

Tuning et benchmark avec THELIA

(par psai 30/09/2007)

Ce week-end, j'ai décidé de passer **THELIA** à la casserole.

Plus sérieusement, n'ayant aucune connaissances en développement je ne peux apporter mon aide à la communauté et au projet dans ce domaine. J'ai donc décidé de l'apporter dans un domaine où je suis plus à l'aise, en espérant que des personnes parmi vous trouveront des informations utiles dans ce petit article.

Un bon repas, dans une assiette en carton, avouez que c'est dommage ... Tout le monde à déjà vécu le buffet ou le barbecue chez un ami, où l'assiette en carton se plie sous le poids des merguez et salades. A tout moment vous sentez que tout peut tomber par terre et là vous vous dites *mince j'aurais du en mettre moins* ou encore *il aurait pu prévoir d'autres assiettes...*

Aujourd'hui nous allons voir comment fabriquer une assiette plus solide pour éviter que tout ne se casse la figure :)

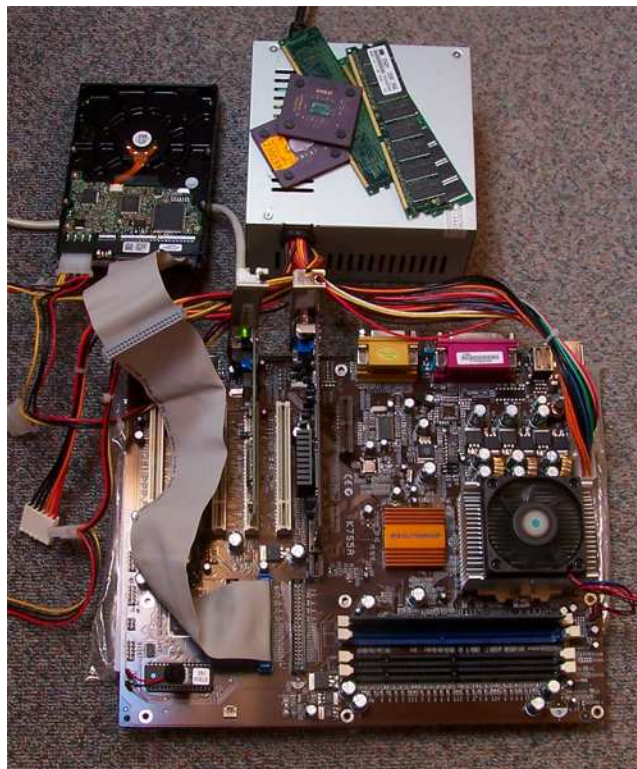
Après cette entrée en matière sur un arrière gout de soirées d'étés (ou de pré-automne étant donné l'été maussade de cette année) abordons le domaine qui nous intéresse ici, la plateforme qui hébergera votre (ou vos) projet(s) THELIA, je parle bien entendu de **LAMP**.

Pour ceux à qui ce nom ne dit pas grand chose je vais vous éclairer; LAMP signifie tout simplement **Linux Apache MySQL PHP**.

Dans cet article je ne parlerai que d'optimisations concernant Apache car ce sont, à mon avis, les plus faciles à mettre en œuvre, mais également celles qui apporteront les gains les plus significatifs.

De nos jours, les offres de serveurs dédiés ou VDS (*Virtual Dedicated Server*) sont vraiment abordables, et bon nombre de personnes s'orientent (et parfois à mon grand regret sans connaissances) vers un serveur dédié plutôt qu'un hébergement mutualisé. Les raisons sont diverses et ne sont pas l'objet de cet article. La plupart des hébergeurs proposent des distributions prêtes à l'emploi, avec des configurations relativement basiques qui ne sont pas forcément optimisées.

J'ai voulu, pour ma plateforme de bench, une config hardware à mi-chemin entre une offre entrée de gamme de serveur dédiés et une offre moyenne gamme de VDS. La plateforme de test utilisée comportait un Athlon XP1800+ underclocké à 1,2GHz, une DDR 256 (2100), un disque dur 40Go 7200tr/min IDE avec 2Mo de cache, et une carte mère tout ce qu'il y a de plus banal. La distribution utilisée était une Debian Etch minimaliste dans sa version 4.0 avec Apache dans sa version 2.2.3, MySQL en version 5.0.32, et PHP5 5.2.0-8.



Le bench s'est déroulé en 3 phases chacune de 3 heures à l'aide de l'outil de bench **siège**, en sollicitant au maximum le processeur qui, pendant les 3 phases, à fonctionné à 100%.

Le bench a été effectué sur une centaine de pages différentes comportant des boucles telles que PRODUIT, RUBRIQUE, ACCESSOIRES.

