

Date de publication Vendredi 02 avril 2010 à 09:04:26 par Cyril
Catégorie Informations

Maximus 2010 de Cyril Levert, comparatif performances PHPNuke

De longue date je me suis toujours dit, un jour Marcel tu devrais t'amuser à faire un petit comparatif de performances des deux CMS sachant que PHPNuke est la base de Maximus (même s'il faut reconnaître que c'est bien loin et que maintenant nous en sommes de plus en plus loin, vu que maximus avance clairement dans le sens optimisation/sécurité alors que phpnuke est resté le même depuis plusieurs années déjà).

Le hasard à fait ce matin que je suis tombé sur une ancienne installation de PHPNuke 8.1 puis de PHPNuke extrem 8.1.1, cette seconde version étant ni plus ni moins un PHPNuke de base doté de Nukesentinel et de NSN Groups les deux scripts les plus gourmands du coeur de phpnuke.

Bien évidemment un PHPNuke Extrem avait des performances nettement inférieures à PHPNuke d'origine du fait de la gourmandise un peu abyssale de nukesentinel et du fait de la gestion des groupes qui est elle aussi très consommatrice ...

Bref, ce type de PHPNuke avait à l'époque suite à l'idée de PHPNuke Extrem fait fleurir sur la toiles des dizaines de projets comme celui ci (de mémoire j'ai commencé phpnuke extrem en 2003, ça en dit long), la dernière génération en date sur la toile de ce type de fork étant phpnuke pro

D'ailleurs à titre informatif un phpnuke pro était exactement pareil qu'un phpnuke extrem 8.1.1 (d'autres amateurs s'étaient à l'époque lancé dans l'aventure, mais faute de moyen ou de connaissances tous les autres projets ont été progressivement abandonnés).

Bref de tous ces CMS un seul est resté: Maximus

Alors pourquoi Maximus est il toujours d'actualité et pas les autres

Tout d'abord il faut bien être conscient que maintenir un CMS est un travail, un vrai travail, un travail quotidien dans lequel il faut s'investir.

Quand je dis investir, c'est dans tous les sens du terme, investir en temps (et là il en faut beaucoup, comptez 4 à 6 heures minimum par jour si un jour vous avez le désir réel de créer un projet de ce type), investir en argent car il vous faudra un nombre impressionnant d'outils, un serveur de développement, une bande passante qui va bien, un hébergement qui tient la route, des noms de domaine etc

Vous l'avez compris maintenir un projet gratuit du type CMS, ce n'est pas gratuit pour le développeur et c'est aussi pour ceci que nombre de projets sont morts de leur belle mort.

Reste bien évidemment ceux qui faisaient ça en bricolage et qui n'avaient rien à fiche des utilisateurs quitte à les laisser tomber comme on laisse tomber son chien en partant en vacance ...

Bref nous en sommes en 2010, et de la floppée de projets du type phpnuke extrem que l'on a pu voir fleurir à l'époque il n'en reste qu'un ... comme quoi finalement il y a des gens fidèles (pour suivre les évolutions), et des petits développeurs qui s'acharnent sur leur temps libre :)

Alors ce matin je me suis remémoré (en tuant la dernière requête SQL de nukesentinel de la mort qui tue, une requête bouvlé de 11 requêtes le tout sur toute les pages visitées), qu'à l'époque je me disais déjà la seule solution pour avoir un extrem au moins aussi rapide qu'un phpnuke il faudrait virer sentinel ... impossible donc :(

Donc pour résumer au mieux, les projets type phpnuke extrem étaient tous aussi bien les uns que les autres, mais avaient un gros défaut ils étaient encore plus gourmands que phpnuke lui même, en lui imposant un surcout de ressources important, très important même.

Il était donc devenu impossible de faire un phpnuke amélioré sans occasionner en même temps une explosion en requêtes SQL, en besoin en RAM, en temps de calcul bref déjà phpnuke était le CMS le plus lourd à l'époque mais avec ces forks on étaient tous dans la mauvaise direction en primant les fonctions du cms mais en omettant le point primordial: les performances.

J'avais donc dès 2003 commencer à envisager un système de cache pour phpnuke, par le biais de Jpcache, ce qui fût une erreur de par le fait que la classe n'étaient pas prévue pour phpnuke et ne pouvait pas s'y greffer sans contraintes énormes et des défauts de fonctionnement insolubles, j'ai donc basculé progressivement sur une autre classe de cache 'cache light' pour arriver aux mêmes conclusions.

Bref, il a fallu attendre Maximus BS donc en 2005/2006 pour que j'intègre non plus un cache pris au hasard sur le web, mais un système de cache (des systèmes de cache devrais je dire) développé spécifiquement pour maximus.

