TP3 — Découvrir Wireshark : capturer et filtrer des trames

Prendre en main l'analyseur de paquets Wireshark : capturer le trafic, appliquer des filtres d'affichage, observer ping, HTTP et la différence entre HTTP et HTTPS.

Plateforme : poste Windows avec Wireshark installé et un accès au réseau local et à Internet.

Wireshark est un analyseur de protocoles (analyseur de paquets) utilisé pour dépanner les réseaux, effectuer des analyses, développer des logiciels et des protocoles et s'informer. Un analyseur de paquets (ou analyseur de réseaux ou de protocoles) est un logiciel permettant d'intercepter et de consigner le trafic des données transférées sur un réseau de données. Une capture réseau est comme une photo à un instant t de ce qui transite sur un réseau informatique.

Découvrir Wireshark

Pour pouvoir capturer des données, vous devez d'abord vous connecter au réseau depuis l'ordinateur sur lequel Wireshark est installé et exécuter Wireshark.

> **Question 1**
>
> Relevez l'adresse IP de votre PC, l'adresse de la passerelle et la noter

> **Question 2**
>
> Lancer Wireshark et sélectionner l'interface de votre connexion réseau.



Figure 1 – Écran d'accueil de Wireshark : choix de l'interface de capture.

Vous devez obtenir un résultat similaire à celui-ci-dessous composé de 3 fenêtres

Section 1 : La liste des paquets capturés disponibles en dessous de la barre de menu avec un affichage synthétique du contenu des paquets

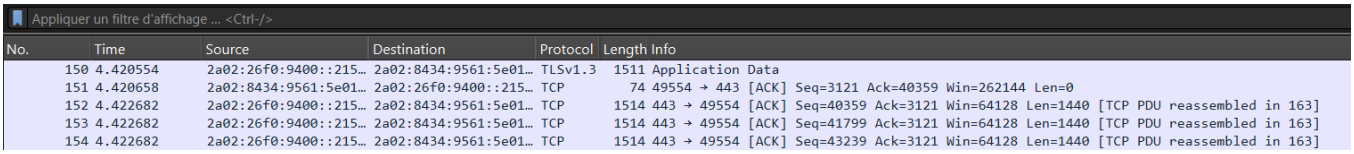


Figure 2 – Section 1 : liste des paquets capturés.

Section 2 : La décomposition exacte du paquet actuellement sélectionné dans la liste. Cette décomposition permet de visualiser les champs des en-têtes des protocoles ainsi que l'imbrication des différentes couches de protocoles connus.

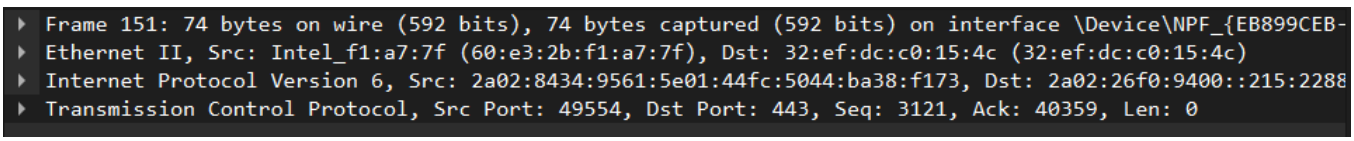


Figure 3 – Section 2 : détail du paquet sélectionné.

Section 3 : Cette zone contient la capture affichée en hexadécimal et en ASCII. Le trafic sur un réseau étant très important, il est nécessaire de mettre en place des filtres, c'est-à-dire sélectionner les communications destinées uniquement à votre carte, ou bien encore sélectionner un type de protocole.

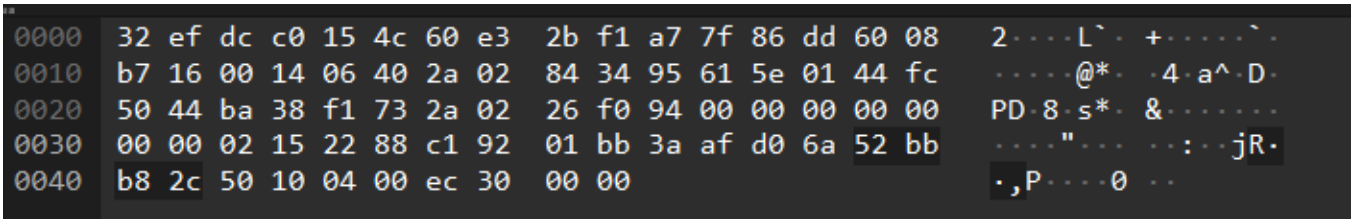


Figure 4 – Section 3 : contenu de la trame en hexadécimal et en ASCII.

