

AJAX | LINUX | MySQL | SQL | UBUNTU | Vmware ESX

Le plus grand magazine sur PHP au monde

phpsolutions

Nouvelles technologies et solutions pour les développeurs PHP

PHP N°10/2010 (46) ISSN 1731-4593

AJAX ET PHP

SÉCURITÉ WEB
PROTÉGEZ VOS CODES SOURCES !

CRÉEZ VOTRE PROPRE HÉBERGEMENT
SERVEUR MYSQL ET PHP

NEXENTA
SOLUTION DE STOCKAGE BASÉE
SUR ZFS ET UBUNTU

POUR LES DÉBUTANTS

SQL
COMMUNIQUEZ
AVEC UNE BASE DE DONNÉES
À PARTIR D'UN SCRIPT PHP

Et vous, pour l'HÉBERGEMENT, vous êtes plutôt :



ou



WEB ?

type	hôtel de 45 chambres ou chalet privé	mutualisation de 45 sites ou serveur dédié
capacité	chambre de 1 à 6 lits	espace Web de 250 Mo à 3 Go en mutualisé
accès	route de montagne	FTP
sécurité	pente raide, extincteur	RAID-1, back-up Mail
réervation	par nuitée	par année
domaine	Méribel, Chamonix...	.fr .com .net .org .info .biz...
les plus	parking, petit-déjeuner	CMS à la demande, interface d'administration

eziHost L'HÉBERGEMENT SIMPLE ET PRATIQUE

- Domaines, Web, Mails, FTP, PHP, Bases MySQL, Statistiques, Panneau d'administration...
- Vous pouvez bénéficier également d'un service d'installation de CMS à la demande, parmi un panel de CMS courants.

Toutes nos offres sont personnalisables à volonté. Plus de renseignements sur notre site www.ezihost.fr ou appelez au 04 76 22 68 94. eziHost est un service édité par la société Amoks.

4,92€
HT/mois
soit
59€
HT/an

PACK
BASIC

*30% de réduction globale pour la première facture aux **50 premiers clients**

-30%*
code promo
phpsol10

voir détails de l'offre sur notre site web.
www.ezihost.fr

LINAGORA

présente

OBM Online

LA MESSAGERIE COLLABORATIVE OPEN SOURCE HÉBERGÉE !

NEW

Agendas partagés

Synchronisation

Synchronisation et mobilité

<http://online.obm.org>

Le périodique *phpsolutions* est publié par
Software Press Sp. z o.o. SK
Bokserka 1, 02-682 Varsovie, Pologne
Tél. 0975180358, Fax. +48 22 244 24 59
www.phpsolmag.org

Président de Software Press Sp. z o.o. SK :
Paweł Marciak

Directrice de la publication :
Ewa Łozowicka

Dépôt légal :
à parution
ISSN : 1731-4593

Rédacteur en chef :
Łukasz Bartoszewicz

Couverture :
Sławomir Sobczyk

DTP :
Sławomir Sobczyk Studio2W@gmail.com

Composition :
Sławomir Sobczyk

Correction :
Valérie Viel, Thierry Borel, Barbara Bourdelles

Bêta-testeurs :

Brice Favre, Valérie Viel, Cyril David,
Christophe Milhau, Alain Ribault, Stéphane Guedon,
Eric Boulet, Mickael Puyfages, Christian Hernoux,
Isabelle Lupi, Antoine Beluze, Timothée Neullas,
Yann Faure, Adrien Mogenet, Jean-François Montgaillard,
Turmeau Nicolas, Jonathan Marois, Wilfried Ceron,
Wajih Letaief, François Van de Weerdt, Eric Vincent,
Franck Michaël Assi, Francis Hulin-Hubard,
Nicolas Dumas, David Michaud.

Les personnes intéressées par la coopération
sont priées de nous contacter :
editor@phpsolmag.org

Publicité :
publicite@software.com.pl

Pour créer les diagrammes on a utilisé le programme



AVERTISSEMENT

Les techniques présentées dans les articles
ne peuvent être utilisées qu'au sein des réseaux
internes. La rédaction du magazine n'est pas
responsable de l'utilisation incorrecte des techniques
présentées. L'utilisation des techniques présentées peut
provoquer la perte des données !

TABLE DES MATIÈRES

VARIA

6 Actualités

Actualités du monde du développement.
Christophe Villeneuve

PROJETS

8 Solution de stockage basée sur ZFS et Ubuntu

Olivier Olejniczak

La virtualisation de PC a amené de nouveaux besoins. Alors que la concentration de serveurs fonctionnels virtualisés par serveur physique de virtualisation s'accroît, la demande en termes de capacité de stockage et de fiabilité des équipements s'accroît également. Dans cet article vous verrez comment mettre en œuvre une solution de stockage de très large capacité et sécurisée accessible en NFS ou iSCSI depuis le réseau local (exemple : serveur de virtualisation).

DOSSIER

20 AJAX et PHP

Jean Tatareau

L'augmentation des débits Internet et de la diversité d'applications sur PC mais aussi de différents terminaux, mobiles, tablettes, assurent la présence incontournable d'AJAX. Dans cet article, nous apprendrons pourquoi et comment est né le concept Ajax, son utilisation avec PHP. Nous aborderons les avantages de la communication asynchrone et ses limites, les solutions apportées et sa mise en œuvre.

PRATIQUE

34 Créer son propre hébergement

Alexandre Bazzet

Nous verrons dans cet article comment créer un serveur d'hébergement web avec un minimum de sécurité. Dans une première partie nous installerons/configurerons un AMP (Apache + MySQL + PHP), puis nous installerons des logiciels tiers pour les serveurs FTP, DNS, etc.



POUR LES DÉBUTANTS

39 SQL : langage de définition des données

Cilia Mauro, Magali Contensin

De nombreuses applications Web (forums, galeries d'images, sites marchands, ...) reposent sur des bases de données. L'interaction avec la base de données, la création de la base et des tables, la manipulation des données, ainsi que le contrôle des droits d'accès aux données sont réalisés avec le langage SQL. Les bases de données sont très utilisées dans les applications Web. La création, l'interrogation et la manipulation des données de la base sont réalisées en SQL. Dans cette série d'articles vous apprendrez le langage SQL ainsi que les bases nécessaires pour communiquer avec une base de données à partir d'un script PHP.

SÉCURITÉ

49 Introduction à la sécurité web

Nicolas Turmeau

Un développeur quel qu'il soit peut réaliser de grands projets. Mais quelle chute fait-il lorsque le dit projet vient d'être détruit par une attaque ? De très haut, tellement haut que certains en abandonnent tout espoir. Dans cet article nous étudierons les bonnes pratiques les plus simples à avoir pour éviter au maximum cette situation.

OpenTBS 1.4

OpenTBS est un outil PHP qui vous permet de générer des fichiers *OpenOffice* et *MS Office* à partir de modèles que proposent ces logiciels. Il s'appuie sur le moteur *TinyButStrong* et utilise la librairie *zLib*. Les documents générés peuvent être produits sous forme d'un fichier enregistré sur le serveur, d'un téléchargement direct, ou même d'un chaîne binaire si besoin.

<http://www.tinybutstrong.com/fr/>

eZ Publish 4.4

La nouvelle version du CMS *eZ Publish* est disponible. Elle apporte de nombreuses nouveautés comme l'amélioration de l'interface d'administration pour faciliter la gestion de contenu, l'évolution de la partie édition des images avec la possibilité de modifier directement le contenu, le soutien du format HTML 5, des évolutions du côté de l'utilisateur... Par ailleurs, *eZ Systems* créatrice du CMS *eZ Publish* invite la communauté à participer au développement de celui-ci. C'est-à-dire la compagnie ouvre le processus de développement de son cœur.

<http://ez.no>

L'AFUP fédère la communauté PHP

Le rendez-vous du moment concerne le *Forum PHP*, qui se déroulera le 9 et 10 novembre 2010. Pour cet événement l'AFUP réunit de nombreux acteurs PHP (Rasmus Lerdorf, Zeev Suraski, Derrick Rethans, et Ilia Alshanetsky...), SQL avec la présence des créateurs de *SkySQL* (Kaj Arnö, Michael « Monty » Widenius). Et propose aussi un espace Projets PHP et beaucoup d'autres chose à découvrir

<http://www.afup.org/forumphp>

Magix CMS

Magix CMS passe sous licence Open Source, tout en gardant les fonctionnalités premières du CMS : basé sur le référencement. Cette version propose plus de fonctionnalités de base avec une interface plus simplifiée.

<http://www.cms-php.be>

Speed test mini PHP

Il s'agit d'une application multi-langage et qui a été libérée pour vous permettre de tester votre bande passante. Celle-ci fonctionne avec différents langages et peut être installée sur votre propre serveur sous la forme d'un nouveau service supplémentaire.

<http://www.speedtest.net/mini.php>

SabreDAV 1.3.0

SabreDAV vous permet d'ajouter facilement WebDAV à une application PHP. En déployant *WebDAV* (ou *SabreDAV* précisément) vous pouvez utiliser votre application web comme s'il s'agissait d'un système de fichiers pour les utilisateurs. Cette nouvelle version apporte une évolution du cache et de la classe.

<http://code.google.com/p/sabredav/>

SkySQL Ab, l'alternative MySQL à Oracle

En ce mois d'octobre, les yeux se sont penchés sur le secteur des bases de données et principalement MySQL. SkySQL Ab est la nouvelle société dont le but principal est de s'engager dans l'Open Source. Elle est fondée et financée par 100% d'anciens cadres et investisseurs de MySQL, et est en train de devenir le nouveau centre de l'écosystème MySQL. L'offre de *SkySQL Ab* repose sur une alternative de logiciels, services et assistance pour la base de données MySQL, c'est à dire :



- *MySQL Server*, est une offre alternative parfaitement fiable, sécurisée et économique pour les utilisateurs de solutions de base de données propriétaires. Le serveur MySQL peut être utilisé pour toutes les applications de base de données en production.
- *Maria DB Server*, est une alternative compatible MySQL, qui peut aussi intégralement remplacer le serveur de base de données MySQL. En plus, de nouveaux drivers apparaissent pour être utilisés avec Zend Framework, Drupal...
- *SkySQL Enterprise Monitor*, est un administrateur de base de données MySQL dans une boîte.
- *SkySQL Visual Editor*, est un éditeur de requête SQL et permet aux administrateurs de base de données de déployer plus rapidement et efficacement leurs serveurs de base de données MySQL.
- L'assistance technique SkySQL permet aux clients de maximiser la disponibilité et la performance de leurs applications de base de données. SkySQL offre à la fois une assistance de résolution des problèmes (erreurs générales, coupures de service et usage général) et une assistance consultative (pour aider à l'amélioration d'un déploiement spécifique).

Enfin, Kaj Arnö, co-fondateur de *SkySQL Ab* et Monty Widenius, créateur de MySQL et fondateur de *Monty Program Ab*, seront présents en exclusivité mondiale au Forum PHP 2010

<http://www.skysql.com/>

Rédaction des actualités :
Christophe Villeneuve



Forum PHP Paris 2010

9 et 10 Novembre 2010

15 ans de PHP/
10 ans de l'AFUP

www.afup.org/forumphp

Le seul rendez vous PHP !

php

9 & 10 novembre 2010

Cités des Sciences et de l'Industrie
30, avenue Corentin-Cariou
75019 Paris

afup

Solution de stockage basée sur ZFS et Ubuntu

La virtualisation de PC a amené de nouveaux besoins. Alors que la concentration de serveurs fonctionnels virtualisés par serveur physique de virtualisation s'accroît, la demande en termes de capacité de stockage et de fiabilité des équipements s'accroît également.

Cet article explique :

- Comment mettre en œuvre une solution de stockage de très grande capacité et sécurisée accessible en NFS ou iSCSI depuis le réseau local (exemple: serveur de virtualisation).

Ce qu'il faut savoir :

- Cet article demande une bonne connaissance de l'architecture de stockage des PC et d'un PC disposant d'au moins trois disques durs.

Les solutions professionnelles de virtualisation proposent des options de haute disponibilité entre serveurs de virtualisation mais, à la condition que, les disques durs des machines virtuelles soient stockés dans un espace accessible à tous les serveurs de votre ferme de virtualisation. Cet espace de stockage partagé doit être suffisamment volumineux pour pouvoir héberger l'intégralité des disques durs de toutes les machines virtuelles. Il doit pouvoir accroître sa capacité de stockage dès l'apparition de nouveaux besoins. Cet espace doit être extrêmement fiable car, en cas de panne, c'est cette fois l'intégralité des machines de tous les serveurs de virtualisation qui seront en panne. Les performances de cet espace de stockage devront être suffisantes pour assurer les demandes en I/O de tous les serveurs virtualisés. La mise en œuvre de la baie doit évidemment être économique afin de ne pas remettre en cause les avantages financiers de la virtualisation de serveurs.

Dans un environnement idéal, deux serveurs de virtualisation seront reliés au travers de deux switches à deux baies de stockages redondantes :

§

Dans le cadre de cet article, nous nous contenterons d'un serveur de virtualisation Vmware ESXi¹ et d'un baie de stockage équipé de l'OS *Nexenta*².

Lors d'un précédent article, j'avais déjà décris une architecture un peu similaire en utilisant la distribution FreeNAS³; laquelle simplifiait grandement la gestion des disques, des volumes de stockage et de la redondance des disques et l'administration de la baie au travers d'une interface WEB⁴.

Dans un autre article⁵, j'ai également présenté le ZFS⁶ qui est un système de fichier développé par SUN pour SOLARIS. Pour des raisons de licences, ZFS ne peut être intégré au noyau Linux. ZFS présente de très nombreux avantages en termes de protection des données, volumétrique et de performance⁷.

1. ZFS mémorise la somme de contrôle des données stockées sur le disque sous la forme d'un code en 256 bits. Ce code est vérifié à chaque opération de sorte que toute corruption des données par un matériel défaillant par exemple ne puisque passer inaperçu (DMA, cache de disque dur...).

2. ZFS utilise le *copy on write* ce qui signifie qu'un bloc de données n'est jamais modifié directement sur le disque. En cas de modification, les écritures sont réalisées dans un espace indépendant des données initiales et si toutes les sommes de contrôle coïncident, alors seulement, les nouvelles données sont validées.

³ <http://freenas.org>

⁴ 2009/09 – Stockage déporté avec FreeNAS: Vmware ESXi, iSCSI et NFS

⁵ 2009/09 – Gestion avancée des disques sous Linux: Raid, LVM & ZFS

⁶ <http://www.sun.com/software/solaris/zfs.jsp>

⁷ http://opensolaris.org/os/community/zfs/docs/zfs_last.pdf

Tableau 1. Comparaison de quelques dossiers systèmes entre Linux et OpenSolaris

Linux	OpenSolaris
/home	/export/home
/var/log	/usr/adm, /var/adm, /var/log
/tmp	/var/tmp
/sys	/devices
/dev	/dev
/lib/modules/foo/*	/kernel/drv/*
/boot/grub	/rpool/boot/grub

Tableau 2. Comparaison des quelques commandes système entre Linux et OpenSolaris

sudo	pfexec
apt-cache search	pkg search -r foo
apt-get install foo	pkg install SUNWfoo
apt-get dist-upgrade	pkg image-update
lsmod	modinfo
insmod	modload
rmmmod	modunload
top	prstat
free	vmstat
cat /proc/cpuinfo	psrinfo -v
ifconfig	ifconfig -a
parted	format

3.ZFS inclut un système de snapshot ; c'est à dire que l'on peut accéder aux données telles quelles étaient à un instant donné et ceci même si des changements ont été opérés ensuite.

4.ZFS organise les disques sous la forme de *pool*. Ces équipes de disques peuvent être redimensionnées et le niveau de redondance entre disques est ajustable (STRIPPING: pas de redondance; tous les disques sont fusionnés – MIRROR: deux disques ont un contenu identiques actualisés en temps réel – RAIDZ/RAIDZ2: le pool peut perdre un ou deux disques sans pertes de données). La répartition des données entre les disques est optimisée de façon totalement automatique en fonction des performances de chaque disque. On peut par exemple mélanger des disques SSD pour la vitesse et SATA pour la capacité.

5.ZFS est un système de fichier 128 bits. Ne cherchez pas à compter quelle capacité de disque cela fait; c'est plus que vous n'en aurez jamais besoin !

6.ZFS peut contrôler manuellement l'état des disques (SCRUB soit l'équivalent d'un *chkdsk*) en temps réel – sans dégradation de performances. Les réparations sont effectuées à la volée et inscrites dans un journal consultable par l'administrateur.

7.Le jeu de commandes administratives utilisées par ZFS est très simple à appréhender comme va le démontrer cet article.

8.ZFS est un système qui évolue puisque sont prévus sous peu le cryptage et la déduplication des blocs de données en temps réel.

Juste à titre d'information, ZFS-FUSE (<http://zfs-on-fuse.blogspot.com/>) propose aujourd'hui un portage sous GNU/Linux de ZFS en attendant la réponse libre à ZFS: BTRFS⁸.

Car en effet, ZFS n'est pas un système de fichier libre. Il est disponible dans le noyau de OpenSolaris⁹, donc gratuit, mais son code source n'est pas disponible.

Qu'à cela ne tienne me direz-vous ! Il est vrai que les caractéristiques de ce système ont de quoi lever bien des objections à un logiciel non libre.

Le problème est que passez à OpenSolaris, c'est aussi réapprendre tout un jeu de commande durement acquit sous GNU Linux.

Je vous donne un petit aperçu des différences entre les chemins systèmes et les jeux de commandes:

Heureusement, la communauté du libre a entrepris la tâche de migrer les outils GNU vers OpenSolaris. Ceci prend la forme d'une distribution hybride entre OpenSolaris et Ubuntu Hardy 8.04 : NEXENTA. Nexenta existe en trois versions :

⁸ <http://fr.wikipedia.org/wiki/Btrfs>

⁹ <http://fr.wikipedia.org/wiki/OpenSolaris>

Projets

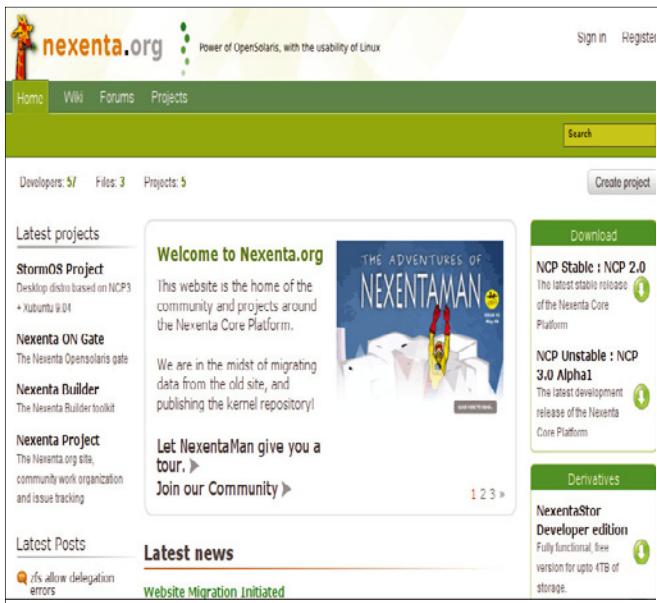


Figure 1. Site de Nexenta Core



Figure 2. Site de NexentaStor

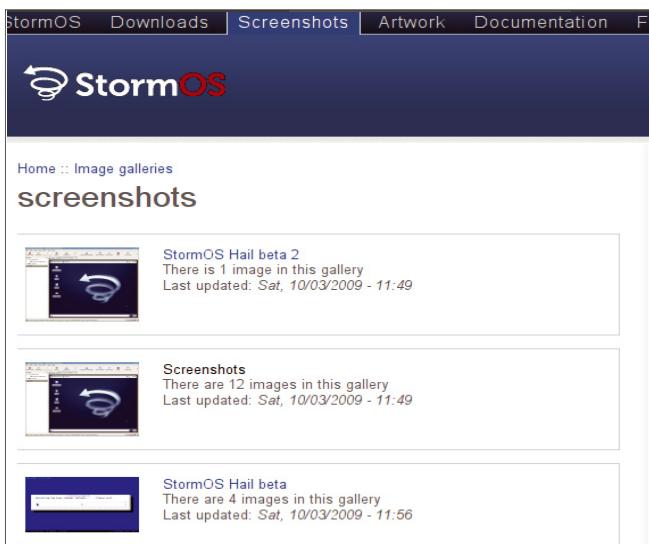


Figure 3. Site de StormOS

- **NexentaCore** : il s'agit d'une distribution en ligne de commande (<http://www.nexenta.org/>). C'est cette version gratuite (license CDDL - <http://www.sun.com/cddl/cddl.html>) que nous utiliserons pour mettre en œuvre un serveur de stockage réseau NAS utilisant la technologie ZFS.
- **NexentaStor** : Il s'agit d'une distribution qui va décliner le PC où elle est installée pour réaliser une baie de stockage (<http://www.nexenta.com/corp/>). Cette version est entièrement administrable à travers une interface WEB et dispose d'extensions dédiées (agent Vmware, client de backup Windows...). Cette version de Nexenta est payante mais dispose d'un support professionnel. Sachez qu'il existe une version *développeur* gratuite jusqu'à 4To de données utiles.
- **StormOS** : Cette distribution propose une station de travail gratuite utilisant l'environnement de fenêtre XFCE (<http://stormos.org>). Nexenta fonctionne sur les architectures 32bits et 64bits.

Installation de Nexenta

Dans ce chapitre, nous allons présenter les différentes étapes d'installation de la distribution Nexenta Core 3. Cette étape représente la grande différence avec une installation OpenSolaris typique. Par exemple, Nexenta permet l'installation de l'OS sur un miroir de disques. Nexenta installe et configure automatiquement le service de console à distance sécurisé SSH. Nexenta active par défaut le service de partage iSCSI (*target*). OpenSolaris aurait nécessité des étapes de post-installation plutôt laborieuses

Dès que vous avez téléchargé puis gravé le CD d'installation de Nexenta Core, démarrez votre PC depuis ce Live-CD. Ensuite, choisissez la langue d'installation. Choisissez la zone géographique de votre PC. Choisissez le pays de localisation du PC. Vérifiez et validez l'ensemble de ces paramètres.

Nexenta détecte ensuite l'intégralité des disques présents dans votre PC. Comme vous pouvez le constater, les conventions de dénomination des disques (C1t0d0) sont très différentes de ce que l'on trouvera sous Linux (/dev/sda1). Par exemple, C1t0d0 est le premier disque de la première nappe du contrôleur 1. Je vous invite à noter précisément ces informations dès maintenant car, si vous avez beaucoup de disques et/ou de contrôleurs, elles vous seront bien utiles lors de la création des *pool* de disques.

Dès maintenant, comme je le mentionnais plus tôt dans cet article, vous pouvez sélectionner deux ou trois disques afin d'installer l'OS Nexenta sur un *Mirror* (deux disques) ou un *raidz* (trois disques – équivalent à un RAID5). Validez le fait que vous êtes bien d'accord pour formater les disques précédemment sélectionnés. Il est temps de définir le mot de passe de l'utilisateur *root*. Attention, bien que vous ayez choisi une installa-

tion en français, le clavier est resté en anglais (*bug* !?). Confirmez que vous avez bien fini de définir le mot de passe. Saisissez maintenant le nom d'un nouvel utilisateur sans priviléges système. La procédure est identique à celle de l'utilisateur *root* à l'exception que vous devez définir l'identifiant. Confirmez les paramètres de création du nouvel utilisateur. Maintenant, la séquence d'installation de NexentaCore va définir les caractéristiques réseaux de votre distribution. Indiquez tout d'abord le nom et le domaine auquel appartient ce PC. Confirmez les informations saisies.

L'installateur va maintenant détecter toutes les cartes réseaux installées. Attention à veiller à ce que votre matériel entre bien dans la grille de compatibilité de Solaris. Sun propose d'ailleurs sur un site une applet JAVA qui contrôle la compatibilité avec Solaris du matériel sur lequel elle est exécutée¹⁰. Dans le cadre de cet article, le PC ne contient qu'une carte réseau fonctionnant au Confirmation avant validation du mot de passe de *root* gigabit. Bien évidemment, en production, vous avez tout intérêt à multiplier le nombre de cartes afin d'ajouter équilibrage de charge et autre disponibilité à votre baie de stockage.

Vous devez décider si la carte réseau se verra attribuer une adresse IP par le serveur DHCP de votre réseau ou bien si vous souhaitez lui affecter une adresse IP statique. Il est préférable de choisir une IP statique, préservant le fonctionnement de la baie même en cas de défaillance du serveur DHCP (lequel pourrait très bien être virtualisé et stocké dans la baie, par exemple).

Saisissez l'adresse et le masque de sous réseau que vous souhaitez pour la baie. Si votre réseau n'utilise pas IPV6, il est inutile de demander le réglage suivant. Validez que vous souhaitez bien appliquer les paramètres précédents. Vous devez alors indiquer la passerelle réseau qui sera appliquée à toutes les cartes de la baie. Si vous n'indiquez pas de passerelle, les mises à jour par internet de NexentaCore ne seront pas possibles. Saisissez l'adresse IP de la passerelle. Vous devez aussi préciser l'adresse d'un serveur de nom DNS afin de terminer le réglage de la connexion de la baie à Internet. Saisissez l'adresse IP du ou des serveurs de noms de votre réseau.

Les réglages sont terminés ! L'installation de la distribution va pouvoir commencer. Il ne vous reste plus qu'à patienter. Sachez qu'à configuration matérielle égale, NexentaCore (ou OpenSolaris) est moins rapides que Linux. ZFS est également très gourmand de mémoire vive.

Une fois l'installation terminée, relancez le PC. Retirez le CDROM et, au bout de quelques secondes, vous vous trouverez devant l'invitation à saisir vos identifiants pour accéder à la console d'administration.

