Architecture d'hébergement Web en masse

Frédéric Pauget

Architecture d'hébergement Web en masse

Frédéric Pauget
Télécom ParisTech
46, rue Barrault
75 634 Paris Cedex 13

Résumé

Depuis quelques années la demande de création de sites Web explose. Chaque projet de recherche, chaire, laboratoire doit avoir une visibilité sur Internet. Les besoins d'enseignement imposent aux étudiants de concevoir des sites ou services Web. Enfin les "pages perso" traditionnellement servies dans le répertoire `public_html` *des utilisateurs sont peu fiables et peu sécurisées.*

Pour répondre à ces problématiques nous avons cherché à mettre en place une solution permettant à l'utilisateur de gérer son hébergement de manière totalement autonome (création, destruction) en proposant plusieurs modèles de systèmes et de bases de données.

La solution mise en œuvre se base sur les conteneurs ("container" en anglais) LXC. L'utilisateur accède à une interface Web pour paramétrer son conteneur puis une fois celui-ci créé il administre son site via une connexion SSH ou SFTP avec ses identifiants habituels. Il peut alors déposer ses fichiers ou exécuter des commandes dans le conteneur.

Les modèles peuvent être démarrés en mode d'édition, l'accès se fait alors par les administrateurs en s'attachant au conteneur. Le modèle une fois modifié est mis en test puis en production via un script qui synchronise le système de fichiers et si besoin effectue un redémarrage des processus serveurs indiqués par l'administrateur.

Mots clefs

web, conteneur, LXC

1 Ancienne architecture et problématiques

1.1 Pages personnelles

Depuis de très nombreuses années l'école a mis en place un système d'hébergement permettant à toute personne ayant un compte informatique (environ 3000) d'héberger un site personnel. La solution technique retenue à l'époque est un classique : l'utilisation du serveur Web apache avec le *mod_userdir*. Pour les administrateurs le paramétrage est simple. Pour les usagers il suffit de déposer les fichiers dans le dossier `public_html` de leur répertoire personnel. Télécom ParisTech propose depuis de nombreuses années ce type d'hébergement permettant de servir des fichiers statiques et des sites en PHP. La figure 1 représente l'infrastructure.

Cette solution a toutefois de nombreux inconvénients. L'utilisateur système exécutant le service apache doit pouvoir lire les fichiers à servir, donc lire tout le contenu des répertoires `public_html`. Cela se traduit généralement en octroyant des droits de lecture pour tous. Dans le cas d'un site en PHP ayant besoin d'écrire des données cela impose même des répertoires avec accès en écriture. De trop nombreux usagers

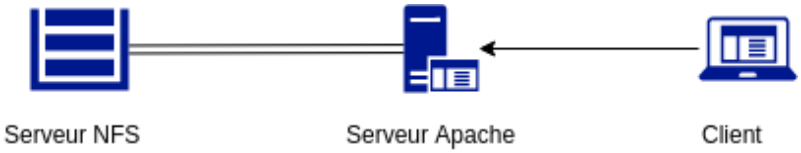

Figure 1 - Ancienne architecture pages Web personnelles

solutionnent ces problèmes en étendant les droits d'écriture pour tous à l'ensemble du contenu du répertoire `public_html`.

Ces répertoires en écriture sont autant de moyens de placer un script malicieux pouvant désinformer les internautes, héberger des activités illicites ou encore moissonner les données confidentielles accessibles depuis le service Web. Ce dernier point est particulièrement sensible, en cas de faille système ou tout simplement de droits mal positionnés, l'intégralité des données des répertoires personnels des utilisateurs hors `public_html` peuvent être exposées.

Un autre inconvénient est la fiabilité de l'hébergement : un site mal programmé peux monopoliser les ressources serveur bloquant ainsi les pages de tous les usagers.

Enfin ce mode d'hébergement est inadapté dans les cas d'un site avec plusieurs gestionnaires ou d'un site devant être pérenne, l'hébergement étant lié au compte informatique.

1.2 Pages pour projet

En parallèle l'école propose un autre type d'hébergement de sites : projet de recherche, support pédagogique, communication… On appellera ce type de sites des "pages pour projets". La mise en place est à chaque fois manuelle et doit être réalisée par la DSI.

