Bismuth Samuel : 342533064
Thuillier Johann : 336104120

- Assignment 3 -

HTTP+DNS (Wireshark)

Question 1 -

HTTP belongs to the layer application the seven layer of the OSI model. It's in the layer application that we find most of the network program. Usually, applications use udp or tcp, and are associated with a famous port. *For HTTP, it's the port TCP 80*.

Question 2 -

GET /form.html HTTP/1.1
Accept: */*
Accept-Language: en-us
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath. 3; .NET4.0C; .NET4.0E)
Accept-Encoding: gzip, deflate
Host: 192.168.2.103
Connection: Keep-Alive

This text represent a GET message, differently said, a request-URI.

Explanation of the previous message:

GET is the method used to send the request. This method is cacheable and safe.

/form.html is the path.
Note that the path cannot be empty, if none is present in the original URI, it must be given as "/".

HTTP/1.1, is the latest version of Hypertext Transfer Protocol.

The accept request HTTP header advertises which content types (MIME types : **M**ultipurpose **I**nternet **M**ail **E**xtension), the client is able to understand. Here "*/*" means any MIME type.

The Accept-Language request HTTP header advertises which languages the client is able to understand, and which locale variant is preferred. Here english and us (United states) are mentioned.

The User-Agent request header contains a characteristic string in our case : "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729; .NET CLR 3.0.30729; Media Center PC 6.0; InfoPath.3; .NET4.0C; .NET4.0E)" that allows the network protocol peers to identify the application type, operating system, software vendor or software version of the requesting software user agent.

The Accept-Encoding request HTTP header advertises which content encoding, usually a compression algorithm, the client is able to understand. Here gzip and deflate are allowed.

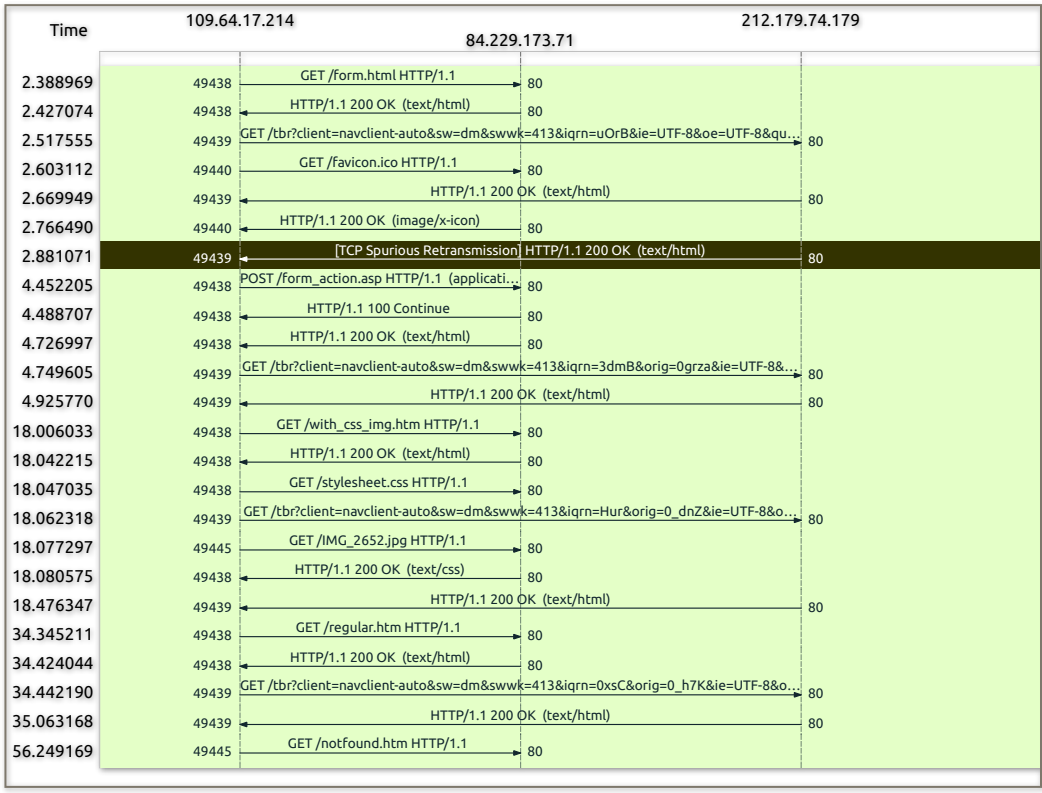
The Host request header specifies the domain name of the serve. Here it's 192.168.2.103.

