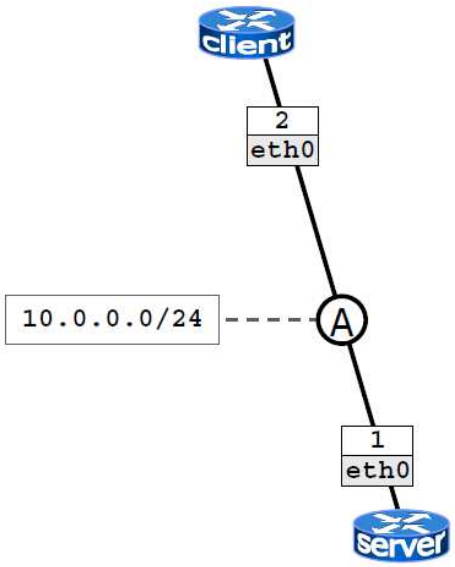
7. Web Server and Browser

http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_webserver/netkit-lab_webserver.pdf

Topologie du réseau :



Nous allons utiliser le laboratoire disponible ici :

http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_webserver/netkit-lab_webserver.tar.gz

Sur la machine réelle utiliser les commandes :

cd netkit-lab_webserver

lstart

Server

- il utilise apache2 (avec une configuration par default)

Client

- le client va utiliser un browser text-based (commande "*links*") pour communiquer avec le serveur

Vérifiez que le serveur apache2 est actif, en allant sur la machine "server" et en utilisant la commande : **/etc/init.d/apache2 status**

Par default, apache2 setup offre une page de test html page, qui se trouve dans : **/var/www/index.html**

Q7.1 : Vérifiez le contenu de cette page html.

Coté "client", vous pouvez utiliser le browser "links". Pour accéder au menu bar, utiliser la touche F10, et ensuite sélectionner “Go to URL” (Enter). Ensuite il faut rentrer l'URL : http://10.0.0.1/

Q7.2 : Vérifiez le résultat de cette commande.

Vous pouvez monitorer les acces au serveur apache2 à l'aide du fichier "access.log", qui se trouve en /var/log/apache2 (et, par example, de la commande "tail pour le lire).

Q7.3 : Lisez le contenu de ce fichier.

Vous pouvez monitorer les erreurs (très utile en phase de debugging) sur le web server à l'aide du fichier "error.log", qui se trouve dans le meme repertoirs que access.log (et, par example, de la commande "tail pour le lire).

Q7.4 : Lisez le contenu de ce fichier.

La plupart de fonctionnalités apache sont déjà "built-in".

Vous pouvez en voir la liste avec la commande : **apache2 -l**

D'autres peuvent être ajoutées avec des modules, qui peuvent être activés comme suit : **a2enmod rewrite**

Q7.5 : Vérifiez le résultat de cette commande.

A noter qu'il faut redémarrer apache juste après.

Les modules disponibles se trouvent en : /etc/apache2/mods-available

Les modules activés en : /etc/apache2/mods-enabled

La commande a2enmod crée un lien symbolique pour le(s) fichier(s) concerné(s) en

/etc/apache2/mods-available to

/etc/apache2/mods-enabled

La commande "a2dismod" sert à éliminer ces lien symboliques.

Voici quelques modules apache utiles :

userdir	enables per-user web sites
rewrite	implements URL rewriting
proxy	implements URL rewriting
cgi/cgid	supports execution of CGI scripts

Nous allons activer maintenant ele module "userdir" :

Q7.6 : Ecrire la commande nécessaire pour l'activation.

Par défaut, userdir cherchera "per-user sites" dans /home/username/public_html

Il suffit de regarder dans le fichier de configuration: /etc/apache2/mods-enabled/userdir.conf

Q7.7 : Créez un site pour "guest" sur le serveur.

- Il faut tout d'abord créer le répertoire /home/guest/public_html
- Créez un fichier index.html à mettre dans ce dossier.
- Accédez à cette page à partir du "client" avec links, l'URL etant maintenant **http://10.0.0.1/~guest/**

Per-directory configuration :

Apache permet de configurer les accès pour chaque directory.

Il faut crées le fichier spécial /some/path/.htaccess , qui contient des "statement" de configuration d'apache, et ces "statements" seront appliqués à tous les sub-directories de /some/path.

Les fichiers .htaccess peuvent etre "nested" dans un arbre, avec la règle : "nested files override their parents"

Quelques exemples de "configuration statements " :

- Restrict access from specific hosts

  Deny from example.org test.com 10.0.0 192.168.0.0/24

- Perform URL rewriting

  (transparent) redirect to other sites

- Restrict access to a specific subdirectory

  enable client-side authentication

- Change name of file containing the default page

  DirectoryIndex pippo.html

- Enable/disable directory indexing

  Options -Indexes

Q7.8 : Exercice : per-directory configuration.

Editez le fichier /home/guest/public_html/.htaccess en ajoutant la directive suivante :

DirectoryIndex custom_file.html

Ensuite, renommez le fichier /home/guest/index.html en --> custom_file.html

Essayez d'accéder, à partir du client, à http://10.0.0.1/~guest/

Renommez custom_file.html à nouveau en --> index.html et essayez d'accéder à nouveau à la page.

- Ecrivez dans le compte rendu les résultats de ces actions.

8. Common Gateway Interface (CGI)

http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_cgi/netkit-lab_cgi.pdf

Nous allons utiliser le laboratoire disponible ici :

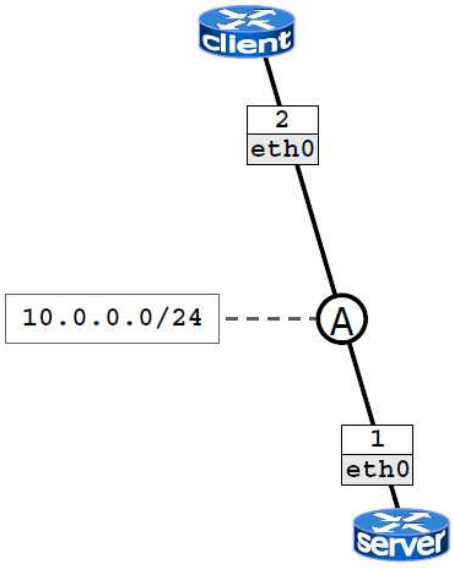
http://wiki.netkit.org/netkit-labs/netkit-labs_application-level/netkit-lab_cgi/netkit-lab_cgi.tar.gz

Sur la machine réelle utiliser les commandes :

cd netkit-lab_cgi

lstart

Topologie du réseau :



Server

- il utilise apache2 (avec une configuration par default)
- héberge une page web et un simple script shell qui montre comme les paramètres en input sont passés à l'application CGI

Client

- le client va utiliser un browser text-based (commande "*links*") pour expérimenter avec CGI

Dans ce laboratoire, apache2 est déjà " up and running" après que la machine virtuelle a été créée.

La page par default (/var/www/index.html) se compose de 3 sections, qui permettent d'expérimenter avec les différents mécanismes de passage de paramètres

Le script CGI (/usr/lib/cgibin/test.sh) montre comment le paramètres ont été reçus.

Pour expérimenter, vous pouvez utiliser sur le client le browser "links" pour vous connecter au serveur : links http://10.0.0.1/

Q8.1 : Ecrivez un compte rendu des résultats de votre expérimentations.