Question 3

Repérez les différentes colonnes de la section 1, indiquez le type de contenu affiché dans ces différentes colonnes.

Question 4

Effectuer une recherche de façon à définir les termes : trame Ethernet et protocole Ethernet (citez vos sources)

Question 5

Donner le format d'une trame Ethernet, citez les différents champs.

1. Lancez une capture à l'aide de Wireshark en sélectionnant l'interface « Ethernet ».

Question 6

Énumérez 3 différents protocoles exécutés par votre PC. et faites recherches sur ces protocoles.

Question 7

(voir le tableau ci-dessous pour répondre aux questions)
Appliquez les filtres d'affichage suivants à votre capture :

a) Les paquets avec un port TCP de destination égal à 443.

b) Tous les messages TCP sauf ceux dont le port de source ou de destination est 80.

c) Seulement les paquets envoyés par votre machine.

d) Tout sauf les paquets icmp.

Objectif	Filtre Wireshark	Explication
Afficher tout le trafic d'une IP source	`ip.src == 192.168.1.10`	Affiche les paquets **envoyés par** cette IP
Afficher tout le trafic d'une IP destination	`ip.dst == 192.168.1.10`	Affiche les paquets **reçus par** cette IP
Afficher tout le trafic d'une IP (source ou destination)	`ip.addr == 192.168.1.10`	Combine les deux cas précédents
Filtrer un protocole particulier	`http` ou `dns` ou `arp`	Affiche uniquement les paquets de ce protocole
Exclure un protocole	`not arp`	Masque tous les paquets ARP

Objectif	Filtre Wireshark	Explication	
Combiner plusieurs conditions	`ip.src == 192.168.1.10 and tcp.port == 80`	Filtre sur une IP **et** un port	
Ou logique	`tcp or udp`	Affiche les paquets TCP **ou** UDP	
and	`ip.src == 192.168.1.10 and tcp.dstport == 80`	Filtre les paquets venant de l'IP 192.168.1.10 **ET** dont le port de destination est 80	Seuls les paquets HTTP provenant de 192.168.1.10
or	`ip.src == 192.168.1.10 or ip.src == 192.168.1.11`	Filtre les paquets venant de l'IP 192.168.1.10 **OU** 192.168.1.11	Tous les paquets provenant de l'une ou l'autre IP
not	`not arp`	Exclut tous les paquets ARP	Affiche tous les paquets sauf ARP
and + not	`tcp and not tcp.dstport == 22`	Sélectionne tous les paquets TCP **sauf** ceux destinés au port 22 (SSH)	Tout le trafic TCP sauf SSH
or + and + not	`(ip.src == 192.168.1.10 or ip.src == 192.168.1.11) and not tcp.port == 22`	Paquets venant de 192.168.1.10 **ou** 192.168.1.11 **mais pas** sur le port 22	Tous les paquets des deux IP sauf SSH

Lancez une capture de paquets dans Wireshark. Pendant la capture, exécutez la commande **ping** vers une machine du réseau local.

Une fois les paquets capturés, filtrez les trames correspondant à l'échange entre votre poste et la machine cible en saisissant dans la barre de filtre :

ip.dst == "adresse_IP_de_la_machine_cible"

> **Question 8**
>
> Quel est le protocole utilisé suite à l'exécution du « ping » ? A quelle couche appartient-il ?

Démarrez une nouvelle capture de paquets dans Wireshark. Pendant la capture, ouvrez votre navigateur web et accédez à l'adresse suivante :

http://w3.uqo.ca/iglewski/ens/inf1493/src/html2/html2_http.php

Une fois la page entièrement chargée, arrêtez la capture.

