

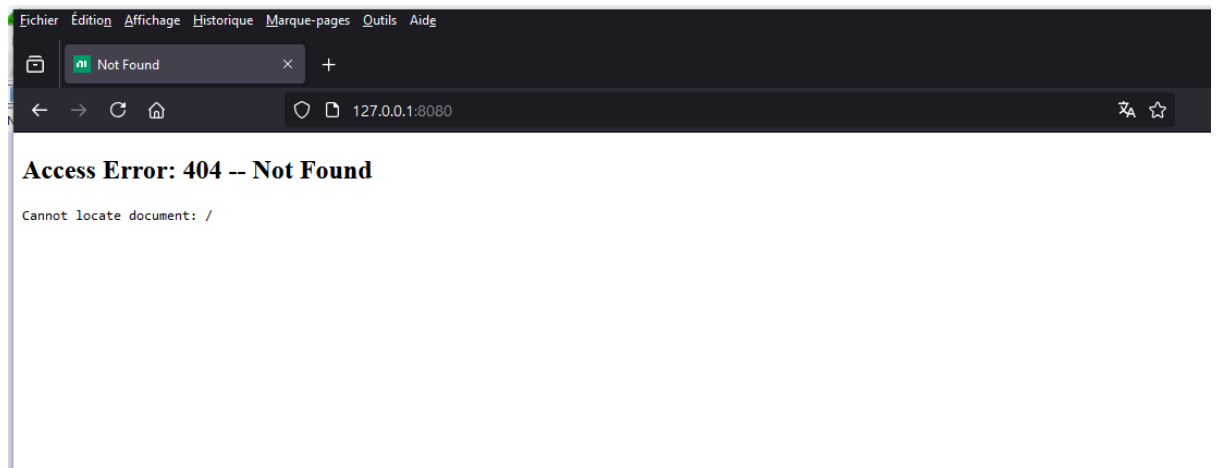
Laboratoire 1

REPONSES À QUESTION NUMERO 7 :

Nom : Bah Prénom : Mamadou Sanoussy Code Permanent : BAHM02080500

Nom : Bah Prénom : Kadiatou lamarana Code Permanent : BAHK30629900

a) **Capture d'écran :**



b) **Message affiché :**

L'erreur **404 - -Not Found** (cannot locate document).

c) **Explication de l'affichage du message :**

L'erreur 404 –Not found (cannot locate document) signifie que la page ou la ressource demandée n'a pas été trouvée sur le serveur web. Cette erreur se produit le plus souvent lorsque L'URL demandée ne correspond à aucun fichier ou ressource sur le serveur.

Voici quelques raisons de cette erreurs :

URL incorrecte ou mal orthographiée : L'URL que vous avez saisie pourrait être incorrecte

Page ou fichier supprimé ou déplacé : La page ou le document que vous tentez de trouver a peut-être été supprimé ou déplacé sans que l'URL ait été mise à jour.

Lien cassé : Si vous avez cliqué sur un lien, il est possible que le lien soit cassé, c'est-à-dire qu'il mène vers une ressource qui n'existe plus.

Accès restreint : Certaines pages peuvent être restreintes ou protégées.

Problème temporaire du serveur : Parfois, le serveur peut rencontrer des problèmes temporaires qui l'empêchent de localiser ou d'afficher la page demandée.

d) **A partir des détails de paquet, trouvons les champs les plus pertinents du protocole**

Accept language: Fr FR ,fr ;q=0.9, en-US ;0.8, en ; q=0.7

Host : Ce champ spécifie l'adresse du serveur auquel la requête est adressée. Ici, `127.0.0.1` est l'adresse locale (localhost) de la machine, et `8080` est le port du serveur web qui écoute la requête. C'est typiquement utilisé lors du développement local.

3. Connection : keep-alive

- **Connection** : Ce champ spécifie si la connexion TCP entre le client et le serveur doit être maintenue ouverte après la réponse. La valeur `keep-alive` signifie que la connexion doit être maintenue ouverte pour de futures requêtes, ce qui améliore la performance.