¹⁰ http://www.sun.com/bigadmin/hcl/hcts/device_detect.jsp

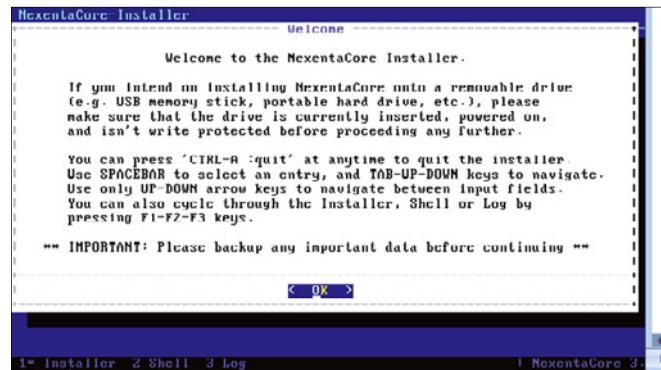


Figure 4. Écran de démarrage du Live-CD NexentaCore



Figure 5. Menu de choix de la langue d'installation

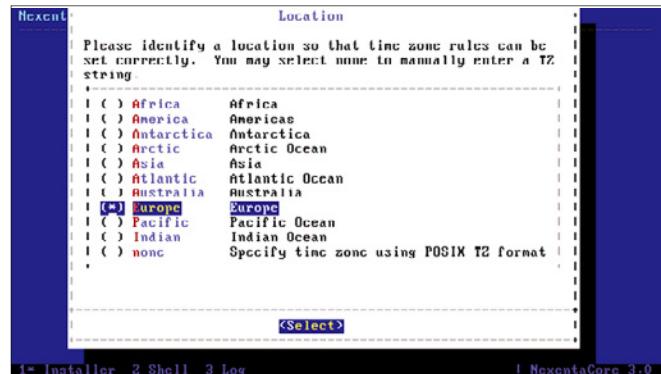


Figure 6. Menu de choix de la zone géographique

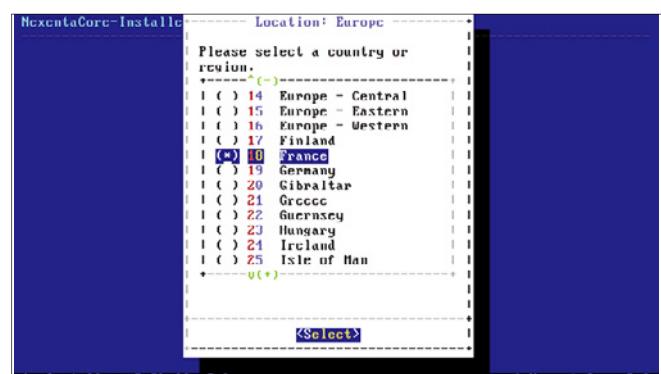


Figure 7. Menu de choix du pays du PC

Connectez-vous avec l'identifiant de l'utilisateur non privilégié et validez le fonctionnement de la couche réseau.

Pour cela, tapez la commande suivante pour visualisez l'état de vos cartes réseau (Figure 27).

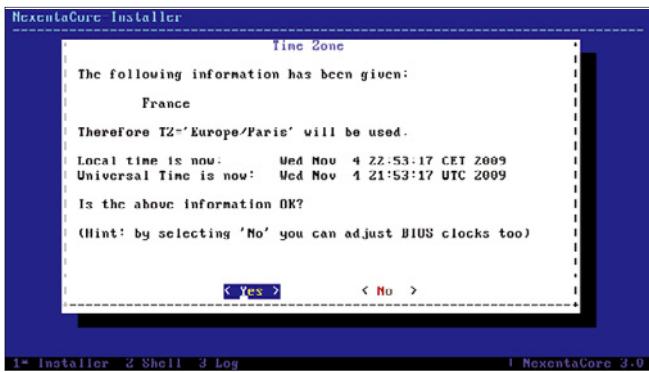


Figure 8. Validation des paramètres géographiques

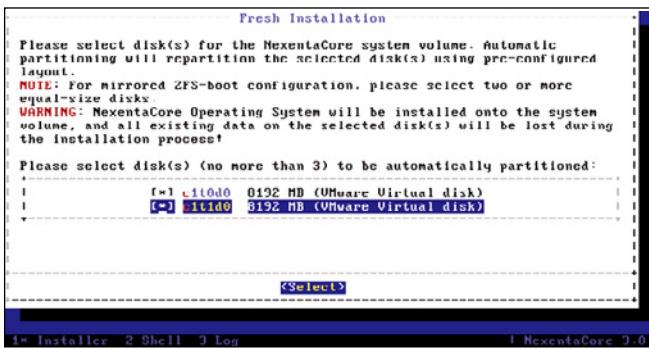


Figure 9. Choix des disques où sera stocké l'OS NexentaCore

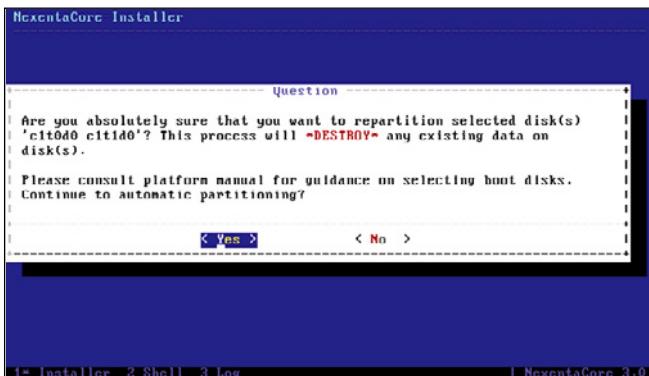


Figure 10. Dernière validation avant formatage des disques

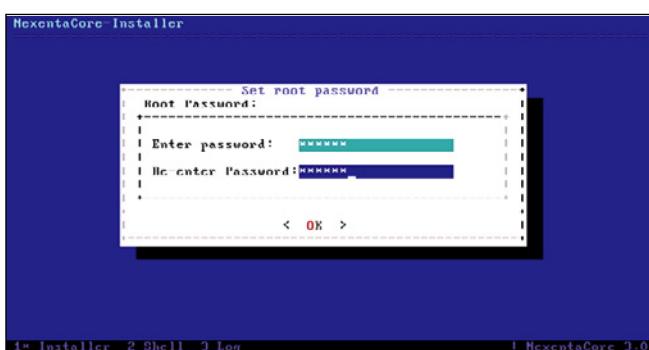


Figure 11. Définition du mot de passe de root

```
dla star-link
LINK CLASS MTU STATE BRIDGE OVER
bnx0 phys 1500 up -- -
```

Pour voir le détail des réglages de vos cartes réseau, tapez la commande (Figure 28).

```
ifconfig -a
```

Si vous souhaitez modifier à posteriori les caractéristiques réseaux de votre PC, modifiez le fichier de configuration `/etc/hostname.bn0` avec l'éditeur `nano`.

```
sudo nano /etc/hostname.bn0
```

Ce fichier contient une ligne qui ressemble à ceci :

```
192.168.1.38 netmask 255.255.255.0 broadcast
+ up
```

Si tout fonctionne correctement, vous pouvez lancer la mise à jour de la distribution avec la séquence de commande suivante :

```
sudo apt-get update
sudo apt-get upgrade
```

Vous noterez l'usage de `sudo` et de `apt-get` portés de Ubuntu vers NexentaCore (ou `aptitude` si vous préférez cet outil).

Si vous devez relancer la machine, alors saisissez la commande suivante:

```
sudo reboot -r
```

Création des pools de disques

Lors de son installation, NexentaCore a créé un pool de disque nommé `syspool` dans lequel il a placé les deux ou trois disques sélectionnés.

Vous pouvez visualiser ce *pool* en tapant la commande :

```
sudo zpool list
NAME      SIZE    USED  AVAIL  CAP
HEATH  ALTROOT
syspool  330G   1,13G   239G
0%      ONLINE -
```

L'OS utilise 1,3 Go du *pool* et celui-ci est en parfait état de marche (status `ONLINE`). Vous l'avez peut-être remarqué mais ZFS retourne parfois des valeurs étranges en ce qui concerne l'espace libre (*bug* ?!). Il ne faut pas oublier qu'un *pool* organisé en `raidz` utilise une partie de l'espace disque pour stocker des sommes de contrôle (tout le comme le ferait un RAID5). De plus, le système de fichier hébergé par ce *pool* peut être compressé en temps réel.

On peut visualiser l'organisation des disques au sein de ce *pool* avec la commande suivante:

```
sudo zpool status syspool
pool: syspool
state:      ONLINE
```

```

scrub: none requested
config:
        NAME          STATE
READ      WRITE  CHKSUM
        syspool
ONLINE 0      0      0
                mirror    ONLINE 0
                0      0
                c0t0d0s0
ONLINE 0      0      0
                c0t1d0s0
ONLINE 0      0      0

```

Les lignes ci-dessus signifient que *syspool* est composé de deux disques placés en *miroir* (RAID1). Tous les disques, le *miroir* et le *pool* fonctionnent parfaitement (ONLINE). Aucun contrôle d'état n'est en cours (scrub: none requested). Les 0 correspondent au Lecture (READ) / écriture (WRITE) réalisées sur chaque disques par seconde. Pour l'instant, il ne se passe donc pas grand chose. Pour information, *c0t0d0s0* fait référence à la première partition *s0* du premier disque *d0* de la première nappe *t0* du premier contrôleur *c0*.Création d'un pool de données

Vous allez maintenant créer un *pool* de disques destiné à accueillir vos données que l'on nommera *data*. Comme la machine de test possède deux disques de 500 Go chacun, vous allez créer un *pool* permettant d'exploiter cet espace en volume et en redondance. Nous allons configurer ce *pool* en *raidz*.

Pour cela, vous utilisez la commande suivante :

```

sudo zpool create data raidz c1t0d0 c1t1d0
c1t2d0 c1t3d0 c1t4d0 c1t5d0 c1t6d0 c1t7d0
c1t8d0 c1t9d0 c1t10d0 c1t11d0

```

Après quelques secondes seulement, le *pool* est disponible. On s'assure que le *pool* est bien fonctionnel avec la commande ci-dessous :

```

sudo zpool status data
pool: syspool
  state: ONLINE
  scrub: none requested
  config:
        NAME          STATE
READ      WRITE  CHKSUM
        data
ONLINE 0      0      0
                mirror    ONLINE 0
                0      0
                c0t0d0s0
ONLINE 0      0      0
                c0t1d0s0
ONLINE 0      0      0
                c0t1d0s0

```

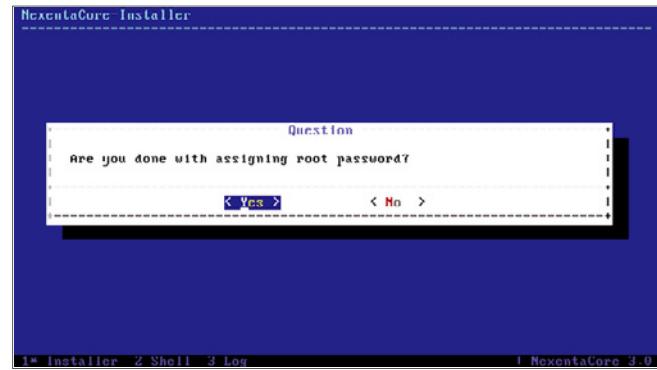


Figure 12. Confirmation avant validation du mot de passe de root

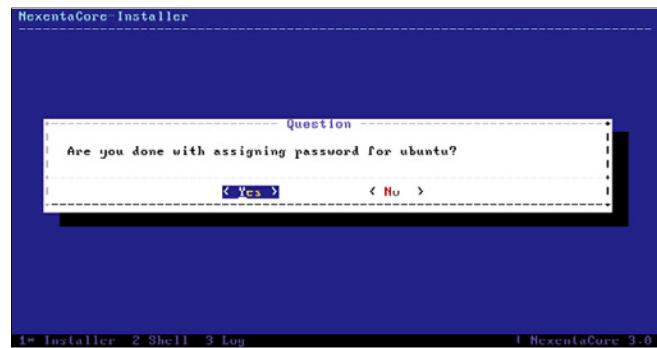


Figure 13. Confirmation avant validation du mot de passe de ubuntu

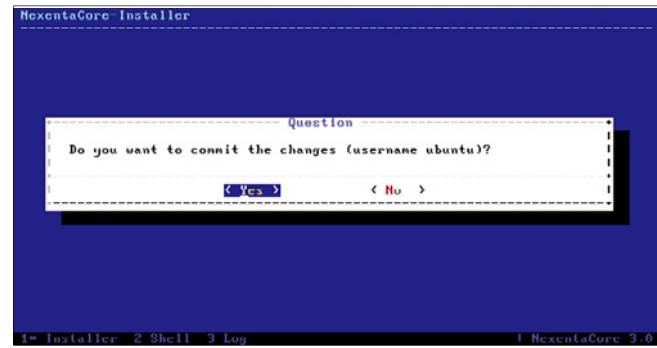


Figure 14. Validation avant création de l'utilisateur ubuntu.

```

ONLINE 0      0      0
                c0t1d0s0

```

Projets

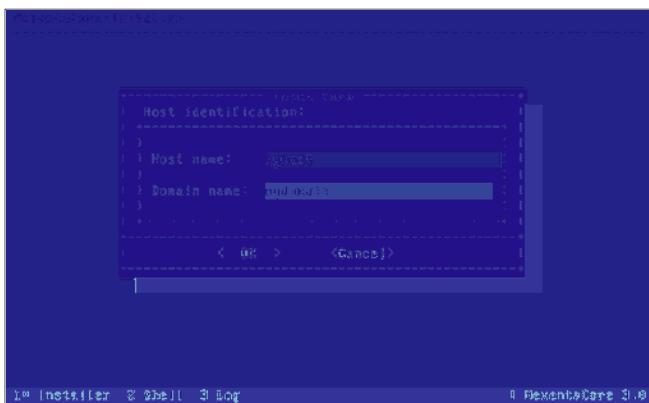


Figure 15. Définition du nom et du domaine du PC

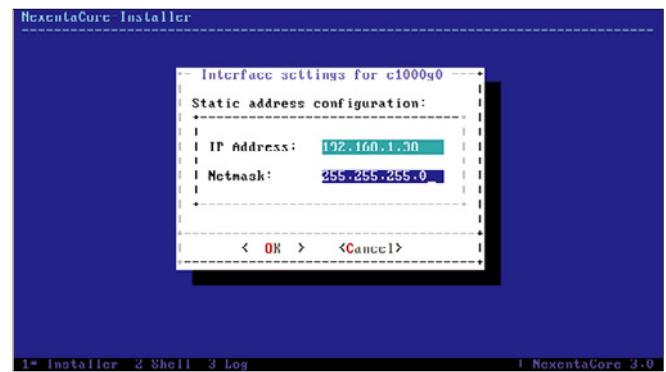


Figure 19. Saisie de l'adresse IP de la baie.

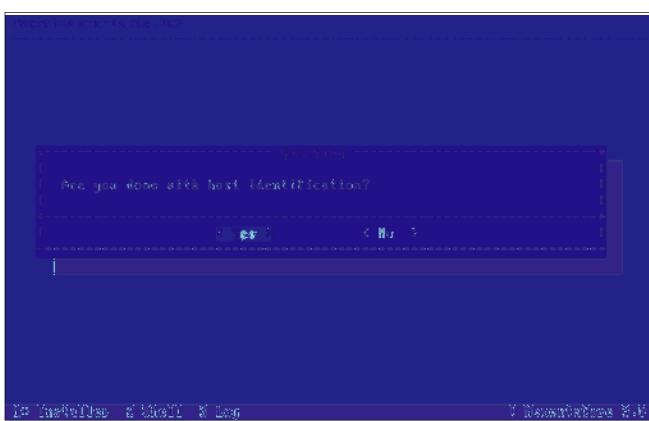


Figure 16. Validation avant configuration des paramètres réseau du PC

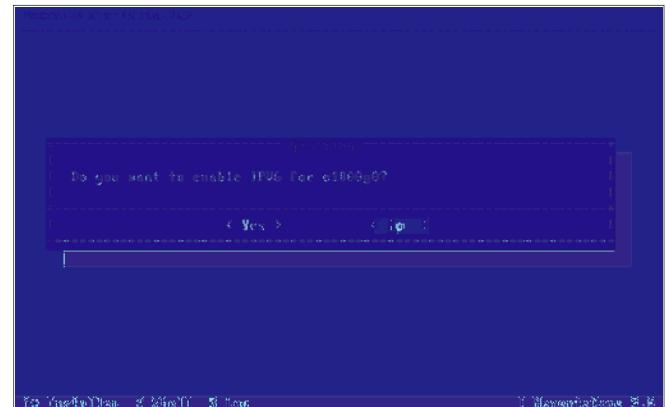


Figure 20. Choix de la configuration IPV6



Figure 17. Confirmation du lancement de la procédure de configuration de la carte réseau

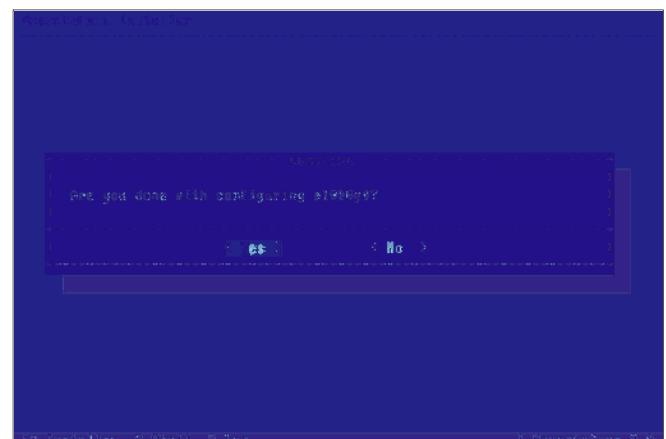


Figure 21. Validation avant application des réglages réseau.



Figure 18. Choix d'attribuer une adresse IP statique à la carte réseau de la baie.

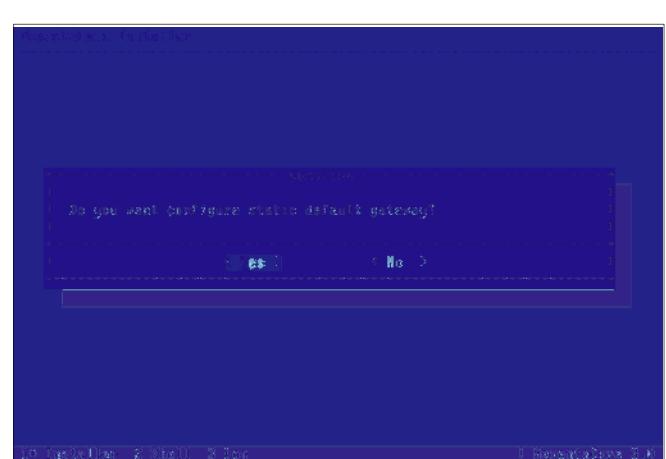


Figure 22. Choix de configuration de la passerelle.

De la même façon que précédemment, on contrôle la capacité avec la commande :

```
sudo zpool list
NAME      SIZE    USED   AVAIL  CAP
HEATH    ALTROOT
data      5.44T   3.21M   5.44T
0%        ONLINE -
```

Vous allez ensuite régler le *pool* afin qu'il répare automatiquement les problèmes qu'il pourra détecter. Pour cela, on utilise la commande :

sudo set zpool set autoreplace = on
dataautoreplace est une des nombreuses options disponibles pour un *pool* .. Si vous souhaitez faire apparaître la totalité des options disponibles et les valeurs associées, tapez la commande :

```
sudo sudo zpool get all data
```

Si vous souhaitez demander un contrôle manuel de l'état des disques, utilisez l'option *scrub* avec la commande *zpool* :

```
sudo zpool scrub data
```

La commande *zpool status data* vous permettra de suivre l'état de l'opération.

Vous pouvez rappeler l'historique de toutes les commandes réalisées sur un *pool* avec la commande :

```
sudo zpool history data
```

Enfin, l'option *iostat* vous permettra de suivre l'utilisation de votre *pool* en nombre de lecture/écriture et le taux de transfert par disques :

```
sudo zpool iostat -v data
```

Ajoutez *-s 1* pour rafraîchir l'affichage chaque seconde.

Accroître la taille d'un disque

Si l'espace venait à manquer, vous pouvez agrandir votre *pool* de disques sans perte d'activité. ZFS propose la commande *zpool add*. Ainsi, pour ajouter les disques *c0t12d0* et *c0t13d0* montés en miroir à *data*, tapez:

```
zpool add data mirror c0t12d0 c0t13d0
```

Création de dataset au sein d'un pool

Jusqu'à présent, vous avez fusionné tous les disques dans un disque virtuel de très grande capacité. Il vous reste à organiser cet espace. On ne peut pas réelle-

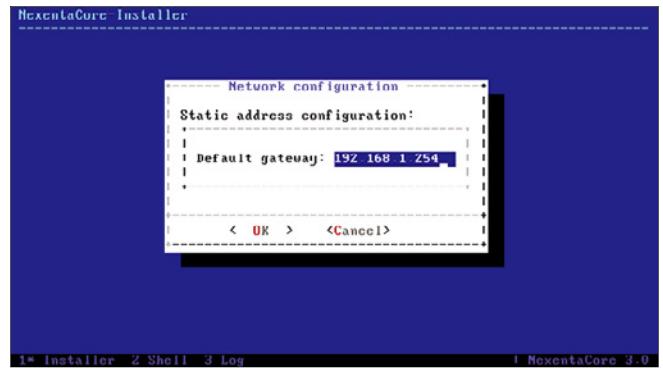


Figure 23. Saisie de la passerelle réseau



Figure 24. Choix de la configuration d'un serveur de nom

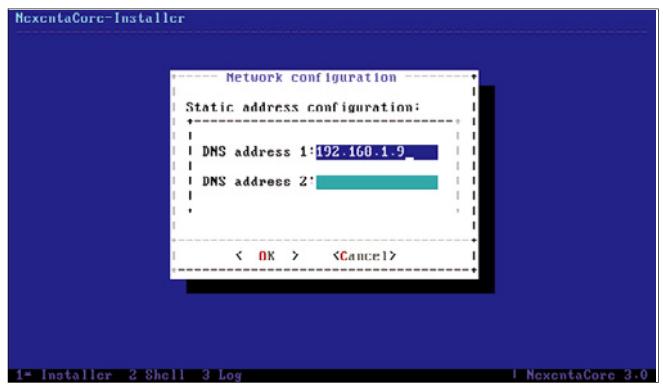


Figure 25. Saisie de l'adresse IP du serveur de nom DNS

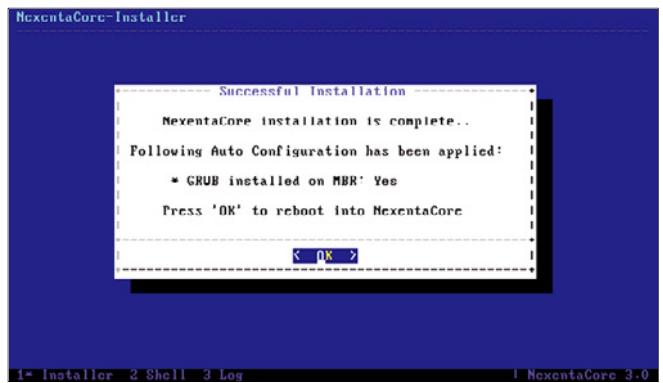


Figure 26. L'installation de NexentaCore est terminée!

ment parler de partitions où cela s'entend avec d'autres systèmes de fichiers sous Linux ou Windows. Il s'agit d'espaces dynamiques, ajustables à volonté et potentiellement limités uniquement par la taille du *pool* ; lequel



Figure 27. Etat des cartes réseau

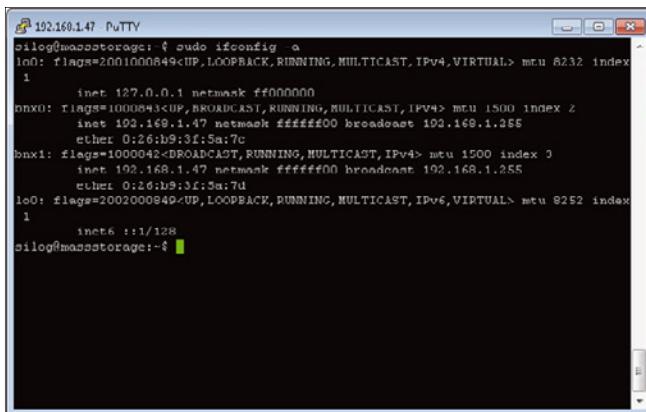


Figure 28. Configuration de la carte réseau

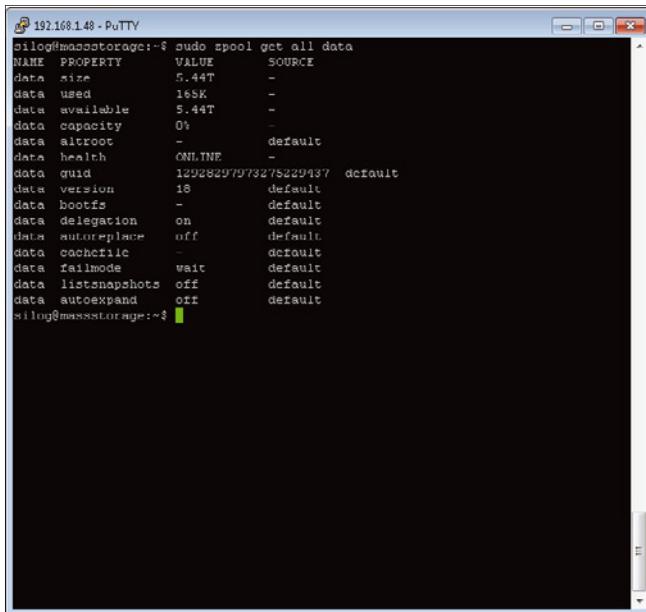


Figure 29. Listing des options du zpool

est également extensible. Ces espaces sont appelés des *datasets*.

Vous allez maintenant créer un premier *dataset* nommé *nfs-share* dans le *pool* nommé *data*

```
sudo zfs create data/nfs-share
```

La commande `zfs list` affiche l'intégralité des partitions du disque virtuel.

```
sudo zfs list
```

NAME	USED	AVAIL	REFER
MOUNTPOINT			
data/nfs-share	5.44T	3.21M	
/data/nfs-share			

Vous noterez que le contenu du *dataset* est directement accessible depuis le point de montage `/data/nfs-share`.

La capacité maximale du contenu de ce *dataset* est celle du *pool data* soit 5To.

Gestion de la taille des partitions

Les partitions ZFS ne sont pas limitées en taille individuellement. Ceci signifie que toutes les partitions partagent la totalité du disque. L'espace est alloué en fonction des besoins. Toutefois, il peut être nécessaire de limiter la taille d'une partition.

Pour cela, vous pouvez utiliser la commande `zfs set quota`. Dans l'exemple ci-dessous, le data `data/nfs-share` sera limité à une taille de 10 Go.

```
zfs set quota=10G data/nfs-share
```

La commande `zfs get quota` permet d'afficher les quotas attribués à une partition.

```
zfs get quota data/nfs-share
```

Inversement, il peut être utile de réserver un minimum d'espace pour une partition et ainsi s'assurer que l'espace ne viendra pas à manquer le jour où il sera utilisé. Cette opération peut être réalisée avec la commande `zfs set reservation`. Dans l'exemple ci-dessous, la partition `data/nfs-share` se voit réserver 1Go sur le disque.

```
zfs set reservation=1G data/nfs-share
```

Tapez la commande `zfs get all data/nfs-share` pour s'assurer que les modifications ont bien été appliquées (Listing 1).

Au passage, vous noterez que la liste des options est assez longue. On y trouve notamment l'option de compression des données en temps réel.

ZFS supporte la compression native des *datasets*. Vous pouvez commander la compression du *dataset* `nfs-share` avec la commande :

```
sudo zfs set compression=on data/nfs-share
```

Vous pourrez facilement atteindre un taux de compression de 1.4 sans dégradation notable des performances.

Depuis fin décembre 2009, un *dataset* dispose également d'une option de *déduplication de données en ligne*. Contrairement à la compression, la déduplication factorise la réduction de volume en ne stockant qu'une fois un bloc de donnée déjà présent dans un *dataset*.