Bien évidemment ceci allait révolutionner le CMS coté performances, mais allait demander une revue extrêmement profonde du logiciel pour arriver à ce que nous sommes aujourd'hui.

Pour moi, ce fût une aventure fabuleuse, plaisante, pleine de rebondissements, enrichissante intellectuellement (car coté argent ça a été un peu le contraire, et je n'estime pas le temps passé ce serait mortel :().

Alors parlons simple, parlons performances ...

Je ne vais pas comparer PHPNuke d'origine puisque de toute façon il n'était pas utilisable en l'état, je vais donc me baser sur un phpnuke 8.1 doté de nukesentinel + nsn groups, version qui a servi de base pour Maximus BS donc.

Nous devrions donc retrouver des chiffres similaires à configuration

identique, et c'est là que ce matin je me suis amusé...

Ces tests ont été réalisés sur ma propre machine personnelle (pas de sites hébergés dessus, la machine ne me sert que pour le développement), j'ai pris soin de vérifier qu'aucun process ou cron n'était lancé, j'avais donc les 8 coeurs de mon Bi-Xéon libre, avec un serverload de 0.03 - 0.05 - 0.03 (machine sans consommation donc totalement disponible) et les sur les 12 gigas de ram du serveur 10 étaient disponibles.

J'ai donc pris le plus simple la page d'accueil vide

- PHPNuke : 70 requêtes SQL
- Maximus 2010: 1 requête SQL
- Maximus 2009: 2 requêtes SQL
- Maximus BS: 10 requêtes SQL

On voit ici clairement le travail effectué sur le besoin SQL 70 fois inférieur sur un CMS vide de données, premier élément important mais ce n'est pas le seul.

Temps de génération moyen enregistré pour une page d'accueil:

- PHPNuke: 0.0905 sec
- Maximus 2010: 0.035

Là encore un travail de fond important permet de couper en trois le temps de génération du CMS vide (je le répète)

Dernier point, mais là c'est extrêmement variable d'une machine à l'autre, de la ram disponible, de la charge machine enfin de pleins de paramètres, le besoin e ram par page:

- PHPNuke: 565 ko
- Maximus 2010: 355 ko

Cette dernière information est à prendre avec des pincettes toutefois, car j'ai pu avoir un phpnuke en top à 2.2 mégas et un maximus à 1.3 mégas en pic ceci est justifié car c'est vraiment en fonction de l'instant T de la machine et c'est très différente d'une page à l'autre.

On voit de toute façon clairement un besoin en RAM grosso-modo de moitié inférieur.

Bien évidemment, ce comparatif est réalisé avec une version dernier cri d'un cms et d'un autre qui date un peu, mais c'est un comparatif que j'avais déjà fait il y a longtemps et qui m'avait donc orienté vers un autre développement que phpnuke extrem puisqu'il était impossible de continuer en ce sens.

Le temps a prouvé mes tests de l'époque puisqu'aucun CMS du type PHPNuke extrem n'est plus présent sur la toile.

Il est d'évidence que ce type de test n'est faisable qu'à vide pour ne pas rendre les tests incomparables, car par exemple (et c'était très fréquent

avec phpnuke), si je met quelques bloc et quelques news sur la page d'accueil je vais me retrouver avec 300 requêtes sql de moyenne, à l'époque nous atteignons très souvent 500 requêtes sql par page !
Moi ça me faisait peur mais pas à d'autres :)

Bien évidemment, ce type d'explosion de besoins SQL n'est plus présent dans maximus, par exemple avec le module blog et cache sur le blog activé, le cms peut encore tourner avec seulement une requêtes SQL durant 99% de son temps.

En plus cerise sur le gâteau, maximus s'est vu doté de dizaines de fonctions supplémentaires dans le même temps !

Alors que les performances étaient nettement améliorées au fil du temps, les fonctions elles étaient nettement supérieures en nombre ce qui nous donne finalement un produit à tout faire sans devoir nécessairement lui greffer autre chose par dessus.

Vous l'aurez compris, le temps est passé ... et les choses ont bien changées ...

La question qui me vient à l'esprit est : dans 5 ans où en seront nous ?

Au final de cette aventure fabuleuse, je me serai non seulement éclaté mais j'aurai quand même réussi mon challenge initial à savoir garder les meilleurs éléments de la communauté phpnuke autour du projet, animer un site super sympa, et maintenir un cms de plus en plus performant, de plus en plus fiable et de plus en plus agréable :)

Bon allez, bonne bourre et au prochain comparatif ... dans 5 ans :)

Billet issu du site internet PHP Maximus CMS:

<http://www.php-maximus.org>

URL du billet

http://www.php-maximus.org/blog_maximus_2010_de_cyril_levert_comparatif_performances_phpnuke-347.html