4. User-Agent : Mozilla/5.0 (Windows NT 10.0; Win64 ; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/130.0.0.0 Safari/537.36

- **User-Agent** : C'est une chaîne qui permet au serveur de savoir quel navigateur et quel système d'exploitation sont utilisés par le client. Cela permet de personnaliser la réponse en fonction du client. Dans ce cas :
 - **Mozilla/5.0** : Traditionnellement utilisé pour indiquer une compatibilité avec Netscape, mais de nos jours utilisé par la majorité des navigateurs.
 - **Windows NT 10.0** : Le système d'exploitation est Windows 10.
 - **Win64 ; x64** : L'architecture du processeur est 64 bits.
 - **AppleWebKit/537.36** : Le moteur de rendu utilisé, ici WebKit (utilisé par Safari et Chrome).
 - **Chrome/130.0.0.0** : Le navigateur est Google Chrome version 130.
 - **Safari/537.36** : Le moteur de rendu Safari (WebKit) est également mentionné, car Chrome utilise WebKit à l'origine avant de migrer vers Blink.

5. Accept: text/html, application/xhtml+xml, application/xml; q=0.9, image/webp, image/apng, /*; q=0.8, application/signed-exchange; v=b3 ; q=0.7

- **Accept** : Ce champ indique les types de contenu que le client accepte de recevoir du serveur. Chaque type de contenu est suivi d'un paramètre *q* (qualité), qui indique la priorité :
 - **text/html** : Le client accepte les pages HTML.
 - **application/xhtml+xml** : Le client accepte aussi le format XHTML.
 - **application/xml** : Le client accepte le format XML.
 - **image/webp, image/apng** : Le client accepte les images au format WebP ou APNG (type d'images animées).
 - **/*** : Le client accepte tous les types de contenu avec une priorité plus faible ($q=0.8$).
 - **application/signed-exchange** : Le client accepte aussi le format signed-exchange, utilisé pour la validation des échanges HTTPS.

Accept-Encoding: gzip, deflate, br, zstd

- **Accept-Encoding** : Ce champ indique les types de compression que le client accepte. Cela permet de réduire la taille des données envoyées et d'améliorer les performances de chargement des pages :
 - **gzip** : Compression standard.
 - **deflate** : Autre méthode de compression.
 - **br** : Brotli, une compression plus moderne et plus efficace.
 - **zstd** : Zstandard, un autre algorithme de compression moderne.

Accept-Language: fr-FR, fr; q=0.9, en-US; q=0.8, en; q=0.7

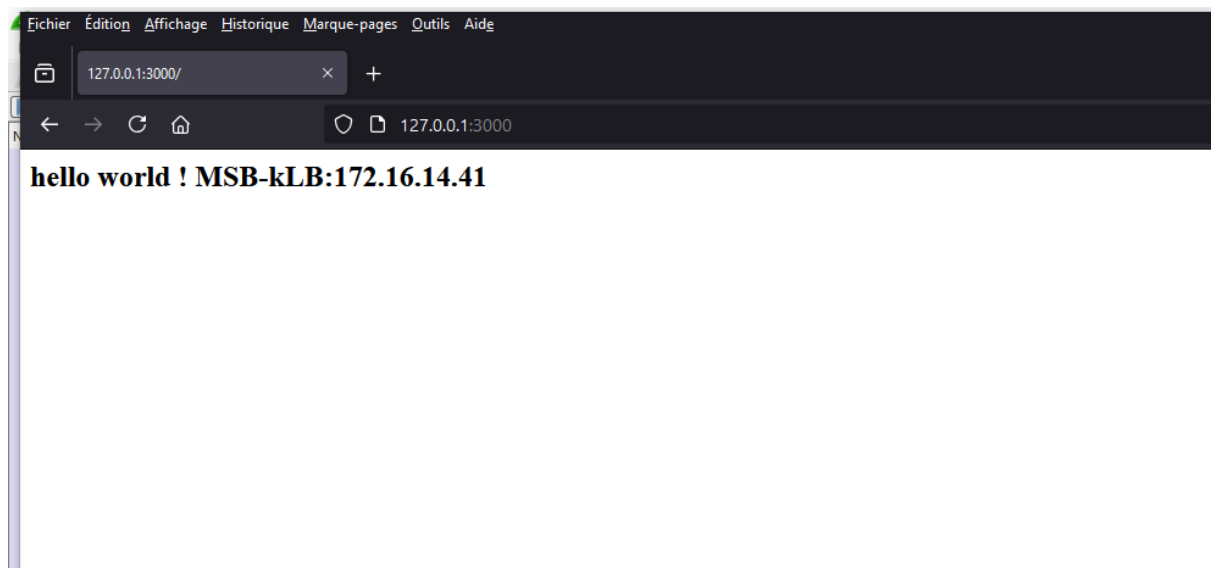
- **Accept-Language** : Ce champ indique les préférences de langue du client pour la réponse :
 - **fr-FR** : Le client préfère le français (France).
 - **fr** : Le client accepte également le français en général.
 - **en-US** : Le client accepte l'anglais (américain).
 - **en** : Le client accepte aussi l'anglais, en général, avec une priorité plus faible.

