TD 4

## CAS D'ATTAQUE PAR LA CORRUPTION D'UN SCRIPT PHP

*Pour illustrer les concepts de sécurité informatique vus en cours de façon concrète et pour vous faire toucher du doigt les dangers d'un réseau mal sécurisé présentant une ou des failles, nous allons détailler une méthode d'attaque d'un serveur Web via les failles existantes dans les scripts exécutés du coté serveur. Nous utiliserons dans notre cas une faille dans une page écrite en PHP pour prendre la main sur le serveur. Ce TD est assez long, ne perdez pas de temps...*

## 1. L'ENTREPRISE FIOUP

L'entreprise propose des services en bio-informatique. Ses clients sont des laboratoires pharmaceutiques qui veulent tester les effets de nouvelles molécules sur le fonctionnement cellulaire. Elle réunit des docteurs biologistes et des spécialistes en informatique pour la génétique. malheureusement, ces derniers sont bien moins experts en sécurité.
l'entreprise à ouvert un site internet accessible à partir de http://www.fioup.net/
Pour ce TP, Fioup va constituer notre cible, nous allons chercher à entrer dans le serveur Web et à en prendre le contrôle (le but étant de **remplacer la page d'accueil du site** par une de notre choix).

## 2. PRISE D'EMPRUNTE

Comme pour toute attaque, il nous faut collecter un maximum d'informations concernant la cible. Effectuez une prise d'empreinte du serveur Web de Fioup en utilisant les outils de réseau "`nslookup`", "`whois`", "`traceroute`", "`ping`".

**Question 2.1 :**

- Adresse IP du serveur
- Gestionnaire du nom de domaine
- Route vers le serveur

Pour des raisons de sécurité, beaucoup de logiciels utiles à une bonne prise d'emprunte sont désactivés sur le site de l'école. Tenez pour acquis les résultats suivant de la commande `nmap`

```
$ nmap fioup.net

Starting nmap 3.81 ( http://www.insecure.org/nmap/ ) at 2007-01-25 13:08 CET
Interesting ports on sd-6345 (88.191.38.225):
(The 1655 ports scanned but not shown below are in state: closed)
PORT    STATE SERVICE
21/tcp  open  ftp
22/tcp  open  ssh
25/tcp  open  smtp
80/tcp  open  http
```

```
110/tcp open  pop3
111/tcp open  rpcbind
143/tcp open  imap
443/tcp open  https
```

**Questions 2.2 :**

- Liste des ports ouverts sur le serveur, à quoi correspondent-ils ?

## 3. Page Web de Fioup

Rendez vous sur la page d'accueil de l'entreprise.

**Questions 3.1 :**

- Parcourez les pages et proposez un plan rapide du site

**Opération 3.1 :**

- Effectuez une copie locale du site avec `wget`

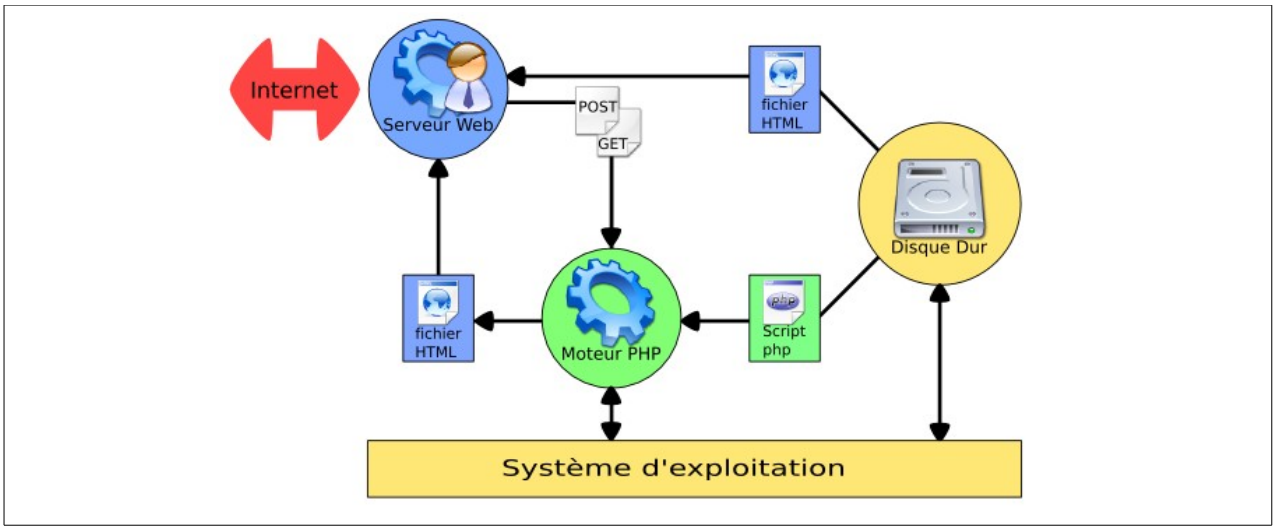
Nous allons nous intéresser plus particulièrement à la page "Network toolbox". Sur cette page figurent 2 liens et 3 formulaires qui semblent être des interfaces à des commandes bien connues de tests d'un réseau (`nslookup`, `ping` et `nmap`)