Suivant les cas l'hébergement était :

— dédié sur une machine virtuelle pour les sites les plus sensibles ;

— mutualisé à l'aide d'une simple configuration d'hôte virtuel ("virtualhost") ;

— mutualisé avec l'utilisation du module *suPHP*, celui-ci permet d'exécuter les scripts PHP avec les droits de l'utilisateur afin d'améliorer un peu la sécurité. Malheureusement il ne prévient en rien le laxisme des droits d'accès choisis par certains utilisateurs.

Ce dernier type d'hébergement était assez satisfaisant d'un point de vue isolation entre les utilisateurs mais le développement de *suPHP* étant abandonné depuis 2013 il fallait trouver une solution de remplacement.

2 Nouvelle architecture

2.1 Cahier des charges et premiers choix

Nous avons identifié les besoins et contraintes suivantes :

— isolation totale entre chaque utilisateur et chaque site ;

— possibilité de créer sa page perso en libre service pour l'utilisateur ;

— pouvoir proposer plusieurs configurations d'hébergement ;

— pouvoir héberger plus de 3000 sites.

Le souhait d'isolation totale entre les utilisateurs nous a orienté vers les solutions de cloisonnement incluses dans le noyaux Linux pour les raisons suivantes :

— utilisation simple des systèmes de fichiers présents sur l'hôte ;

— très faible impact de l'isolation sur les performances ;

— la partie système d'exploitation hébergé reste minimaliste et consomme très peu de ressources ;

— utilisation possible dans une machine virtuelle de type KVM.

De plus nous souhaitons utiliser au maximum les briques présentes dans Debian GNU/Linux. C'est la distribution de la majorité de nos serveurs et l'utilisation de paquets inclus dans la distribution nous garantit une simplicité de gestion des mises à jour de sécurité.

2.2 Infrastructure retenue

Au final l'architecture choisie est composée des briques suivantes :

— les conteneurs sur lesquels les utilisateurs se connectent et qui servent les pages Web ;

— isolation des conteneurs avec LXC ;

— une infrastructure de proxies inverses (*reverses-proxies*) permettant l'accès aux pages Web des conteneurs ;

— de quoi gérer les éléments précédents.

La plupart des choix techniques ont été réalisés au début de l'été 2016 au commencement du projet.

2.3 Hébergement des conteneurs

Afin de pouvoir anticiper la montée en charge nous avons choisi de répartir les conteneurs sur plusieurs hôtes. Ceux-ci sont des machines virtuelles KVM hébergées sur notre infrastructure Proxmox préexistante comme indiqué sur la figure 2. Cela nous permet de bénéficier des avantages de la virtualisation classique : allocations de ressources, migration de VM, sauvegardes…

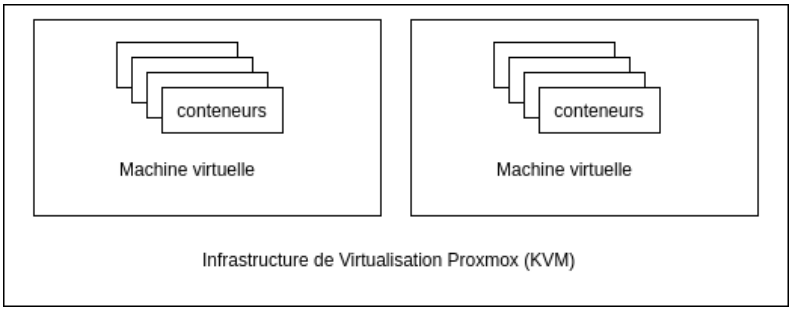

Figure 2 - Hébergement des conteneurs

2.4 Système de fichiers racine du conteneur

Afin de constituer le système de fichiers du conteneur nous voulions dissocier la partie système d'exploitation (gérée exclusivement par la DSI et non modifiable pendant l'exécution) et les données des utilisateurs. Pour cet usage nous avons choisi *aufs* car disponible sur Debian 8. Dans le futur nous seront probablement amenés à évaluer *overlayfs* depuis son introduction en Debian 9.