Ces informations permettent au serveur de personnaliser la réponse en fonction du client (navigateur, langue, compression, etc.).

REPONSES À QUESTION NUMERO 8 :

Après avoir saisi l'adresse sur le navigateur ;

a) **Capture d'écran est la suivante :**



b) **Le message donner par le navigateur est le suivant :**

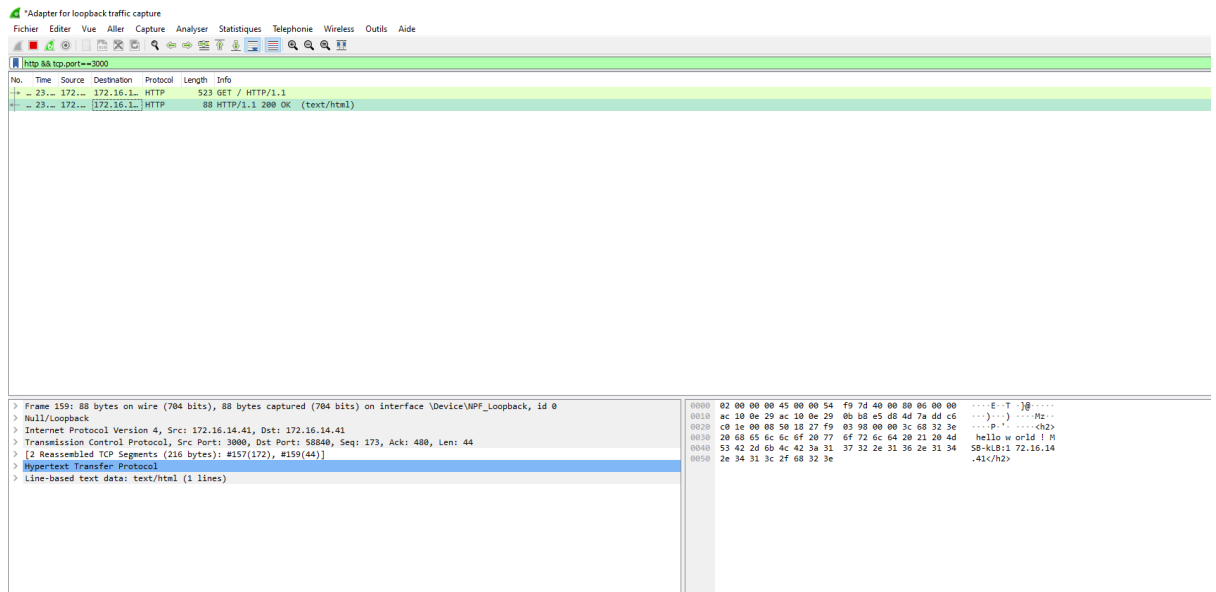
Hello world ! MSB-kLB :172.16.14.41

c) **J'explique en quelque phrase ce qui s'est passé :**

Le serveur qui écoute sur <http://127.0.0.1:3000> répond avec ce message pour indiquer que tout fonctionne correctement. Le serveur est configuré pour afficher le message "Hello world !" suivi de l'ID de la machine ou du nom d'hôte **MSB-kLB** et de l'adresse IP **172.16.14.41**.