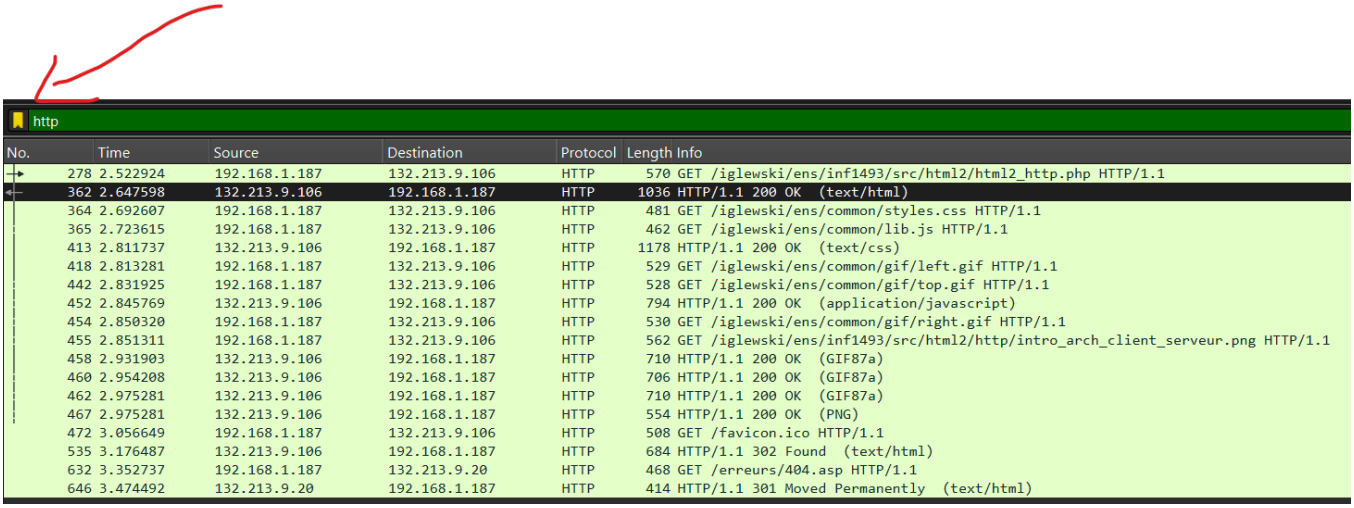
Appliquez un filtre d'affichage afin de ne conserver à l'écran que les paquets correspondant au protocole **HTTP**

Repérez et sélectionnez le premier message **HTTP** contenant la mention **"OK"** (réponse du serveur).

Dans la **zone 2** (détails du paquet), cliquez sur la ligne **"Line-based text data : text/html"**, puis faites défiler jusqu'en bas pour lire rapidement le contenu.

> **Question 9**
>
> Le protocole http est de quelle couche ?

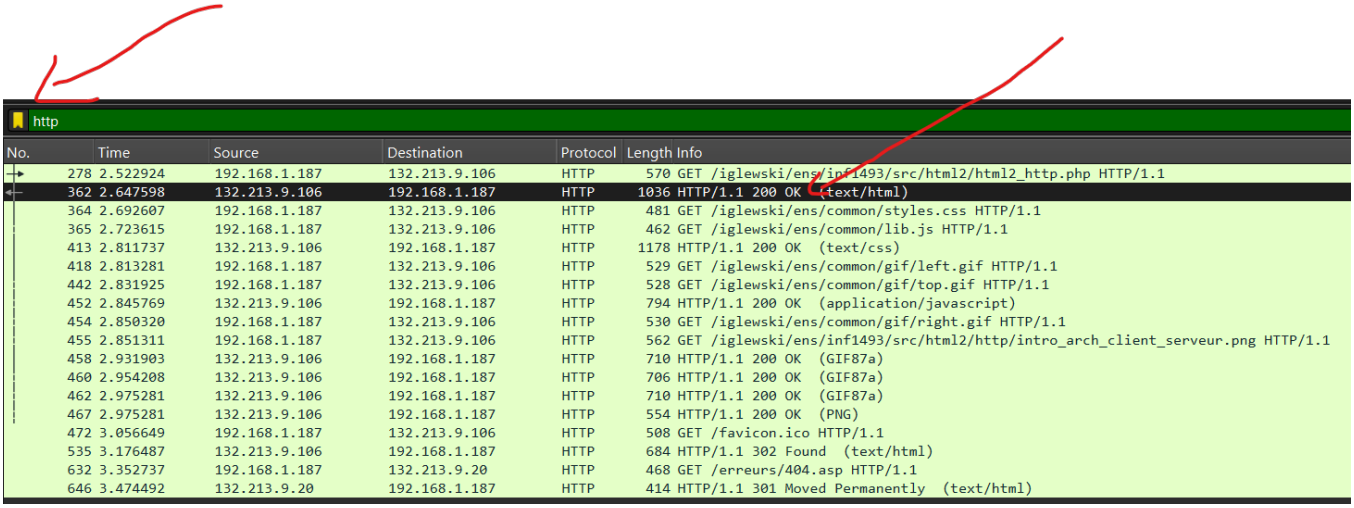
> **Question 10**
>
> Quel protocole de transport utilisé par http ?



