NE302 – Projet Réseau
Projet « HTTP Server » - étape 1

1 Travail préparatoire

Lisez la documentation de l'outil netcat (dans un terminal, faire man netcat).

Une première introduction à la RFC HTTP est préférable si le sujet vous est inconnu.

Pour cela allez sur le site :

http://www.bortzmeyer.org/9110.html

http://www.bortzmeyer.org/9112.html

2 Objectifs de la séance 1 – Projet HTTP

Cette séance 1 va permettre :

- la (re)découverte du protocole HTTP en utilisant un serveur web « complet ».
- d'effectuer des captures réseaux vous permettant de mieux comprendre les spécifications des RFCs sur HTTP. Vous devrez conserver ces captures comme références pour vos travaux ultérieurs.

Pour les captures réseaux utilisez wireshark et sauvegardez les paquets au format pcap.

3 Pourquoi lire une specification ?

Lisez le chapitre 1.1.3 du document https://www.rfc-editor.org/rfc/rfc3986#page-7 et indiquez votre compréhension de la différence entre URI URL et URN.

4 Installation d'un serveur web « complet »

Sur votre station de travail (en tant qu'utilisateur root), installez un serveur web avec le package apache2 avec les commandes suivantes :

#apt-get update

#apt-get install apache2

Vérifiez que le serveur écoute sur le port 80, avec la commande

 # netstat -antp

ou

 # ss -lntp

Q1 : Que signifie l'adresse 0.0.0.0 ? Est-ce le cas pour tous les serveurs installés sur cette station ?

Q2 : Concrètement sur quelle(s) adresse(s) IP pouvez-vous contacter votre serveur HTTP ?

Vous avez maintenant installé le serveur HTTP, via les packages Debian, celui-ci est donc installé avec une configuration par défaut, il vous faut maintenant comprendre cette configuration.

Les fichiers de configuration se situent dans /etc/apache2 et celui qui nous intéresse plus particulièrement est le fichier « /etc/apache2/sites-enabled/000-default.conf ».

Editez ce fichier et grâce à la documentation de apache https://httpd.apache.org/docs/2.4/fr/server-wide.html, trouvez ce que signifie chaque directive de configuration :

Q3 : Dans quel répertoire se situent les fichiers correspondant aux pages html servies par le serveur ?

Q4 : Ouvrez le fichier index.html dans ce répertoire, en quel langage est-il écrit ?

Q5 : Où peut-on trouver les spécifications de ce langage ?

Lancer un navigateur et connectez-vous sur le serveur que vous venez de lancer. Trouvez dans le fichier index.html la partie qui s'affiche dans le navigateur.

Nous allons maintenant mettre en place plusieurs sites sur notre serveur HTTP, pour cela il faut créer un nouveau « VirtualHost »

Créer un répertoire /var/www/html2

Modifiez le contenu de manière à pouvoir le différentier de l'autre fichier index.html.

Ajouter les lignes suivantes à la fin du fichier de configuration

/etc/apache2/sites-enabled/000-default

```
<VirtualHost *:80>
        ServerName www.toto.com
        DocumentRoot /var/www/html2
</VirtualHost>
```

et redémarrez le serveur :

```
# systemctl restart apache2
```

Vérifiez que le serveur est bien démarré avec la commande « ss » précédente, si cela n'est pas le cas regardez les erreurs remontées par la commande systemctl status apache2

Pour que tout fonctionne correctement vous devez modifier le fichier /etc/hosts (à la fin de la première ligne)

```
127.0.0.1       buster.b120.esisar.fr   buster www.toto.com
```

Q6 : Pourquoi devez-vous faire ceci ? Que se passerait-il si vous ne le faisiez pas ? (vous pouvez faire des captures réseau avant et après la modification)

5 Schéma

Faites un schéma dans lequel figurent :

Le client, le serveur, le navigateur, le resolveur DNS du client, le serveur HTTP, le système de fichier, et le fichier. Indiquez par des flèches et des labels sur ces flèches les interactions effectuées entre ces entités et les protocoles de communication.

6 Utilisation de HTML

HTML est le format de données permettant de représenter les pages web.

Voici un exemple de fichier HTML simpliste :

```
<html>
 <head>
  <title>
   Exemple de HTML
  </title>
 </head>
 <body>
  Ceci est une phrase avec un <a href="cible.html">hyperlien</a>.
  <p>
   Ceci est un paragraphe où il n'y a pas d'hyperlien.
  </p>
 </body>
</html>
```

Dans /var/www/html2, écrire une première page HTML simple (index.html) à votre convenance, et afficher ensuite cette page dans votre navigateur. Via l'url www.toto.com/index.html

Q7 : combien de requêtes HTTP faut-il pour afficher l'ensemble de la page ?

Ecrivez une deuxième page contenant une image balise <img>, exemple :

Faites les captures réseau correspondant à ces pages HTML.

Q8 : combien de requêtes HTTP faut-il pour afficher l'ensemble de la page ?