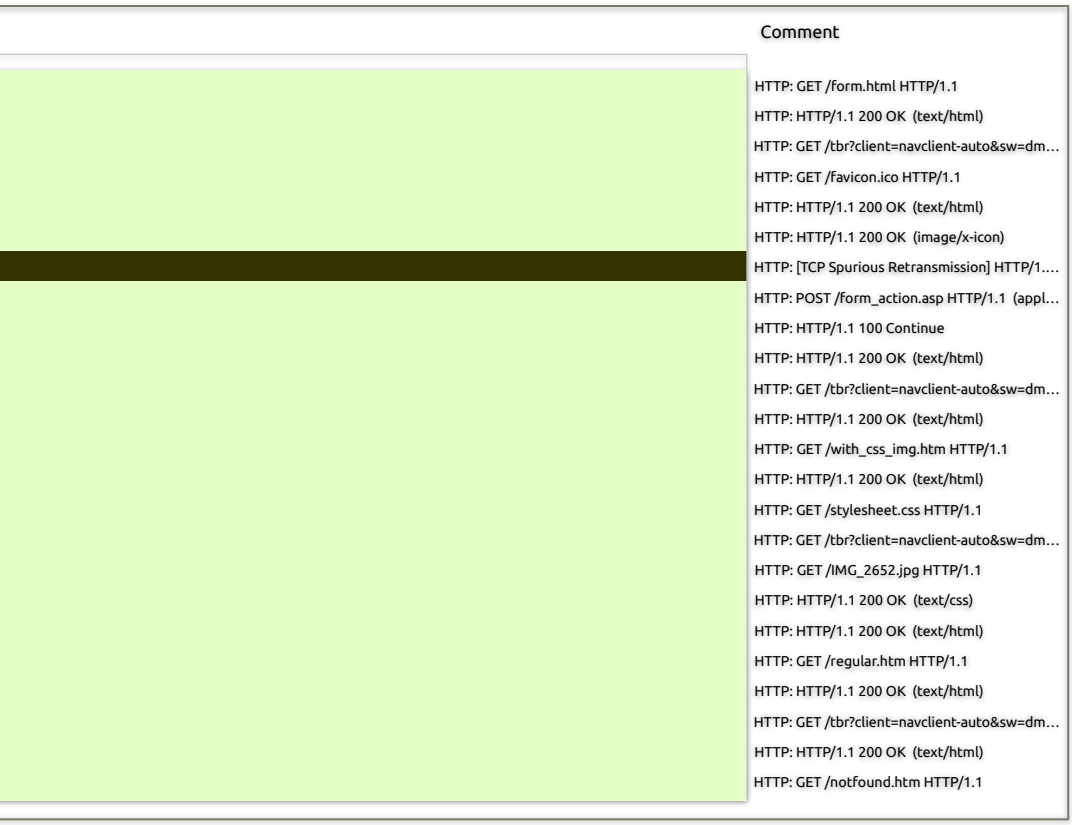
The Connection request header controls whether or not the network connection stays open after the current transaction finishes. Here the value sent is keep-alive, ie, the connection is persistent and not closed, allowing for subsequent requests to the same server to be done.

.