Il nous fallait aussi pouvoir attribuer des quotas disque à chaque utilisateur. Plusieurs solutions ont été envisagées :

— utiliser *NFS* avec un partage par conteneur, ce que permet simplement notre infrastructure NAS. Cette solution n'a pas été retenue, les conteneurs avec un tel montage ne pouvaient pas démarrer ;

— utiliser une partition *brtfs* avec des quotas par *subvolumes*. Ce choix initial nous a posé problème lors de la mise à jour de nombreux fichiers dans le conteneur ;

— utiliser une partition *xfs* avec les quotas de projets, cette solution est actuellement en production ;

— créer une partition séparée pour chacun des conteneurs. Non testé, très probablement fonctionnel mais plus compliquée à mettre en place et posant des problèmes de fragmentation sur les petits systèmes de fichiers. Cette solution a été envisagée comme une solution de repli en cas d'échec des précédentes.

Les chemins d'accès ont été choisis comme indiqué en figure 3

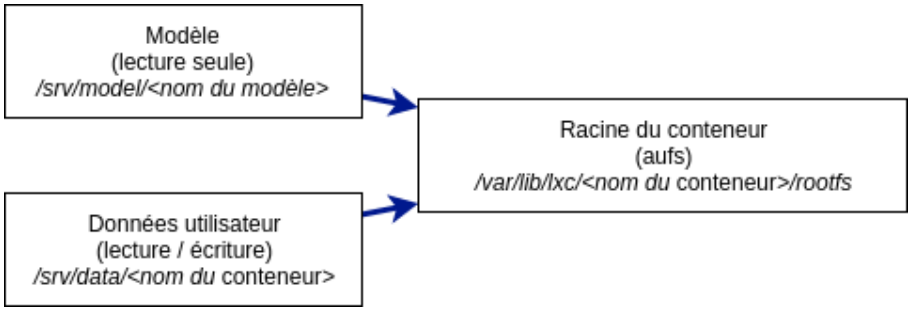

Figure 3 - Systèmes de fichiers utilisés par les conteneurs

La partition de "données utilisateur" est sauvegardée chaque nuit.

2.5 Les conteneurs

Les conteneurs sont basés sur un système Debian. Le modèle standard utilise un serveur OpenSSH afin de permettre les accès utilisateurs. Ceux-ci sont authentifiés sur l'infrastructure Kerberos de l'école ou avec des comptes locaux spécifiques dans le cas de prestataires extérieurs. Cron et logrotate sont également présents sur celui-ci.

Les droits sont positionnés pour que les utilisateurs ne puissent déposer des fichiers que dans leur répertoire personnel et à la racine servie par apache. Ils ont les droits de lecture sur les logs.

L'accès Web aux conteneurs passe obligatoirement par l'infrastructure de proxies inverses de l'école. Ils permettent de réaliser le chiffrement SSL (nous avons imposé que tous les sites hébergés soient uniquement disponible avec ce protocole). Dans le cas des pages perso ils permettent de "proxifier" les requêtes vers le bon conteneur à partir de l'URL `https://perso.telecom-paristech.fr/<login>`. Enfin ils proposent optionnellement une authentification des visiteurs via Shibboleth.

Les proxies inverses sont configurés pour transmettre les requêtes sur le port 8080 des conteneurs. Ce choix d'un port non privilégié a été fait afin de permettre le lancement de processus serveur sans droits particuliers sur le conteneur. Sur la plupart des modèles le serveur Web Apache est configuré pour écouter sur ce port et servir le répertoire `/srv/www`.

Les conteneurs sont tous sur le même sous-réseau IP et ont uniquement une adresse IPv6. Nous avons fait ce choix pour deux raisons : les accès d'administration sont exclusivement réalisés depuis l'école (ou via une connexion VPN) où ce protocole est généralisé et cela prépare à un futur sans IPv4. Les accès Web utilisant les proxies inverses, le service est quand à lui disponible aussi bien en IPv4 qu'en IPv6.

Les conteneurs sont protégés par un filtrage IP sur les routeurs. De plus un firewall local au noeud d'hébergement restreint les communications inter-conteneurs.

Pour des conteneurs spécifiques, notamment ceux hébergeant un site géré par un prestataire extérieur nous avons prévu la possibilité d'affecter une IPv4 en plus et d'ouvrir l'accès SSH depuis l'extérieur.

