ISEN ALL IS DIGITAL! BREST | yncréa	**Apache** **TP 4** **V1.1**	**CIR 2**

Utilisation des machines distantes

1. Sous Windows, utiliser le logiciel putty à télécharger à l'URL suivante (https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html) , une fois téléchargé il faut l'installer
2. connectez vous avec putty sur le serveur qui vous est associé avec le login : user et la password qui vous est fourni. Le mot de passe est à modifier immédiatement, n'oubliez pas de le mémoriser, le serveur vous servira de nombreuses fois.
3. Vérifiez si Apache est lancé sur votre serveur distant. Utilisez les commandes suivantes

```
sudo systemctl start apache2        => lancement
sudo systemctl restart apache2      => redemarrage
sudo systemctl stop apache2         => arret
sudo systemctl reload apache2       => rechargement de la conf, sans arrêt
```

4. créer sur le serveur distant un répertoire /var/www/cir2/apache/tp4.

   Dans ce répertoire, créez un fichier **index.html** (contenu libre) et un fichier **test.php** (voir modèle en annexe).

   Créez un virtualhost permettant d'accéder au répertoire **/var/www/cir2/apache/tp4** avec le ServerName correspondant à : ap-tp4-ex1-XXX.serveur (XXX correspondent à vos initales). Et faites que ce virtual-host soit accessible depuis votre navigateur

   L'accès avec votre navigateur à ce « serveur name » doit afficher la page **index.html** que vous avez insérée.
5. Copiez cette page **index.html** en **test.html** (dans le même répertoire) et changez le texte afin de les différencier.

   Vérifiez que vous visualisez la page **test.php**

Gestion de .htaccess

Éléments de cours

Apache permet de délocaliser la gestion de la configuration, au moyen de fichiers spéciaux appelés par défaut **.htaccess (attention au 'point' qui précède le nom)**.

Chaque fichier .htaccess est placé directement dans le répertoire dont il doit gérer la configuration particulière et éventuellement protéger l'accès.

Ce mécanisme de délégation doit au préalable être soumis à autorisation comme décrit ci-dessous.

• La syntaxe à employer dans ces fichiers .htaccess est identique à la syntaxe utilisées dans apache2.conf. On peut en particulier y inclure des restrictions d'accessibilité par hôte et des autorisations d'accès par utilisateur.

• Fonctionnement : Les fichiers .htaccess étant lus dynamiquement au moment de chaque requête qui concerne son répertoire, toute modification de ces fichiers prend effet immédiatement, contrairement à apache2.conf, pour lequel il est nécessaire de faire relire la configuration au serveur (donc intervention de root !)

Mais alors n'y aurait-il pas possibilité de conflit avec les directives placées dans apache2.conf dans un conteneur de directives <Directory chemin-rép> ...</Directory> ?

C'est le rôle de la directive **AllowOverride** de préciser la manière selon laquelle les directives contenues dans un fichier .htaccess doivent être prises en compte, c'est-à-dire si ces directives ont "le droit" de supplanter ou non celles qui sont incluses dans la présente directive.

• Ainsi, l'administrateur a le dernier mot ! S'il veut inhiber totalement l'action de .htaccess, il précisera **AllowOverride None** pour le répertoire. Sinon, il peut accorder des droits complets au fichier .htaccess avec **All** (prise en compte totale) ou des droits limités en ne positionnant que certaines valeurs. On limite souvent cette délégation de gestion à **AllowOverride AuthConfig** ou **AuthUserFile**, ce qui est suffisant pour protéger l'accès à un site privé par une authentification par exemple.

Informations complémentaires :
http://httpd.apache.org/docs/current/howto/htaccess.html

Application

6. Dans le vhost créé précédemment ajoutez une directive associée au répertoire /var/www/cir2/apache/tp4 qui spécifie le droit d'utilisation de .htaccess
7. créez un fichier .htaccess qui indique que le fichier index est le fichier test.html et vérifiez le bon fonctionnement
8. modifiez le vhost pour interdire l'utilisation de .htaccess et rechargez le site. Qu'est-ce qui a changé ? Pourquoi ?

Gestion des authentifications

Apache intègre un système d'autorisation simple de protection des répertoires au travers de directives spécifiques.

Dans un premier temps créer un répertoire « private » sous le répertoire /var/www/cir2/apache/tp4 et copiez y la page index.html dans laquelle vous changerez le texte pour la distinguer des autres pages (par exemple indiquez « page privée »).

9. créez dans le vhost une règle spécifique pour ce répertoire autorisant uniquement votre poste et celui d'un de vos voisins pour y accéder et interdisant la visualisation des fichiers dans le répertoire.

10. Ajoutez la règle permettant d'accéder directement au fichier index.html

En utilisant .htaccess

En vous référant à la documentation Apache, vous allez utilisez la méthode « AuthType Basic » : https://httpd.apache.org/docs/2.4/howto/auth.html

11. créez (avec htpasswd) un utilisateur **isenuser** avec son mot de passe et créez la configuration adéquate dans le vhost pour que seul cet utilisateur (**isenuser**) puisse accéder au répertoire **private**.
Quelle est la différence un utilisateur créé de cette manière et votre utilisateur user1 ?

12. créez un groupe d'utilisateurs (**usergroupisen**) contenant **isenuser** et **isenuser2** (nouvel utilisateur) et modifiez le vhost pour que les membres de ce groupe puissent accéder au répertoire **private.**
Note : il est nécessaire de charger un module supplémentaire dans Apache (situé dans mods-available) avec la commande **a2enmod** pour pouvoir profiter de cette fonctionnalité.

13. commentez les instructions relatives à l'authentification dans le vhost et placez les dans un nouveau fichier .htaccess et vérifiez le fonctionnement

14. *Question subsidiaire : Les données du dossier **private** sont-elles totalement protégées ? Comment un pirate pourrait-il y avoir accès malgré la protection mise en place dans les questions précédentes ?*

ANNEXE

fichier test.php

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