7 Découverte de HTTP

Le protocole HTTP/1.x fonctionne au dessus du protocole TCP, il permet de manipuler des ressources distantes. Le client envoie des requêtes avec le format suivant :

Ligne de requête (Methode, URL, Version de protocole)
En-tête de requête
[Ligne vide]
Corps de requête

La ligne de commande indique ce que le client attend, par exemple obtenir la page d'accueil : `GET /index.html HTTP/1.0`

L'entête de requête contient une succession de couples (variable : valeur) , par exemple `User-Agent: Mozilla`.
Elle permet au client de donner des informations à son propos ou de préciser la requête.

La ligne vide est très importante elle distingue le header HTTP du reste du message.
Le corps de la requête peut contenir des données additionnelles (dans le cas de messages de type POST par exemple)

Le serveur répond avec un format similaire :

Ligne de statut (Version, Code-réponse, Texte-réponse)
En-tête de réponse
[Ligne vide]
Corps de réponse

La ligne de status indique au client si sa requête a réussi. Parmi les réponses possibles:

- HTTP/1.0 200 OK indique le succès de l'opération
- HTTP/1.0 404 Not Found, indique que la page demandée n'existe pas, etc...

L'entête permet au serveur de préciser la réponse par exemple en indiquant la taille et le type de la ressource demandée. La ligne vide sépare ici aussi l'entête de la ressource demandée dans le corps du message.

Naviguer sur votre station de travail avec un navigateur classique. Pour les 2 URL suivantes :

http://127.0.0.1/index.html
http://www.toto.com

Faites des captures réseaux afin de répondre aux questions suivantes :

Q9 : quelle version de HTTP est utilisée par votre navigateur ?
Q10 : Pour la page contenant l'image que vous avez créée, pour chaque requête, quelle est l'URL ?

Q11 : Combien d'entêtes y a t-il dans la requête ? A quoi servent-ils ?
Q12 : Combien d'entêtes y a t-il dans la réponse ?

8 Visualisation du protocole avec netcat

Pour mieux comprendre les requêtes effectuées, vous allez les refaire « à la main » avec l'utilitaire netcat. Effectuez la même navigation, en tapant directement à la main les commandes HTTP

Ex : nc -C 127.0.0.1 80
GET / HTTP/1.0

Q13 : A quoi sert l'option -C pourquoi est-elle nécessaire ?
Q14 : Pourquoi faut-il appuyer 2 fois sur la touche <RETURN>

Faites des captures réseaux en utilisant le protocole HTTP/1.0 et HTTP/1.1 en utilisant une requête GET.

Q15 : Quels sont les seuls entêtes nécessaires pour obtenir une réponse de type 200 OK de la part du serveur pour chacune des versions ?

Q16 : Quels sont les entêtes permettant d'accéder au site correspondant à www.toto.com que vous avez ajouté en début de séance.

Vous allez tester les cas suivants avec netcat, et vous noterez quelles sont les réponses apportées par le serveur (code retour) (vous resterez avec le protocole HTTP/1.0).

Utilisez une méthode non connue	*ex : HELLO / HTTP/1.0*
Ne pas mettre de version de protocole	*ex : GET /*
Mettre une version de protocole non supportée	*ex : GET / HTTP/3.3*
	ex : GET / WEB/2.0

Pour l'instant on s'est contenter de la ressource /, vous allez maintenant expérimenter différentes ressources :

```
GET toto HTTP/1.0
GET /toto HTTP/1.0
GET /%?? HTTP/1.0
GET /index.html HTTP/1.0
GET /%69%6E%64%65%78%2E%68%74%6D%6C HTTP/1.0
GET index.html HTTP/1.0
```

```
GET http://www.facebook.com/index.html HTTP/1.0
etc..
```

Notez pour chacun les codes retournés. Essayez avec d'autres requêtes...

9 Gestion de la connexion

Vous allez utiliser GET /index.html HTTP/1.x (avec x qui vaut 0 ou 1) et netcat.

Dans la version 1.0 la connexion est tout de suite fermée dès que l'on à reçu la réponse valide.

Envoyez une requête valide en version HTTP/1.1 et attendez la réponse 200.

Q17 : Que se passe t-il au bout de quelques secondes.

Q18 : Lisez la section 6.3 de la RFC7230 trouvez l'entête HTTP permettant de recevoir la réponse, et de refermer tout de suite la connexion.

10 - Gestion du champ content-type

Dans le répertoire référencé par DocumentRoot, placez une image de type gif, puis dupliquez plusieurs fois ce fichier en changeant l'extension du fichier → html, jpg, txt, ou autres.. et un fichier sans extension.

Créez aussi un fichier de contenu aléatoire

dd if=/dev/urandom of=random bs=512 count=10

Pour chacun des fichiers effectuez une requête via le navigateur et observez avec le debug firefox ou wireshark les valeurs du champ content-type dans la réponse.

Est-ce que l'extension à une influence sur ce champ, est-ce que cela modifie l'affichage de l'image sur le navigateur ? → donnez une conclusion sur la génération du champ content-type.

11 - Simulation du serveur

Maintenant que vous connaissez un peu le fonctionnement d'apache, essayez avec netcat de simuler le fonctionneent de celui-ci pour servir une page html ou une image.

Pour cela vous pourrez utiliser un script shell (demandez à l'enseignant)