Question 3 -

1/2





2/2

Screen-shot of sequence diagram from Wireshark

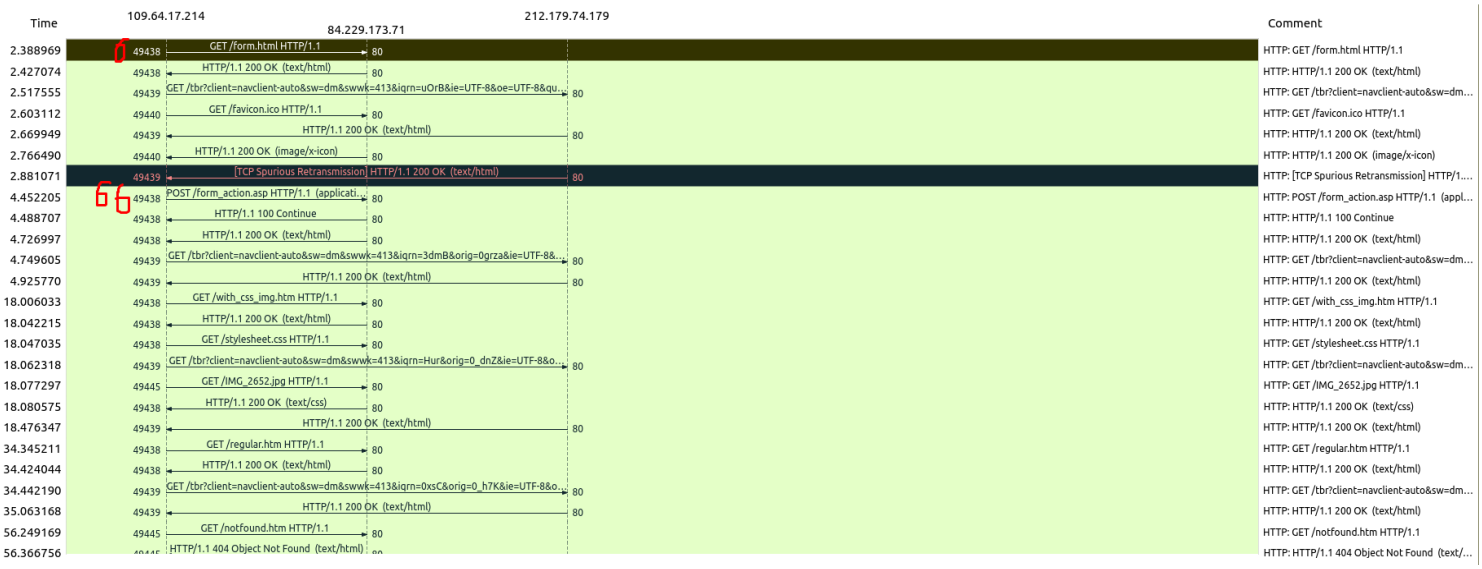
.

Question 4 -

From the trace, we can notice two types of requests : GET and POST.

The GET method requests a representation of the specified resource. Requests using GET should only retrieve data.

