir de ta zone de confort!



... L'informatique est tellement vaste... y a trop de choses à savoir...!



J'ai constamment l'impression de ne rien savoir...

Moi aussi... Il y a toujours quelqu'un de plus calé que soi!



C'est angoissant... Mais j'apprends tellement chaque jour!



Je me demande ce que demain me réserve...



... J'adore mon métier!

Hehe!



Merci à tous pour ces moments de richesse!

Danya

MERCI à Edouard qui a lancé le projet, à la tribu Ops pour m'avoir donné les moyens de travailler dessus, à OCTO et à la Comm' pour la publication de l'ouvrage et à mes managers pour leur soutien. MERCI à tous les Octos et à tous ceux et celles qui m'ont relue, challengée et encouragée. C'est grâce à vous que mes planches sont ce qu'elles sont.



*There
is
a Better
Way*

OCTO Technology

▶ CABINET DE CONSEIL ET DE RÉALISATION IT ◀

« Nous croyons que l'informatique transforme nos sociétés. Nous savons que les réalisations marquantes sont le fruit du partage des savoirs et du plaisir à travailler ensemble. Nous recherchons en permanence de meilleures façons de faire. **THERE IS A BETTER WAY !** »

– Manifeste OCTO Technology





Dépot légal : Octobre 2021

Conçu, réalisé et édité par OCTO Technology.

© OCTO Technology 2021

Les informations contenues dans ce document présentent le point de vue actuel d'OCTO Technology sur les sujets évoqués, à la date de publication. Tout extrait ou diffusion partielle est interdit sans l'autorisation préalable d'OCTO Technology.

Les noms de produits ou de sociétés cités dans ce document peuvent être les marques déposées par leurs propriétaires respectifs.