Suppression d'une partition

La suppression du *dataset* *nfs-share* du *pool* nommé *data* est aussi simple que :

```
sudo zpool destroy data/nfs-share
```

Partage d'un dataset en NFS

Si vous souhaitez partager tout le contenu du *dataset* *nfs-share* au travers du protocole NFS, il vous suffira de taper la commande :

```
sudo zfs set sharenfs=anon=0 data/nfs-share
```

La séquence *sharenfs=anon=0* nous affranchi de tout problème de lecture/écriture sur le partage lié à des droits d'utilisation du "dataset". Si vous aviez saisi *sharenfs=on*, une authentification avec un compte utilisateur NexentaCore valide serait requis pour écrire dans le *dataset*. Si votre baie dispose de plusieurs cartes réseau, le partage est disponible depuis chacune des adresses IP. Vous pouvez équilibrer la charge entre les deux cartes réseaux configurant vos serveurs pour qu'ils s'adressent à la baie au travers des multiples cartes.

Partage d'un dataset en iSCSI

Si vous souhaitez partager un *dataset* au travers du protocole ISCSI¹¹, sa taille doit impérativement être figée. Il faut s'imaginer que le *dataset* est en réalité un fichier stocké dans le *pool* et que c'est le contenu de ce fichier qui est présenté au réseau au travers de ISCSI.

La création d'un *dataset* de 1To s'effectue avec la commande suivante :

```
sudo zfs create -V 1T data/iscsi-share
```

On partage ensuite ce *dataset* avec la commande :

```
sudo zfs set shareiscsi=on data/iscsi-share
```

Il faut bien avoir en tête que si ZFS ne connaît pas de limitations pratiques pour la taille des *pool* et des *datasets*, ce n'est pas le cas des autres systèmes d'exploitation. Aussi, dans la pratique, il est inutile d'exporter un *dataset* de plus de 2To car il ne pourra pas être exploité par d'autres systèmes que Solaris.

¹¹ http://fr.wikipedia.org/wiki/Internet_Small_Computer_System_Interface

pool	capacity	operations	bandwidth
	used avail	read write	read write
data	168K 5.44T	0 12	429 20.9K
data	168K 5.44T	0 0	0 0
data	168K 5.44T	0 0	0 0
data	160K 5.44T	0 0	0 0

Figure 30. Analyse de l'usage des disques en temps réel

pool	capacity	operations	bandwidth
	used avail	read write	read write
data	168K 5.44T	0 11	382 18.6K
raidz1	168K 5.44T	0 11	382 18.6K
c1t1d0	- -	0 5	21.2K 73.5K
c1t2d0	- -	0 4	21.3K 73.1K
c1t3d0	- -	0 5	21.3K 73.4K
c1t4d0	- -	0 4	21.2K 73.1K
c1t5d0	- -	0 5	21.2K 73.5K
c1t6d0	- -	0 4	21.3K 73.1K
c1t7d0	- -	0 5	21.3K 73.5K
c1t8d0	- -	0 4	21.2K 73.2K
c1t9d0	- -	0 5	21.2K 73.5K
c1t10d0	- -	0 4	21.0K 70.0K
c1t11d0	- -	0 5	21.3K 73.5K
c1t12d0	- -	0 4	21.2K 73.2K

Figure 31. Usage d'un pool en temps réel

TARGET NAME	STATE	SESSIONS
ign.1986-03.com.sun:02:341f332d-8f40-625e-fcc8-c0067eb246d6	online	0

Figure 32. création d'un partage ISCSI

Pour valider que le partage est bien actif et connaître son identifiant de partage, tapez la commande :

```
sudo itadm list-target
```

Listing 1.

NAME	PROPERTY	VALUE	SOURCE
data/nfs-share	type	filesystem	-
data/nfs-share	creation	ven mai 15 12:44 2009	-
data/nfs-share	used	24,0K	-
data/nfs-share	available	10,0G	-
data/nfs-share	referenced	24,0K	-
data/nfs-share	compressratio	1.00x	-
data/nfs-share	mounted	no	-
data/nfs-share	quota	10G	local
data/nfs-share	reservation	1G	local
data/nfs-share	recordsize	128K	default
data/nfs-share	mountpoint	/data/nfs-share	local
data/nfs-share	sharenfs	off	default
data/nfs-share	checksum	on	default
data/nfs-share	compression	off	default
data/nfs-share	atime	on	default
data/nfs-share	devices	on	default
data/nfs-share	exec	on	default
data/nfs-share	setuid	on	default
data/nfs-share	readonly	off	default
data/nfs-share	zoned	off	default
data/nfs-share	snapdir	hidden	default
data/nfs-share	aclmode	groupmask	default
data/nfs-share	aclinherit	restricted	default
data/nfs-share	camount	on	default
data/nfs-share	shareiscsi	off	default
data/nfs-share	xattr	on	default
data/nfs-share	copies	1	default
data/nfs-share	version	3	-
data/nfs-share	utf8only	off	-
data/nfs-share	normalization	none	-
data/nfs-share	casesensitivity	sensitive	-
data/nfs-share	vscan	off	default
data/nfs-share	nbmand	off	default
data/nfs-share	sharesmb	off	default
data/nfs-share	refquota	none	default
data/nfs-share	refreservation	none	default
data/nfs-share	primarycache	all	default
data/nfs-share	secondarycache	all	default
data/nfs-share	usedbysnapshots	0	-
data/nfs-share	usedbydataset	24,0K	-
data/nfs-share	usedbychildren	0	-
data/nfs-share	usedbyrefreservation	0	-

Si votre baie dispose de plusieurs cartes réseau, le partage est disponible depuis chacune des adresses IP. Vous pouvez équilibrer la charge entre les deux cartes réseaux configurant vos serveurs pour qu'ils s'adressent à la baie au travers des multiples cartes.

Configurer Vmware ESXi pour utiliser NexentaCore

Maintenant que NexentaCore est fonctionnel et qu'il exporte vos disques vers le réseau au travers des protocoles NFS et iSCSI, il reste à configurer Vmware ESXi afin d'utiliser ces espaces de stockage distants.

La configuration des *datastore* est réalisée depuis le menu *configuration*.

Création d'un datastore NFS

La création d'un *datastore* NFS est extrêmement simple. Depuis l'interface d'administration de Vmware ESXi, cliquez sur l'option *storage* dans le menu de gauche puis cliquez sur le lien *add storage* en haut à droite. Un assistant va vous guider dans la création du *datastore*. Choisissez tout d'abord l'assistant pour un *network file System*. Cliquez sur le bouton *NEXT*.

Vous devez saisir l'adresse IP (192.168.1.38) de votre NexentaCore dans le champ *server* et nom de votre partage NFS (*data/nfs-share*) dans le champ *folder*. Nommez votre *datastore* dans le champ *datastore name*. Cliquez ensuite sur *next*. Votre *datastore* est prêt et fonctionnel.

Il ne vous reste plus qu'à cliquer sur le bouton *finish*. Désormais, lorsque vous créez une machine virtuelle, il vous sera proposé d'entreposer ces disques *virtuels* dans le *datastore* distant *NFS datastore*. Le reste de l'utilisation de ESXi est totalement inchangée.

Création d'un datastore iSCSI

La création d'un *datastore* iSCSI est légèrement plus complexe car elle requiert deux étapes :

- l'établissement d'un lien entre ESXi et le serveur iSCSI (c'est à dire NexentaCore),
- la création du *datastore* à proprement parler.

La première étape est accessible depuis le menu de droite nommé *storage adapters*. Sélectionnez ensuite la ligne en dessous de *iscsi storage adapters*. Cliquez

enfin sur le lien *properties* en haut à droite de la zone *details*. Assurez-vous que le protocole iSCSI est bien activé sur votre serveur ESXi en cliquant sur l'onglet *général*. Dans l'onglet *dynamic discovery* du formulaire de paramétrage du client iSCSI de ESXi, cliquez sur le bouton *add*.

Indiquez dans la zone *iSCSI server*, l'adresse IP du serveur NexentaCore (192.168.1.38).

Fermez cet assistant. ESXi va vous proposer de parcourir tous les serveurs iSCSI définis afin de détecter les disques disponibles. Acceptez mais sachez que cette étape nécessaire peut prendre plusieurs minutes.

Retourner dans le menu droit *storage* et relancer l'assistant de création de *datastore* en cliquant sur le lien *add storage* en haut à droite.

Cette fois ci, choisissez l'option *disk/lun*.

L'assistant va vous proposer la liste des disques détectés sur l'opération de *rescan* mentionnée plus haut dans cet article. Sélectionnez le disque que vous souhaitez utiliser comme *datastore*. Le champ *san identifier* se termine par le nom de l'*extend* utilisé dans NexentaCore (commande = `sudo itadm list-target`). L'étape suivante vous rappelle que le disque est vierge et qu'il va être formaté. Saisissez le nom que vous souhaitez donner à ce *datastore*. Les paramètres du formatage du disque doivent être ajustés en fonction de la taille maximale du disque de machine virtuelle qui y sera déposé. Cliquez sur *next*. Le *datastore* iSCSI est désormais opérationnel.

Il ne vous reste plus qu'à cliquer sur le bouton *finish*. Désormais, lorsque vous créerez une machine virtuelle, il vous sera proposé d'entreposer ces disques *virtuelles* dans le *datastore* distant *NFS datastore*. Le reste de l'utilisation de ESXI est totalement inchangé.

Si vous souhaitez créer d'un *datastore* de plus de 2To, il faut suffit de créer plusieurs *datasets* partagés en iSCSI sur NexentaCore puis de les fusionner en cliquant dans la zone *extents* disponible avec un clic sur le bouton droit de la souris sur le *datastore* iSCSI.

L'expérience montre que les accès à un disque virtuel stocké dans un *datastore* iSCSI sont 5 à 10 fois plus rapide que pour un *datastore* NFS, surtout en charge. Toutefois, cette mesure dépend grandement de l'utilisation qui est faite de la machine virtuelle.

Conclusion

Au terme de cet article, vous disposez désormais d'un baie de stockage de très haute performance mue par la distribution NexentaCore et portée par le système de fichier ZFS. Le choix entre OpenSolaris ou NexentaCore est purement pratique car NexentaCore permet à l'utilisateur de Ubuntu de retrouver une ligne de commande familière. Les puristes préféreront sûrement la version SUN OpenSolaris, gage de compatibilité et de rapidité de réplication des correctifs.

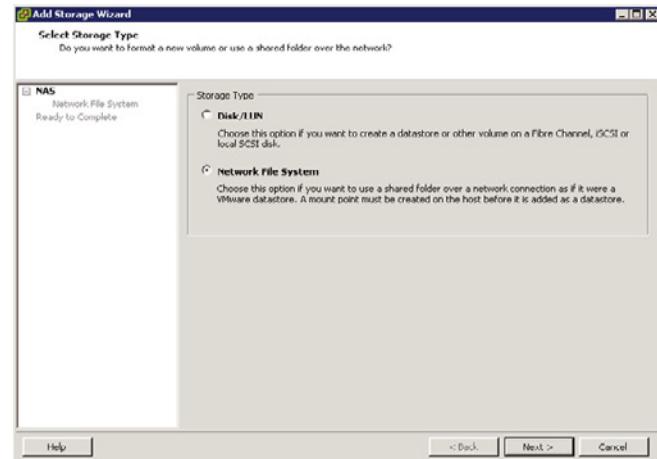


Figure 33. Choisir entre la création d'un datastore NFS ou iSCSI

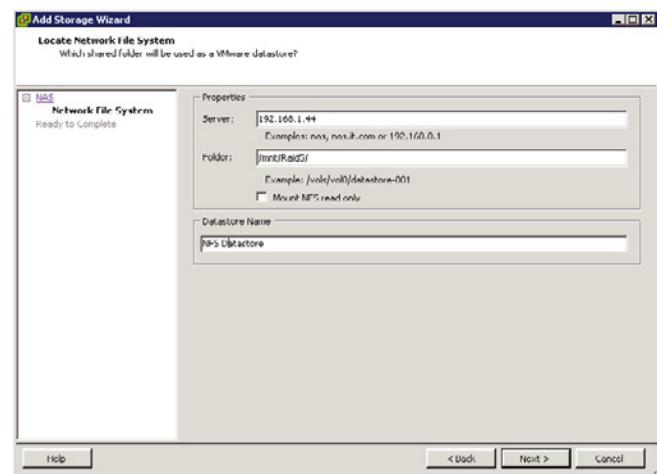


Figure 34. Ajout d'un datastore NFS

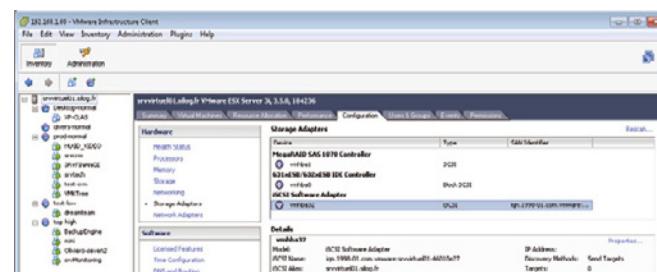


Figure 36. Recherche de partages iSCSI sur le réseau

De très nombreux aspects relatifs à l'OS (Haute disponibilité des cartes réseau, synchronisation entre baies...) et à ZFS (snapshot, clones, backup, disques de secours, réparation de pools....) n'ont pas été abordé dans le cadre de cet article mais devront être mis en œuvre avant la mise en exploitation d'une telle baie de stockage.

OLIVIER OLEJNICZAK

Responsable informatique de l'éditeur logiciel silog.fr.

Cofondateur du blog www.synergeek.fr.

Membre de Calvix.org.

AJAX et PHP

'Il n'y a pas au monde de pire malheur que la servitude!' Sophocle (extrait de Ajax). Dans cet article, nous apprendrons pourquoi et comment est né le concept Ajax, son utilisation avec PHP. Nous aborderons les avantages de la communication asynchrone et ses limites, les solutions apportées et sa mise en œuvre.

Cet article explique :

- Historique et description technique AJAX.
- Différentes implémentations de AJAX.
- Communication entre AJAX et PHP.
- Mise au point avec Firebug.

Ce qu'il faut savoir :

- Base de la communication HTTP.
- Bonne connaissance des fondements JavaScript.
- Bonne connaissance PHP 5.2.
- Notion du DOM et CSS.

AJAX en tant que réponse à une problématique. Description de l'évolution du web des années 1990 à 2005

Le Web dans les années sombres de 1990 présentait des pages HTML statiques. La plupart des pages étaient du texte mises en forme à grand renfort de balises HTML, et nous étions fascinés par les liens hypertextes. Certains fous du design y ajoutaient des images et quelques effets visuels, aussitôt nous désactivions tout cela pour laisser respirer nos modems asthmatiques.

En quelques années, la tendance à porter la charge sur le poste client en applications diverses s'accentua, il fallait alors alléger le plus possible la communication Web, au vue de l'expansion du réseau Internet. Les technologies de script côté client dont certaines propriétaires aux navigateurs, des solutions à base de plugins furent florès et l'ignorance des recommandations du W3C alourdirent considérablement la tâche du développeur.

Au tout début du siècle, la baisse des coûts des serveurs, l'amélioration des débits et des infrastructures ont favorisé le transfert des traitements dynamiques des pages vers le serveur Web. Le développement de technologies serveurs et des langages serveurs étaient assurés. En revanche, la charge augmenta côté serveur et engorga le réseau. Dans ce western sauvage, surgit Jesse James Garett, auteur d'un article et d'un compromis sur la relation client-serveur.

Présentation de l'article fondateur Ajax de Jesse James Garrett du 18 février 2005

Jesse James Garrett a l'idée de réunir sous un acronyme AJAX un ensemble de technologies qui font le Web de son époque. Mais c'est un peu plus que cela, il comprend qu'il ne faut pas interrompre l'utilisation de la page Web par des requêtes au serveur, qui entraînent un gel de la page et nuisent à son utilisation. Il faut faire circuler sur le réseau le strict nécessaire d'information pour la demande de la page du client et en tâche de fond, *Magister dixit* (Le maître l'a dit). Un langage servira de glu à l'ensemble des technologies avec la possibilité de communiquer en asynchrone : the winner is JavaScript.

Définition du RIA (Rich Internet Application)

Aujourd'hui, l'utilisateur ne conçoit pas (et ne veut pas) de différence entre une application en locale et sur réseau web, la technologie s'efface pour laisser place à une expérience utilisateur qui se focalise sur l'objet de sa demande et non sur la façon de l'adresser. L'utilisateur s'attend alors aux mêmes fonctionnalités de part et d'autre, quelque soit le support, l'environnement ou le type d'informations demandés. Le web doit être capable de fournir toutes ces fonctionnalités souhaitées avec une réactivité toujours plus accrue. L'ensemble des technologies qui permettent ce paradigme répond à l'acronyme de RIA (*Rich Internet Application*) : application internet riche.

Énumération de techniques alternatives sans Ajax

D'autres techniques alternatives existent et peuvent sauver la mise lors de spécifications bien particulières.

On peut citer :

- Les applications Java pour le web : les applets (sur le client) et servlets (sur le serveur),
- Flex - Flash player et son plugin,
- Cadre caché avec l'utilisation de iframe, elle fait office de cache et une fois prête renvoie la partie utile à la page cliente.

Description technique de AJAX

Rappel (AJAX = *Asynchronous JavaScript+CSS+DOM+XMLHttpRequest*). Jesse James Garrett définit AJAX comme un ensemble de technologies et donne un cadre de développement à base de standard, ou plutôt de recommandations selon la terminologie W3C. En énumérant les différentes couches, on définit la séparation entre la présentation des données et leurs interrogations, ce qui n'est pas sans rappeler le modèle MVC (Modèle – Vue – Contrôleur).

- La présentation est assurée par HTML/XHTML et CSS et sa structure dynamique repose sur le DOM (*Document Object Model*),
- La manipulation des données par XSLT et l'échange par XML (et plus),
- L'orchestration est contrôlée par JavaScript utilisant son objet fétiche *XMLHttpRequest*.

Description de l'acronyme AJAX en approfondissant l'asynchrone et le X

Asynchronous JavaScript+CSS+DOM+XMLHttpRequest, on prend le début et la fin, on obtient un nom un peu marketing AJAX. C'est vendu ! En détail, regardons la différence entre une communication synchrone et asynchrone.

Premier cas de figure le modèle classique synchrone

L'utilisateur à partir de la page web fait une demande HTTP sur le serveur, par exemple, il veut connaître la description de l'article qu'il désire acheter. La page se gèle et attend la réponse du serveur. Le serveur web reçoit la demande, interroge une base de données, refait entièrement la page d'achat du client. Le serveur renvoie la totalité de la page au navigateur qui l'affiche (HTML et CSS). La page se rafraîchit en clignotant, ce qui indique le rechargeement entier de la page.

L'utilisateur peut à nouveau naviguer.

Deuxième cas de figure le modèle Ajax asynchrone

L'utilisateur à partir de la page web fait une demande HTTP sur le serveur, il veut toujours connaître la description de l'article. En interne, le navigateur fait appel

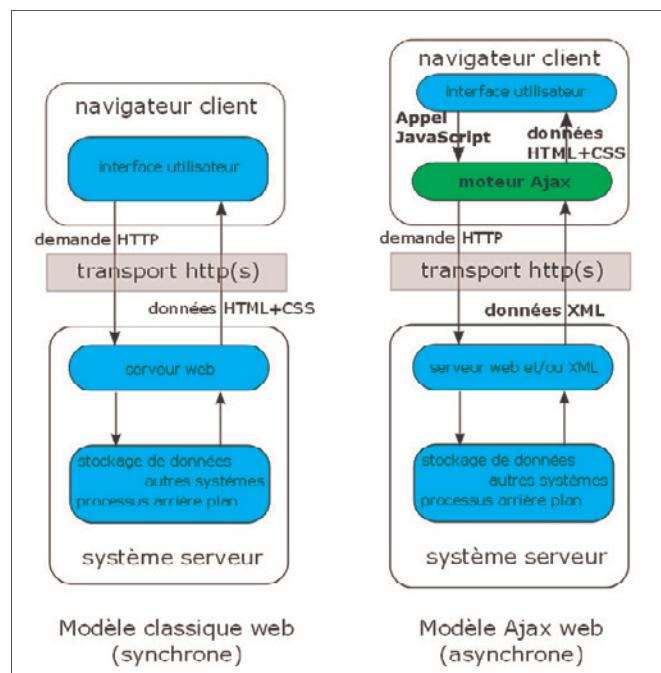


Figure 1. Communication synchrone et asynchrone



Figure 2. Google et sa complétion automatique

au script client et avec *XMLHttpRequest* se connecte au serveur. L'utilisateur peut pendant ce temps continuer à utiliser la page, par exemple, pour voir des articles similaires en faisant défiler une zone sur la même page.

Le serveur web reçoit la demande, interroge une base de données, construit la réponse nécessaire uniquement, ici dans une chaîne HTML contenant la description. Le serveur renvoie la partie qui satisfait à la demande, et le navigateur l'affiche (HTML et CSS). L'utilisateur n'a été à aucun moment contraint d'attendre cette réponse.

Vous pouvez expérimenter Ajax sur Google, vous verrez que la recherche s'effectue à chaque événement d'appui de touche et la liste s'enrichit très rapidement au fur et à mesure des lettres successives en donnant cet aspect de fluidité.

Listing 1. Choix de l'objet XHR

```
function createXHR() {
  var xmlhttpReq = null;
  if(window.XMLHttpRequest) { //IE7 et plus, Firefox, Safari, Opera...
    var xmlhttpReq = new XMLHttpRequest();
    return xmlhttpReq;
  } else if (window.ActiveXObject) { //on part au pays de Microsoft
    try {
      return new ActiveXObject("Msxml2.XMLHTTP");
    //Internet Explorer version > 5.0
    } catch (err) {}
    try {
      return new ActiveXObject("Microsoft.XMLHTTP");
    // Internet Explorer version 5.0 !
    } catch (err) {}
  }
  throw new Error("Impossible de créer l'objet XMLHttpRequest pour le navigateur.");
}
```

Listing 2. Le moteur AJAX

```
// on implémente le callback comme fonction anonyme
objXHR.onreadystatechange = function() {
// on test les flags readyState (4 : complete) et
status (200 : OK du HTTP)
  if(this.readyState == 4 && this.status == 200){
//Si c'est une réponse texte
  if(this.responseText != null) {
//On récupère la valeur et on l'affiche
    var resultText = this.responseText ;
    alert(resultText);
  } else return ;
  }
}
```

Listing 3. Les appels de fonctionnalités

```
//Appel du script PHP avec une méthode GET sans
paramètre.
objXHR.open("GET", "reponseScript.php", true) ;
//On envoie la requête
objXHR.send(null) ;
```

Listing 4. Le script reponseScript.php

```
<?php
//Type de la réponse à renvoyer
header("Content-Type: text/plain");
echo "Bonjour du script".$_SERVER['SCRIPT_NAME'] ;
?>
```

Listing 4B. Manipulation du DOM

```
innerHTMLContent(id, newContent) {
  var elt = document.getElementById(id) ;
  if(elt != null) {
    while(elt.firstChild) {
      elt.removeChild(elt.firstChild) ;
    }
    var inner = document.createTextNode(newContent) ;
    elt.appendChild(inner) ;
  }
}
```

Listing 5. Chargement startAjax

```
<script type="text/javascript">
  window.onload = function() {
    startAjax ;
    ... // autres scripts à charger
  }
</script>
```

Comment l'implémenter, présentation de la classe XMLHttpRequest et ActiveXObject pour Microsoft

Depuis longtemps les navigateurs, au début l'Internet Explorer de Microsoft, peuvent faire de l'asynchrone. Cet objet existe sous deux entités différentes : un *ActiveX* pour Microsoft nommé *ActiveXObject* et *XMLHttpRequest* pour les autres. Microsoft a décliné son objet sous deux formes pour la version supérieure à 5.0 et Internet Explorer 5.0. Nous allons écrire le code pour faire le choix et la détection à notre place. Code choix de l'objet : écriture du `createXHR()` l'appel se fera simplement par : `var objetXHR = createXHR() ;`

Tableau des attributs et méthodes

Un objet, c'est un ensemble d'attributs (états) et de méthodes (comportements).

Notion de callback : par méthode de *callback*, nous appellerons une fonction qui observera en tâche de fond le changement d'un état donné et exécutera la fonction passée en paramètre.

Valeurs de readyState

Un exemple simple

L'exemple s'écrit en JavaScript, il peut s'appeler au chargement de la page côté client.

On utilise la méthode de création de l'objet `createXHR()` implémentée précédemment.

Les conditions de l'appel sont les suivantes :

- appel par la méthode `GET`,
- un script `reponseScript.php` (le script est censé bloquer l'interaction utilisateur sur le poste client) retourne un texte,
- La communication client-serveur est asynchrone.

Ce script ne prend pas de paramètre pour la méthode `GET`. On instancie l'objet `XMLHttpRequest`

```
var objXHR = createXHR() ;
```

That's all folks!

Pour rester simple, nous ne faisons pas intervenir le DOM, ni CSS, ce qui ne correspond pas stricto sensu à la définition d' AJAX. A l'aide de ce script concis, on peut se rendre compte que l'inscription d'une requête AJAX relève d'une méthodologie et d'un ordonnancement précis. Cette méthodologie est à suivre à chaque fois, selon l'orientation que nous voulons donner à notre requête. En d'autres termes, le traitement des erreurs et des états de l'instance `XMLHttpRequest` peut être assez varié.

Après l'exécution de `send()`, on peut afficher les en-têtes de réponses HTTP. Par exemple :

```
client.onreadystatechange = function() {
  if(this.readyState == 2) print(this.getAllResponseHeaders()); }
```



Patrice MALÉ

Le Spécialiste MAC

**Ordinateurs APPLE (Mac neuf & Occasion) affilié AppleStore
Logiciels et Accessoires pour Mac - iPod - iPhone - iPad
Périphériques (imprimantes, scanners, disques durs, ...) ...etc**

Vente - installation - Formation - Réparation



Revendeur officiel



Mon Mac & Moi

Sur R.D.V. : 51, avenue Georges Pompidou • local FPI • 19100 BRIVE

✓ : 06 08 60 96 59 • e-mail : patrice@e-sei.com • Site : www.e-sei.com

NOUVEAU : Dès le 15 Novembre 2010, retrouvez le site marchand :

www.TOUTPOURLEMAC.com



c'est les services informatiques autour du MAC (uniquement)

mais c'est aussi :

- **la Création Graphique** (logos, communications),
- **la Création de Sites** (sites vitrines, sites marchands),
- **l'Hébergement**.