2.6 Orchestration

Pour chaque conteneur il faut :

- un nom, pour les conteneurs de page perso c'est *perso-<login>* ;
- les quotas RAM et disque ;
- le nœud d'hébergement ;
- une adresse IP, pour les conteneurs de page perso l'identifiant Unix (*uid*) de la personne est utilisé pour le calcul de l'IP (concaténation du préfixe avec la conversion en hexadécimal de l'uid) ;
- pour les conteneurs de projet :
  - l'URL d'accès publique ;
  - les personnes autorisées à se connecter ;
- optionnellement des directives spécifiques à inclure dans la configuration des proxies inverses (choix pour l'authentification notamment).

L'objectif initial étant le remplacement de l'infrastructure de pages personnelles nous avons choisi de stocker ces paramètres dans l'annuaire LDAP en ajoutant des attributs personnalisés à l'entrée de la personne. L'administration de ces attributs s'effectue avec la même application que celle gérant les comptes informatiques. Cette liaison assure l'expiration automatique du conteneur avec l'expiration du compte. Pour les conteneurs de projets le stockage est réalisé dans une branche spécifique.

Des scripts paramètrent les différents composants à partir des données de configuration :

- enregistrement de l'IP du conteneur dans le DNS ;
- génération de la configuration des proxies inverses ;
- création des bases de données ;
- création d'un script de gestion du démarrage et de la configuration du conteneur qui sera exécuté sur le nœud cible.

Le script de démarrage du conteneur effectue les opérations suivantes :

- création du répertoire `/var/lib/lxc/<nom du conteneur>`
- s'il n'existe pas, création du répertoire de stockage des données de l'utilisateur `/srv/data/<nom du conteneur>`
- montage de la racine du conteneur `/var/lib/lxc/<nom de conteneur>/rootfs`
- écriture du fichier de configuration `/var/lib/lxc/<nom de conteneur>/config`
- ajout des règles de firewall
- démarrage du conteneur avec `lxc-start`

Ce même script effectue ensuite les configurations suivantes pendant le fonctionnement du conteneur :

- configuration du nom d'hôte ;
- mise en place des quotas disques et RAM ;
- configuration des utilisateurs ayant accès au conteneur en SSH ;
- configuration du serveur Web ;
- configuration de l'accès à l'éventuelle base de données et écriture d'un fichier contenant les identifiants de connexion à celle-ci ;

Nous avons choisi d'effectuer ces opérations de configuration en cours de fonctionnement pour permettre des reconfigurations sans avoir besoin de redémarrer le container lors de la modification des paramètres.

C'est ce même script qui est utilisé pour effectuer les arrêts et redémarrage des conteneurs en cas de redémarrage de l'hôte.

2.7 Gestion des modèles

Les modèles sont à la base de tous les conteneurs. Ils sont initialement créés comme un conteneur LXC standard : un système de fichiers classique monté en lecture écriture. La configuration est effectuée en s'attachant au conteneur en fonctionnement via la commande *lxc-attach*. Une machine virtuelle est dédiée à l'hébergement et à la modification de ces modèles. Un moyen d'édition pourrait être de directement modifier les fichiers depuis l'hôte. Cela est source d'erreurs sur les droits d'accès car nous utilisons la translation d'uid : sur le système de fichiers vu depuis l'hôte les uid et gid sont décalés de +100000 par rapport à ceux dans le conteneur. Nous avons fait ce choix afin d'exécuter les conteneurs de la manière la plus sécurisée possible.

Un modèle, une fois finalisé, est tout simplement copié sur la partition de modèles des autres nœuds dans `/srv/models`. La copie se fait d'abord sur un nœud de test et après validation du bon fonctionnement par l'administrateur, sur les nœuds de production.

Après la mise en service, les modèles continuent d'évoluer, notamment pour les mises à jour de sécurité, les ajouts de logiciels ou les changements mineurs de configuration. Les évolutions sont immédiatement prises en compte par tous les conteneurs au moment de la copie. En cas de mises à jour nécessitant un redémarrage des processus, il faut par contre en complément lancer l'opération sur tous les conteneurs par une simple boucle en shell.

Un historique des modèles est conservé grâce à l'utilisation des instantanés btrfs sur le nœud maître.

Afin de garantir la bonne application des mises à jour il est impératif qu'aucune modification n'ai été apportée au système dans les conteneurs. En effet dans le cas d'une modification, celle-ci serait prépondérante sur celle du modèle. Les quelques opérations de configurations devant toucher au système (nom d'hôte par exemple) sont réalisées par le script de lancement. L'utilisation courante par les usagers ne peut pas générer ce type de déviation, les fichiers systèmes sont naturellement protégés par les droits standard. Seul un administrateur pourrait le faire. Les procédures doivent bien mentionner ce point. Il est facile de surveiller de tels écarts en analysant le contenu du répertoire de données du conteneur : il ne doit pas contenir de fichiers systèmes.