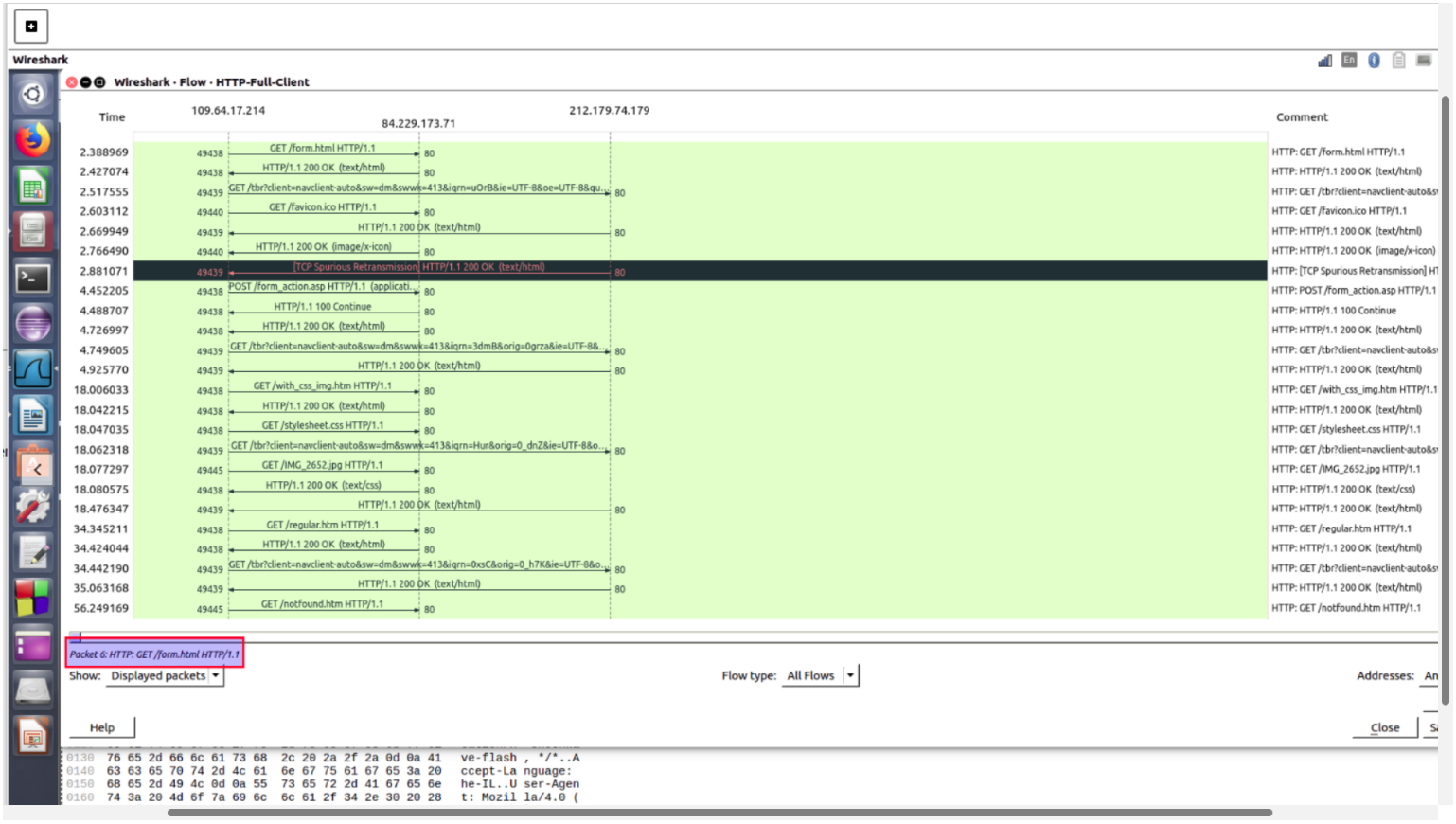
The POST method is used to submit an entity to the specified resource, often causing a change in state or side effects on the server