- **127.0.0.1** fait référence à mon propre ordinateur (localhost), donc le serveur fonctionne sur ma machine.
- Le message peut être utilisé pour identifier des informations sur le serveur ou le réseau qui héberge l'application web.

d) **A partir des détails de paquet, trouvons les champs les plus pertinents du protocole :**



Explication :

Protocole :

HTTP (HyperText Transfer Protocol).

Est le protocole de communication utilisé sur le web. C'est ce qui permet aux navigateurs (comme Chrome, Firefox, etc.) et aux serveurs web de s'échanger des informations.

Détails HTTP :

Statut : HTTP/1.1 200 OK.

Cela signifie que la requête a été traitée avec succès et qu'il n'y a pas eu d'erreur. Le code de statut 200 OK est la réponse standard pour une requête HTTP réussie.

Serveur : Werkzeug/3.0.6 Python/3.8.4.

Werkzeug est un serveur web utilisé avec le framework Flask pour le développement local. Ici, il est utilisé pour exécuter l'application Flask.

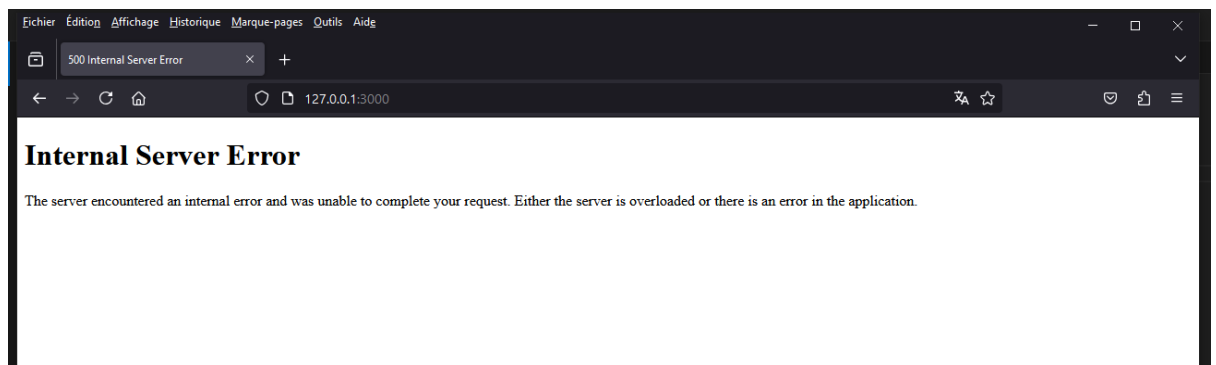
Type de contenu : text/html ; charset=utf-8.

Cela signifie que le contenu de la réponse est du type HTML (HyperText Markup Language), qui est utilisé pour structurer les pages web.

charset=utf-8 précise que le jeu de caractères utilisé pour encoder le texte dans la réponse est UTF-8, un encodage largement utilisé pour les caractères internationaux.

Réponse à la question N9:

a) **La capture d'écran obtenu est :**



b) **Le message donné par le navigateur est le suivant:**

Internal Server Error :

<<The server encountered an internal error and was unable to complete your request .Either the server is overloaded or there in an error in the application>>