http

No.	Time	Source	Destination	Protocol	Length	Info
278	2.522924	192.168.1.187	132.213.9.106	HTTP	570	GET /iglewski/ens/inf1493/src/html2/html2_http.php HTTP/1.1
362	2.647598	132.213.9.106	192.168.1.187	HTTP	1036	HTTP/1.1 200 OK (text/html)
364	2.692607	192.168.1.187	132.213.9.106	HTTP	481	GET /iglewski/ens/common/styles.css HTTP/1.1
365	2.723615	192.168.1.187	132.213.9.106	HTTP	462	GET /iglewski/ens/common/lib.js HTTP/1.1
413	2.811737	132.213.9.106	192.168.1.187	HTTP	1178	HTTP/1.1 200 OK (text/css)
418	2.813281	192.168.1.187	132.213.9.106	HTTP	529	GET /iglewski/ens/common/gif/left.gif HTTP/1.1
442	2.831925	192.168.1.187	132.213.9.106	HTTP	528	GET /iglewski/ens/common/gif/top.gif HTTP/1.1
452	2.845769	132.213.9.106	192.168.1.187	HTTP	794	HTTP/1.1 200 OK (application/javascript)
454	2.850320	192.168.1.187	132.213.9.106	HTTP	530	GET /iglewski/ens/common/gif/right.gif HTTP/1.1
455	2.851311	192.168.1.187	132.213.9.106	HTTP	562	GET /iglewski/ens/inf1493/src/html2/http/intro_arch_client_serveur.png HTTP/1.1
458	2.931903	132.213.9.106	192.168.1.187	HTTP	710	HTTP/1.1 200 OK (GIF87a)
460	2.954208	132.213.9.106	192.168.1.187	HTTP	706	HTTP/1.1 200 OK (GIF87a)
462	2.975281	132.213.9.106	192.168.1.187	HTTP	710	HTTP/1.1 200 OK (GIF87a)
467	2.975281	132.213.9.106	192.168.1.187	HTTP	554	HTTP/1.1 200 OK (PNG)
472	3.056649	192.168.1.187	132.213.9.106	HTTP	508	GET /favicon.ico HTTP/1.1
535	3.176487	132.213.9.106	192.168.1.187	HTTP	684	HTTP/1.1 302 Found (text/html)
632	3.352737	192.168.1.187	132.213.9.20	HTTP	468	GET /erreurs/404.asp HTTP/1.1
646	3.474492	132.213.9.20	192.168.1.187	HTTP	414	HTTP/1.1 301 Moved Permanently (text/html)

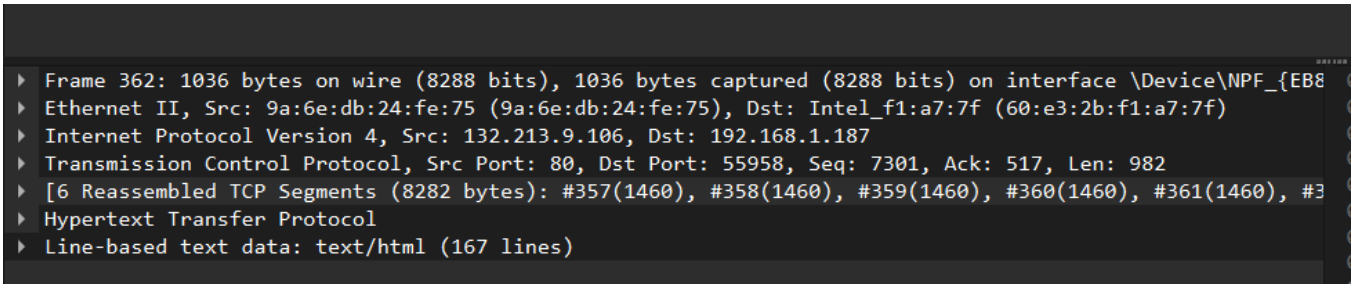
Figure 5 – Filtre d'affichage « http » appliqué à la capture.