**Opération 3.2 :**

- Testez le fonctionnement des 3 formulaires avec l'IP du serveur de Fioup
- Testez maintenant les commandes correspondantes depuis votre console linux (avec `localhost` si l'IP de Fioup ne marche pas). Que constatez vous ? Que pouvez en déduire sur le fonctionnement de cette page Web ?

## 4. Page Web Script en php

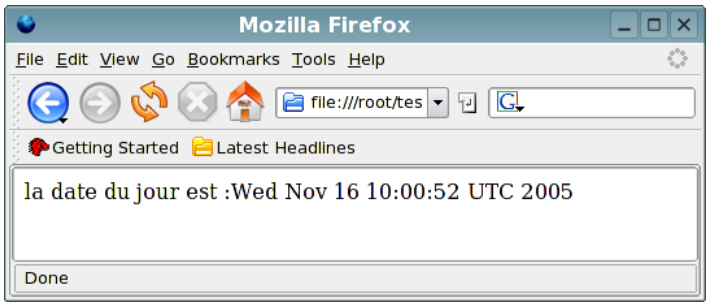
Un serveur Web permet la distribution de pages Web à des clients. En général, les pages Web sont écrites en HTML et dans ce cas, le serveur distribue ces pages directement au client, sans les modifier au préalable. Ces pages sont dites statiques. En utilisant PHP, il est possible d'intégrer des variables dynamiques au sein des pages Web, comme par exemple la date du jour, l'IP du client et bien d'autres. PHP est un langage de programmation complet qui permet aussi de traiter les données en provenance des formulaires des pages WEB. Le schéma simplifié ci dessous vous donne une idée de l'architecture d'un serveur Web utilisant PHP. L'accès aux pages Web se fait via une demande du client auprès du serveur. Deux cas de figure se présentent, soit il est demandé une page statique et le serveur récupère celle ci directement sur le disque dur du serveur, soit la page demandée est dynamique.




Comme le serveur ne sait gérer que de l'HTML statique, il délègue la tache de transformation de la page dynamique en HTML au moteur PHP. Le moteur php récupère le script à exécuter sur le disque dur et va traiter les instructions qu'il contient les une après les autres et envoyer la sortie de l'exécution de ce script au serveur Web. Le moteur PHP est une porte d'entrée vers le système d'exploitation. En fait un script php peut via le système d'exploitation accéder au disque dur, effacer ou créer des fichiers, lister des répertoires... comme pour tout langage de programmation, avec de la méthode, il est possible de tout faire. Normalement, le système d'exploitation, le disque dur et tout autre matériel ou service bas niveau ne sont pas accessibles à un utilisateur depuis Internet, seul le serveur Web doit faire l'interface.

```
<?php
echo "<html><body>\n";
$commande='date';
echo "la date du jour est :".exec($commande)."\n";
echo "</body></html>";
?>
```

```
<html><body>
la date du jour est :Wed Nov 16 10:00:52
UTC 2005
</body></html>
```




*Exemple de script PHP et sa sortie en HTML et enfin la vue dans le navigateur*

**Question 4.1 :**

- Comment pouvons nous savoir que la page "outils pour le réseau" à été générée à partir d'un script php ?

Le script donné en exemple utilise la fonction "`exec()`" avec pour argument la variable `$commande` qui contient la chaîne "`date`".

**Question 4.2 :**

- Quel est la fonction et le paramètres de "exec()" ? (cherchez sur php.net)
- Repérez sur la page la notion de "safe-mode", à quoi cela correspond-il ?

