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

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

Il s'agit ici de l'adresse à laquelle s'attache l'application (plus précisément la socket via la fonction bind de l'API des sockets Linux). Effectivement une station de travail possède plusieurs interfaces (lo, eth0, eth1, etc..) et donc souvent autant d'adresse IP. Spécifier l'adresse 0.0.0.0 (aussi appelée INADDR_ANY) indique que l'application s'accroche à TOUTES les adresses IP configurées sur la station.

Référence : man 7 ip.

When a process wants to receive new incoming packets or connections, it should bind a socket to a local interface address using bind(2). In this case, only one IP socket may be bound to any given local (address, port) pair. When INADDR_ANY is specified in the bind call, the socket will be bound to all local interfaces.

C'est le cas pour toutes les application (par exemplle serveur ssh), c'est un comportement classique.

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

Toutes les adresses IP configurées : 127.0.0.1 et 192.168.130.X (suivant le PC)

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

Puisque la doc indique que la directive correspondante est DocumentRoot, il s'agit donc de /var/www/html

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

Comme aussi son extension l'indique (même si ce n'est pas toujours vrai) il est écrit en HTML= HyperText Markup Langage (pour être plus précis en XHTML) Comme indiqué dans la première ligne du fichier

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

C'est le W3C (World Wide Web Consortium) qui édite les spécifications de ces langages www.w3.org (allez voir)

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)

Les manipulations qu'on vient de faire permette à notre serveur de savoir comment répondre quand il reçoit une requête adressée à www.toto.com, mais pour que le test fonctionne, il faut que notre navigateur arrive à envoyer les requêtes pour www.toto.com à notre à IP 192.168.130.X ou 127.0.0.1

Puisque notre station est configurée pour faire une résolution locale (fichier) puis DNS pour les noms de domaine (cf fichier /etc/nsswitch.conf) (ligne host, lisez le man si vous ne connaissez pas), si on cherche à résoudre www.toto.com, tant qu'il n'existe pas dans le fichier de résolution locale, on utilise le DNS.

www.toto.com. 600 IN A 202.232.69.147

Nous allons forcer la résolution de www.toto.com à 127.0.0.1 (ce qui est fait en mettant cette ligne dans le fichier /etc/hosts)

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

Une requête pour la page sans image

Deux requêtes pour la page avec image (une pour le fichier HTML) puis ensuite une pour l'image (gif ou jpg)

Q9 : quelle version de HTTP est utilisée par votre navigateur ?

On le voit dans les traces réseau sur le premier TCP comportant des DATA (GET /index.html HTTP/1.1)

Q10 : Pour la page contenant l'image que vous avez créée, combien de requêtes HTTP y a t-il, quelles sont les URL?

Deux requêtes pour la page avec image (une pour le fichier HTML)

(index.html)

puis ensuite une pour l'image (gif ou jpg) donc bien sûr il faut avoir copier l'image sur le serveur ….

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

Ca dépend du navigateur, mais on retrouve généralement ces entêtes

Accept	text/html,application/xhtml+xm…plication/xml;q=0.9,*/*;q=0.8
Accept-Encoding	gzip, deflate
Accept-Language	en-US,en;q=0.5
Connection	keep-alive
Host	192.168.130.X
User-Agent	Mozilla/5.0 (X11; Linux x86_64…) Gecko/20100101 Firefox/68.0

Notez que l'entête utilise un ':' comme séparateur entre le nom de l'entête et le contenu.
Il servent soit à préciser la requête (host RFC723 5.4. Host), soit à indiquer les caractéristiques du client pour que le serveur fasse une réponse le mieux adaptée possible (ex Accept ou Accept-Encoding RFC7231 5.3.2 et 5.3.4), soit à indiquer le comportement que doit avoir le serveur (Connection RFC7230 6. Connection Management)

Q12 : Combien d'entêtes y a t-il dans la réponse ?
Idem cela dépend de la requête mais on trouvera par exemple :

Connection	Keep-Alive
Content-Length	1024
Content-Language	en-US
Content-Type	text/html;charset=UTF-8
Date	Thu, 13 Feb 2020 15:22:27 GMT
Server	Apache/2.4.6 (Red Hat Enterpri…SSL/1.0.2k-fips mod_qos/11.64
Transfer-Encoding	chunked

On remarquera aussi que ces entêtes servent soit à mieux décrire le contenu renvoyé par le serveur, soit à donner des informations sur le serveur. (il peut y avoir aussi d'autres usages que l'on verra plus tard)

Q13 : A quoi sert l'option -C pourquoi est-elle nécessaire ?

Le man de netcat nous indique « -C Send CRLF as line-ending »

Effectivement sous Linux le passage à la ligne suivante est uniquement constitué de LF (0xA ou \n)

Or la grammaire de HTTP nous indique qu'une requête est de la forme (RFC 7230 Appendix B)

Au passage vous pouvez déjà regarder la syntaxe de cette grammaire (ABNF) (RFC 5234)

HTTP-message = start-line *(header-field CRLF) CRLF [message-body]
start-line = request-line / status-line
request-line = method SP request-target SP HTTP-version CRLF

Il faut donc envoyer en fin de chaque ligne les 2 caractères CR LF (0xA et 0xD) C'est pourquoi netcat va traduire l'appui sur return en CR LF grace à l'option -C

Q14 : Pourquoi faut-il appuyer 2 fois sur la touche <RETURN>

Idem, la grammaire indique
HTTP-message = start-line *(header-field CRLF) CRLF [message-body]

Il y a donc une ligne vide entre la fin des entêtes et le corps du message (qui est optionnel et souvent inexistant dans les requêtes de type GET), donc une fois <RETURN> pour la fin des entêtes, une fois <RETURN> pour la ligne vide.

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

Pour HTTP/1.0 aucun entête n'est strictement nécessaire.
Pour HTTP/1.1 seul l'entête Host est strictement nécessaire.

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

Il suffit d'ajouter l'entête
Host : www.toto.com

pour que cela fonctionne.

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

Avec un netcat la connexion reste active, on n'a pas le prompt « $ » qui indique que le programme est toujours en exécution car le serveur de son côté n'a pas fait la demande de fermeture de connexion. Mais au bout de quelques secondes on s'aperçoit que le programme se termine et que le prompt apparaît, il semblerait donc que le serveur (avec les traces réseau) ferme finalement la connexion.

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.

Il s'agit de

Connection : close

Il faut lire le paragraphe de management de la connexion pour bien comprendre à quoi cela sert, je vous laisse voir.