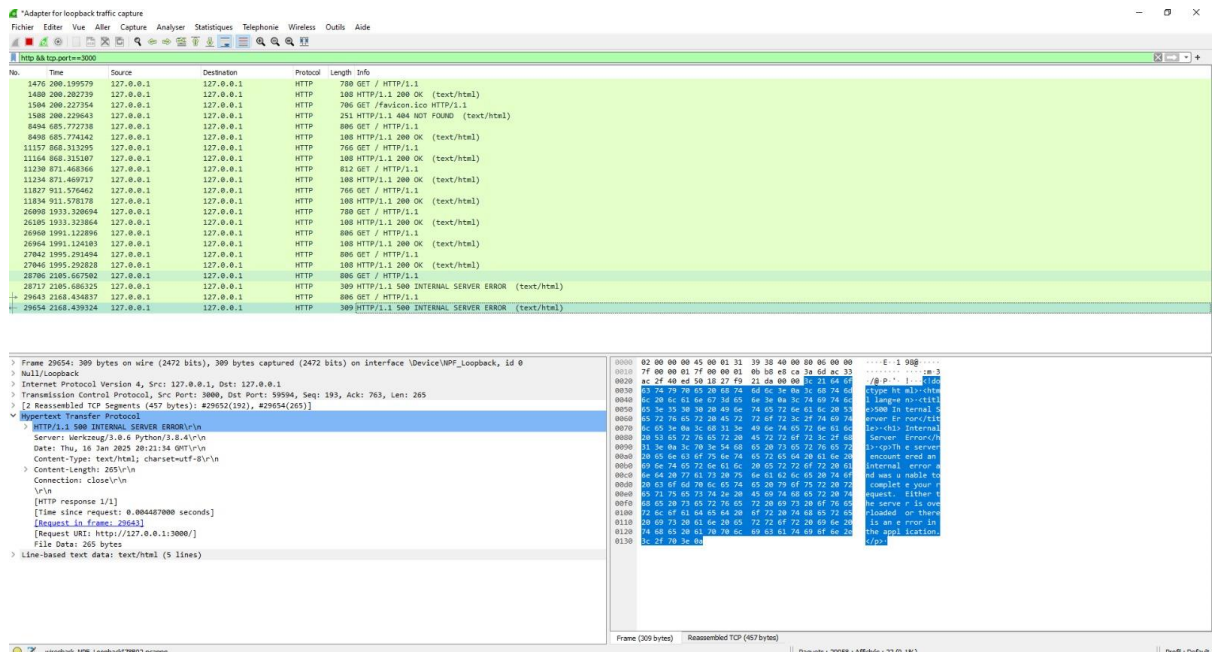
c) **Explication de ce qui s'est passé :**

Le message "**Internal Server Error**" (erreur interne du serveur) signifie que le serveur auquel on essaye d'accéder a rencontré un problème lors du traitement de la demande. Cela peut être causé par plusieurs raisons, telles que :

1. **Erreur dans l'application web** : Il peut y avoir un bug dans le code de l'application qui empêche le serveur de traiter correctement la requête.
2. **Problème de configuration du serveur** : La configuration du serveur peut être incorrecte, ce qui empêche son bon fonctionnement.
3. **Surcharge du serveur** : Le serveur peut être trop sollicité, avec trop de requêtes à traiter simultanément, ce qui conduit à un échec du traitement.

Ce message indique qu'un problème a eu lieu sur le **serveur** et que la requête ne peut pas être traitée.

d) **À partir des détails de paquet, trouvons les champs pertinents du protocole :**



Les champs pertinents sont :

Protocole HTTP :

Statut de réponse :500 INTERNAL SERVER ERROR

EN tête HTTP :

Server :Werkzeug/3.0.6 Python/3.8.4 : indique que le serveur utilise Python et le micro Framework flask(via werkzeug).

Content-type : texte/html ;charset= utf-8 : le type de contenu renvoyé est du HTML.

Connection:close : la connexion est fermée après la réponse.

Contenu HTML :

La réponse inclut une page HTML basique avec un titre (500 Internal Server Error) et un message détaillé :
 <p>The server encountered an internal error or misconfiguration and was unable to complete your request...</p>

Client/serveur :

Source : 127.0.0.1 (client local).

Destination : 127.0.0.1, port 3000 (serveur local).