Tableau 1. Liste des méthodes et attributs

Méthode et attribut	Paramètre et description
open(method, url, async)	Ouvre une connexion à l'URL du script côté serveur method = HTTP (GET, POST, etc.) url = url à ouvrir, peut inclure une chaîne de requête async = précise si la connexion doit être asynchrone (true) ou bien synchrone (false)
onreadystatechange	Affecte une fonction objet comme callback (syntaxe identique à l'affectation du modèle événement du navigateur)
setRequestHeader (name,value)	Ajoute un header à la requête HTTP pour préciser la nature du corps de la requête passée par send() en cas de POST
send(body)	Envoi de la requête body = null pour GET, une chaîne pour POST
abort()	arrête le XHR qui écoute la réponse
readyState	État du cycle de vie de la réponse (renseigné après que send() soit appelée)
Status	Code de retour de HTTP (un entier)
responseText	Corps de la réponse du serveur sous forme de texte
responseXML	Corps de la réponse du serveur sous forme XML
getResponseHeader (name)	Lecture de la réponse de l'en-tête précisé par son nom
getAllResponseHeaders()	Retour de tous les en-têtes HTTP dans une seule chaîne

Tableau 2. Liste des valeurs readyState

État	Valeur	Description
0	Uninitialized	La requête n'est pas encore envoyée (open() non appelée)
1	Loading	Initialisée mais la réponse n'est pas arrivée car la méthode send() n'a pas été exécutée
2	Loaded	Les en-têtes de réponse peuvent être lus, send() a été exécutée
3	Interactive	La réponse est en cours de réception, elle est incomplète, mais elle peut être lue.
4	Complete	La réponse du serveur est complète. La propriété responseText ou responseXML contient le résultat en fonction du type de données.

On peut désirer une politique de granularité des exceptions sur les paramètres en lecture seule comme `readyState` et `status`, pour les traiter de façon différenciée. Pour `status`, les codes de retour HTTP avertissent l'utilisateur ou le script du problème : 200 OK, 404 page not found, 403 forbidden, par exemple. Une liste exhaustive des codes de retour se trouve sur le site Internet du W3C.

Restrictions et solutions

Tout n'est pas merveilleux dans le monde AJAX. La première condition sera que la requête s'effectue sur le même serveur, restriction qui ne tardera pas à tomber au regard de la nouvelle implémentation de `XMLHttpRequest 2`. Les frameworks JavaScript proposent déjà des solutions quasi-automatiques. Il existe plusieurs méthodes alternatives, certaines sont citées succinctement ci-après.

Schéma serveur différent : méthode proxy – le serveur peut interroger un autre serveur, il n'y a pas de restriction.

Méthode à base de `iframe` :

On utilise une balise `<iframe></iframe>` que l'on rend invisible grâce à ses propriétés CSS. L'intérêt est que l'ifra-

me ne souffre pas de la restriction de l'interrogation d'un service sur un serveur différent du serveur hébergeur.

Méthode `cross domain` :

Microsoft avec `XDomainRequest` et `XMLHttpRequest 2` propose en plus de la simplification de syntaxe le `cross domain`, il suffit d'indiquer dans l'URL, le domaine distant à atteindre : par exemple `objXHR.open("GET", "http://www.domainesistant.net/script.php", true);`

Le protocole HTTP

Le protocole HTTP permet un transfert de fichiers. Ce qui importe, c'est l'écriture de l'URL et l'écriture d'en-tête. Il est défini par une RFC, la rfc2616 pour le protocole HTTP/1.1 du W3C.

Une URL get avec ses paramètres visibles :

`http://<HOST>:<PORT>/lescript.php?param1=valeur1¶m2=valeur2`

HOST c'est l'adresse de l'hôte de la ressource à atteindre.

PORT est par défaut égal à 80, il peut être omis.

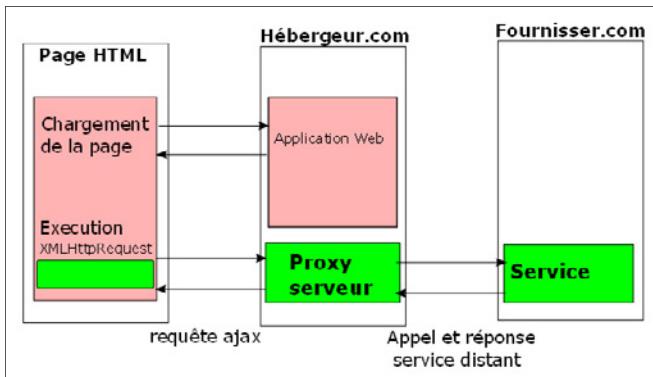


Figure 3. Méthode proxy

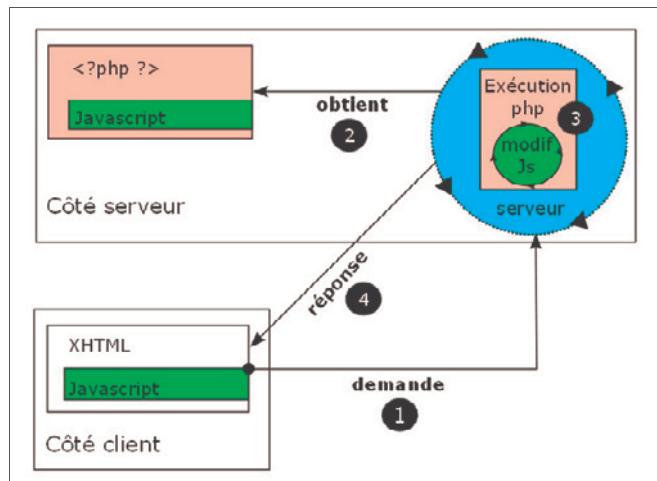


Figure 4. Schémas PHP sur serveur Javascript sur client

Les en-têtes HTTP

Les en-têtes (headers), nous serons intéressés essentiellement par :

- *Content-Type* : par exemple `text/xml`, pour préciser quel type de données est envoyé,
- *Cache-Control* : contrôle du cache, pour le désactiver le cache vaut `no-cache` pour HTTP 1.1,
- *Pragma* : pour le désactiver le cache vaut `no-cache` pour HTTP 1.0.

Rappels GET/HEAD/POST

- GET

C'est une méthode pour demander une ressource URL. Les paramètres, s'ils existent, sont passés dans l'URL et sont donc visibles,

- HEAD

C'est une demande d'information de la ressource elle-même,

- POST

Soumission de formulaire, les paramètres ne sont pas visibles dans l'URL.

PHP et AJAX. Pas de communication PHP avec JavaScript

Il est souvent question pour les débutants de vouloir faire communiquer ces scripts, par exemple, que JavaScript utilise une fonction PHP. Ce n'est pas possible! Le PHP s'exécute sur le serveur, le JavaScript côté client. De plus l'instant n'est pas le même, quand Javascript s'exécute côté client le PHP 'n'existe plus', le serveur l'a renvoyé sous forme HTML + Javascript. Et pourtant... PHP peut construire du JavaScript à la volée, qui sera renvoyé par le serveur, donc on peut dire qu'il y a relation. De la même façon, un formulaire HTML construit par PHP peut réagir à des événements JavaScript.

JavaScript peut construire aussi un formulaire, le soumettre et faire une requête à un script PHP, donc il y a aussi interaction. Mais ce sont des interactions indirectes, qui nécessitent des aller-retours entre le client et le serveur.

- 1 le navigateur client demande au serveur une page php soit par lien HTML ou par JavaScript,
- 2 le serveur obtient la page,
- 3 le serveur exécute le php (qui peut écrire ou modifier du texte JavaScript),
- 4 la réponse amène un fichier interprétable par le navigateur (XHTML + JavaScript par exemple).

Ce schéma, plutôt symbolique, permet de comprendre que JavaScript du client n'est jamais en relation directe avec le PHP du script php.

Ajax appelle 'script php' et interagit sur le DOM :

- Le DOM, permet de reconstituer le document sous forme d'un arbre et de nœuds. On peut ajouter, modifier, supprimer un nœud ou un ensemble de nœuds, ses attributs et valeurs,
- JavaScript autorise la manipulation de l'arbre syntaxique du DOM.

Introduction à innerHTML et l'alternative JavaScript de manipulation du DOM

La méthode la plus commune pour afficher un résultat de type texte dans une balise HTML avec JavaScript est `innerHTML` :

pour une balise HTML avec un Id 'textAjax'

```
<span id="textAjax">&nbsp;</span>
```

Code JavaScript avec `innerHTML` :

```
document.getElementById("textAjax").innerHTML
= objXHR.responseText ;
```

Cependant l'attribut `innerHTML` ne fait pas partie des spécifications du DOM. Toutefois, cet attribut est largement

diffusé sur les navigateurs récents. Malgré tout, il peut être judicieux d'écrire correctement les choses...

Manipulation des nœuds DOM

Pour remplacer innerHTML, en pure manipulation du DOM, on écrit la fonction du Listing 4B. L'appel devient donc : innerHTMLContent("textAjax", objXHR.responseText); Il nous manque pour un AJAX complet, un traitement CSS. Dans le fichier client, on trouve le HTML suivant :

 . À inclure dans un fichier CSS à part ou bien en début de script entre les tags <head></head>

```
<style type='text/css'>
span.message {
  color: red ;
  font-size: bold ;
}
</style>
```

Maintenant les textes des span qui ont une classe message seront en rouge et gras.

Différents types d'envois

Pour déclencher l'AJAX, nous avons besoin d'un événement. Dans la page une fois chargée (voir Listing 5), on exécute notre fonction startAjax (voir Listing 6).

- Sans paramètre,

On reprend le code précédemment écrit et on modifie l'appel à alert() pour conserver le résultat dans la page (voir Listing 6).

- Avec GET paramétré texte,

Le problème avec AJAX, c'est la mise en cache automatique de certains navigateurs. Pour éviter cela, il faut un peu lui mentir, faire croire que notre requête est nouvelle à chaque fois.

Sinon, le navigateur renvoie son cache !

Comme toujours en programmation, le temps va nous permettre d'obtenir un résultat variable.

Il suffit de passer ce paramètre par GET, après le point d'interrogation, on écrit ces lignes avant l'appel à open() (voir Listing 7).

- Avec POST paramétré texte.

Avec POST, les paramètres de l'URL disparaissent, mais comme rien ne se perd mais tout se transforme, on définit les paramètres dans une variable :

```
var param = "utilisateur=root&codePasSecret=1234" ;
```

Listing 6. Le moteur AJAX

```
function startAjax() {
  var objXHR = createXHR() ;
  objXHR.onreadystatechange = function() {
    if(this.readyState == 4 && this.status == 200 ){
      if(this.responseText != null ) {
        var resultText = this.responseText ;
        document.getElementById("textAjax").innerHTML =
resultText ; // à la place de alert()
      } else return ;
    }
  }
}
L'appel
objXHR.open("GET", "script.php", true);
objXHR.send(null) ;
```

Listing 7. Appel du script GET

```
<?
var nocache = new Date().getTime() ;
objXHR.open("GET", "script.php?cache=" + nocache,
true);
objXHR.send(null) ;
```

Listing 8. Appel du script POST paramétré

```
<?
objXHR.setRequestHeader( "Content-Type", "application/x-www-form-urlencoded" );
objXHR.open("POST", "script.php" , true);
objXHR.send(param) ;
```

Listing 9. Renvoi de la page

```
<?php
header("Content-Type: text/plain; charset=utf-8");
header("Cache-Control: no-cache , private");
header("Pragma: no-cache") ;
if(isset($_POST['utilisateur']) && $_POST['utilisateur']=='root' )
  echo $_POST['utilisateur']." vous êtes un idiot!" ;
?>
```

Listing 10. Renvoi de la page

```
<?php
header("Content-Type: text/plain");
if(isset($_POST['nom']) && trim($_POST['nom'])!="") $nom =
$_POST['nom'] else $nom = 'M.John Doe' ;
$reponse = "Bonjour ".$nom . " ! " ;
echo $reponse ;
?>
```

Listing 11. objet JSON

```
//javascript objet JSON
{
"nomObjet": {
  "attribut_1": "valeur_1",
  "attribut_2": "valeur_2"
}
}
```

Listing 12. Parse la réponse

```
function parseResponse(objXHR){
  var jsonObj=eval(
  "("+
  +objXHR.responseText
  +")");
  var result1 = jsonObj.nomObjet .attribut_1;
  return (result1) ;
}
document.getElementById("result1").innerHTML =
parseResponse(objXHR) ;
```

Listing 13. Tableau PHP au format JSON

```
<?php

$arrayPHP = array ( "attribut_1"=>"valeur_1",
"attribut_2"=>"valeur_2");
$arrayJSON = array ("nomObjet" => $arrayPHP) ;
$reponse = json_encode($arrayJSON) ;
echo $reponse ;
?>
```

Listing 14. Réponse PHP au format XML

```
<?php

header("Content-Type: text/xml; charset=utf-8");
header("Cache-Control: no-cache , private");
header("Pragma: no-cache") ;
$reponse = '<?xml version="1.0" encoding="utf-8"?>' ;
$reponse .= "<reponse><attribut_1>Bonjour</attribut_1><attribut_2>foo</attribut_2></reponse>" ;
echo $reponse ;
?>
```

Listing 15. Récupération en JavaScript au format XML

```
objXHR.onreadystatechange = function() {
  if(this.readyState == 4 && this.status == 200 ){
    if(this.responseXML != null ) {
      var resultXML = this.responseXML ;
      var rootXML = resultXML.firstChild ; //racine du document xml
      var attribut_1 = rootXML.getElementsByTagName("attribut_1")[0].firstChild.nodeValue ;
      document.getElementById("textAjax").innerHTML =
attribut_1 ;
    } else return ;
  }
}
```

Listing 16. ScriptProxy.php

```
<?php

header("Content-Type: text/xml; charset=utf-8");
header("Cache-Control: no-cache , private");
header("Pragma: no-cache") ;
readfile("http://domainRss.com/infoRSS.xml"); // PROXY
?>
```

Listing 17. Extrait d'un flux RSS

```
...
<item>
  <title> RSS</title>
  <link>http://domain.net/index.htm</link>
  <description>texte du flux RSS </description>
</item>
...
```

Listing 18. Traitement JavaScript du RSS

```
objXHR.onreadystatechange = function() {
  if(this.readyState == 4 && this.status == 200 ){
    if(this.responseXML != null ) {
      var resultXML = this.responseXML ;
      var lstItemRSS = resultXML.getElementsByTagName("item");
      for( i = 0 ; i < lstItemRSS.length ; i++){
        var title = lstItemRSS[i].getElementsByTagName("title")[0].firstChild.nodeValue ;
        var link = lstItemRSS[i].getElementsByTagName("link")[0].firstChild.nodeValue ;
        var out = title + " : " + link ;
        traite (out) ; //la fonction qui traitera les données pour affichage (manipulation du DOM )
      }
    } else return ;
  }
}
```

nous allons avertir le serveur que nous postons, donc on l'indique à l' header (voir Listing 8).

Le script php :

Il récupère les paramètres et envoie une appréciation (voir Listing 9).

Note : les headers donnent le type de la page renvoyé (text/plain) ainsi que la façon dont le cache est géré (ici no-cache). Dans la balise span textAjax du client le message 'root vous êtes un idiot!' doit apparaître en rouge et gras. Les exemples ci-dessus ont tous des paramètres d'envoi texte simple. Toutefois, les paramètres transmis au serveur peuvent être au format JSON, XML, fragment du DOM (XML), etc. Ceci modifie le type de header à envoyer et ajoute un traitement de l'argument supplémentaire, mais la logique de l'AJAX reste la même. PHP dans sa version supérieure à 5.2 permet une approche simplifiée de ces traitements grâce aux fonctions json_* et simplexml_*. Dans tous les cas, on peut lire les données brutes (raw data) envoyées par POST par :

```
<?php $postdata = file_get_contents("php://input"); ?>
```

Différents types de format de données pour les réponses

L'objet XMLHttpRequest a deux propriétés qui contiennent les réponses :

- 1 `responseText` contient les données sous forme de chaîne texte,
- 2 `responseXML` contient les données en xml valide. Dans le cas contraire contient null, et le retour sera du texte dans `responseText`.

Firefox supporte un attribut `overrideMimeType`, à écrire juste à la suite de l'instanciation de XMLHttpRequest. Ceci permet d'informer le navigateur que les données en retour auront un Content-Type: «text/xml».

```
if (objXHR.overrideMimeType) objXHR.overrideMimeType('text/xml') ;
```

La méthodologie du script de réponse repose sur un ordre précis d'étapes à respecter :

- 1 Envoyer un header correspondant à la nature de la ressource retournée. Sans aucune instruction de sortie le précédent(pas de echo, balise html...), PHP vous le rappellera le cas échéant,
- 2 Récupération des données du POST ou GET. Les variables globales et méthodes suivantes seront vos amies : `$_REQUEST`, `file_get_contents()`, etc.,
- 3 Les données seront traitées si besoin, pour finalement correspondre au format de sortie voulu. Par

- exemple pour JSON, nous utiliserons la fonction `json_encode()`.
- 4 Écriture de la réponse à la suite du header, par exemple `<?php echo $reponse ; ?>`.

Texte

Jusqu'à présent, le serveur a renvoyé les réponses sous forme de texte. Pour les scripts retournant ou traitant du texte, il serait plus judicieux de recourir à un script CGI en Perl. Exemple simple est présenté dans le Listing 10.

JSON

Définition succincte : JSON est du texte au formatage convenu de données (nom/valeur) qui est extrêmement facile d'emploi pour JavaScript. JSON est la notation objet de JavaScript. Il se prête à l'évaluation (`eval()`) par JavaScript. JSON est sans doute plus léger et aussi polyvalent en terme de communication inter-application que XML. Exemple format JSON est montré dans le Listing 11.

Le résultat retourné est une chaîne de caractères, elle sera dans l'attribut `responseText`.

On écrit la fonction pour traiter la réponse dans la fonction de `callback` liée à `onreadystatechange` (voir Listing 12). On remarque l'emploi de la fonction `eval()` pour retourner l'objet décrit dans `responseText`.

Avantages et inconvénients

Le format JSON est un moyen simple d'échanger des données tout en s'affranchissant du langage.

Les spécifications JSON, sont prisent en compte par PHP grâce aux fonctions *natives* `json_*`.

En revanche, en fonction de la version de PHP, on peut être obligé de passer par des bibliothèques tierces, celles-ci peuvent ne pas être disponibles sur le serveur, ou bien encore, le serveur n'autorise pas de sortie JSON. Format JSON PHP avec `json_encode()`, `json_decode()` et `json_last_error()`

Soit un tableau `array()` PHP présenté dans le Listing 13.

`$reponse` correspond à l'objet JSON définit précédemment. La fonction réciproque `json_decode()`, transforme un objet JSON en un tableau PHP. On aura intérêt à vérifier que le format JSON est correct en testant la dernière valeur de retour de `json_last_error()`. Par exemple, cette vérification se fera avant `echo` de la réponse et juste après `json_encode()` :

```
if (json_last_error() != JSON_ERROR_NONE) //  
traiter l'erreur  
return; else echo $reponse ;
```

XML

Définition succincte : le format XML devient un standard de fait dans la plupart des échanges de données. Ce format bien que rigoureux et verbeux, reste lisible

à l'humain et permet de hiérarchiser les données parfois issues de système complexe. De plus, rien n'empêche de convertir ensuite les données XML en JSON, dans ce cas il faudra récupérer le XML sous format texte dans `responseText`. Exemple, la réponse du serveur est présenté dans le Listing 14. On récupère dans la fonction de `callback` la réponse XML (voir Listing 15).

Avantages et inconvénients

On remarque que le code peut devenir vite touffu, mais il peut se factoriser dans une boucle, ou bien être déplacé dans une fonction dédiée. Il existe aussi, des solutions avec des frameworks qui pourront alléger l'écriture. Le traitement nécessite de connaître parfaitement la structure du fichier xml reçu. Dans le cas contraire, il faut rendre un peu plus abstrait ou générique le code au risque de le rendre moins compréhensible.

par exemple :

```
var attribut _ 1 = rootXML.getElementsByTagName("attribut _ 1")[0].firstChild.nodeValue ;
```

peut être remplacé par son équivalent :

```
var attribut _ 1 = rootXML.childNodes[0].  
firstChild.nodeValue ;
```

On note la disparition explicite du nom de l'attribut.

RSS exemple d'application XML

Définition succincte : RSS (*RDF Site Summary*) est un fichier de syndication sous format xml et qui contient des informations complémentaires de descriptions de la source. Nous avons deux parties, les métadonnées (description du *channel*) et le contenu (l'info qui nous intéresse).

Comme le flux RSS ne se trouve pas sur le même serveur en général, nous allons utiliser le script appelé sur le serveur comme `proxy`. Exemple `scriptProxy.php` sur le serveur (appelé par `open()`) (voir Listing 16).

L'appel se fait comme une fonction GET :

```
objXHR.open('GET', 'http://www.domainservice.  
net/scriptProxy.php', true) ; objXHR.  
send(null) ;
```

Le flux RSS comprend une succession d'informations sur le modèle en extrait du Listing 17.

La réponse est traitée en XML : `resultXML.getElementsByTagName("item")` ; retourne une liste des items du flux (voir Listing 18).

Exemple complet

Un exemple d'agenda simple. On utilise la base de données SQLite, qui peut servir de SGBD intermédiaire, el-

Listing 19. Exemple complet AJAX

Fichier index(xhr.php)

```

<?php

echo "<div>";
echo "<select id='nom'>
      <option>Hochon</option>
      <option>Tatareau</option>
    </select>" ;
echo "</div>" ;
echo '';
echo "<span id='contactClient'>&nbsp;</span>";
?>

<script type="text/javascript" >
window.onload = quelNom ;
document.getElementById('nom').onclick = quelNom ;

function creationXHR() {
var resultat=null;
try {
  var xmlhttpReq = new XMLHttpRequest();
}
catch(err) {}

if(window.XMLHttpRequest){
var xmlhttpReq = new XMLHttpRequest();
if (xmlhttpReq.overrideMimeType){
  xmlhttpReq.overrideMimeType("text/xml");
}

return xmlhttpReq;
} else if (window.ActiveXObject) {
  try {
    return new ActiveXObject("Msxml2.XMLHTTP");
  } catch (err) {}
  try {
    return new ActiveXObject("Microsoft.XMLHTTP");
  } catch (err) {}
}
throw new Error("Impossible de créer
l'objet"+"XMLHttpRequest pour le navigateur.");
}

function quelNom() {
objetXHR = creationXHR();
var parametres = new Array() ;
var quelNom = document.getElementById('nom').
options[document.getElementById('nom').selectedIndex].
text ;
var nocachetime = new Date().getTime();
parametres.push("nom=");
parametres.push(quelNom) ;
parametres.push("&nocache=");
parametres.push(nocachetime) ;
var params = parametres.join("");
objetXHR.open("get","test_mysqli.php?"+params, true);
document.getElementById('anim').style.
visibility="visible" ;
objetXHR.onreadystatechange = actualiserClient ;
objetXHR.send(null);
}

function actualiserClient(){
if(objetXHR.readyState == 4){
  if(objetXHR.status == 200) {
    var clientXML = objetXHR.responseXML ;
    var rootClientXML = clientXML.firstChild ;
    var nom = rootClientXML.getElementsByTagName("nom")[0].
firstChild.nodeValue ;
    var prenom = rootClientXML.getElementsByTagName("pr
énom")[0].firstChild.nodeValue ;
    var telephone = rootClientXML.getElementsByTagName(
"telephone")[0].firstChild.nodeValue ;
    document.getElementById('anim').style.
visibility="hidden" ;
    document.getElementById("contactClient").innerHTML =
nom + ' ' + prenom + ' ' + telephone ;
  }
}
}

```

```

else
{
var erreurXHRstatus = objetXHR.status;
var erreurXHRstatusText = objetXHR.statusText;
objetXHR.abort() ;
objetXHR=null;
}

</script>

fichier test_mysqli.php

<?php

header("Content-Type: text/xml ; charset=utf-8") ;
header("Cache-Control: no-cache , private") ;
header("Pragma: no-cache") ;

echo '<?xml version="1.0" encoding="UTF-8"?>' ;
$dsns = 'sqlite:BaseTestSQLite' ;
$user = '' ;
$pass = '' ;

try{
  $dbh = new PDO($dsns) ;
}

catch (PDOException $e)
{
  die ( 'Erreur ! : '.$e->getMessage() ) ;
}

if (isset($_GET['nom']) && trim($_GET['nom'])!=
"") {
  $nom = strtoupper( $_GET['nom'] ) ;
  $sql = "SELECT * FROM client WHERE nom LIKE :
nom ORDER BY nom " ;
  $sth = $dbh->prepare($sql, array(PDO::ATTR_
CURSOR => PDO::CURSOR_FWDONLY ) );
  $sth->execute(array(':nom' => $nom));
  $XML = new XMLWriter();
  $XML->openMemory();
  while($row = $sth->fetchObject() ){
    $XML->startElement("client");
    $XML->writeElement("nom",
$row->nom);
    $XML->writeElement("prenom",
$row->prenom);
    $XML->writeElement("telephone",
$row->tel);
    $XML->endElement();
  }
  echo $XML->outputMemory();
}
else echo 'Demande impossible.';
$dbh = NULL ;
?>

```

Listing 20. Mise en place de AJAX avec JQuery

```

$("#success").load("script.php", { 'nom[]': ["Adam", "Eve"] },
, function(response, status, xhr) {
  if (status == "success") {
    var msg = "resultat OK !";
    $("#result").html(msg);
  });
});

// JQuery laisse la possibilité d'une programmation
// complète d' AJAX .

$.ajax({
  type: "POST",
  url: "script.php",
  data: "prenom=John&nom=Resing",
  success: function(msg){
    $("#result").html(msg);
  }
});

```

le est très légère et elle est constituée d'un seul fichier. Son accès se fait par PDO et nous permet de revoir la requête paramétrée. On suppose que la liste déroulante des noms clients est alimentée par une demande SQL de PHP. Quand le nom est sélectionné, la requête AJAX appelle un script php qui retourne en XML le nom, prénom et numéro de téléphone du client.

La demande XMLHttpRequest se fait par une requête GET et les paramètres sont passés par URL. La réponse est XML en utilisant les fonctions de XMLWriter(). Le Javascript décode le XML par le DOM.

La table client est créée par :

```
CREATE TABLE "client" ("id" INTEGER PRIMARY
KEY NOT NULL , "prenom" VARCHAR, "nom"
VARCHAR, "tel" VARCHAR)
```

On peut utiliser le plugin SQLite Manager pour Firefox pour construire et alimenter la base pour les tests. La base est constituée d'un seul fichier que l'on placera de façon à pouvoir être accédé par PHP.

Mise au point et analyse dans Firefox avec Firebug 1.5.4

Firefox est un formidable outil de débogage. Il possède plusieurs plugins, dont certains très performants : 'Web developper', 'Firebug' et IETab 2 pour émuler Internet Explorer dans Firefox 3.6. Onglet Console pour voir Header et Response de la page avec Firebug. Firebug possède un onglet en-têtes qui permet de visualiser les en-têtes de réponse et de requête. Il est intéressant de noter le Content-Type du type de donnée acceptée par la requête et de voir évidemment que XMLHttpRequest a fonctionné.