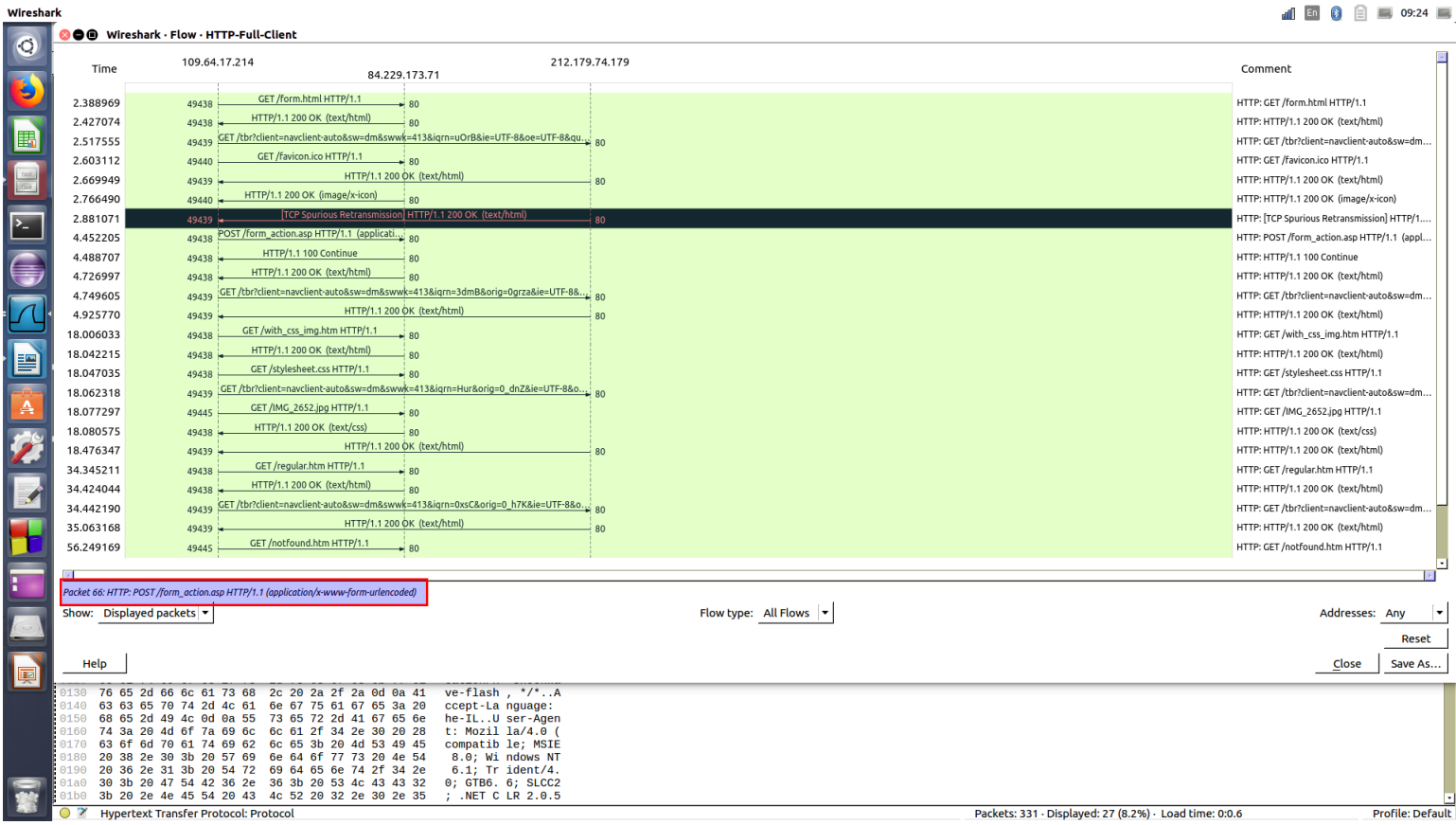
Screen-shot from Wireshark who display the packet number from one of the gate message and the post message.

As we can see in the following pictures, we raise that the first get request get a packet with the number 6. The only post request of the trace get a packet with the number 66. This number provide from the legend, see like on the pictures :

.



Screen-shot from Wireshark. The rectangle focus on the packet number.



Screen-shot from Wireshark. The rectangle focus on the packet number.

.

Here are all the HTTP request types except of GET and POST , HEAD, PUT, DELETE, CONNECT, OPTIONS, TRACE, PATCH.

More explanations to follow :

The HEAD method asks for a response identical to that of a GET request, but without the response body.
The PUT method replaces all current representations of the target resource with the request payload.

The DELETE method deletes the specified resource.

The CONNECT method establishes a tunnel to the server identified by the target resource.

The OPTIONS method is used to describe the communication options for the target resource.
The TRACE method performs a message loop-back test along the path to the target resource.

The PATCH method is used to apply partial modifications to a resource.

Question 5 -

There are a lot of differences between the two method GET and POST.

The fundamental difference is that they correspond to different HTTP requests.

The submission process for both methods begins in the same way a form data set is constructed by the browser and then encoded in a manner specified by the enctype attribute. Here are some of the many differences :

The GET request can be cached whereas the POST is never cached.

The GET request remain in the browser history whereas the POST don’t.

The GET request can be bookmarked whereas the POST can’t.

.

The GET request have length restriction whereas the POST have no restriction on data length.

Question 6 -

GET is basically for just getting data whereas "POST" may involve anything, like storing or updating data, or ordering a product, or sending E-mail.

GET is recommended when submitting "idempotent" forms. In other words, forms that involve database queries only. Also recommended if the interaction is more like a question, such as query, read operation or lookup.