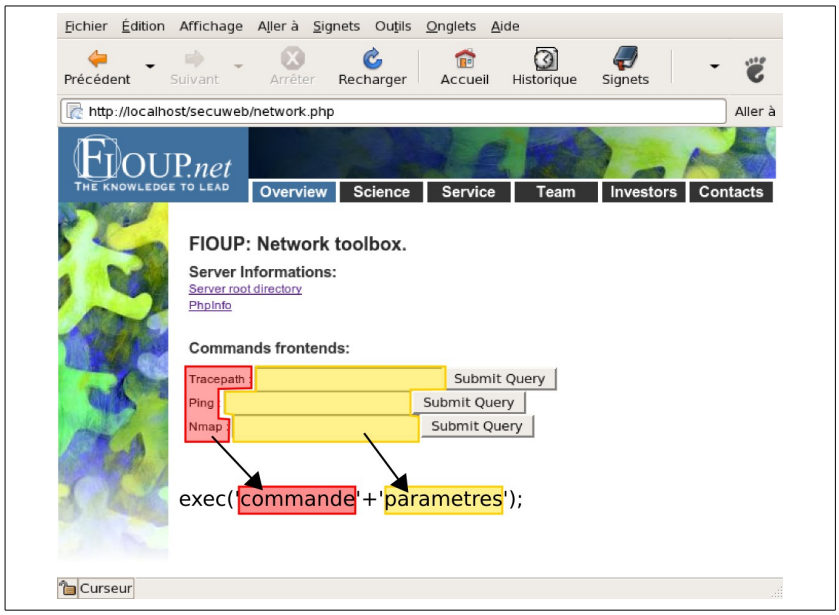
La page "Phpinfo" du site de Fioup résume la façon dont a été installé le moteur PHP sur le serveur. Bien entendu, ce genre d'information constitue une mine d'or pour tout attaquant...

**Question 4.3 :**

- Le moteur php est il en "safe-mode" ?
- Quelles sont les conséquences pour l'utilisation de fonctions telles que "exec" dans un script ?

## 5. Trou de sécurité

Les formulaires permettent de saisir des arguments spécifiés par les utilisateurs (dans le cas du formulaire, on attends des IPs). Cela signifie aussi que l'utilisateur, via ce qu'il va rentrer dans le champ de saisie, va pouvoir influencer le comportement de l'appel "exec()". Voici la décomposition de l'appel "exec()" que l'on peut déduire après l'analyse du fonctionnement du script :




*lien entre le formulaire et la fonction Php "`exec()`"*

L'appel serait composé d'une première partie, la commande (`ping`, `nslookup` ou `nmap`), sur laquelle nous ne pouvons pas intervenir directement. La partie paramètre correspond par contre directement au contenu du champ des formulaires de la page Web.
Après comparaison des sorties de commandes console et des sorties sur la page Web (question 2), il semble clair (même sans avoir vu le code) que le

script "network.php" utilise des appels système de commandes via la fonction PHP "`exec()`".

**Opération 5.1 :**

- En vous inspirant du code exemple, écrivez le code PHP qui renverrait un résultat à l'un des trois outils de la page

A priori, rien n'empêche - sauf peut être l'éthique - d'envoyer par le formulaire tout autre chose qu'une IP. On peut penser que tout texte saisi dans le champs sera passé en argument de la commande correspondante. Bien que nous n'ayons pas pour le moment de moyen d'en être sur, il est envisageable que cette page nous offre un moyen très simple et extrêmement courant d'exécuter des commandes sur le serveur : la prise de contrôle par "escape shell" (c'est avec le DOS, l'attaque technique de serveur la plus fréquente).
Vous avez peut être entendu parlé en programmation C d'attaque par dépassement de tampon, le "escape shell" est le pendant du "buffer overflow" pour les langages interprétés. Ces deux attaques sont possibles des lors que les entrées des utilisateurs ne sont pas ou alors mal vérifiées.
Les différents shells unix ("sh", "bash", "csh", etc) implémentent en général la notion de tuyaux ("pipes") pour enchaîner les commandes. "`|`" placé entre deux commandes va prendre le résultat de la première et en faire l'entrée de la seconde. La commande "`ls ~/ | grep test`" donne la liste de tout les fichiers ou répertoires de votre répertoire personnel qui contiennent la chaîne "`test`". La sortie de commande "`ls`" est l'entrée de la commande "`grep`" (Vous devriez connaître ces notions).

**Opération 5.2 :**

- Testez dans la console une commande chaînée qui après un "`nslookup`" affiche la date
- Testez dans la console une commande chaînée qui après un "`ping`" affiche le contenu de la racine

**Question 5.1 :**

- Quel est le résultat qui s'affiche ? où est passée la sortie des premières parties de la commande ?

**Opération 5.3 :**

- Sur la page "Network toolbox", dans le formulaire de la commande "`ping`", complétez l'habituelle IP par un pipe et une commande d'affichage du contenu de la racine (comme ce que vous venez de faire dans la console). Validez le formulaire

**Question 5.2 :**

- Le résultat retourné par le formulaire est-il celui attendu ?
- Essayez de passer la commande de pipe directement (sans l'IP attendue par "`ping`"). Cela fonctionne-t-il ? Pourquoi ?