Utilisation l'onglet DOM pour observer le comportement de l'objectXHR

On retrouve tous nos amis de AJAX, un objet objectXHR qui instancie XMLHttpRequest, puis les différents états et réponses : readyState=4 du traitement terminé, la responseText et son contenu, responseXML à null, le status à 200 du serveur avec le texte OK. Il est évident que cette fonctionnalité est très pratique, surtout pour visualiser les erreurs et états de ressources demandées.

L'onglet réseau (voir Figure 7) permet d'obtenir le statut de la requête (ici GET) avec 200 OK, son temps d'exécution (2,13s) et ses paramètres mois et no-cache. En plus la sélection par le bouton XHR trie les appels AJAX uniquement!

Profilage pour les temps d'accès et nombres d'appel des méthodes

On peut avoir un profilage plus précis encore, avec le bouton console et profiler (voir Figure 8). On peut lire que la fonction callback actualiserPage à été appelée 5 fois.

Figure 5 shows the Firebug Network tab with the 'Headers' tab selected. It displays the response headers for a GET request to `http://fr.wikipedia.org/wiki/Special:BannerListLoader?language=fr`. The 'Content-Type' header is circled in red. The 'Request' tab shows the 'Host' and 'User-Agent' headers, also with 'Content-Type' circled in red.

Figure 5. Lecture des en-têtes sur client

Figure 6 shows the Firebug DOM tab with the 'objectXHR' node expanded. It shows various properties and methods of the XMLHttpRequest object. The 'onuploadprogress' method is highlighted with a red box and shows a status of 4, responseText of '16 course(s) baby', and statusText of 'OK'.

Figure 6. Observation de objectXHR

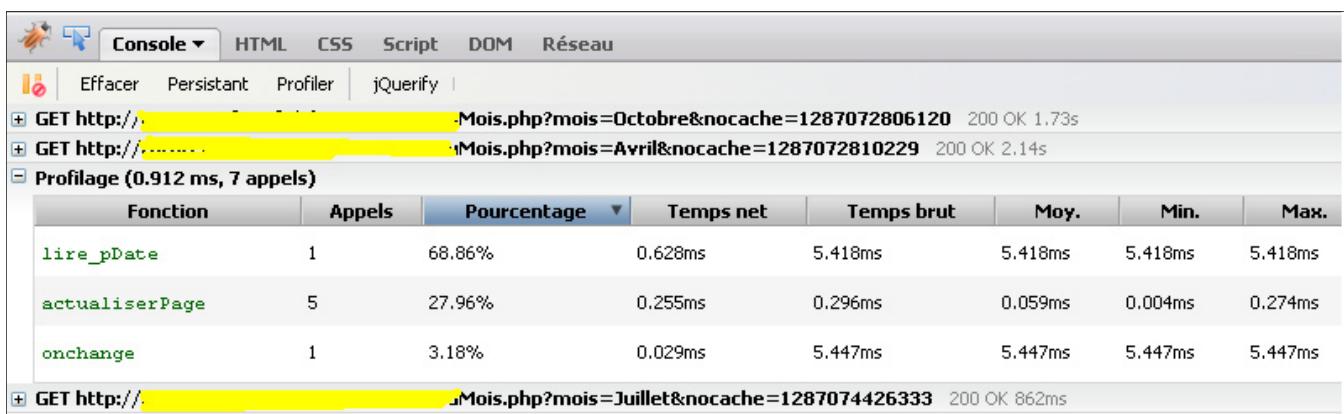
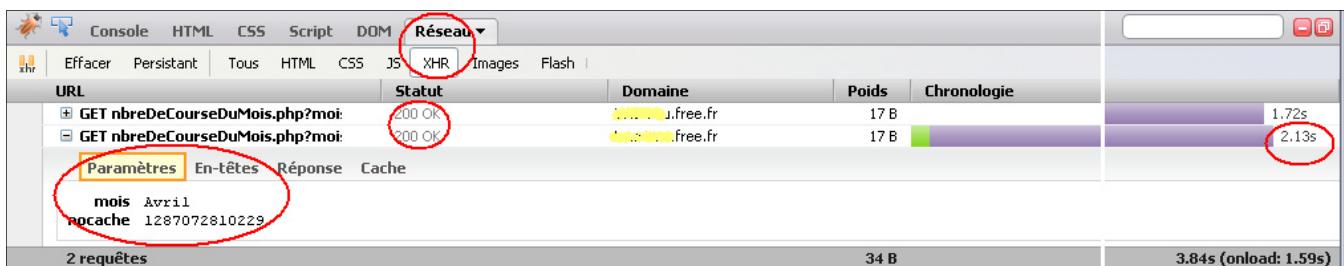
Examen des CSS

Pour finaliser l'examen AJAX, la structure du document ainsi que son CSS peut être évaluée et modifiée in-situ. L'état des choses ne sera pas persistant sur le serveur, c'est uniquement sur le navigateur ! On peut obtenir le script CSS en cours avec l'onglet CSS, (voir Figure 9), nous avons les onglets CSS, style calculé (c'est-à-dire après le cascading). Les propriétés sont éditables et l'on peut observer l'effet des modifications sur la page.

Note :

A partir du JavaScript, on peut écrire des lignes d'information ou de débogage dans la console :

```
console.log("débogage: %s!", erreur1);
```



L'objet console supporte d'autres fonctions puissantes comme les assertions avec `console.assert()`. L'onglet script permet d'intervenir sur le JavaScript avec des points d'arrêt et des valeurs à suivre appelées *Espions*.

La valeur 'Espions' recherchée de `document.getElementById("pdate")` donne en bleu `input#pdate` (voir Figure 10).

Ce n'est qu'un maigre aperçu de Firebug, qui est une aide indispensable au développement.

Un framework JavaScript : implémentation avec Jquery. Présentation et avantages d'utiliser un framework Javascript

Un framework offre un cadre de travail sécurisé et permet de façon méthodique, de s'abstraire de certaines contraintes récurrentes de programmation. Cela favorise la concentration sur les parties réellement fonctionnelles du code et facilite la maintenance de ce dernier. Un framework permet généralement une écriture plus concise du code dans un cadre déterminé en apportant des fonctionnalités supplémentaires qui simplifient la tâche du développeur. Toutefois, il est important de connaître la façon de coder sans le framework, *from scratch* avec JavaScript, car cela permet d'une part, de maîtriser les techniques mises en jeu, et d'autres part de remédier à des bugs, des défaillances et limites du framework.

Parmi les problèmes rencontrés, un framework peut résoudre :

L'implémentation de l'objet XMLHttpRequest dans les différents navigateurs qui n'est pas homogène. Nous avons vu les différents objets à instancier pour

Firefox et Internet Explorer Microsoft qui sont *XMLHttpRequest* et *ActiveXObject*. Le code est souvent redondant pour l'exploration du DOM, par exemple de nombreuses boucles sont nécessaires pour explorer l'arbre DOM.

Exemple de la simplification

Parmi les méthodes remarquables de JQuery :

```
.load( url, [ data ], [ complete(responseText, textStatus, XMLHttpRequest) ] )
```

Cette méthode va gérer seule l'envoi par POST ou GET en se basant sur le format du data. Si le data est un objet, alors la méthode sera POST sinon GET.

Exemple :

```
//En xHTML
<div id = "result" ></div>
```

Le script va être appelé de façon asynchrone, ce passage de paramètres oblige un POST, il n'y a pas de fonction de callback.

```
//javascript
$("#result").load("script.php", { 'nom[]': ["Adam", "Eve"] } );
```

Le script va être appelé de façon asynchrone, ce passage de paramètres oblige un POST, la fonction de callback est appelée en fin de traitement et affiche le succès de l'appel AJAX.

Ce qu'il ne faut pas faire et pourquoi

Les techniques aperçues sont séduisantes et peuvent à peu de frais booster l'interactivité d'un site. Malgré ses atouts, AJAX n'est pas la panacée, il est même contre-indiqué dans certains cas de figure.

Ne pas utiliser l'asynchrone

Les appels asynchrones sont facilement réalisables et l'on peut même 'oublier' que l'on appelle un script de cette façon. La fonction `.load` de JQuery peut avoir cet inconvénient. Il peut être nécessaire de synchroniser des scripts, la contrainte existe lorsque ceux-ci doivent être exécutés dans un ordre précis. C'est le cas lorsque un script dépend du résultat d'un autre, ou lorsque des accès à une ressource sont critiques et doivent être verrouillées. Le fait qu'une méthode de *callback* est mise en place, peut laisser dans l'ignorance l'utilisateur sur le résultat de la réponse. Au cas où la réponse tarde, l'utilisateur peut continuer à agir et modifier son environnement et le retour du script appelé ne pourrait plus être en adéquation avec l'état de la page à l'instant zéro.

Ne pas utiliser Ajax

On peut ne pas être amené à utiliser AJAX, si les navigateurs sont trop vieux et ne supportent pas un JavaScript récent a fortiori *XMLHttpRequest*. Un utilisateur peut avoir désactivé son interpréteur JavaScript. Cas un peu plus rare, les navigateurs en mode texte pourraient avoir quelques difficultés avec la modification du DOM...

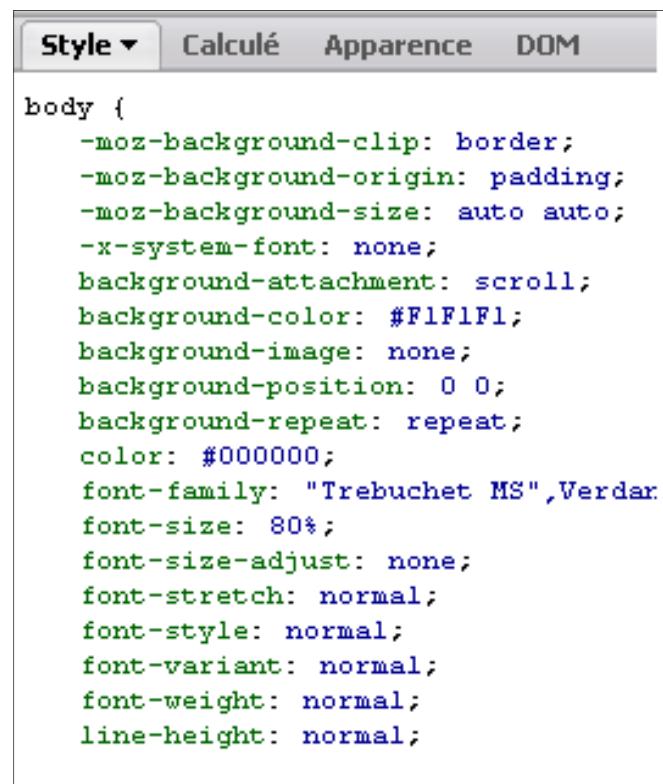
Des problèmes de navigabilité

L'indexation des pages se font sur la page principale, mais pas sur les contenus amenés par AJAX. Les fonctionnalités du navigateur 'précédent, suivant' ne peuvent s'exercer sur le contenu dynamique, et ainsi pénaliser l'utilisateur.

Des surcharges de maintenance et complexité du code

La course sans cesse a une meilleur ergonomie, interactivité avec le serveur, un SGBD, et une surenchère de fonctionnalités *desktop-like*, peut a contrario, surcharger le navigateur et le réseau.

La maintenance du code peut devenir problématique et des conflits peuvent apparaître si on utilise plusieurs frameworks, engendrant des bugs difficilement détectables, surtout en mode asynchrone.



```
body {  
  -moz-background-clip: border;  
  -moz-background-origin: padding;  
  -moz-background-size: auto auto;  
  -x-system-font: none;  
  background-attachment: scroll;  
  background-color: #F1F1F1;  
  background-image: none;  
  background-position: 0 0;  
  background-repeat: repeat;  
  color: #000000;  
  font-family: "Trebuchet MS", Verdana;  
  font-size: 80%;  
  font-size-adjust: none;  
  font-stretch: normal;  
  font-style: normal;  
  font-variant: normal;  
  font-weight: normal;  
  line-height: normal;
```

Figure 9. Visualisation du CSS

Dans le cas de plusieurs objets asynchrones, les problématiques d'accès concurrentiel et de gestion des ressources complexifieront les tâches d'écriture, de tests et de débogages.

La nature même d'AJAX, un ensemble de technologies, devrait interpeller le développeur sur la nécessité de séparer les scripts de natures différentes, d'utiliser si possible un modèle MVC, et d'avoir une approche pragmatique de son emploi certes nécessaire, mais pas toujours adéquat.

Avenir

AJAX est une conception et une articulation autour d'un objet *XMLHttpRequest*. L'évolution de cet ensemble sera la résultante de l'évolution et de la cohérence de ses composants. DOM et CSS continuent leurs progressions au rythme des recommandations du W3C, l'intégration de HTML 5 est en route. Les fonctions de timing, le retour d'objet *BLOB*, ainsi que l'intégration du *cross-domain*, dans de nouveau objet comme *XDomainRequest* de Microsoft ou bien la nouvelle *XMLHttpRequest 2*, permettront de franchir nativement ces barrières.

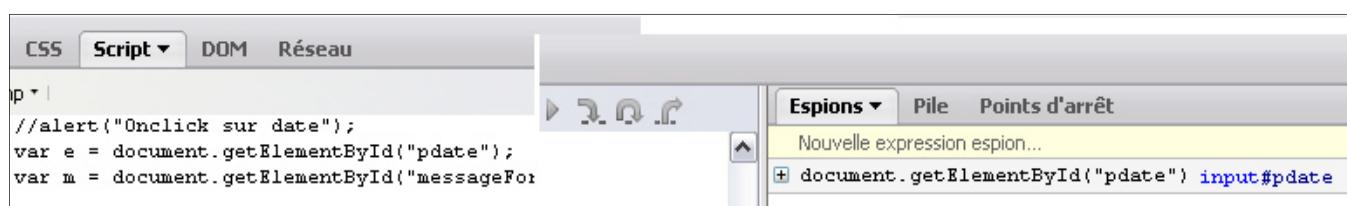


Figure 10. Débogueur du JavaScript

Sur Internet

- <http://www.adaptivepath.com/ideas/essays/archives/000385.php> – Article de référence,
- <http://ajax.developpez.com/> – Site d'information sur le développement Ajax entre autres,
- <http://www.w3.org/TR/XMLHttpRequest/> – W3C du XMLHttpRequest,
- <http://fr.php.net/manual/fr/book.simplexml.php> – PHP : simplexml,
- <http://fr.php.net/manual/fr/book.json.php> – PHP : JSON,
- <http://tools.ietf.org/html/rfc4627> – Mémo sur JSON,
- <http://msdn.microsoft.com/en-us/library/ms535874%28v=VS.85%29.aspx> – XMLHttpRequest Object,
- <http://msdn.microsoft.com/en-us/library/cc288060%28v=VS.85%29.aspx> – XDomainRequest Object,
- <https://developer.mozilla.org/en/XMLHttpRequest> – XMLHttpRequest de Mozilla,
- <http://code.google.com/p/sqlite-manager/> – plugin Mozilla.

La Figure 12 d'attraction BLOB (BAG *blob attract graph*), montre l'évolution suggestive de XMLHttpRequest vers XMLHttpRequest version 2. L'objet asynchrone appelé XHR devient un attracteur plus orienté cross-domain, ce qui nous amène à redéfinir AJAX en crossAJAX. Le cross-domain est maintenant natif à AJAX, ce qui correspond à la demande de croissante de nouveaux services externes et technologies distantes (Yahoo, Google, Amazon, Twitter). Par cette modification de conformité, l'attraction des autres technologies, telles que CSS 3 et HTML 5 forment autour de JavaScript une plus grande cohésion.

Pour HTML 5, l'intégration audio-vidéo simple et robuste, contrôle de formulaire, *drag and drop*, gestion de

l'historique ne sont plus des rôles dévolus ou de façon moindre à Javascript. Pour CSS 3, graphisme sophistiqué et une interaction accrue avec le DOM. L'écriture du code devient plus rigoureuse et cela ne peut qu'améliorer la précision et la concision des pages ainsi que leurs robustesses.

D'autres part, ces dernières prennent à leurs charges des tâches qui auparavant étaient dévolues à JavaScript. On peut voir, sur le schéma un nouvel équilibre qui renforce la robustesse de l'application dans la répartition volumétrique de ses rôles propres. En ce sens crossAJAX à l'instar de AJAX à de l'avenir, s'il délaisse un peu les rôles de widgets-gadgets que l'on lui fait tenir trop souvent. Note : le BAG (*blob attract graph*) est un concept graphique intuitif et suggestif développé chez Kiwi-Engineering, de l'expression d'idées reposant sur l'attractivité – répulsion, compression – extension de ses constituants.

Conclusion

L'augmentation des débits Internet et de la diversité d'applications sur PC mais aussi de différents terminaux, mobiles, tablettes, assurent la présence incontournable d' AJAX. En revanche la pérennité d'une solution informatique est un leurre, celle-ci doit être adaptée, avec pragmatisme, à la problématique présente.

AJAX et le langage de l'éléphant peuvent faire bon ménage et cela de manière élégante. Cette interaction dynamique entre les deux mondes du serveur et du client peuvent donner l'illusion d'une communication inter-langage, une alchimie qui décuple la puissance de l'ensemble en s'appuyant sur les points forts respectifs de l'un et de l'autre.

JEAN TATAREAU

Diplômé en ingénierie et qualité du logiciel au CNAM de Bordeaux, l'auteur s'intéresse à la programmation et à la réalisation logicielle depuis de nombreuses années. Depuis plus de deux ans, il a créé et dirige la société Kiwi Engineering. Les domaines de prédilection sont le travail à façon d'applicatifs pour le professionnel ou le particulier au travers d'interfaces Web.

Contact <http://www.kiwi-engineering.net>.

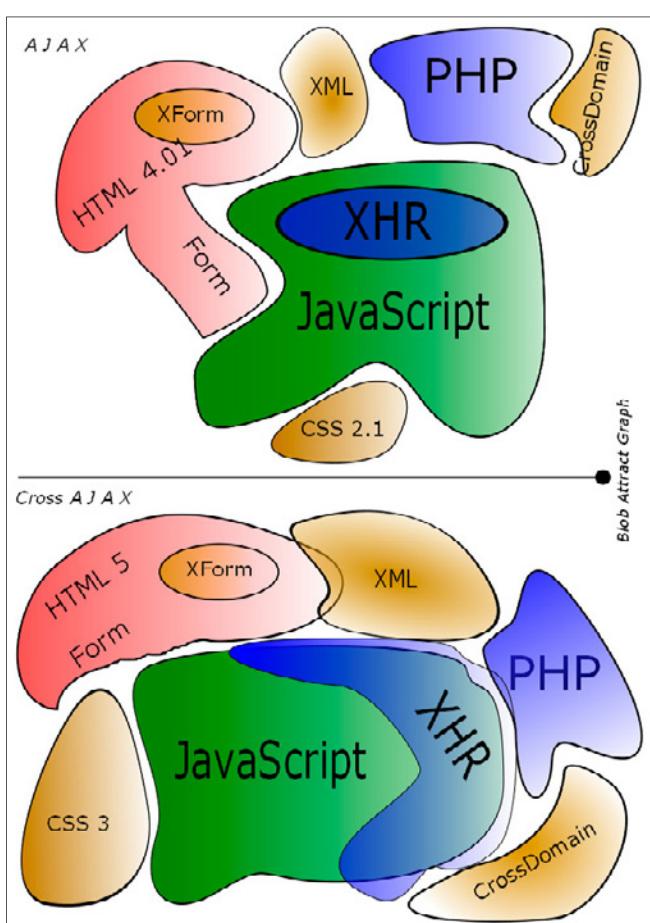


Figure 12. Graphe d'attraction BLOB

Créer son propre hébergement

Nous verrons dans cet article comment créer un serveur d'hébergement web avec un minimum de sécurité.

Dans une première partie nous installerons/configurerons un AMP (Apache + MySQL + PHP), puis nous installerons des logiciels tierces pour les serveurs FTP, DNS, etc.

Cet article explique :

- Installer et configurer un serveur Web (Apache, MySQL, PHP).
 - Installer et configurer des services d'hébergement (FTP, DNS).
 - Installer et configurer les utilitaires (phpMyAdmin).
 - Mettre un minimum de sécurité.
 - Créer un compte utilisateur.
 - La législation et l'hébergement web.

Ce qu'il faut savoir :

- Utiliser la ligne de commande en console d'un linux (debian ou pas).
 - Nous admettrons que IPS représente l'IP du serveur où vous installez votre serveur d'hébergements de site (127.0.0.1 pour une installation sur le poste local).

Nous travaillerons ici sur un serveur Debian 5.0 Lenny installé sans aucun logiciel supplémentaire.

Installation et configuration de l'AMP, le serveur web

Qui dit hébergement de site, dit serveur web. Nous resterons sur le plus utilisé au monde, Apache. Pour cela nous utiliserons la commande aptitude qui gère les paquets de Debian aptitude install apache2. Normalement si vous affichez l'IP de votre machine sur un navigateur vous devriez avoir la mythique page *It works !*.

Le serveur MySQL et PHP

Maintenant nous allons nous attaquer à PHP et MySQL.

```
aptitude install php5 php5-mysql mysql-server  
mysql-client mysql-common
```

Il peut vous demander un mot de passe pour mysql, nous vous conseillons de mettre un mot de passe différent de celui du système. Il faut redémarrer apache pour que les modifications soient prises en compte.

```
/etc/init.d/apache2 restart
```

Maintenant vous avez un serveur web avec PHP et MySQL d'installés. Pour vérifier que tout fonctionne

nous allons créer un fichier de visualisation de la configuration de PHP. En CLI, tapez : *nano /var/www/index.php*. Coller le contenu du Listing 1. Faites ensuite Ctrl X puis O (Y pour les Debian anglaises).

Rendez-vous dans votre navigateur à l'adresse suivante : <http://IPS/index.php>. Rappelons le, le répertoire par défaut d'apache est /var/www/. Vous devriez trouver



Figure 1. Résultat d'un `phpinfo()`

cette page. Si ce n'est pas le cas, c'est que tout ne s'est pas installé correctement. Dans ce cas, redémarrez apache2 et réessayez, si cela ne fonctionne toujours pas recommencez les aptitudes.

Conclusion

Vous disposez dès à présent d'un serveur apache avec PHP/MySQL.

Installation des services d'hébergements

Nous verrons ici comment installer les services minimums liés à l'hébergement de sites web.

Le serveur FTP

Dans le cas d'un hébergement, il est en général plus intéressant de proposer un protocole de dépôt de fichier tel que le FTP. Personnellement je préfère proftpd, mais il en existe de nombreux autres.

Commençons par l'installer : aptitude install proftpd. Il vous demandera de sélectionner le mode d'utilisation, il y a 2 cas de figure :

- Vous pensez accueillir peu de monde, juste pour un hébergement semi personnel alors il est conseillé de sélectionner inetd qui consommera beaucoup moins de ressources.
- Vous allez faire un vrai hébergement sur internet pour héberger toute la toile (c'est viser un peu haut mais qui sait) alors sélectionner standalone qui assure un très grand débit.

Notez que nous pouvons à tout moment changer ce paramètre dans le fichier de configuration de *proftpd*. Nous allons maintenant modifier un peu la configuration, ouvrez le fichier *proftpd* :

```
nano /etc/proftpd/proftpd.conf
```

Ligne 14 : `ServerName <LE-NOMDEVOTRESERVEUR>`
Ligne 33 : retirer le `#` devant `<< DefaultRoot ~ >>`

Enregistrer et quitter le fichier. Si vous avez mis une installation en standalone tapez la commande `/etc/init.d/proftpd restart`.

Le paramètre `DefaultRoot` permet de bloquer l'utilisateur dans son dossier client, en effet sans ce paramètre activé il pourrait voyager dans tout le disque dur du serveur (ou les permissions système le permettent). Vous avez un serveur FTP fonctionnel !

Le serveur DNS

Tout d'abord qu'est-ce qu'un serveur DNS (*Domain Name Server*) ? C'est un serveur qui convertit un nom en une IP. Il peut s'avérer utile pour héberger des sites

Listing 1. Fichier index.php

```
<?php  
phpinfo();  
?>
```

Listing 2. Fichier DNS named.conf

```
zone "mondomaine.fr" IN {  
type master;  
file "/etc/bind/db.mondomaine.fr";  
};
```

Listing 3. Fichier de zone DNS

```
$TTL 86400  
@ IN SOA mondomaine.fr. monadresse.email.fr. (2010090501; Numéro de série sous la forme AAAAMMMJNN 28800; Temps de rafraîchissement 7200 ; Temps avant nouvel essai 864000 ; Temps d'expiration 86400 ; Min TTL )  
NS premier.dns.fr.  
NS second.dns.com.  
mondomaine.fr. A 123.456.789.123  
monsousdomaine CNAME mondomaine.fr.
```

Listing 4. Fichier portsentry.conf

```
ADVANCED_PORTS_TCP="1024" # Valeur du port maximum à utiliser en mode avancé en TCP  
ADVANCED_PORTS_UDP="1024" # Valeur du port maximum à utiliser en mode avancée en UDP  
ADVANCED_EXCLUDE_TCP="113,139" # On exclue ces ports du TCP  
ADVANCED_EXCLUDE_UDP="520,138,137,67" #On exclue ces ports de l'UDP  
IGNORE_FILE="/etc/portsentry/portsentry.ignore" #Fichier des IPs ignorées  
HISTORY_FILE="/var/lib/portsentry/portsentry.history" # Le fichier de logs  
BLOCKED_FILE="/var/lib/portsentry/portsentry.blocked" # Le fichier des utilisateurs bloqués  
RESOLVE_HOST = "0" # on désactive la résolution d'host (charge système)  
BLOCK_UDP="1" # On active le blocage de l'UDP  
BLOCK_TCP="1" # Blocage du TCP  
KILL_ROUTE="/sbin/iptables -I INPUT -s $TARGET$ -j DROP" # Commande de blocage de l'attaquant ($TARGET$) représente l'IP.  
SCAN_TRIGGER="0" # Nombre de tentative tolérée
```

Listing 5. Fichier portsentry

```
TCP_MODE="atcp"  
UDP_MODE="audp"
```

Listing 6. Extrait filter.d

```
[Definition]  
failregex = \(\S+<HOST>\)\[: -]+\ Maximum login attempts \(\d+\) exceeded$ #LE fail regex contient toutes les regex des attaque possible, <HOST> représente l'IP dans la chaîne.  
ignoreregex = # On peut ignorer certaine regex précise, si ce n'est pas le cas (souvent) on laisse vide.
```

Listing 7. Extrait jail.conf

```
[ssh]  
enabled = true # Activé  
port = ssh # Port à bloquer  
filter = sshd # nom du filtre (correspondant au nom du fichier dans filter.d)  
logpath = /var/log/auth.log #Fichier de log  
maxretry = 2 # Nombre de tentative maximum
```

Listing 8. Exécution de la commande adduser

```
ns:~# adduser pseudo
Adding user 'pseudo' ...
Adding new group 'pseudo' (1000) ...
Adding new user 'at' (1000) with group 'pseudo' ...
Creating home directory '/home/pseudo' ...
Copying files from '/etc/skel' ...
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
Changing the user information for pseudo
Enter the new value, or press ENTER for the default
    Full Name []:
    Room Number []:
    Work Phone []:
    Home Phone []:
    Other []:
Is the information correct? [y/n]
```

Listing 9. Fichier de vhost d'apache

```
<VirtualHost *:80>
DocumentRoot /home/pseudo/
ServerName pseudo.domaine.fr
<Location />
php_admin_valueopen_basedir /home/pseudo/
php_admin_valueupload_tmp_dir "/home/pseudo/mtmp/"
</Location>
</VirtualHost>
```

à moins que vous ne souhaitiez donner comme url à vos membres votre IPS. Je n'en connais que 2, le plus répandu est sans nul doute BIND assez simple de configuration. C'est celui que nous utiliserons ici, sinon il y a mydns qui lui utilise MySQL pour les enregistrements (évite la création de fichier de zone que nous allons voir par la suite). Nous ne verrons que le cas où le serveur est DNS primaire puisqu'en général on utilise des serveurs secondaires fournis par un prestataire.