En cas d'évolution importante touchant aux fonctionnalités offertes ou pour les nouvelles versions de logiciels, il faut créer un nouveau modèle.

3 Utilisation du service par les utilisateurs

3.1 Gestion de son conteneur personnel

Chaque personne ayant un compte a l'école a accès à une interface de gestion de son compte informatique. Nous avons inclus dans celle-ci la possibilité de créer son hébergement personnel et de choisir une base de données (figure 4).

La création est immédiate et le conteneur directement accessible. Le changement de modèle est instantanément effectué et permet ainsi à l'utilisateur de maîtriser le moment d'application des mises à jour majeures (cf. 2.7). Nous avons récemment proposé le passage de PHP de la version 5 à la version 7 comme cela.

Figure 4 - Interface utilisateur

Une fois le conteneur créé, l'accès à celui-ci se fait par *ssh*. L'utilisateur peut directement éditer des fichiers via l'interpréteur de commandes mis à disposition ou déposer des fichiers par *scp*, *sftp* ou *rsync* dans son *home* et dans l'espace d'hébergement.

3.2 Migration des pages personnelles

La nouvelle architecture proposant des versions plus récentes des logiciels serveur et utilisant des nouveaux chemins d'accès aux fichiers nous n'avons pas voulu mettre en place de migration automatique de l'hébergement `public_html` vers les conteneurs. L'opération aurait pu "casser" de nombreux sites.

Nous avons profité de l'architecture de proxies inverses pour servir conjointement les pages sur les deux infrastructures : une personne possédant un conteneur a systématiquement sa page personnelle servie par celui-ci, une personne ayant un répertoire `public_html` préexistant a son site servi par l'ancien service. Par contre le chemin retour "conteneur vers répertoire" n'est pas possible. Nous prévoyons une extinction de l'ancien serveur à la fin de l'année.

Une procédure de copie est toutefois fournie aux usagers pour faciliter la migration.

3.3 Conteneur projet

Les conteneurs "projets" sont créés sur demande via notre système de tickets. L'adresse du service est choisie par le demandeur, que ce soit sur un nom de domaine géré par l'école ou extérieur (par exemple dans le cas de projet en collaboration souhaitant avoir leur propre domaine). L'accès d'administration en ssh au conteneur est possible pas plusieurs personnes, aussi bien avec des comptes de l'école ou des comptes spécifiques crées pour des extérieurs.

4 Retour d'expérience

Après une phase initiale de tests techniques débutée durant l'été 2016, une dizaine d'utilisateurs ont pu migrer leur hébergement de pages personnelles fin 2016 sur le seul modèle proposé (Debian 8 avec PHP 5). À cette époque l'interface self-service n'était pas encore en service et ces premiers utilisateurs nous ont permis grâce à leurs retours de faire quelques évolutions de configuration (filtrage réseau des conteneurs, possibilité de connexion http sortantes, catalogue de logiciels disponibles).

En juin 2017 l'interface self-service a été mise en service en proposant un nouveau modèle (Debian 9 et PHP 7). L'arrêt de l'ancien service a été annoncé pour la fin de l'année. Depuis fin août, la création de répertoire `public_html` n'est plus prise en compte dans la configuration des proxies inverses.

Fin septembre, 85 personnes ont créé leur conteneur auxquelles s'ajoutent 60 conteneurs "projets". 518 personnes ont encore un hébergement `public_html`.

Il est très probable qu'un nombre assez important de `public_html` ne soient jamais transférés car utilisés par des étudiants ponctuellement pour des projets d'enseignements. Par contre, nous avons noté une demande importante sur les hébergements de projets. Les personnes apprécient l'accès partagé, la possibilité de choix du nom et la pérennité de celui-ci après le départ du demandeur, notamment dans le cas des doctorants.

La DSI utilise également cette infrastructure pour certains des sites hébergés. Le service de support (GLPI) a été migré en remplacement de l'hébergement en VM pour en simplifier l'administration. Nous envisageons de transférer à terme la plupart des nos sites Web sur ce type d'hébergement.