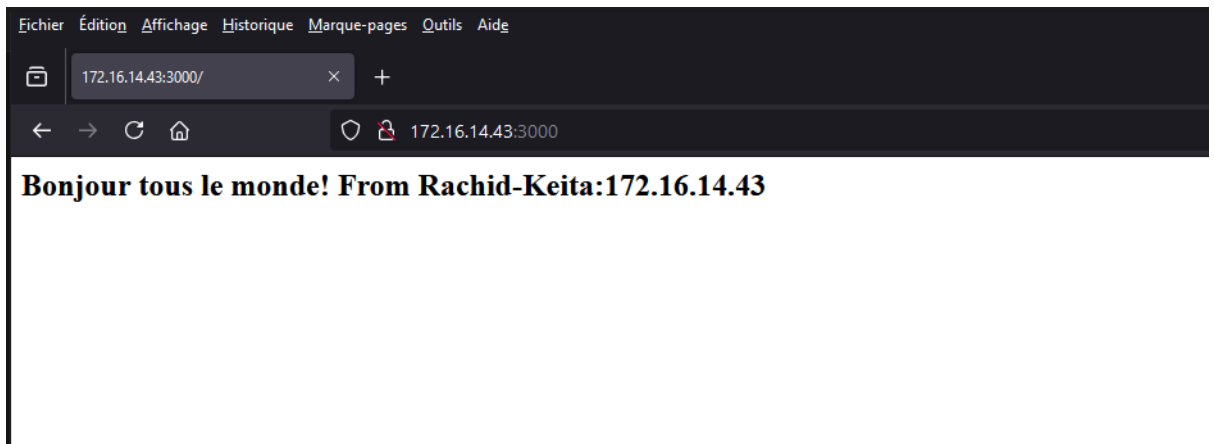
Requête associée :

URI de requête : <http://127.0.0.1:3000/>.

10) la capture d'écran étant arrêté.

RÉPONSE À LA QUESTION N11

a) La capture obtenue est la suivante :



- b) Le message donnée par le navigateur après l'utilisation de l'adresse IP de la machine d'une autre équipe est :

<<Bonjours tous le monde ! From Rachid-Keita : 172.16.14.43

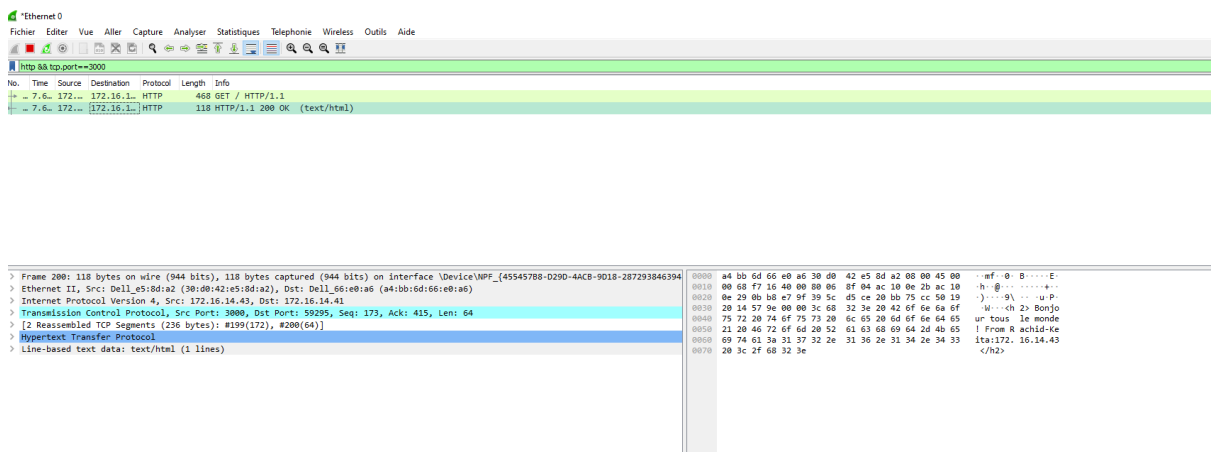
- c) voici l'explication de ce qui s'est passée en quelques lignes :

Le navigateur a envoyé une requête HTTP GET à l'adresse `http://172.16.14.43:3000/`.

Le serveur (probablement un serveur Flask ou autre microframework) a traité la requête et a renvoyé une réponse HTTP 200 (succès).

Le serveur a répondu avec une page HTML contenant le message "Hello, World ! From Rachid-Keita " accompagné de l'adresse IP locale du serveur.

- d) Les champs les plus pertinents du protocole sont:



Protocole HTTP :

Requête : GET / HTTP/1.1

Le navigateur demande la ressource racine /.

Réponse : 200 OK

Le serveur a répondu avec succès.

Contenu HTML :

La réponse contient un message en HTML affichant : Hello, World ! From Rachid-Keita : 172.16.14.43

En-têtes HTTP possibles :

Réponse :

Content-Type : text/html ; charset=utf-8 : Indique que le contenu renvoyé est au format HTML.