La première phase a été réalisée dans une configuration standard sans aucune optimisation d'Apache, la seconde en utilisant le module mod_deflate (mod_gzip si vous êtes sous Apache 1.3), et pour finir la troisième avec eaccelerator.

mod_deflate permet tout simplement de compresser les fichiers avant envoi au client, et c'est le rôle du navigateur de décompresser les données. Aujourd'hui, tous les navigateurs récents sont capables d'effectuer cette tâche, ce qui n'est pas le cas avec de vieux navigateurs. La configuration de mod_deflate permet de spécifier les navigateurs pour lesquels il ne faudra pas utiliser cette fonction de compression, car incompatible, mais pour plus d'infos sur le sujet je vous dirige vers la doc officielle de ce module : http://httpd.apache.org/docs/2.0/mod/mod_deflate.html

Quel est donc l'intérêt de nos jours de compresser une page (de quelques centaines de kilos) alors que presque tout le monde est équipé d'une connexion haut-débit tout en sachant que les hébergeurs mettent à disposition des bandes passantes de plusieurs dizaines de mégas ?

Eh bien non, tout le monde n'est pas logé à la même enseigne, certains sont encore en 56K, d'autres ne peuvent dépasser un débit ADSL de 512K et beaucoup partagent la connexion entre différentes machines. Il y a donc aujourd'hui encore un intérêt à compresser les données avant envoi. Il y a également un autre intérêt dont on entend peu parler concernant mod_deflate, c'est l'affichage des pages. En effet la plupart des navigateurs n'afficheront la page qu'une fois entièrement décompressée. Cela a pour effet de ne pas voir une page se construire sous vos yeux au fur et à mesure de son chargement, mais de voir une page s'afficher d'un seul coup. Ce n'est qu'un avis personnel, mais je préfère une page qui apparaît à l'écran dans son état final, que de voir mon navigateur jouer à un semblant de puzzle ...

Gardez tout de même à l'esprit que c'est au processeur de votre serveur de compresser ces données, aussi si votre trafic est important vous pourriez mettre à genoux votre serveur avec ce module. Etudiez bien la question avant de faire votre choix !

Eaccelerator ... non ? Ca ne vous dit rien ? Tant mieux sinon quel serait l'intérêt pour moi de vous le présenter ! Eaccelerator est tout simplement un cache de contenu dynamique. Aïe ... vous sentez cette migraine ? :)

Son rôle va être de conserver les scripts PHP dans leurs états compilés. Pour mieux comprendre je vais tâcher d'expliquer rapidement le fonctionnement avec et sans.

Si vous n'utilisez pas un tel cache, Apache va, à l'aide de la librairie PHP, compiler vos scripts avant de les traiter, et ce pour chaque page ou script PHP qui vont être appelés. La compilation de ces scripts, est encore une fois le rôle du processeur.

Quel est l'intérêt de compiler des milliers de fois le même script ou la même page ? Eaccelerator est la solution, puisqu'il va conserver les résultats de compilations, et lorsqu'une page ou un script sera appelé, plutôt que de refaire tout le travail, eaccelerator va fournir ces compilations toutes prêtes. Un gain de temps de traitement, un processeur moins sollicité, et donc disponible pour autre chose, comme par exemple servir plus de contenu à plus de clients !

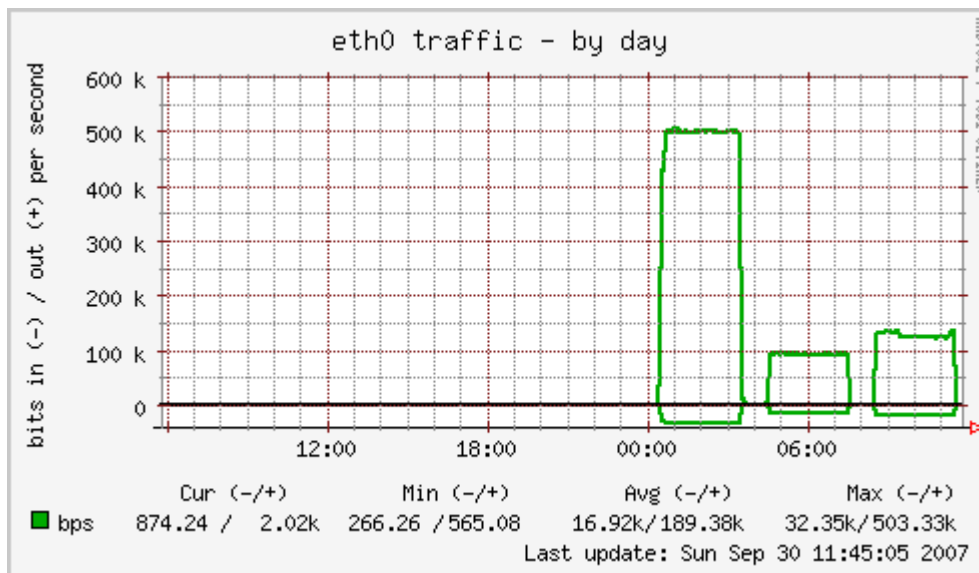
En résulte un affichage plus rapide de vos pages et une machine moins sollicitée.