http

No.	Time	Source	Destination	Protocol	Length	Info
278	2.522924	192.168.1.187	132.213.9.106	HTTP	570	GET /iglewski/ens/inf1493/src/html2/html2_http.php HTTP/1.1
362	2.647598	132.213.9.106	192.168.1.187	HTTP	1036	HTTP/1.1 200 OK (text/html)
364	2.692607	192.168.1.187	132.213.9.106	HTTP	481	GET /iglewski/ens/common/styles.css HTTP/1.1
365	2.723615	192.168.1.187	132.213.9.106	HTTP	462	GET /iglewski/ens/common/lib.js HTTP/1.1
413	2.811737	132.213.9.106	192.168.1.187	HTTP	1178	HTTP/1.1 200 OK (text/css)
418	2.813281	192.168.1.187	132.213.9.106	HTTP	529	GET /iglewski/ens/common/gif/left.gif HTTP/1.1
442	2.831925	192.168.1.187	132.213.9.106	HTTP	528	GET /iglewski/ens/common/gif/top.gif HTTP/1.1
452	2.845769	132.213.9.106	192.168.1.187	HTTP	794	HTTP/1.1 200 OK (application/javascript)
454	2.850320	192.168.1.187	132.213.9.106	HTTP	530	GET /iglewski/ens/common/gif/right.gif HTTP/1.1
455	2.851311	192.168.1.187	132.213.9.106	HTTP	562	GET /iglewski/ens/inf1493/src/html2/http/intro_arch_client_serveur.png HTTP/1.1
458	2.931903	132.213.9.106	192.168.1.187	HTTP	710	HTTP/1.1 200 OK (GIF87a)
460	2.954208	132.213.9.106	192.168.1.187	HTTP	706	HTTP/1.1 200 OK (GIF87a)
462	2.975281	132.213.9.106	192.168.1.187	HTTP	710	HTTP/1.1 200 OK (GIF87a)
467	2.975281	132.213.9.106	192.168.1.187	HTTP	554	HTTP/1.1 200 OK (PNG)
472	3.056649	192.168.1.187	132.213.9.106	HTTP	508	GET /favicon.ico HTTP/1.1
535	3.176487	132.213.9.106	192.168.1.187	HTTP	684	HTTP/1.1 302 Found (text/html)
632	3.352737	192.168.1.187	132.213.9.20	HTTP	468	GET /erreurs/404.asp HTTP/1.1
646	3.474492	132.213.9.20	192.168.1.187	HTTP	414	HTTP/1.1 301 Moved Permanently (text/html)

Figure 6 – Sélection du premier message HTTP contenant « OK ».



```
▸ Frame 362: 1036 bytes on wire (8288 bits), 1036 bytes captured (8288 bits) on interface \Device\NPF_{EB8
▸ Ethernet II, Src: 9a:6e:db:24:fe:75 (9a:6e:db:24:fe:75), Dst: Intel_f1:a7:7f (60:e3:2b:f1:a7:7f)
▸ Internet Protocol Version 4, Src: 132.213.9.106, Dst: 192.168.1.187
▸ Transmission Control Protocol, Src Port: 80, Dst Port: 55958, Seq: 7301, Ack: 517, Len: 982
▸ [6 Reassembled TCP Segments (8282 bytes): #357(1460), #358(1460), #359(1460), #360(1460), #361(1460), #3
▸ Hypertext Transfer Protocol
▸ Line-based text data: text/html (167 lines)
```

Figure 7 – Zone 2 : détail de la réponse HTTP.

Question 11

Combien de temps a-t-il fallu pour recevoir la réponse à la requête HTTP envoyée par votre PC ?

Question 12

Quelle-est l'adresse Internet du serveur du « http://w3.uqo.ca/iglewski/ens/inf1493/src/html2/html2_http.php » ?

Enregistrez la dernière capture effectuée.

HTTP VS HTTPS

Pour voir si une transmission est chiffrée ou non

- un logiciel de capture de trames : ***Wireshark***.
- et plutôt ***Firefox*** (car Chrome refuse parfois les connections non sécurisées).