Connection : keep-alive ou close selon la configuration du serveur.

Requête (du navigateur) :

User-Agent : Indique les informations sur le navigateur utilisé. Cela peut être vu dans la capture Wireshark associée.

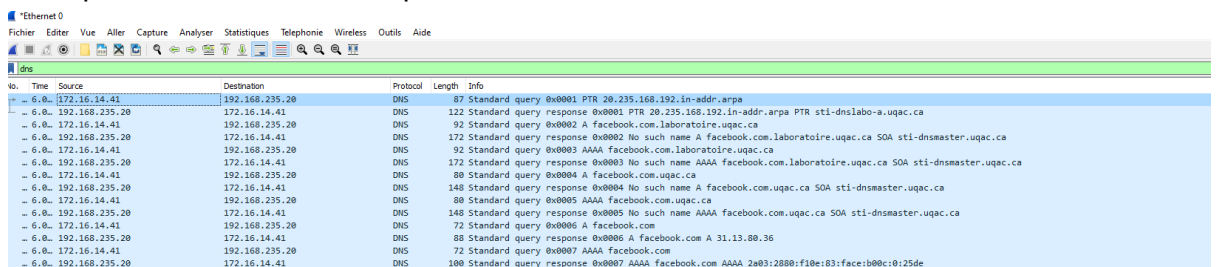
Client/serveur :

Source : Navigateur exécuté par le client à une IP spécifique.

Destination : 172.16.14.44, port 3000 (serveur local sur le réseau privé).

2)Service DNS

La capture d'écran obtenu à partir du wireshark est la suivante :



No.	Time	Source	Destination	Protocol	Length	Info
6.	0.000000	172.16.14.41	192.168.235.20	DNS	87	Standard query 0x0001 PTR 20.235.168.192.in-addr.arpa
7.	0.000000	192.168.235.20	172.16.14.41	DNS	122	Standard query response 0x0001 PTR 20.235.168.192.in-addr.arpa PTR stl-dnsiabo-a.uqac.ca
8.	0.000000	172.16.14.41	192.168.235.20	DNS	92	Standard query 0x0002 A facebook.com.laboratoire.uqac.ca
9.	0.000000	192.168.235.20	172.16.14.41	DNS	172	Standard query response 0x0002 No such name A facebook.com.laboratoire.uqac.ca SOA stl-dnsmaster.uqac.ca
10.	0.000000	172.16.14.41	192.168.235.20	DNS	92	Standard query 0x0003 AAAA facebook.com.laboratoire.uqac.ca
11.	0.000000	192.168.235.20	172.16.14.41	DNS	172	Standard query response 0x0003 No such name AAAA facebook.com.laboratoire.uqac.ca SOA stl-dnsmaster.uqac.ca
12.	0.000000	172.16.14.41	192.168.235.20	DNS	80	Standard query 0x0004 A facebook.com.uqac.ca
13.	0.000000	192.168.235.20	172.16.14.41	DNS	148	Standard query response 0x0004 No such name A facebook.com.uqac.ca SOA stl-dnsmaster.uqac.ca
14.	0.000000	172.16.14.41	192.168.235.20	DNS	80	Standard query 0x0005 AAAA facebook.com.uqac.ca
15.	0.000000	192.168.235.20	172.16.14.41	DNS	148	Standard query response 0x0005 No such name AAAA facebook.com.uqac.ca SOA stl-dnsmaster.uqac.ca
16.	0.000000	172.16.14.41	192.168.235.20	DNS	72	Standard query 0x0006 A facebook.com
17.	0.000000	192.168.235.20	172.16.14.41	DNS	88	Standard query response 0x0006 A facebook.com A 31.13.80.36
18.	0.000000	172.16.14.41	192.168.235.20	DNS	72	Standard query 0x0007 AAAA facebook.com
19.	0.000000	192.168.235.20	172.16.14.41	DNS	100	Standard query response 0x0007 AAAA facebook.com AAAA 2a03:2880:f10e:83:face:b00c:0:25de

a) Identifions les paquets des requêtes DNS qui correspondent à notre requête requête :

- *Source : * 192.168.235.20 (la machine cliente émettrice des requêtes).
- *Destination : * 172.16.14.41 (le serveur DNS interrogé).
- *Info : * Les lignes contenant "*"Standard query"* (A, PTR ou AAAA).