Commençons par l'installer : aptitude install bind9 (Note : Il se peut que ce dernier soit déjà installé sur votre système auquel cas il n'installera rien, ce n'est pas un problème). Donc pour configurer les DNS de nos futurs domaines nous allons avoir besoin de modifier la configuration principale du serveur DNS et de créer le fichier de zone.Modifier le fichier /etc/bind/named.conf nano /etc/bind/named.conf. Allez en bas du fichier et ajouter le texte du Listing 2.

Quitter nano en enregistrant votre fichier. Maintenant nous allons créer le fichier de zone du domaine : nano /etc/bind/db.mondomaine.fr. Quitter nano en enregistrant votre fichier. Maintenant il faut recharger la configuration de bind : pour cela taper /etc/init.d/bind9 reload. Pour vérifier que votre serveur est bien configuré je vous invite à effectuer un zone check via http://www.zone-check.fr/demo/. Pour cela renseignez en zone votre nom de domaine et en DNS vos serveurs DNS. Vous pouvez voir un exemple de fichier de zone dans le listing 3.

Le serveur Mail

Nous verrons ici une configuration très basique d'un serveur e-mail puisque nous allons l'utiliser uniquement pour l'envoi d'e-mail. Il faudrait un article complet pour

expliquer le fonctionnement d'un serveur mail. Vous trouverez à la fin de cet article un lien contenant une installation très détaillée d'un serveur mail et de toute sa sécurité.

```
aptitude install postfix
```

Sélectionnez Site Internet, en nom de serveur *votre-nomdedomaine.fr*. Par défaut un serveur postfix n'envoie que des emails provenant exclusivement de localhost, donc pas de configuration à faire du côté de la configuration générale.

Conclusion

Vous venez d'installer le nécessaire pour procéder à un hébergement mutualisé, vos utilisateurs peuvent se connecter en FTP, vous avez votre propre serveur de DNS et un serveur mail pour l'envoi des e-mails depuis PHP.

Installation des utilitaires

Nous allons maintenant installer des petits utilitaires pour vous aider à gérer votre plateforme.

phpMyAdmin

phpMyAdmin est une application PHP permettant de gérer les bases de données MySQL. Pour l'installer nous allons utiliser comme à notre habitude aptitude, cependant sachez que vous pouvez l'installer en le téléchargeant directement sur le site officiel de phpMyAdmin c'est une simple archive à décompresser dans un dossier web.

```
aptitude install phpmyadmin
```

Dans l'installation sélectionner apache2, une fois l'installation terminée il faut recharger la configuration d'apache via la commande /etc/init.d/apache2 reload. Maintenant rendez-vous sur <http://IPS/phpmyadmin/> et connectez-vous via l'identifiant root et le mot de passe mis dans l'installation.

Logwatch

Logwatch est un programme qui vous envoie tous les matins un récapitulatif analysé de vos logs serveurs. Par exemple pour apache vous recevez toutes les adresses renvoyées en erreur ou même des tentatives de piratages via des injections, il affiche aussi toutes les connexions proftpd et ssh avec leur IP.

```
aptitude install logwatch
```

Nous allons modifier le fichier de configuration pour qu'il envoie son e-mail à notre boite e-mail.

```
nano /usr/share/logwatch/default.conf/
logwatch.conf
```

Ligne 44 : MailTo = root remplacé le par
 MailTo = *monadresse@email.fr*
 Enregistrer et quitter.

La sécurité d'un hébergeur, le pare-feu

La première sécurité d'un serveur réside dans son pare-feu. Je vous conseille d'utiliser iptables pour le configurer. Rappelons-le iptables n'est pas le pare-feu de linux mais juste l'utilitaire pour le configurer. Le pare-feu contenu dans le kernel s'appelle netfilter.

Je ne ferai pas une configuration complète d'un pare-feu (il en existe beaucoup sur la toile), mais juste une explication simple. Iptables est constitué selon 3 tables (on peut en ajouter d'autre) :

- INPUT, c'est le trafic entrant.
- OUTPUT, c'est le trafic sortant.
- FORWARD, c'est le trafic qu'on transfère.

Le fonctionnement est très simple : il applique toutes les règles de ses tables en fonction des paquets en transit s'il y a une règle lui correspondant, il applique ce qu'on lui a demandé. Dans le cas contraire il prend ce qu'on appelle la POLICY de la table et applique l'ordre.

Les mise à jours systèmes et logiciels

Nous allons ici aborder l'aspect sécuritaire d'un hébergement web. Tout d'abord il faut mettre à jour quotidiennement son serveur via ses 2 commandes :

```
aptitude update
aptitude safe-upgrade
```

En cas de changement de kernel pensez à redémarrer le serveur.

En cas d'installations faites en dehors de la commande aptitude, phpMyAdmin par exemple, pensez à vérifier les nouvelles versions, et de surtout mettre à jour si une nouvelle version est annoncée. Pour parler pratique, 200 serveurs se sont fait stopper le 10/08/2010 chez le fournisseur de serveur à la hâche pour une attaque phpMyAdmin.

Portsentry

Nous allons maintenant voir portsentry qui est un utilitaire d'anti scan de port.

```
aptitude install portsentry
```

Modifier la configuration en ouvrant */etc/portsentry/portsentry.conf* et placez y le contenu.

Maintenant mettez le contenu du listing 5 dans le fichier de configuration du mode par défaut de *portsentry* : */etc/default/portsentry*

```
/etc/init.d/portsentry restart
```

Terminologie

- LAMP : Serveur composé de Linux, Apache, MySQL et Php.
- Netfilter : Le module du kernel linux qui sert de par feu
- Iptables : Programme permettant d'interagir avec netfilter
- POLICY : Règle par défaut du par feu

Vous remarquerez que l'on ne fait qu'ajouter un 'a' dans les modes, pourquoi ? Car on le met en mode avancé, ce qui nous permet un scan de tous les ports inutilisé jusqu'à 1024.

Fail2ban

Fail2ban est un programme qui permet de bannir un utilisateur après un certain nombre d'essai de mots de passe erronés.

```
aptitude install fail2ban
```

Maintenant rendons nous dans */etc/fail2ban/*. Vous y trouverez deux fichiers et deux dossiers.

- *action.d* : Ce dossier contient tous les fichiers de configuration des actions à faire pour bannir/en - voyer un email lorsque une attaque est détectée.
- *filter.d* : Ce dossier contient toutes les règles de détection sous le format du Listing 6.
- *fail2ban.conf* : Ce fichier contient la configuration du programme, Il y a 3 paramètres possible de base :
 - *loglevel* : De 1 à 4, 1 représentant le moins d'information, par défaut je recommande 3.
 - *logtarget* : Le chemin du fichier de log des tentatives d'attaque bloquée.
 - *socket* : chemin du fichier de socket.
- *jail.conf* : Le fichier *jail.conf* contient tous les paramètres des règles (activation, fichier de log, tentative maximum, etc). Il est composé en 2 parties :
 - La première est une configuration de base elle commence par un [DEFAULT] on y trouve principalement les paramètres suivants :
 - *ignoreip* : Liste des IPs à ignorer, si vous êtes en IP fixe ne pas hésiter à l'ajouter car 3 essaies ou moins peuvent vite arriver.
 - *bantime* : Temps de bannissement par défaut d'une IP en seconde.
 - *maxretry* : Nombre d'essais maximum.
 - La seconde (cf. Listing 7) est composée des paramètres de la première partie qu'on modifie (si on veut utiliser les valeurs par défaut on ne les redéfini pas) et d'autres paramètres.

Une fois vos règles déterminées, il ne reste plus qu'à redémarrer le service :

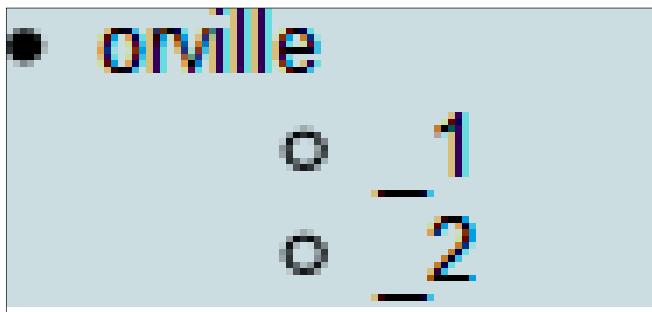


Figure 2. Résultat d'une charte de nommage dans phpMyAdmin

/etc/init.d/fail2ban restart.

Le reste

Il existe énormément de logiciels de sécurité pouvant contrer chacune des attaques de certain type, je vous en conseille quelques autres :

- *rkhunter* : un *anti rookit* puissant,
- *snort* : un NIDS (non pas un nid d'oiseau mais un détecteur d'intrusion),
- *kernel* : Le *kernel linux* contient énormément de sécurité contre les attaques comme le *syn cookie* ou encore le *rp filter* et bien d'autres encore.

Conclusion

Vous l'aurez compris, la sécurité d'un hébergeur est très importante. En effet vous donnez accès à votre serveur à des personnes, il est donc nécessaire de bien se protéger. N'oubliez pas que plus vous installer de programme, plus il y aura de failles potentielles sur votre serveur.

Créer un compte utilisateur

Nous allons maintenant voir comment créer un compte utilisateur pour héberger son site, pour cela nous allons commencer par lui créer un compte utilisateur système : `adduser pseudo`.

En changeant bien sûr le pseudo. Il va vous demander plusieurs informations d'abord le mot de passe du compte à confirmer, puis des informations comme le numéro de téléphone qui sont facultatives. Vous pouvez voir un exemple dans le Listing 8.

Maintenant que l'utilisateur est créé il a accès à son FTP. Il faut maintenant créer son espace web pour cela nous allons créer le fichier `/etc/apache2/sites-available/pseudo` et y mettre le contenu du Listing 9.

Sur Internet

- <http://www.rhien.org/> – Le Réseau d'Hébergeurs Indépendants et ENgagés
- <http://www.starbridge.org/spip/spip.php?article12> – Installer un serveur mail géré par MySQL avec anti-virus, anti-spam, blacklist, etc.

```
mkdir /webuser/pseudo/mtmp/  
a2ensite pseudo  
/etc/init.d/apache2 reload
```

Si vous vous rendez sur <http://pseudo.domaine.fr>, vous y trouverez votre site.

Pour créer une base de données il suffit de vous rendre sur phpMyAdmin, de créer un utilisateur via l'onglet privilège et de cocher l'option *Créer une base portant son nom et donner à cet utilisateur tous les priviléges sur cette base*. Nous vous conseillons d'adopter une charte de nommage pour les bases de données SQL, en effet phpMyAdmin range les bases de données par numéro, par exemple si il y a une base nommée *pseudo_1* et une autre *pseudo_2* il les regroupera sous *pseudo* avec 2 déclinaison *_1* et *_2* comme représenté dans la Figure 2.

Conclusion

L'*open base dir* permet d'empêcher l'utilisateur, d'aller dans d'autres répertoires que le sien ce qui garantit une certaine sécurité. Pour créer quelques utilisateurs cette méthode vous conviendra, cependant en cas d'affluence il faudrait penser à en faire un petit script bash.

La législation et l'hébergement web

Nous avons vu dans les précédents chapitres tout ce qu'il fallait savoir pour lancer un petit hébergement mutualisé. Cependant il ne faut pas s'y prendre à la légère, la législation française est assez claire pour ce genre de cas.

Déjà il faut savoir que vous devez sauvegarder une année de log. En effet vous devez être capable d'identifier la personne qui aura déposé des fichiers sur votre serveur, dans le cas contraire la justice pourra considérer que c'est vous qui les avait déposés. Hors pour sauvegarder de telles données vous devez faire une demande à la CNIL (gratuit et réalisable en 10 minutes sur internet). De plus si vous faites des newsletters ou une inscription demandant une information personnelle il est aussi obligatoire de faire une déclaration à la CNIL. Vous pouvez dans certains cas être dispensé de déclaration à la CNIL pour vous renseigner, vous pouvez les contacter au 01 53 73 22 22.

ALEXANDRE BAZZET

L'auteur est étudiant en 3ème année à SUPINFO et formateur STA Linux ; fondateur d'Ice Groups association d'informatique et d'Ice Héberg services d'hébergement gratuit et payant.

SQL : langage de définition des données

Les bases de données sont très utilisées dans les applications Web. La création, l'interrogation et la manipulation des données de la base sont réalisées en SQL. Dans cette série d'articles vous apprendrez le langage SQL ainsi que les bases nécessaires pour communiquer avec une base de données à partir d'un script PHP.

Cet article explique :

- Définition d'une base de données.
- Présentation du langage SQL.
- Création de bases et de tables.
- Gestion des contraintes.

Ce qu'il faut savoir :

- Aucun prérequis.

De nombreuses applications Web (forums, galeries d'images, sites marchands, ...) reposent sur des bases de données. L'interaction avec la base de données, la création de la base et des tables, la manipulation des données, ainsi que le contrôle des droits d'accès aux données sont réalisés avec le langage SQL.

Dans cette série d'articles sur SQL, vous allez apprendre les notions qui vous permettront de créer une base de données et d'ajouter, consulter, modifier ou supprimer des informations dans cette base. Vous verrez comment communiquer avec elle via une console SQL, l'interface graphique PhpMyAdmin ou depuis un script PHP.

Pour appliquer ce que vous allez apprendre dans cet article, il est nécessaire d'installer un serveur de base de données MySQL, de préférence la version 5, un serveur web avec PHP version 5 et le client graphique phpMyAdmin. Les distributions XAMPP (Windows, Linux, Mac OS), WAMP (Windows), EasyPHP (Windows) ou MAMP (Mac OS) vous fourniront l'environnement de travail nécessaire pour communiquer avec MySQL et créer des applications Web.

L'article sera illustré par la création d'une base de données très simple concernant la gestion d'une bibliothèque privée.

Base de données

Une base de données est un ensemble structuré d'informations centralisées ou réparties. Un ou plusieurs utilisateurs, ou des programmes, peuvent manipuler les

données de la base en fonction des droits qui leur ont été accordés.

SGBD

L'interaction avec la base de données ainsi que la création de la base sont réalisées par un système de gestion de base de données (SGBD). Ce dernier est composé de logiciels permettant :

- la création de la base, le choix du type de données, la gestion des contraintes,
- l'ajout, la suppression, la modification ou la consultation des données,
- la gestion des mots de passe et des priviléges des utilisateurs sur les données,
- le partage des ressources en prenant en compte les accès concurrents,
- l'annulation de transactions,
- la restauration de la base en cas de destruction et la reprise automatique en cas d'incident.

Un des modèles les plus répandus de SGBD est le relationnel, introduit à la fin des années 1970 (Ingres, Oracle, ...). Dans ce modèle, les données sont représentées sous la forme de tables à deux dimensions. Par exemple, un auteur pourrait être stocké dans une ligne d'une table nommée auteur, dont les colonnes représenteraient les champs nom, prénom, date de naissance, etc... Cette table comporterait autant de lignes qu'il y a d'auteurs distincts.

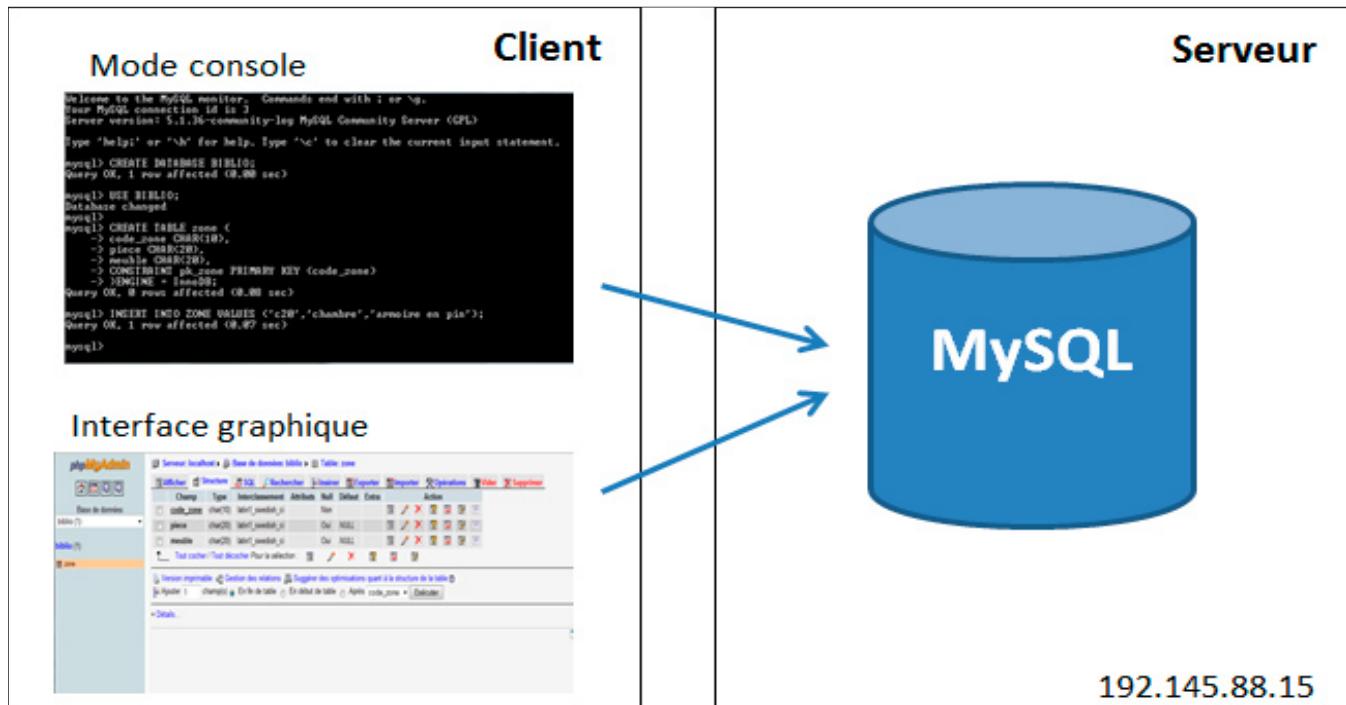


Figure 1. Client/Serveur

SQL

SQL (*Structured Query Language*) est le langage utilisé dans les bases de données relationnelles pour la manipulation des données, leur définition et le contrôle de l'accès aux données. C'est un langage non procédural issu du langage SEQUEL (*Structured English Query Language*) développé par IBM. SQL a été standardisé en 1986 (ANSI SQL/86). Les versions SQL2 et SQL3 sont des révisions du standard SQL, datant respectivement de 1992 et de 1999. SQL2008 est la dernière version de la norme.

Les principaux SGBD relationnels du marché respectent le niveau *Entry Level* de la norme SQL2, c'est-à-dire qu'ils implémentent l'ensemble minimum d'instructions définies dans la norme. Chaque SGBD propose des fonctionnalités supplémentaires (fonctions, types de données, ...).

Le langage SQL comporte plusieurs parties :

- Le langage de définition des données (LDD) permet de définir la structure de la base : création, modification, suppression de tables ou de bases, ajout de contraintes,
- Le langage de contrôle des données (LCD) permet de définir les priviléges des utilisateurs sur les données,
- Le langage de manipulation des données (LMD) permet d'insérer des données dans la base, de les extraire, de les modifier ou de les supprimer.

Dans cet article, vous allez apprendre à utiliser le LDD et le LCD. Le LMD sera quant à lui présenté dans les prochains articles de cette série dédiée à SQL.

MySQL

De nombreuses plateformes de développement d'applications Web reposent sur MySQL. Ce SGBD client/serveur open source est distribué sous licence GPL et sous licence commerciale. MySQL est simple d'utilisation et rapide. Il est disponible pour de nombreux systèmes d'exploitation. C'est pourquoi les exemples qui vont illustrer cet article seront tous testés sous MySQL.

La communication avec un serveur de bases de données MySQL peut s'effectuer depuis un client MySQL en mode console, en utilisant le langage de requêtes SQL, ou en utilisant l'interface graphique web phpMyAdmin (Figure 1). L'utilisateur peut ainsi, depuis son navigateur, envoyer des requêtes à MySQL sans avoir recours au langage SQL. Les requêtes sont automatiquement créées par phpMyAdmin, à partir des données saisies et des actions effectuées dans l'interface. Il sera cependant toujours possible d'envoyer des requêtes à la base en utilisant la console SQL intégrée de phpMyAdmin.

Vous apprendrez à utiliser ces clients dans la dernière partie de l'article.

Création de la base et des utilisateurs

La commande `CREATE SCHEMA` permet de créer une base de données. La marque de fin d'instruction est le caractère point-virgule.

`CREATE SCHEMA nom_base;`

Dans l'exemple de cet article, la base contient les données d'une bibliothèque. La base dont le nom est `biblio` est créée par l'instruction suivante :

`CREATE SCHEMA biblio;`

Pour les débutants

Tableau 1. Différents types de données proposés par MySQL

Classe	Type	Description
Numérique (entiers)	TINYINT	Permet de stocker 2^8 nombres
	SMALLINT	Permet de stocker 2^16 nombres
	MEDIUMINT	Permet de stocker 2^24 nombres
	INTEGER	Permet de stocker 2^32 nombres
Numérique (réels)	FLOAT	Nombre à virgule flottante stocké sur 4 octets
	DOUBLE, REAL	Nombre à virgule flottante stocké sur 8 octets
	DECIMAL(M,D)	DECIMAL(5,2) stocke un nombre à virgule fixe comportant 5 chiffres dont 2 décimaux
Chaîne de caractères	CHAR(n)	CHAR(8) stocke une chaîne de longueur fixe sur 8 caractères
	VARCHAR(n)	VARCHAR(25) stocke une chaîne de longueur variable et de maximum 25 caractères
	TINYTEXT	Permet de stocker 2^8 caractères
	TEXT	Permet de stocker 2^16 caractères
	MEDIUMTEXT	Permet de stocker 2^24 caractères
	LONGTEXT	Permet de stocker 2^32 caractères
Binaire	TINYBLOB	Permet de stocker 2^8 caractères en objet binaire
	BLOB	Permet de stocker 2^16 caractères en objet binaire
	MEDIUMBLOB	Permet de stocker 2^24 caractères en objet binaire
	LONGBLOB	Permet de stocker 2^32 caractères en objet binaire
Liste de chaînes	ENUM('val1', 'val2', ...)	Le champ prend 0 ou une valeur parmi la liste proposée
	SET('val1', 'val2', ...)	Le champ peut prendre 0, une ou plusieurs valeurs de la liste proposée
Date et temps	DATE	'2010-10-23'
	DATETIME	'2010-10-23 09:14:05'
	YEAR	2010
	TIME	'09:14:05'
	TIMESTAMP	1287818045

Tableau 2. Structure de la table ZONE

Colonne	Type de données	Description
code_zone	chaîne	Clé primaire, identifiant de la zone composé de la première lettre de la pièce suivie du numéro de l'étagère
piece	chaîne	Nom de la pièce dans laquelle se situe le livre
meuble	chaîne	Description du meuble dans lequel se situe le livre

Par convention les mots réservés du langage (CREATE, SCHEMA, ...) s'écrivent en majuscules et les mots définis par l'utilisateur en minuscules (biblio).

Une fois la base de données conçue, il faut créer un utilisateur et lui donner des droits afin qu'il puisse accéder à la base. Il est important de donner des droits différents aux utilisateurs de la base pour la protéger, assurer la confidentialité des données et leur intégrité.

Pour accorder des droits, il faut suivre la syntaxe :

```
GRANT privilege1 [, privilege2, ...] ON objet
TO nom_utilisateur1 [, nom_utilisateur2,
...];
```

La liste des priviléges peut comprendre SELECT, DELETE, INSERT, UPDATE ou tous à la fois (ALL PRIVILEGES). Les crochets sont utilisés dans la syntaxe pour indiquer les parties optionnelles de la commande. Les droits SELECT, INSERT, UPDATE et DELETE permettent respectivement de consulter, d'ajouter, de mettre à jour ou de supprimer des données. Ces priviléges peuvent être accordés en une seule commande à un ou plusieurs utilisateurs sur un objet (une ou plusieurs tables).

Dans MySQL, il est possible de créer un utilisateur et de lui donner des priviléges dans la même commande. Les instructions suivantes donnent à l'utilisateur lecteur le droit de lecture sur la table livre de la base

`biblio` et tous les droits au bibliothécaire sur toutes les tables (caractère *) de la base `biblio`.

```
GRANT SELECT ON biblio.livre TO lecteur IDENTIFIED BY 'motdepasse1';
GRANT ALL PRIVILEGES ON biblio.* TO biblio-thecaire IDENTIFIED BY 'motdepasse2';
```

Il est possible de supprimer ultérieurement des priviléges avec la commande `REVOKE` :

```
REVOKE privilege1 [, privilege2, ...] ON objet
FROM nom_utilisateur;
```

Type de données

Les valeurs des colonnes d'une table sont définies sur un domaine. La norme SQL définit quatre grandes classes de types de données : les numériques, les chaînes de caractères, les binaires et les dates. Les différents SGBD du marché proposent un sous-ensemble standard de types de données ainsi que des types spécifiques. La liste des types disponibles est définie dans la documentation de chaque SGBD.

Numérique

Les types SQL2 `SMALLINT` et `INTEGER` stockent des nombres entiers, respectivement sur 2 ou 4 octets. Les types entiers disponibles dans le SGBD MySQL sont listés dans le Tableau 1. Le type peut être suivi par l'option `UNSIGNED`, celle-ci permet de préciser que les valeurs attendues doivent être positives (non signées). Le domaine des entiers positifs sera donc écrit dans MySQL :

```
INTEGER UNSIGNED
```

Suivant le type de données défini pour une colonne, les informations peuvent être modifiées lors d'une insertion ou d'une mise à jour. Par exemple, l'insertion d'un nombre négatif dans une colonne typée `UNSIGNED` peut entraîner le changement de la donnée en 0. Pour pallier cela, il faut paramétriser le mode SQL de MySQL en strict. Pour ce faire, il est nécessaire d'ajouter dans le fichier de configuration de MySQL (fichier `my.ini` sous Windows, `my.cnf` pour Linux et Mac OS X) la ligne suivante :

```
sql_mode = strict_all_tables
```

Ainsi, une mauvaise insertion sera rejetée et la donnée ne sera pas altérée.