Voici 2 pages de connexion, l'une est sécurisée (https) et l'autre ne l'est pas (http) :

Démarrer une capture sur wireshark

Allez sur le site suivant : http://testphp.vulnweb.com/login.php :

- entrer n'importe quel identifiant/mot de passe
- puis filtrer les trames Wireshark avec `ip.dst == 44.228.249.3`

Puis revenez sur wireshark pour arrêter la capture

Tapez la commande "ip.dst == 44.228.249.3" pour filtrer que les échanges de données entre votre machine le serveur qui héberge le site web auquel vous venez de vous connecter.

Puis cliquez la ligne dont la colonne info contient : "754 POST /userinfo.php HTTP/1.1 (application/x-www-form-urlencoded)"

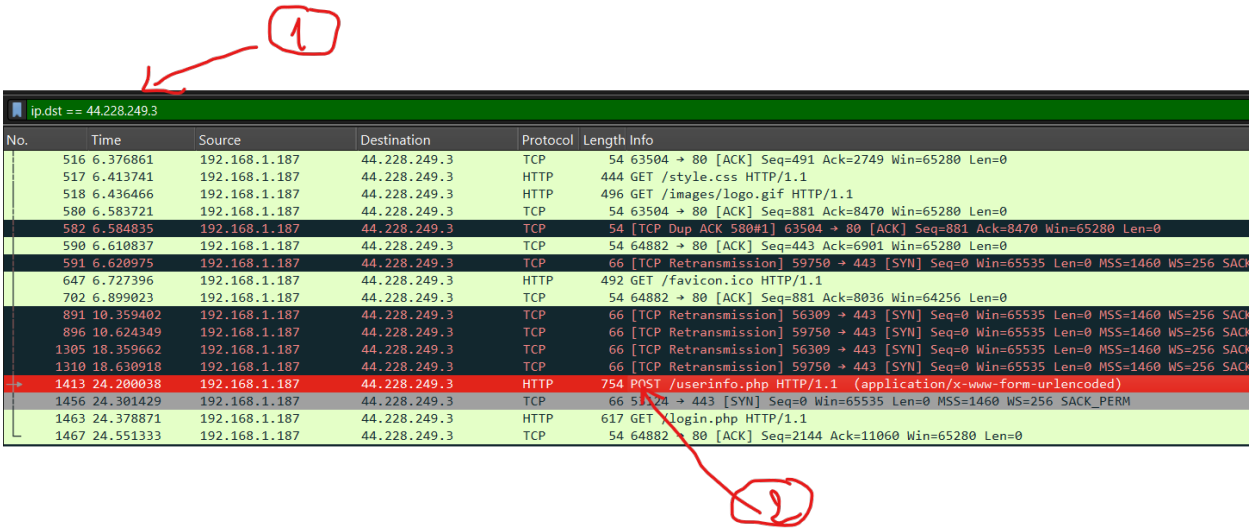


Figure 8 – Filtre sur l'adresse du serveur et ligne POST à sélectionner.

Dans la zone 3 cliquez sur la ligne déroulante "**HTML Form URL Encoded : application/x-www-form-urlencoded**"

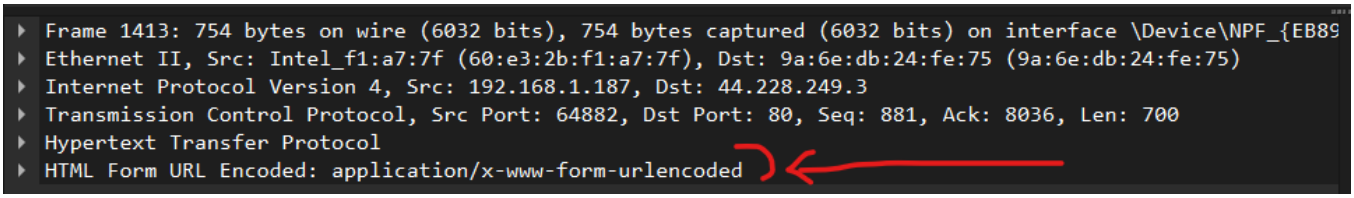


Figure 9 – Ligne « HTML Form URL Encoded » du détail de la trame.

Question 13

Que remarquez-vous ?

Question 14

Que représente `44.228.249`

Refaites la même chose avec le site suivant :

- https://nsirennes.fr/wp-login.php :
  - Lancer une « Connexion » avec un login/password quelconque.
  - puis filtrer les trames Wireshark avec `ip.dst == 164.132.47.68`