Comment ca ? Ca ne vous intéresse toujours pas ? Ok, quelques jolis graphs et quelques chiffres devraient vous faire changer d'avis !

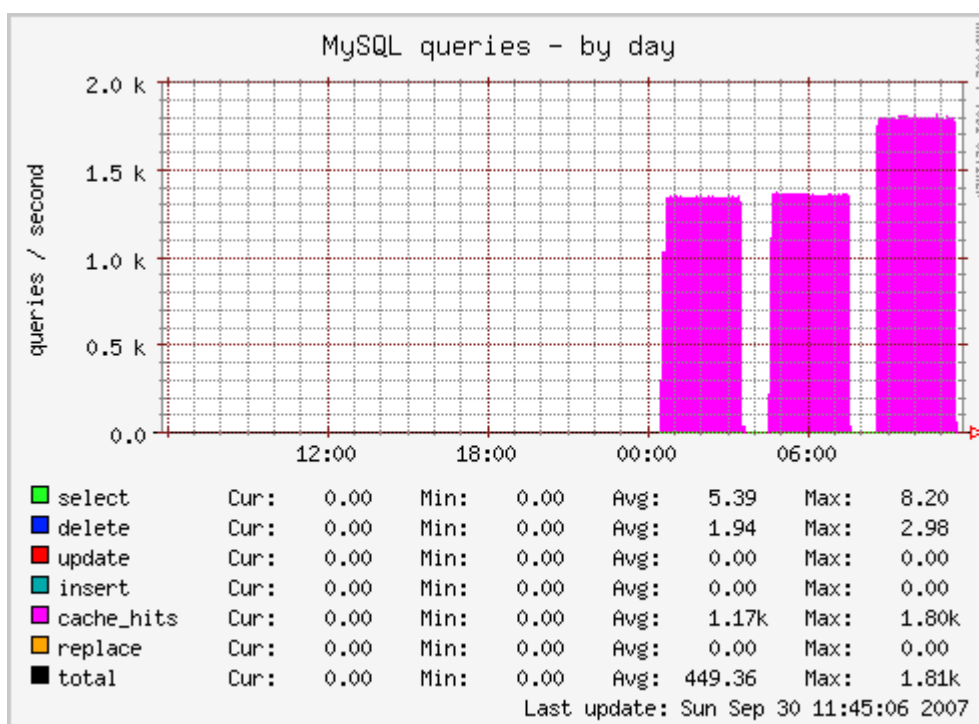
Sur les graphs vous pouvez constater les 3 phases de bench dans l'ordre suivant :

config standard - config avec mod_deflate - config avec mod_deflate et eaccelerator

Regardons dans un premier temps la bande passante. Elle a pu être divisée par 5 (entre la phase 1 et 2) ! Comprenez par là qu'une page pesant 50ko ne représentera que 10ko à télécharger par le client. En 56K si on part sur un débit moyen de 4ko/s, le temps de chargement de la page passe de 12,5 secondes à 2,5 ... Intéressant non ?



Dans la phase 3 la bande passante a augmentée. Cela voudrait dire que eaccelerator consomme plus de bande passante ? Eh bien non ! Rappelez-vous, eaccelerator permet de diminuer l'utilisation du processeur. Donc le processeur est disponible pour traiter plus de requêtes ! C'est ce que vous pouvez constater sur ce graph qui comptabilise le nombre de requêtes MySQL par secondes. Le nombre de requêtes MySQL a augmenté puisque Apache a pu fournir plus de requêtes HTTP (donc plus de pages à la seconde tout simplement). Forcément si on fournit plus de requêtes, on consomme plus de bande passante, tout est lié ...



Toujours pas convaincu par ces jolis graphs ?

Place aux chiffres alors !

Je me dispense de les commenter, ils sont assez parlant ... Mais attardez vous surtout sur les valeurs **Data transferred** ainsi que **Transaction rate**.