Les types SQL2 `DECIMAL` et `FLOAT` stockent respectivement des nombres à virgule fixe (prix, notes, ...) et flottante (pourcentages, fractions, ...). La liste des types numériques disponibles dans le SGBD MySQL est donnée dans le Tableau 1. La définition des nombres à virgule fixe suit la syntaxe :

```
DECIMAL(M, D) [UNSIGNED]
```

M représente le nombre total de chiffres stockés dans la base, dont D décimales. C'est la somme du nombre D de chiffres derrière la virgule et du nombre de chiffres avant la virgule. Ainsi `DECIMAL(5,2)` permettra de représenter des nombres de -999.99 à 999.99 (cinq chiffres dont deux sont placés derrière la virgule).

Chaînes de caractères

Les types `CHAR` et `VARCHAR` stockent respectivement des chaînes de caractères de taille fixe ou variable. La taille maximale est indiquée entre parenthèses et doit être comprise entre 1 et 255 caractères :

```
CHAR(25)
```

Pour des chaînes de plus grande taille, il faut utiliser les types textes MySQL présentés dans le Tableau 1.

Le type `CHAR` est bien adapté pour stocker des informations dont la taille est identique ou varie peu (numéro d'immatriculation, numéro ISBN de livres, ...). Lorsque les valeurs entrées dans la colonne n'atteignent pas la taille définie, la chaîne est complétée par des espaces. Toutes les valeurs de la colonne sont stockées sur le même nombre d'octets.

Le type `VARCHAR` est utilisé lorsque les données ont une longueur variable (titre de livre, nom d'auteur, ...). Le nombre d'octets utilisé pour stocker chaque champ varie en fonction du nombre de caractères de la chaîne, plus un octet pour mémoriser cette taille.

Le choix du type de données a donc un impact sur l'espace disque utilisé par la base de données.

Si l'insertion est supérieure à la taille du champ, le comportement par défaut de MySQL sera de tronquer du texte. Si MySQL est en mode strict, l'insertion sera refusée.

Temps

Les types temporels de la norme SQL sont les suivants :

- `DATE` : une date comprenant le jour, le mois et l'année sur quatre chiffres,
- `TIME` : un horaire avec heure(s), minute(s) et seconde(s),
- `TIMESTAMP` : nombre de secondes depuis le 1er janvier 1970 à ce jour.

Le SGBD MySQL propose des types temporels supplémentaires comme indiqués dans le Tableau 1.

Création de tables

La commande `CREATE TABLE` ajoute une table dans la base. Elle permet de nommer la table, de définir le nom et le type de ses colonnes et de fixer les contraintes d'intégrité. La syntaxe de la commande est la suivante :

Listing 1. création base de données

```
-- creer la base
CREATE SCHEMA biblio;

-- creer les utilisateurs
GRANT SELECT ON biblio.livre TO lecteur IDENTIFIED BY
'motdepasse1';
GRANT ALL PRIVILEGES ON biblio.* TO bibliothecaire
IDENTIFIED BY 'motdepasse2';

-- selectionner la base de travail
USE biblio;

-- creer les tables
CREATE TABLE zone (
code_zone CHAR(10),
piece VARCHAR(20),
meuble VARCHAR(20),
CONSTRAINT pk_zone PRIMARY KEY (code_zone)
)ENGINE = InnoDB;

CREATE TABLE livre (
isbn CHAR(20),
titre VARCHAR(30) NOT NULL,
genre ENUM('roman','policier','theatre','historique','fantastique'),
date_parution YEAR,
nb_pages INTEGER UNSIGNED,
code_zone CHAR(10),
CONSTRAINT pk_livre PRIMARY KEY (isbn),
CONSTRAINT fk_zone FOREIGN KEY (code_zone) REFERENCES
zone(code_zone) ON DELETE SET NULL ON UPDATE CASCADE
)ENGINE = InnoDB;

-- ajouter une colonne
ALTER TABLE livre ADD (langue ENUM('francais','anglais','allemand','espagnol','chinois') DEFAULT 'francais');

-- afficher la liste des tables de la base
SHOW TABLES;

-- afficher la structure des tables
DESCRIBE zone;
DESCRIBE livre;
```

```
CREATE TABLE nom_table (
nom_colonne1 type [option],
nom_colonne2 type [option],
...
nom_colonneN type [option],
CONSTRAINT nom_contrainte1 type _ contrainte1,
CONSTRAINT nom_contrainte2 type _ contrainte2,
...
CONSTRAINT nom_contrainteN type _ contrainteN
);
```

Ces instructions vont être décrites dans les prochaines sections.

Pour créer une colonne, il faut suivre la syntaxe :

```
nom_colonne type [option]
```

Le type de la colonne est une contrainte d'intégrité de domaine. Il indique le domaine des valeurs de la colonne (cf. section *Types de données*). Par exemple, les colonnes `isbn` et `nb_pages` de la table `livre` (Tableau 3), sont créées par l'instruction :

```
isbn CHAR(20),
nb_pages INTEGER UNSIGNED
```

Le type peut être suivi d'une ou plusieurs options :

- **NOT NULL** : le champ doit être obligatoirement rempli, il ne peut pas contenir la valeur `NULL`. Cette valeur représente l'absence d'information, elle ne doit pas être confondue avec la valeur numérique 0 ou une chaîne de caractères vide,
- **UNIQUE** : les valeurs de la colonne doivent être distinctes,
- **DEFAULT valeur** : valeur attribuée par défaut au champ concerné lorsque celui-ci n'est pas renseigné,
- **AUTO_INCREMENT** : lorsqu'une ligne est ajoutée dans la table, une colonne de type `AUTO_INCREMENT` prendra un entier unique auto incrémenté dans ce champ. Cette option ne fait pas partie de la norme SQL mais est disponible dans MySQL.

La colonne `titre` de la table `livre` doit forcément être renseignée et la colonne `langue` peut seulement prendre une des valeurs prédéfinies par le biais d'une liste (`ENUM`), par défaut le français (Listing 1). Ainsi pour les créer, l'instruction est de la forme :

```
titre VARCHAR(30) NOT NULL,
langue ENUM('francais','anglais','allemand','espagnol','chinois') DEFAULT 'francais'
```

Les caractères autorisés dans les noms des objets peuvent varier selon le SGBD. Pour être compatible, il est préférable de commencer par une lettre de l'alphabet et de n'utiliser ensuite que les lettres de l'alphabet en minuscule et majuscule, les chiffres et le caractère souligné. Bien que MySQL supporte l'utilisation de mots réservés du langage (`create`, `insert`, ...), il est toutefois déconseillé de les utiliser pour nommer vos tables ou vos colonnes.

Suivant le système d'exploitation, le nom des tables et des colonnes peut être sensible à la casse.

Définition des contraintes

Les données de la base doivent rester conformes à ce qui est attendu. Afin de vérifier que les données restent cohérentes lors de la manipulation de la base, il est nécessaire de définir des contraintes d'intégrité. Celles-ci permettent de rejeter des actions qui ne respecteraient pas la cohérence de la base et peuvent être définies par la syntaxe :

```
CONSTRAINT nom_contrainte type _ contrainte
```

Il est nécessaire de leur donner un nom (norme SQL2) et de spécifier de quel type de contrainte il s'agit (clé primaire, clé unique, clé étrangère, vérification). La plupart des SGBD acceptent que les contraintes ne soient pas nommées explicitement lors de la création, dans ce cas c'est le SGBD

Tableau 3. Structure de la table LIVRE

Colonne	Type de données	Description
isbn	chaîne	Clé primaire, identifiant du livre composé de son numéro ISBN
titre	chaîne	Titre du livre
genre	énumération	Genre
date_parution	année	Année de parution
nb_pages	entier positif	Nombre de pages
code_zone	chaîne	Clé étrangère référençant la colonne code_zone de la table zone

Tableau 4. Contenu de la table ZONE

code_zone	piece	meuble
c10	Chambre	Armoire en pin
c20	Chambre	Armoire en pin
s8	Salon	Bibliotheque noire

Tableau 5. Contenu de la table LIVRE

isbn	titre	genre	date_parution	langue	nb_pages	code_zone
128-5-56985-5	Fall of giants	historique	2010	anglais	255	c20
598-5-55596-2	Notre Dame de Paris	roman	1831	francais	123	s8
102-2-35419-5	Les fourmis	roman	1991	francais	300	c10
523-5-65472-9	David Copperfield	roman	1850	anglais	458	s8
320-2-02365-5	Le bourgeois gentilhomme	theatre	1670	francais	152	NULL
152-5-55695-2	Le seigneur des anneaux	fantastique	1954	francais	832	c10

qui attribue un nom par défaut. Nommer les contraintes permet d'une part de les modifier plus facilement et, d'autre part, d'obtenir des messages d'erreur plus clairs que ceux qui font référence à une contrainte dont le nom a été attribué automatiquement par le SGBD. Le nom donné à la contrainte doit être unique pour toute la base. Il devrait être composé du nom de la table préfixé par une abréviation représentant le type de contrainte. Par exemple, la clé primaire dans la table `auteur` sera nommée `pk_auteur` (`pk` = *primary key*) et une clé étrangère dans la table `livre` pourra être nommée `fk_zone` (`fk` = *foreign key*).

Clé primaire

Une clé primaire est un sous ensemble minimal de colonnes d'une table dont les valeurs identifient de manière unique une ligne de la table. Toute table d'une base de données requiert une clé primaire qui doit être renseignée pour chaque enregistrement. Par exemple dans la table `livre`, la clé primaire est le numéro ISBN car il représente un seul et unique livre (Tableaux 3 et 5).

Une contrainte de clé primaire peut porter sur une ou plusieurs colonnes et s'écrit de la manière suivante :

```
PRIMARY KEY (col1 [, col2, ...])
```

L'instruction qui suit indique que la clé primaire de la table `livre` est la colonne `isbn` (Tableau 3) :

```
CONSTRAINT pk_livre PRIMARY KEY (isbn)
```

Clé unique

Une clé unique est un sous ensemble de colonnes d'une table pour lequel aucun doublon n'est autorisé.

Une contrainte de type clé unique s'écrit :

```
UNIQUE KEY (col1 [, col2, ...])
```

Une clé primaire est unique et non nulle, c'est-à-dire que la ou les colonnes de la clé ne peuvent pas contenir la valeur `NULL` et qu'une même valeur ne peut pas apparaître plus d'une fois.

Clé étrangère

Une clé étrangère est un sous ensemble de colonnes d'une table qui fait référence à une clé primaire ou unique d'une autre table. Les colonnes qui constituent la clé étrangère doivent avoir le même type que les colonnes référencées. Par exemple, dans la table `livre`, la colonne `code_zone` est clé étrangère et référence la clé primaire `code_zone` de la table `zone` (Tableau 3). Un livre ne peut pas référencer une zone qui ne serait pas enregistrée dans la base. Il est cependant possible d'enregistrer un livre qui ne serait pas encore rangé à sa place en insérant la valeur `NULL` dans le champ `code_zone` (Tableau 5).

L'utilisation d'une clé étrangère empêche le classement de certains livres dans des zones qui n'existent pas, ce qui garantit la cohérence de la base.

Une contrainte de type clé étrangère doit être définie ainsi :

```
FOREIGN KEY (col1 [, col2, ...]) REFERENCES nom_
table (col1 [, col2, ...]) [clause action]
```

A la suite de `FOREIGN KEY`, il faut lister la ou les colonnes de la table qui sont clés étrangères. Ces colonnes référencent les colonnes d'une autre table dont le nom est indiqué à la suite de `REFERENCES`. Généralement, les colonnes clés étrangères portent le même nom que les colonnes qu'elles référencent.

La création d'une table comportant des clés étrangères doit être faite après la création des tables référencées sinon une erreur sera générée.

Dans la table `livre`, la colonne `code_zone` est une clé étrangère qui référence la clé primaire `code_zone` de la table `zone`. L'instruction suivante permet de le spécifier :

```
CONSTRAINT fk_zone FOREIGN KEY (code_zone) RE-
FENCES zone(code_zone)
```

Par défaut, lorsqu'un champ référencé de la table `zone` est modifié ou supprimé, une erreur est générée et empêche la modification ou la suppression. Ce comportement peut-être redéfini. Il est possible de spécifier une clause de la forme `ON UPDATE` ou `ON DELETE`. Derrière ces clauses, il faut indiquer l'action à entreprendre en cas de mise à jour ou de suppression :

- `CASCADE` : toute modification ou suppression d'un champ référencé entraîne la modification de la valeur ou la suppression de la ligne dans la table comportant la clé étrangère
- `SET NULL` : toute modification ou suppression d'un champ référencé entraîne l'attribution de la valeur `NULL` au champ correspondant dans la table comportant la clé étrangère

Par exemple, en cas de suppression de la zone `c10` de la table `zone` (tableau 4), il ne faurait pas supprimer les livres qui référencent cette zone. Une telle suppression doit entraîner l'attribution de la valeur `NULL` aux champs `code_zone` dont la valeur est `c10` dans la table `livre`. Ceci est réalisé grâce à la clause `ON DELETE SET NULL`. En cas de mise à jour, l'instruction ne devra pas être bloquée : les modifications seront reportées dans les champs qui référencent les valeurs modifiées grâce à la clause `ON UPDATE CASCADE`. Par exemple, si `s8` est modifié en `s1` dans `zone`, la modification sera répercutée automatiquement dans `livre`.

```
CONSTRAINT fk_zone FOREIGN KEY (code_zone)
REFERENCES zone(code_zone) ON DELETE SET
NULL ON UPDATE CASCADE
```

Afin que MySQL permette la déclaration de clés étrangères et gère les transactions, il faut indiquer que les

tables sont de type InnoDB. Autant la table référencée que la table déclarant la contrainte de type clé étrangère. Pour indiquer qu'une table est de type InnoDB, dans la syntaxe de création de la table, il faut ajouter l'instruction `ENGINE = InnoDB` avant le `;` (Listing 1).

Contrainte sur le domaine

La contrainte d'intégrité `CHECK` permet de vérifier qu'une valeur insérée est conforme à une condition, par exemple qu'un nombre est compris entre deux bornes.

```
CONSTRAINT nom_contrainte CHECK (condition)
```

MySQL ne gère pas les contraintes de type `CHECK`. Il existe trois manières de contourner l'absence de cette fonctionnalité :

- l'option `UNSIGNED` interdit l'insertion de nombres négatifs (colonne `nb_pages` : Listing 1),
- le type `ENUM` permet de restreindre les valeurs autorisées à une liste (colonnes `langue` et `genre`),
- un `trigger` peut être déclenché pour vérifier une condition lors d'une insertion ou d'une mise à jour.

Modification de la structure

Une fois les tables créées, il est possible d'y ajouter des colonnes, de modifier le type de ces dernières ou d'ajouter une contrainte d'intégrité. Il est également possible de supprimer des éléments de la base.

Les commandes `ALTER TABLE` et `DROP` permettent de modifier la structure de la base.

Ajout de colonne

Pour ajouter une ou plusieurs colonnes, il faut utiliser la commande `ALTER TABLE` avec le mot-clé `ADD` suivi de la définition de chaque colonne.

```
ALTER TABLE nom_table ADD (
nom_col1 type1, nom_col2 type2, ...);
```

L'instruction suivante ajoute une colonne `langue` à la table `livre` :

```
ALTER TABLE livre ADD (langue ENUM('français',
'anglais','allemand','espagnol','chinois'));
```

Modification du type d'une colonne

Pour modifier une ou plusieurs colonnes dans le SGBD MySQL, il faut utiliser la clause `MODIFY` suivie de la nouvelle définition de chaque colonne.

```
ALTER TABLE nom_table MODIFY(
nom_col1 nouveau_type1,
nom_col2 nouveau_type2,
...);
```

Ajout de contrainte

La clause `ADD CONSTRAINT` permet d'ajouter une contrainte d'intégrité. La définition de la contrainte est la même que celle décrite dans la section intitulée *Définition des contraintes*.

```
ALTER TABLE nom_table
ADD CONSTRAINT nom_contrainte type_contrainte;
```

Suppression de contrainte

De même qu'il est possible de modifier la structure de la base, des commandes permettent d'en supprimer des éléments.

La clause `DROP CONSTRAINT` de la commande `ALTER TABLE` permet de supprimer la contrainte d'intégrité dont le nom est indiqué dans l'instruction :

```
ALTER TABLE nom_table DROP CONSTRAINT nom_contrainte;
```

Dans MySQL, les commandes de suppression des clés primaires et étrangères sont de la forme :

```
ALTER TABLE nom_table DROP PRIMARY KEY;
ALTER TABLE nom_table DROP FOREIGN KEY nom_contrainte;
```

L'instruction ci-après supprime de la table `livre` la contrainte de type clé étrangère, dont le nom est `fk_zone`, qui référence la clé primaire de la table `zone` :

```
ALTER TABLE livre DROP FOREIGN KEY fk_zone;
```

Suppression de table

La commande `DROP TABLE` permet de supprimer une table dans la base. Par exemple, l'instruction suivante permet la suppression de la table `livre` :

```
DROP TABLE livre;
```

Si on tente de supprimer une table dont la clé primaire est référencée en tant que clé étrangère dans une autre table, MySQL génère une erreur et la table n'est pas supprimée.

```
DROP TABLE zone;
>>> ERROR 1217 : Cannot delete or update a parent row : a foreign key constraint fails
```

Si vous voulez tout de même supprimer la table `zone`, il faut commencer par supprimer la table `livre` ou la contrainte de type clé étrangère qui lie les deux tables.

Suppression de base

La commande `DROP SCHEMA` supprime définitivement une base, toutes ses tables et leur contenu.

```
DROP SCHEMA nom_base;
```

Clients MySQL

Pour communiquer avec la base de données, il est possible d'entrer les commandes en mode console ou grâce à une interface graphique. Les deux solutions sont équivalentes et vont être présentées ci-après. Il est nécessaire pour leur utilisation d'avoir démarré le serveur de base de données de la distribution WAMP, XAMPP, MAMP ou EasyPHP.

Console

Pour vous connecter au serveur depuis la console, vous devez entrer l'instruction suivante :

```
mysql [-h nom_hote] [-u login] [-p]
```

Lorsque le nom de l'hôte n'est pas spécifié, la connexion est tentée en local sur l'ordinateur, c'est-à-dire que le serveur est considéré comme étant sur le même ordinateur que le client. L'option `-u` indique qu'il va être spécifié un nom d'utilisateur. Lors de votre première connexion en local, l'utilisateur attendu est `root` (administrateur de la base). L'option `-p` précise qu'il est nécessaire de renseigner un mot de passe. Par défaut, l'administrateur n'a pas besoin d'entrer un mot de passe lors de sa première connexion. Si une erreur d'authentification est générée, vérifiez qu'un mot de passe n'a pas été défini par défaut par la plateforme que vous utilisez.

Une fois authentifié, vous pouvez entrer les commandes SQL du Listing 1 dans la console.

L'exécution de toutes les commandes aura permis la création de la base `biblio`, des utilisateurs `lecteur` et `bibliothecaire` et des deux tables `livre` et `zone`.

La commande `QUIT` permet de quitter le client.

PhpMyAdmin

L'interface Web `phpMyAdmin` permet de créer des bases, des tables et de manipuler les données sans avoir de connaissances SQL. Chaque action effectuée via cet outil envoie une ou plusieurs requêtes SQL au serveur MySQL. La ou les requêtes envoyées sont affichées en haut de la page dans l'interface graphique. L'interface propose également une console SQL qui permet de rentrer des commandes SQL directement.

Si vous utilisez XAMPP, WAMP, MAMP ou EasyPHP, vous avez sur la page d'accueil du serveur web un lien vers `phpMyAdmin`. Sinon entrez l'URL de votre application `phpMyAdmin` dans votre navigateur.

L'interface graphique de `phpMyAdmin` comporte deux cadres. Le cadre gauche permet de sélectionner une base de données parmi la liste des bases hébergées sur le serveur MySQL. Une fois qu'une base a été sélectionnée, la liste de ses tables est affichée sous le nom de la base.

Le cadre de droite permet d'agir sur les bases de données : création de bases et de tables, modification, suppression, gestion des priviléges, consultation et manipulation des données.

Pour les débutants

phpMyAdmin

Serveur: localhost > Base de données: biblio > Table: livre

Champ	Type	Taille/Valeurs ¹	Défaut ²	Interclassement	Attributs	Null
isbn	CHAR	20	Aucun			
titre	VARCHAR	30	Aucun			
genre	ENUM	'fantastique'	Aucun			<input checked="" type="checkbox"/>
date_parution	YEAR		Aucun			<input checked="" type="checkbox"/>
nb_pages	INT		Aucun		UNSIGNED	<input checked="" type="checkbox"/>
code_zone	CHAR	10	Aucun			<input checked="" type="checkbox"/>

Commentaires sur la table: Moteur de stockage: InnoDB Interclassement:

Figure 2. PhpMyAdmin : déclaration des types

Serveur: localhost > Base de données: biblio > Table: livre

Afficher Structure SQL Rechercher Insérer Exporter Importer Opérations Vider Supprimer

La table 'biblio'.livre a été créée.

```
CREATE TABLE `biblio`.`livre` (
  `isbn` CHAR(20) NOT NULL,
  `titre` VARCHAR(30) NOT NULL,
  `genre` ENUM('roman', 'policier', 'theatre', 'historique', 'fantastique') NULL,
  `date_parution` YEAR NULL,
  `nb_pages` INT UNSIGNED NULL,
  `code_zone` CHAR(10) NULL
) ENGINE = INNODB;
```

Champ Type Interclassement Attributs Null Défaut Extra Action

isbn	char(20)	latin1_swedish_ci	Non	Aucun						
titre	varchar(30)	latin1_swedish_ci	Non	Aucun						
genre	enum('roman', 'policier', 'theatre', 'historique', 'fantastique')	latin1_swedish_ci	Oui	NULL						
date_parution	year(4)		Oui	NULL						
nb_pages	int(10)		Oui	NULL						
code_zone	char(10)		Oui	NULL						

Tout cocher / Tout décocher Pour la sélection

Version imprimable Gestion des relations Ajouter 1 champ(s) En fin de table

Annonce de la page http://localhost : Voulez-vous vraiment effectuer : ALTER TABLE 'livre' ADD PRIMARY KEY('isbn')

OK Annuler

Aucun index n'est défini!

Figure 3. PhpMyAdmin : structure de la table

Serveur: localhost > Base de données: biblio > Table: livre

Afficher Structure SQL Rechercher Insérer Exporter Importer Opérations

Relié à

FOREIGN KEY (INNODB)

isbn	ON DELETE	ON UPDATE
titre	Aucun index n'est défini!	
genre	Aucun index n'est défini!	
date_parution	Aucun index n'est défini!	
nb_pages	Aucun index n'est défini!	
code_zone	biblio.zone.code_zone	ON DELETE SET NULL ON UPDATE CASCADE

Figure 4. PhpMyAdmin : déclaration d'une clé étrangère

Sur Internet

- <http://www.mysql.fr> – MySQL,
- <http://www.wampserver.com> – Outil Wamp,
- <http://sourceforge.net/projects/xampp> – Outil Xamp,
- <http://www.mamp.info> – Outil Mamp,
- <http://www.easypHP.org> – Outil EasyPHP,
- <http://www.phpmyadmin.net> – phpMyAdmin.

La page d'accueil comporte un champ texte permettant d'indiquer le nom de la base à créer. L'envoi du formulaire génère et soumet automatiquement à MySQL la commande :

```
CREATE SCHEMA nom_base;
```

Il est maintenant possible de créer une table dans la base grâce au champ intitulé *Créer une nouvelle table*. Il faut indiquer le nombre de colonnes que celle-ci contiendra. Pour finir de créer la table, il faut indiquer le nom des champs et leur type ainsi que le moteur de la table (InnoDB : Figure 2). PhpMyAdmin génère la commande de création de table, l'envoie à MySQL et affiche sa structure (Figure 3). Les clés primaires et étrangères sont définies à ce niveau. Pour indiquer la clé primaire, il suffit de cliquer sur l'icône en forme de clé (Figure 3). Un clic sur le lien *Gestion des relations* permet d'enregistrer les clés étrangères dont les colonnes concernées doivent avoir été indexées au préalable par un clic sur l'icône en forme d'éclair (Figure 4).

L'onglet *Priviléges* permet de gérer les utilisateurs et leurs droits. Une fois la base sélectionnée, vous pouvez cocher les priviléges accordés par rapport au LMD, au LDD et au LCD. Il est possible de définir des priviléges sur toute la base ou sur des tables.

Conclusion

Vous avez appris dans cet article les bases des langages de définition et de contrôle de données : création de bases et de tables, définition de contraintes, gestion des utilisateurs. Dans le prochain numéro, vous apprendrez à insérer et manipuler des données.

CILIA MAURO, MAGALI CONTENSIN

Cilia Mauro est gestionnaire de bases de données et développeur d'applications web au CNRS. Elle enseigne les bases de données et PHP à l'université.

Contact : cilia.mro@gmail.com

Magali Contensin est chef de projet en développement d'applications au CNRS. Elle enseigne depuis plus de dix ans le développement d'applications web à l'université et est l'auteur de nombreux articles sur le développement web en PHP.

Contact : <http://magali.contensin.online.fr>

Rejoignez le Club .PRO

Pour plus de renseignement : editor@phpsolmag.org

Stonfield Inworld



Stonfield Inworld propose aux entreprises des solutions globale d'intégration d'Internet et des Univers Virtuels dans leur stratégie de développement. Au-delà de ses services, la société consacre 30% de ses ressources à des travaux de R&D sur le e-Commerce et le e-Learning dans les Mondes Virtuels

COGNIX Systems



Conseil, conception et développement d'applications évoluées pour les systèmes d'informations Internet/intranet/extranet. Alliant les compétences d'une SSII et d'une Web Agency, Cognix Systems conçoit des applicatifs et portails web à l'ergonomie travaillée et des sites Internet à forte valeur ajoutée.
<http://www.cognix-systems.com>

WEB82



Création et hébergements de sites web pour particuliers, associations, entreprises, e-commerce. Développement entièrement aux normes W3C (www.w3.org) de sites web de qualité, au graphisme soigné et employant les dernières technologies du web (PHP5, MySQL5, Ajax, XHTML, CSS2).
<http://www.web82.net>

Core-Techs



Expert des solutions de gestion et de communication d'entreprise en Open Source, Core-Techs conçoit, intègre, déploie et maintient des systèmes de Gestion de Contenu Web, de Gestion Documentaire, de Gestion de la Relation Client (CRM), d'e-commerce et de travail collaboratif.
<http://www.core-techs.fr>

POP FACTORY



Pop Factory, SSII spécialisée Web. Développement de solutions applicatives spécifiques ; offre de solutions packagées : catalogue numérique, e-commerce, livre/magazine numérique, envoi SMS. Nous accompagnons nos clients tout au long de leur projet : audit, conseil, développement, suivi et gestion.
<http://www.popfactory.com/info@popfactory.fr>

Blue Note Systems



Spécialistes en CRM Open Source, nous proposons une offre complète de prestations sur la solution SugarCRM. Notre valeur ajoutée réside dans une expertise réactive et une expérience des problématiques de la GRC. Nous vous aidons à tirer le meilleur parti de votre solution CRM.
<http://www.bluenote-systems.com>

Intelligence Power



Conseil, Expertises, Formations et Projets E-business centrés au tour du cœur de métier : la Business Intelligence. Intelligence Power vous propose des solutions innovantes pour aligner la technologie sur la stratégie de votre entreprise.
<http://www.intelligencepower.com>

Web Alliance



Vous souhaitez être en première page des moteurs de recherche ? Rejoignez-nous, 100% des clients Web Alliance sont en 1ère page de Google. Web Alliance, société de conseil spécialisée dans le référencement internet, vous propose son expertise (référencement, liens sponsorisés, web-marketing).
www.web-alliance.fr

Club .PRO

Introduction à la sécurité web

Un développeur quel qu'il soit peut réaliser de grands projets. Mais quelle chute fait-il lorsque le dit projet vient d'être détruit par une attaque ? De très haut, tellement haut que certains en abandonnent tout espoir. Dans cet article nous étudierons les bonnes pratiques les plus simples à avoir pour éviter au maximum cette situation.

