Présentation du Fab PHPTopUnix

Introduction

Après vous avoir montré les possibilités d'exécution de programme par le serveur Unix (Fab IPFW PHP-Manager), ceci sans utiliser le retour de cette exécution, je vais maintenant montrer les possibilités d'exécution en récupérant les données rendues par le serveur (par exemple un 'ls' pour lister les fichiers dans un répertoire).

Je vais donc maintenant vous présenter un script PHP permettant d'afficher les infos en dynamique sur un serveur Unix (grace à la commande 'top').

Explication de l'utilité

Ce script a un but pédagogique afin de faire comprendre qu'on peut faire exécuter des commandes par le serveur et récupérer le résultat pour ensuite le traîter dans la suite du script PHP.
On peut aussi lui trouver une autre utilité qui est de pouvoir contrôler facilement l'état du serveur WEB (utile pour l'administrateur qui n'a donc pas besoin de se connecter en SSH sur celui-ci).

Source simplifiée

Pour le TOP :

```
<?
exec ("top",$test);
for ($index = 0; $index < count($test); $index++)
{
    if (STRPOS($test[$index], "root") == FALSE )
    {
        echo $test[$index] ."<BR>";
    }
    else
    {
        echo "<font color='red'>". $test[$index] ."</FONT><BR>";
    }
}
?>
```

A ce top, on peut maintenant rajouter la liste des utilisateurs et la charge du proc de la machine (Ceci pour montrer que le résultat de l'exec est transformé en une variable tout ce qu'il y a de plus classique).

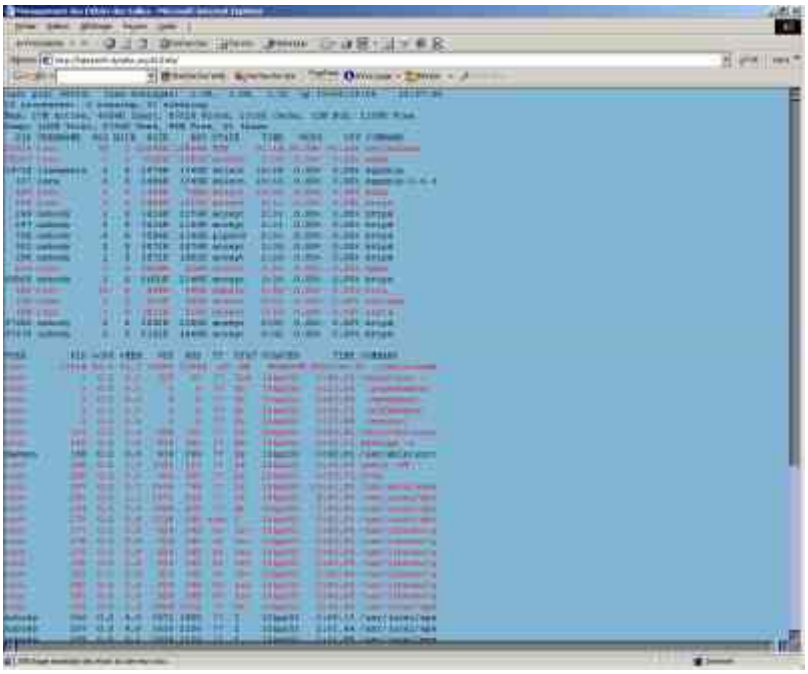
```
<?
exec ("uptime", $uptime);

list ($up, $temps, $user, $average, $av2, $av3) = split (", ", $uptime[0], 50);

$upt = split (" ", $up, 50);
echo "Allumé depuis : ". $upt[4] ." jours et". $temps ."<BR>";

$usr = split (" ", $user, 50);
echo $usr[0] ." Utilisateur(s) sur le serveur actuellement<BR>";

$avmoy = split (" ", $average, 50);
echo "Charge moyenne : ". $avmoy[2] ., ". $av2 ., ". $av3;
?>
```

Pour afficher le résultat à l'écran de façon correcte, il faut mettre avant l'affichage du top un un balise PRE qui va garder la même organisation que possédait le résultat (à l'écran en mode console).

Sources totales

Dans ces sources, vous aurez aussi un exemple avec un 'ps' en dynamique en même temps que le top.
Le téléchargement se fait ici :

- Scripts PHP3
- Scripts PHP4



ATTENTION : L'exemple en téléchargement est différent avec le refresh automatique réglé à 5 secondes. Ceci peut être très dur pour un serveur, car chaque exec lancé sur celui-ci par PHP lance en réalité un process de plus dans la liste donc prend un PID (base du système UNIX qui correspond au numéro unique de chaque processus). Il faut donc veiller à ne pas saturer la table des processus.

Conclusion

Pour la raison expliquée juste au dessus, je ne mets pas d'exemple de ce source en ligne, car je ne souhaite pas mettre mon serveur par terre.
Ce script (de la même facon que le Fab IPFW PHP-Manager) est présenté ici pour vous montrer que le PHP permet aussi d'exécuter du code sur des machine Unix en lancant des commandes (dès que le serveur APACHE en a l'autorisation) et de pouvoir en récupérer le résultat pour le traîter par ailleur. C'est une possibilité qui n'est pas existante en natif en ASP, car le serveur IIS ne peut pas exécuter de commande système. En ASP, la méthode est de développer une DLL (en VB ou C++) qui va être recensée sur le serveur Windows et qui correspondra à un objet en ASP et cet objet fera les exécutions souhaitées.

F___