Phase 1

Transactions:	23837 hits
Availability:	100.00 %
Elapsed time:	10800.02 secs
Data transferred:	628231449 bytes
Response time:	1.36 secs
Transaction rate:	2.21 trans/sec
Throughput:	58169.47 bytes/sec
Concurrency:	3.00
Successful transactions:	23837
Failed transactions:	0
Longest transaction:	53.56
Shortest transaction:	0.34

Phase 2

Transactions:	24112 hits
Availability:	100.00 %
Elapsed time:	10799.80 secs
Data transferred:	103658792 bytes
Response time:	1.34 secs
Transaction rate:	2.23 trans/sec
Throughput:	9598.21 bytes/sec
Concurrency:	3.00
Successful transactions:	24112
Failed transactions:	0
Longest transaction:	33.97
Shortest transaction:	0.34

Phase 3

Transactions:	31887 hits
Availability:	100.00 %
Elapsed time:	10799.68 secs
Data transferred:	137110845 bytes
Response time:	1.02 secs
Transaction rate:	2.95 trans/sec
Throughput:	12695.83 bytes/sec
Concurrency:	3.00
Successful transactions:	31887
Failed transactions:	0
Longest transaction:	141.47
Shortest transaction:	0.24

Vous avez fait le choix en utilisant THELIA de proposer à vos clients une belle boutique modulable à souhait et robuste. Vous avez maintenant la possibilité de la rendre plus réactive et plus accessible. Pourtant ce ne sont que quelques optimisations qui vous prendront quelques heures à mettre en place pour les plus novices d'entre vous, et 10 minutes pour ceux qui ont un peu plus de bouteille. Il reste encore beaucoup d'optimisations possibles, tant au niveau d'Apache avec sa multitude de modules, qu'au niveau de MySQL avec un peu de tuning au niveau des caches, des réglages de buffers, etc ... Ces optimisations feront peut-être l'objet d'un prochain article si celui-ci vous a plu et apporté une aide.

Le but de cet article n'était non pas de vous expliquer comment mettre en place ces solutions, car il existe déjà bon nombre d'articles là-dessus, mais simplement de présenter ces solutions là avec des arguments, des graphiques et des chiffres à l'appui ce qui d'après moi manquait sur la toile. Je mets tout de même à disposition les configurations de mod_deflate et eaccelerator utilisées pour ces benchmarks. Je ne peux bien évidemment pas être responsable des dommages que pourraient causer la mise en pratique de ces optimisations et vous conseille vivement de les mettre en application sur un environnement de test avant de faire quoi que ce soit dans votre environnement de production.

J'espère avoir apporté une petite pierre à l'édifice et remercie encore une fois Yoan et la communauté THELIA pour ce très bon projet, en croisant les doigts pour les Trophées du Libre ! ;)

Config de mod_deflate :

```
SetOutputFilter DEFLATE
BrowserMatch ^Mozilla/4 gzip-only-text/html
BrowserMatch ^Mozilla/4.[0-9] no-gzip
BrowserMatch \bMSIE !no-gzip !gzip-only-text/html
SetEnvIfNoCase Request_URI (?:gif|jpe?g|png)$ no-gzip dont-vary
Header append Vary User-Agent env=!dont-vary
```

Config de eaccelerator :

```
extension="eaccelerator.so"
eaccelerator.shm_size="32"
eaccelerator.cache_dir="/tmp"
eaccelerator.enable="1"
eaccelerator.optimizer="1"
eaccelerator.check_mtime="1"
eaccelerator.debug="0"
eaccelerator.filter=""
eaccelerator.shm_max="0"
eaccelerator.shm_ttl="0"
eaccelerator.shm_prune_period="0"
eaccelerator.shm_only="0"
eaccelerator.compress="1"
eaccelerator.compress_level="9"
```

Copyright (c) 2007 psai

La permission vous est accordée de copier, distribuer et/ou modifier (dans le sens d'améliorer...) ce document suivant les termes de la GNU Free Documentation License, Version 1.1, ou de toute version ultérieure de cette même licence publiée par la Free Software Foundation.

La licence étant respectée il n'y a pas de sections invariantes.

La page de garde (Front-Cover) devra spécifier le nom du document, le nom de l'auteur (détenteur des copyrights) et mentionner explicitement la licence.

Afin de faire en sorte qu'une bonne action reste, de temps en temps, impunie, prenez connaissance de l'avertissement suivant :

Ce document vous est fourni dans l'espoir qu'il vous sera utile, mais SANS AUCUNE GARANTIE D'AUCUNE SORTE, NI EXPLICITE, NI IMPLICITE, QUE CE SOIT SUR SA PERTINENCE, SON EXACTITUDE OU L'ADEQUATION DES INFORMATIONS DONNEES A LA RESOLUTION D'UN PROBLEME PARTICULIER.

SON UTILISATION SE FAIT A VOS RISQUES ET PERILS, SANS QUE LES AUTEURS PUISSENT ETRE RENDUS RESPONSABLES DES DOMMAGES DIRECTS OU INDIRECTS, MATERIELS OU IMMATERIELS, QUI POURRAIENT EN RESULTER. CE DOCUMENT EST LIBRE DANS LE SENS DEFINI PAR LA LICENCE, ET VOUS ETES LIBRES EN PARTICULIER DE NE PAS L'UTILISER...