Cet article explique :

- Comment contrer les méthodes utilisées par certaines personnes pour détruire une base de données.
- Comment contrer les méthodes pour voler des sessions.
- Comment se prémunir de bien d'autres soucis.

Ce qu'il faut savoir :

- Connaître les systèmes de base de données.
- Programmer en Php et le langage SQL.
- Utilisation des formulaires.

Pour simplifier la compréhension de cet article au public débutant, la POO (*Programmation Orientée Objet*), ainsi que l'utilisation de PDO (*PHP Data Object*) pour la connexion à la base de données n'ont pas été utilisés. De plus, comme beaucoup de cours utilisent encore les fonctions *mysql_xxx* pour les opérations sur base de données Mysql, j'ai choisi de les utiliser mais il est vrai qu'il serait grand temps d'utiliser *mysqli_xxx* ou PDO.

Never trust user inputs

Quoi de mieux que de débuter par l'exemple? Nous allons en premier lieu étudier le Listing 1.

Celui ci est un fichier php contenant un formulaire et une partie de code qui permet le traitement de la réponse de ce formulaire. Nous voyons qu'il contient quelques champs devant contenir du texte afin de permettre à un utilisateur de s'inscrire. Il récupère les informations reçues dans les variables *POST* et les utilise pour les insérer en base de données. Nous voyons déjà ici une erreur très grave brisant la règle la plus importante du programmeur : *Never trust user inputs*. Ne jamais faire confiance aux informations que l'utilisateur a saisies. Ainsi, l'utilisateur a-t-il rentré une adresse mail *valide* ? Qui dit qu'il n'a pas entré du texte dans le champ *année*? Voire n'a-t-il pas tenté autre chose ?

J'espère que vous n'êtes pas dans ce cas là, mais si vous débutez et que vous vous retrouvez malgré tout dans cette situation, ne vous alarmez pas,

nous allons étudier comment vérifier les saisies de l'utilisateur.

Premièrement vérifier les données. Il peut y avoir des données texte ou de données telles que des chiffres. Soit l'un soit l'autre bien qu'il existe un troisième type : les fichiers que nous aborderons très rapidement plus loin. Cela peut être un prénom, une ville, un commentaire, ou alors, une année, un numéro de téléphone...

La première règle que doit respecter tout programmeur PHP est de vérifier les données externes. Tout ce qui ne provient pas de la base de données devra être passé 'à la moulinette' afin d'éviter d'une : de mauvaises informations, de deux : des attaques pouvant mettre à mal votre base de données, votre site, voire aussi vos visiteurs.

La première erreur concerne le pseudonyme: la fonction *trim()* est utilisée sur ce champ pour supprimer les espaces en trop, en encadrant la valeur saisie par l'utilisateur.... Mais l'utilisateur pourrait ne rien avoir saisi !

Pour vérifier qu'une variable n'est pas *vide* nous pouvons utiliser la fonction *empty()* qui renvoi un booléen : 0 si la variable passée en paramètre contient quelque chose, sinon 1 si c'est vide.

De cette manière nous entrons dans la condition suivante si la variable *\$pseudo* contient des *informations* après le *trim();*

```
if (!empty($pseudo))
```

```

{
    echo 'Pseudo saisi';
}
else
    echo 'Formulaire à resaisir !';

```

Cette fonction sera à utiliser sur chacun de nos éléments réellement nécessaires comme le *pseudo*, mot de passe et mail.

```

If ( !empty($pseudo) ) AND !empty($_
POST['password']) AND !empty($_POST['email'])
)
{
    echo 'on peut continuer les
vérifications du formulaire';
}
else
    echo 'formulaire à ressaisir';

```

Nous nous approchons déjà de la fin ! Avec cela nous sommes 'sûrs' que nos champs ne sont pas vides. Pour le pseudo c'est bon, il nous reste mots de passe et email dans la catégorie champs à valider, puis année à vérifier. Oui, l'adresse mail devra être vérifiée. Mais revenons sur la comparaison des mots de passe *Password* et *Password2*. Dans le *listing1a.php*, la comparaison est réalisée par un simple `==`. Je vous invite à utiliser la fonction `strcmp ()` qui permet de comparer 2 variables et renvoi 0 si celles-ci sont identiques !

Tentez de valider un formulaire avec le code du *listing1a.php* avec 000000 pour premier password et 000 pour le second. Vous aurez une mauvaise surprise car PHP pourra supposer que les deux variables sont égales à 0 !

Il nous reste donc l'adresse mail à vérifier. Pour cela il y a ici deux écoles : la vieille utilisant les expressions régulières et une toute récente utilisant la librairie magique permettant de 'filtrer' les données. Nous allons utiliser une expression régulière, la plus fastidieuse à mon goût.

Ce cours n'ayant pas pour but d'expliquer les 'regex' nous ferons simple. Une adresse mail c'est quoi ? *bi-dule@monsite.fr, machin.moi@supersite.com*. On peut donc en déduire qu'une adresse mail commence par une suite de caractères pouvant aussi être des chiffres voire certains caractères précis puis un arobase @ ensuite un nom de domaine (pouvant contenir un sous domaine) donc là aussi des caractères du même type qu'au début puis un '.' (point) séparant du com, fr ou autre.

Nous pouvons former avec cela un *patron*, une sorte de modèle qui permettra de comparer une chaîne de caractères à celui-ci. Si les deux ne correspondent pas, l'inscription ne sera pas possible.

Listing 1a. Exemple à ne pas suivre

```

include('includes/fonctions.php ');
connexion(); // on se connecte à la BDD

If isset($_POST['formulaire'])
{
    $pseudo=trim($_POST['login']);
    $mot_passe=trim($_POST['password']);
    $email=trim($_POST['email']);
    $annee=trim($_POST['annee']);
    if($_POST['password']==$_POST['password2'])// compare les passwords
    {
        mysql_query("INSERT INTO comptes (pseudo,
        pass, mail, annee, ) VALUES ('".$pseudo."', '".$mot_passe."',
        '".$email."', '".$annee."')");
        // on réalise des traitements pour le reste de
        l'inscription (envoi d'un mail ou autre)
        // .....
        //fin des traitements
    }
    else
    {
        echo '<form method="post" action="listing1a.
        php">
            <h1>Inscrivez vous !</h1>
            Identifiant: <input name="login" value="" type="text"/><br/>
            Mot de passe: <input name="password" value="" type="password"/><br/>
            Mot de passe (confirmation): <input name="password2" value="" type="password"/><br/>
            Email : <input name="email" value="" type="text"/><br/>
            Année de naissance: <input name="annee" value="" type="text"/><br/>
            <input class="button" name="formulaire" value="Connexion" type="submit">
        </form>';
    }
    ?>

```

Listing 1b. Mieux mais....

```

<?php

include('includes/fonctions.php ');
connexion(); // on se connecte à la BDD

If isset($_POST['formulaire'])
{
    $log=trim($_POST['login']);
    $pass=trim($_POST['password']);
    if (!empty($log) AND !empty($pass))
    {
        $recup=mysql_query(' select * from comptes
        where login= "'.$log. ' ' . and pass= "'.$pass. ' ' ');
        // on réalise des traitements pour la
        connexion
        // .....
        //fin des traitements
    }
    else
    {
        echo '<form method="post" action="listing1b.
        php">
            Identifiant: <input name="login" value="" type="text"/><br/>
            Mot de passe: <input name="password" value="" type="password"/><br/>
            <input class="button" name="formulaire" value="Connexion" type="submit">
        </form>';
    }
}

```

L'exemple suivant vous montre comment utiliser cela.

```
if (preg_match("#^([a-z0-9_.-]+@[a-z0-9_.-]{2,}\.[a-z]{2,4}$#", $email))
{
    echo 'Adresse valide';
}
else
{
    echo 'Adresse non valide!';
}
```

Les regex peuvent être utiles pour reconnaître des mots (le système de *Bbcode* des forums). Mais cela n'est pas toujours évident de les utiliser.

Passons à notre nouvelle fonction `filter_var()`. Elle prend 2 paramètres : le premier est la variable à examiner, le second est un 'filtre' permettant de signaler à quoi doit ressembler notre variable passée en premier paramètre. Exemple : `filter_var('bidule@monsite.com', FILTER_VALIDATE_EMAIL);`

Cette fonction renvoi la valeur du champ passée en paramètre si elle correspond avec le filtre, sinon `false`. Je vous invite à faire plus ample connaissance avec cette fonction qui pourrait bien vous simplifier la vie contrairement aux *regexp*.

Il nous reste enfin le champ année. Celui-ci ne doit contenir que des nombres. Concernant les nombres, il y a 2 solutions : l'utilisation du cast, ou alors une fonction existante.

Le cast s'utilise de la manière suivante : `$annee=(int) $_GET['année'];`. Sinon il y a là aussi une fonction bien pratique renvoyant un booléen si la chaîne passée en paramètre est un nombre. Cette fonction magique c'est `ctype_digit()`.

```
if (ctype_digit($_GET['annee']))
{
    echo 'C\'est un nombre';
}
else
    echo 'Ca n\'en n\'est pas un !';
```

À la place des echo, il suffit de faire soit une nouvelle affectation, soit ne rien changer.

Concernant ce dernier champ, il serait encore possible de vérifier que l'utilisateur a saisit une date plausible mais cela est de votre ressort. Chacun est libre de choisir sa méthode favorite même si la première tient en une ligne.

La vérification des contenus est donc une phase très importante car elle peut permettre d'éviter de fausses données voire éviter un peu de spam (dans les formulaires de contact par exemple). Cependant nous sommes loin d'avoir réellement fini notre travail.

La protection des bases de données

Voyons maintenant comment se protéger côté serveur. La plus grande catastrophe qui pourrait arriver est la perte de la base de données. Comment font-ils ? Comment se sont retrouvés avec un site vidé de son contenu ? Une chose que beaucoup ignorent, c'est combien cela peut être facile. L'objectif de votre ennemi est le traitement du formulaire ou plus précisément lors des requêtes SQL.

Nous avons dans le premier script vérifié les informations saisies par l'utilisateur de votre site web. Ce que nous avons pour l'instant seulement réalisé c'est garder une cohérence des informations reçues afin d'éviter tout traitement inutile et informations à jeter. Cependant il nous reste à protéger notre base de données car pour l'instant niveau sécurité, disons que nous n'avons absolument rien fait. Les chaînes de caractères comme le pseudo et le mot de passe sont encore sensibles dans notre script précédent alors il faudra bien appliquer les nouvelles règles étudiées dans cette partie et les appliquer au *listing1a.php*. Regardons le *listing1b.php*.

Il récupère les informations reçues dans les variables `POST` et les utilise pour les comparer à ce qu'il existe en base de données.

```
mysql_query(<< select * from comptes where
login=' >>.$log. >> and pass=' >>.$pass. >> << )
```

Cette partie de code est censée vérifier les données saisies sur le compte de la même manière que nous l'avions réalisée dans la première partie. Voyons comment outre passer cela : Saisissons `or a = a` dans le champ `login` et `password`. Cette requête ressemblera désormais à : `mysql_query(<< select * from comptes where login='or 'a' = 'a' and pass='or 'a' = 'a' <<)`. Ceci est ce qu'on appelle de l'injection SQL. C'est l'exemple même de l'injection la plus simpliste possible. Cette requête retournera le premier enregistrement qu'elle trouvera et comme vous le savez, lors de la création d'un site web, le premier compte créé est souvent le compte administrateur... N'est-ce pas ?

Dans cette situation l'utilisateur devrait se retrouver connecté avec les droits de votre compte... Pas très plaisant n'est-ce pas ? Bien sûr il est possible d'aller encore plus loin ! Saviez vous que le `#` pouvait servir à mettre des commentaires dans le sql (suivant la configuration). En mettant `admin #` dans le champ `login`, la requête ressemblerait à : `select * from comptes where login='admin#';`. Avec cette astuce vous pouvez modifier la requête comme bon vous semble... Il est grand temps d'empêcher cela !

```
$log=trim($_POST['login']);
$pass=trim($_POST['password']);
```

```

$log=mysql_real_escape_string($log);
$pass=mysql_real_escape_string($pass);

mysql_query(<< select * from comptes where
login=' ».$log. »' and pass=' ».$pass. »'
<< );

```

La fonction `mysql_real_escape_string()` permet d'échapper tout caractère pouvant être 'exécuté' par le langage SQL tels que les apostrophes, guillemets, # et d'autres... En pensant utiliser cette unique fonction, vous mettez à l'abri bien des informations censées rester dans votre base de données...

Cependant cela ne sert à rien si vous ne passez pas au chapitre suivant.

La base sécurisée avec un compte non sécurisé.

Je me rappelle mes débuts en programmation web. Je suivais un tutoriel où l'auteur préconisait de mettre le code des fichiers inclus dans des fichiers `.inc`. Je crois que l'auteur avait sauté lui même une étape dans son cours. Un site bien développé possédera une certaine organisation de fichiers

À la racine il y aura forcément un fichier `index.php` ou `.html` et sûrement d'autres pages ainsi que des dossiers `images` ou `design` voire `css` et `js`. Il y aura aussi très souvent un dossier `include` dans lequel l'on mettra des fichiers contenant diverses fonction du site mais aussi un script contenant les identifiants de connexion.

Souvent sous cette forme:

```

function connexion()
{
// on se connecte à MySQL
@mysql_connect(<<localhost >, «logindebbd»,
«motdepasse»);
mysql_select_db(<<bddprotegee»);
}

```

Si ce code est ainsi placé dans le fichier `truc.inc`, toutes les protections réalisées dans le paragraphe précédent n'auront servies à rien ! Les plus malins (et pas forcément uniquement ceux là) s'en sont doutés depuis le départ, il suffit de taper l'adresse: `http://www.votresite/include/truc.inc` et surprise, le fichier s'affiche dans le navigateur affichant joliment le code avec le login mot de passe du compte BDD... Voyons déjà comment sécuriser cela. Premièrement changer cette extension `.inc` en `.php` c'est sûr. Si vous avez du code exécutable dans un tel fichier `*.inc`, il faut alors empêcher son exécution lors de l'utilisation de l'url directe ; j'ai une seconde étape : `.htaccess`.

Le `.htaccess` permet de spécifier des règles. Créez un fichier `.htaccess` vide dans le dossier `include`.

Sous Windows, le système d'exploitation risque de refuser. Dans ce cas, ne mettez pas le point mais pensez à le modifier lorsque vous l'enverrez via FTP en le renommant. Éditez maintenant le contenu de ce fichier pour y mettre le code suivant :

```

deny from all
allow from 127.0.0.1
<Files maintenance.html>
allow from all
</Files>

```

Ce code permet de signaler que ce dossier doit être inaccessible pour tout le monde (donc à partir du web), sauf le serveur lui même. De cette manière les fichiers sont disponibles à partir d'un include du serveur mais pas à partir de l'extérieur. Deux autres points assez simples : dans tout dossier qui existe, il doit y avoir un fichier `index.html` ou `index.php` même vide pour empêcher l'affichage du contenu du dossier. De plus pensez à utiliser 2 comptes mysql différents : un compte admin possédant certains droits lors de vos manipulations SQL, et un autre compte ayant moins de droits (pas de DROP par exemple) pour le site.

Coté serveur Web !

Nous avons vu comment protéger votre base de données. Il existe pourtant d'autres failles qu'il ne faut pas négliger.

La configuration même de votre machine peut vous porter défaut : `Register_globals` peut en effet vous poser bien des soucis. En effet, si elles sont activées elle peuvent permettre à quiconque de modifier des valeurs de variable de votre code. Il vaut donc mieux placer cette fonctionnalité sur OFF. Cette directive devait normalement disparaître dans PHP 6. Vous pouvez éditer la configuration soit dans le `PHP.ini` de votre serveur si vous y avez accès soit dans un `.htaccess`.

Pensez à désactiver les erreurs... ou plutôt les message d'erreurs en passant la fonction `display_errors` à OFF. Pourquoi cela ? Pour éviter que lorsque quelqu'un de mal intentionné testera les failles de votre site, il puisse avoir des messages d'erreurs pouvant parfois le guider dans sa recherche. Devenu obsolète depuis PHP5.3, les `magic_quotes` avaient au départ été imaginés pour simplifier la sécurité en réalisant un échappement automatique. Pour cela, vaut mieux réaliser sa sécurité soit même que via cette fonction qui ne fait pas tout le nécessaire.

Vous pouvez la aussi mettre dans un `.htaccess` :

```

php_flag magic_quotes_gpc off
php_flag magic_quotes_runtime off

```

ou alors votre fichier `php.ini`.

Listing 2. Exemple pour faille include

```
<?php
if(isset($_GET['page']))
{
    include $_GET['page'].'.php';
}
else
{
    include 'default.php';
}

?>
```

Sécurité sur les fichiers uploadés

Uploader des fichiers sur un site, cela se fait surtout sur des forums pour mettre son avatar préféré. Seulement l'upload de fichiers peut se révéler ultra dangereux s'il est mal manipulé.

Premièrement, pensez à examiner l'extension du fichier (.jpg, .bmp .gif ou autre) afin de ne pas laisser passer d'exécutables. Cependant, cela ne vérifiera en rien le contenu du fichier ! Donc ne pensez pas qu'en ayant filtré les fichiers vous serez à l'abri. Il est facile de renommer *moncode.php* en *moncode.jpg*. Combiné à la faille CSRF (astuce visant à faire réaliser une tâche de modération/administration comme surprimer un message par un autre compte qui en possède les droits) cette faille doit pouvoir créer des ravages sans que personne ne s'en rende compte. Ensuite il suffit de renommer le fichier ! Voici un bout de code si simple qui pourrait éviter bien des défacing (action de changer le contenu d'un site).

```
$alphanum = 'abcdefghijklmnopqrstuvwxyz01234
56789';
$melange = str_shuffle($alphanum);
$nouveau_nom = substr($lettres_chiffres_
melanges, 1, 8);
```

Il suffit d'ajouter l'extension adéquate pour que le fichier puisse être gardé. S'il contient du code malicieux comme du PHP, le fichier sera téléchargé par la navigateur et non exécuté par le serveur.

La faille include

Certains développeurs se croyant un peu plus malins, voire plutôt faignant ont développés la page index de leur site qui se consulte de cette manière : www.site.com/index.php?page=abcd.

Le pirate va donc procéder ainsi : il écrit une adresse url telle que

www.site.com/index.php?page=http://www.sitedupirate.com/pageultradangereuse
ou même
www.site.com/index.php?page=admin,

en essayant plusieurs chose pour trouver la page d'administration du site. Si votre serveur utilise PHP 4, le script du visiteur sera exécuté par le serveur disposant de cette faille. Pour la contrer il suffit de mettre la fonction `php allow_url_fopen à OFF`.

Vous pouvez aussi (sans vous passer pour autant de la technique précédente) modifier votre code pour vérifier l'existence du fichier à inclure :

```
<?php
if(isset($_GET['page']) AND file_exists($_
GET['page'].'.php'))
{
    include $_GET['page'].'.php';
}
else
{
    include 'default.php';
}
?>
```

Cependant, il faudra vérifier que le visiteur ne peut pas inclure des pages d'administration ! Malgré tout, je vous déconseille l'utilisation de cette méthode pour naviguer dans votre site. Au pire, utilisez un entier pour déchiffrer chaque page. De cette manière il suffit de caster la valeur `$_GET['page']` ; et créez un array numéroté avec pour valeur les noms de pages. Soyez très prudents ! On ne le dira jamais assez.

Sécurité extérieure

Nous avons maintenant vu comment améliorer la sécurité interne à votre site web. Il y a aussi des sécurités externes : protéger vos visiteurs !

La première chose sera le mot de passe. Les visiteurs ont des mauvaises habitudes : ils choisissent des mots de passe communs : test, lechien, superman et d'autres... Vous me direz que vous avez pris la protection de *hasher* vos mots de passe avec MD5 ou SHA1. C'est bien ; de grands sites ne le font même pas. Mais ce n'est pas suffisant. Imaginez qu'un hacker arrive à pénétrer votre base de données. Il peut alors s'imprimer la liste des comptes avec *pseudo + mot de passe + mails*. Sachez que sur le web il existe des dictionnaires inversés comme pour l'annuaire. Vous y mettez le hash, vous obtenez le mot de passe ! Il faut donc empêcher cela.

Lors de l'inscription du membre, vous faites sûrement un :

```
$pass=md5($_POST['pass']);
ou
$pass=sha1($_POST['pass']);
```

Je vous invite à choisir une sorte de clé supplémentaire sans aucun sens tel que : KNSDI768ITDSR que vous concaténeriez avec le mot de passe du visiteur.

```
$pass=K4NSDI768ITDSR+$_POST['pass'];
$pass=sha1($pass);
```

De cette manière, cela formera un hash bien différent de ceux contenus dans les dictionnaires.

Certains proposent une génération aléatoire de ce second hash puis l'insérer en base de données. Seulement si une personne bien malhonnête arrive à s'y connecter, celle-ci aura le hash du password concaténé avec le hash généré ainsi que la chaîne du hash généré... D'autres encore proposent une utilisation telle que : `md5(sha1($pass))`. A chacun de voir quelle méthode utiliser ici.

XSS

Une autre sécurité à prendre en compte est lors de l'affichage des données saisies dans les formulaires. En effet, si l'utilisateur entre du code Html ou javascript dans un champ texte, avez vous essayé d'afficher le résultat ? Il sera exécuté et croyez moi, avec du javascript, on peut faire de très méchantes choses comme rediriger vers un autre site en transmettant quelques informations en paramètres et voler des sessions.

Comment afficher les informations de la base de données alors ? Je vous présente là aussi une fonction toute simple : `htmlspecialchars()`. Elle s'utilise simplement : `echo htmlspecialchars($sql['pseudo']);`. Elle échappera les caractères pouvant être exécutés par un navigateur. Par cette méthode vous évitez ainsi le tristement célèbre *Cross-Site Scripting (XSS)*.

Tester sa sécurité

Il existe bien les tests classiques tels que les tests unitaires et de recette. Ceux-ci peuvent parfois s'appliquer aux tests de sécurité grâce à des outils comme PHPUnit en saisissant volontairement des données pouvant porter atteinte à votre site web. Vous pourriez bien y prendre goût. Vous connaissez le code alors tentez de le contourner. Saisissez des url normalement réservées à l'administration, tentez de poster des données de formulaire via `POST` et `GET` (firefox et l'addon webdeveloper peuvent vous être d'une grande utilité). Nous pourrions vous recommander PhpSeclInfo, Rats et bien d'autres...

Pour aller plus loin

Vous commencez à avoir de bonnes notions PHP ? Savez qu'il existe des frameworks qui peuvent vous faire gagner du temps lors de vos développements tout comme vous simplifier la vie pour la sécurité à savoir : Zend framework, code Igniter, Symphony, et bien d'autres. Plusieurs articles ont été publiés dans les numéros précédents sur ces frameworks. La liste n'est bien entendu pas exhaustive et il en existe bien d'autres tous aussi complets que ceux nommés.

Conclusion

Cet article est loin d'être exhaustif voire même très loin. La sécurité est un domaine tellement vaste qu'un numéro complet pourrait ne pas suffire tant les attaques sont différentes et évoluent. J'espère cependant avoir intéressé votre esprit sur l'importance de la sécurité dans votre site et les futurs projets que vous aurez à mener. Perdre du temps sur la sécurité, c'est aussi en gagner à l'avenir. Méditons bien cela !

Un grand merci à Arnaud Lemaire pour sa relecture très attentive et ses remarques qui m'ont grandement aidé dans la réalisation de cet article.

NICOLAS TURMEAU

Nicolas Turneau est étudiant en informatique (en 3ème année de licence). Passionné, il désire aussi vulgariser certaines notions pour le plus grand public. nturneau@iia-laval.fr.



**Bishamonten
Technologies**

La sécurité pour tous.

CRÉATION - MODIFICATION -SÉCURISATION D'APPLICATIONS WEB

- + PHP, MySQL, PgSQL
- + Sous-traitance
- + E-mailing
- + Visio-conférence
- + Modification de solutions Open Source
- + Suivi et Gestion de solutions Open Source

Bishamonten Technologies

23 rue du 19 Mars 1962
58 660 Coulanges-les-Nevers

Tél. : 06 30 57 82 85 - E-mail : contact@bsmt.fr
Site internet : www.bsmt.fr

Génialement rapide

Votre site est-il vraiment rapide ?

Les sites WebGazelle sont conçus pour s'afficher en moyenne 20% plus vite que la majorité des sites Internet

En savoir plus : www.webgazelle.net

► N°Azur **0 810 810 815**

PRIX APPEL LOCAL



Grand jeu concours 12 sites internet à gagner

Tirage au sort tous les mois

Lot d'une valeur de 2498.20 € TTC

Vous gagnez un site Internet qui comprend :

- le conseil et l'étude de votre projet,
- le graphisme de votre site,
- l'optimisation du référencement,
- l'intégration des contenus,
- le logiciel WebGazelle CMS 2.0,
- le support et l'hébergement.

Extraits de règlement

Jeux Concours sur tirage au sort, organisé par la société Cognix Systems, à destination des personnes morales et personnes physiques ayant la qualité de commerçant, artisan et auto-entrepreneur, ayant leurs sièges sociaux en France métropolitaine (Corse comprise ; DOM-TOM compris). Pour participer, il faut s'inscrire sur Internet à l'adresse <http://www.webgazelle.net/jeu-concours>. Le règlement est disponible sur notre site Internet : <http://www.webgazelle.net/reglement.php>. Il est déposé chez un huissier de justice et peut être demandé gratuitement sur demande écrite à l'adresse suivante : Cognix Systems, 65 avenue Aristide Briand, 35000 Rennes.

Présentation des lots

12 sites Internet sont à gagner, basés sur un Pack WebGazelle CMS 2.0 « Essentiel » (valeur d'environ 2500 € TTC). Ces sites Internet comprennent : Les services compris dans un pack WebGazelle « Essential », un espace d'hébergement et un nom de domaine rattaché au site remporté, pour une durée de un an.