Exemples :

1. *Paquet 1 : * Standard query 0x0001 PTR (recherche d'un nom à partir de l'IP).
2. *Paquet 3 : * Standard query 0x0008 A facebook.com.
3. *Paquet 5 : * Standard query 0x0009 AAAA facebook.com.

b. identifions les paquets des requêtes DNS qui correspondent à notre requête

Les paquets correspondant aux réponses DNS sont ceux où :

- Source : 172.16.14.41 (le serveur DNS qui répond).
- Destination : 192.168.235.20 (la machine cliente qui reçoit la réponse).
- Info : Les lignes contenant "Standard query réponse".

Exemples :

1. Paquet 2 : Réponse PTR pour l'adresse IP recherchée.
2. Paquet 4 : Réponse indiquant "No such name" pour facebook.com.laboratoire.uqac.ca.
3. Paquet 7 : Réponse contenant une adresse IP valide pour facebook.com* (31.13.80.36).

C) Identifions les champs pertinents de chaque paquet :

Pour les requêtes DNS :

1. Transaction ID : Identifiant unique pour lier requête et réponse (exemple : 0x0008 pour *A facebook.com*).
2. Flags : Champ indiquant une requête (Query = 0).
3. Questions : Contient le nom de domaine demandé (exemple : facebook.com).
4. Type : Indique le type de recherche (A pour IPv4, AAAA pour IPv6, PTR pour reverse DNS).

Pour les réponses DNS :

1. Transaction ID : Identique à celui de la requête correspondante (exemple : 0x0008).
2. Flags : Indique une réponse (Réponse = 1).
3. Answers : Contient les enregistrements de réponse (adresse IP, ou message d'échec).
4. Type : Le type de réponse (A, AAAA, ou SOA pour les erreurs).

Dans l'image, on observe une capture Wireshark montrant des paquets DNS (Domain Name System).

Port 53 :

Le *port 53* est utilisé par le protocole DNS. Il s'agit du port standard pour les requêtes et réponses DNS, que ce soit en UDP (pour la plupart des requêtes) ou en TCP (pour les transferts de zone ou les requêtes volumineuses). Ici, il est utilisé pour envoyer une réponse DNS au client.

Port 60592

Le *port 60592* est un *port éphémère* ou temporaire utilisé par le client. Lorsque le client initie une requête DNS, il choisit un port aléatoire (dans la plage des ports éphémères) comme port source. Dans cet exemple, c'est le port 60592. Ce port est unique pour cette session et permet de recevoir la réponse correspondante depuis le serveur DNS.

- Le *port 53* est celui du serveur DNS (destination pour la requête, source pour la réponse).
- Le *port 60592* est un port temporaire choisi par le client pour établir la communication avec le serveur DNS.

Résumé de ce que nous avons compris

Nous avons compris qu'avec le premier message d'erreur 404, cela indique qu'une ressource demandée sur le serveur web n'a pas été trouvée, souvent en raison d'une URL incorrecte, d'un fichier déplacé ou d'un lien cassé. L'exemple d'une requête HTTP montre des champs importants comme le verbe HTTP GET, le type de contenu et le statut de la réponse. Avec le serveur qui répond avec le message "Hello World !" Cela nous a appris une autre expérience qui nous indique que le message qui a été envoyé à belle et bien eu du succès. Nous avons évidemment appris qu'avec l'erreur (500) cela veut dire que le problème qui s'est produit a pour source à l'intérieur de l'application. Dans le cadre du DNS nous avons constaté et appris que les paquets DNS capturés illustrent les échanges entre un client et un serveur DNS, où des requêtes et réponses identifient des noms de domaine ou des adresses IP via des transactions spécifiques, avec des ports comme 53 pour le DNS et des ports éphémères pour le client. Ces informations montrent comment les serveurs web et DNS fonctionnent pour traiter et répondre aux requêtes des clients.