## 6. Exploitation de la faille

Le plus technique est réalisé. Nous avons vu qu'il suffit d'envoyer un pipe suivi d'une commande de notre choix pour récupérer l'affichage du résultat dans le navigateur. Comme le moteur PHP n'est pas installé en safe-mode, nous avons la possibilité d'exécuter n'importe quelle commande avec les mêmes droits d'utilisation que le serveur WEB, c'est à dire que l'on fait à peu près ce que l'on veut. Pour vous en convaincre, testez les commandes suivantes depuis le formulaire :

**Opération 6.1:**

- Commande "`date`"
- Commande "`df -h`"
- Commande "`mount`"
- Commande "`whoami`"
- commande "`ps-aux`"
- Commande "`cat /etc/passwd`"

Bien sûr, nous pourrions être beaucoup plus nocif dans les commandes que nous envoyons .. un "`rm -fr /`" par exemple efface l'intégralité du disque dur !

**Question 6.1:**

- Décrivez le rôle et le résultat de chacune des commandes précédentes

Intéressons nous maintenant au système de fichier du serveur.

**Opération 6.2:**

- Listez avec `ls` les répertoires de "/home", puis "/home/fioup.net"
- Il semble que "/home/fioup.net" contient des fichiers au noms évocateurs ... Afficher avec `cat` le contenu de "passftp.txt"
- Continuez à parcourir les repertoires et trouvez le Graal ...

**Question 6.2:**

- Comment pourrait-on assurer une meilleure sécurité du fichier "passftp.txt" ?
- Quel fichier va-t-on chercher a modifier pour changer la page d'accueil du site (donner son emplacement sur le serveur) ?

Nous avons récupéré un compte et un mot de passe vers un FTP. la sortie de la commande "`nmap`" nous à indiqué qu'un serveur FTP sur port 21 était en route sur ce même serveur.
Dans la console, utilisez la commande "`ftp`" pour vous connecter avec les indications contenues dans le fichier. la session FTP s'ouvre sur une ligne de commande, "`help`" affiche la liste des commandes disponibles.

**Opération 6.3:**

- Utilisez les commandes "ls" et "cd" pour parcourir les répertoires du compte FTP
- Récupérez le fichier d'index du site (commande "get nom_fichier")

En local maintenant, éditez le contenu du fichier récupéré

**Opération 6.4:**

- avec un éditeur de votre choix modifiez le contenu de la page récupérée (rajoutez votre nom à la fin)
- renvoyez la version modifiée via le ftp (commande "put nom_fichier")

**Question 6.3:**

- Qu'advient-il de la page d'accueil de Fioup ? Le fichier avait-il déjà été modifié avant vous, par l'un de vos collègues ?

*Si vous êtes le premier à ajouter votre nom au fichier, félicitations, vous avez été le plus rapide (+2)*

## 7. Conclusion

Nous avons vu lors de ce TP comment un script mal écrit pouvait corrompre totalement la sécurité d'un serveur. PHP est un langage souvent proposé par des fournisseurs d'espace Web. Il est utilisable avec un minimum de formation par n'importe qui, certains éditeurs de pages Web comme "Dreamweaver" permettent même de générer automatiquement du code Php. Il résulte de cette apparente simplicité d'utilisation, un succès croissant auprès du grand public. Malgré cela, écrire des script sécurisés est une tache complexe que l'on aurait tort de négliger. Des lors que l'on interagit avec un utilisateur (via un site internet par exemple), il ne faut pas perdre de vue que celui-ci n'a pas forcement les meilleures intentions.

**Question 7.1 (option du concepteur):**

- Sans rentrer dans des détails trop techniques, listez les points faibles pour la sécurité du système (nous en avons vu beaucoup). Enfin, proposez vos idées pour améliorer la sécurité de ces différents points.

**Question 7.1 (option du technicien):**

- En ouvrant la page "Network toolbox" du site, on exécute en fait un script php. Ce dernier, "network.php" est récupérable via ftp. Étudiez son fonctionnement et expliquez pourquoi il est permit de douter de la fortuité de la faille que nous avons exploité. en particulier expliquez pourquoi les malins qui auraient voulu tester la commande d'effacement du disque dur en sont revenu bredouille ?

*Pour en apprendre plus sur la sécurisation des scripts PHP :*
*http://www.linux-pour-lesnuls.com/securiserscript.php*