Another perspective is that several idempotent queries will have the same effect as a single query. If database updates or other actions such as triggering emails are involved, the usage of POST is recommended. In others words, if the interaction is more like an order, or the interaction changes the state of the ressource in a way that the user would perceive. Or if the user be held accountable for the result of the interaction.

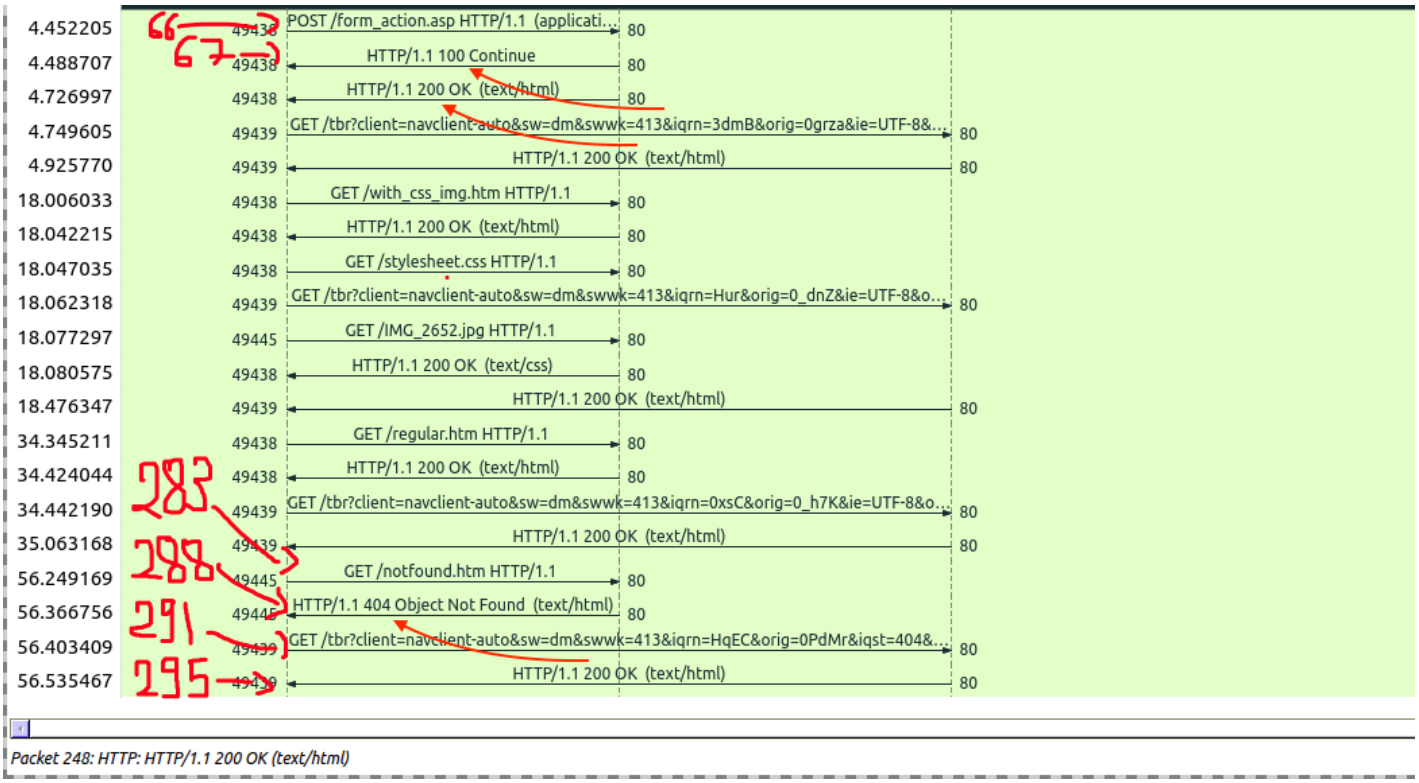
Question 7 -

From the trace we notice three types of answers : 200 ok, 100 continue, 404 Object Not Found.

200 ok means the request has succeeded.

100 continue means the client should continue with its request.

404 Not found means that the server has not found anything matching the Request-URI.



Screen-shot from wireshark. The numbers determines the packets numbers.

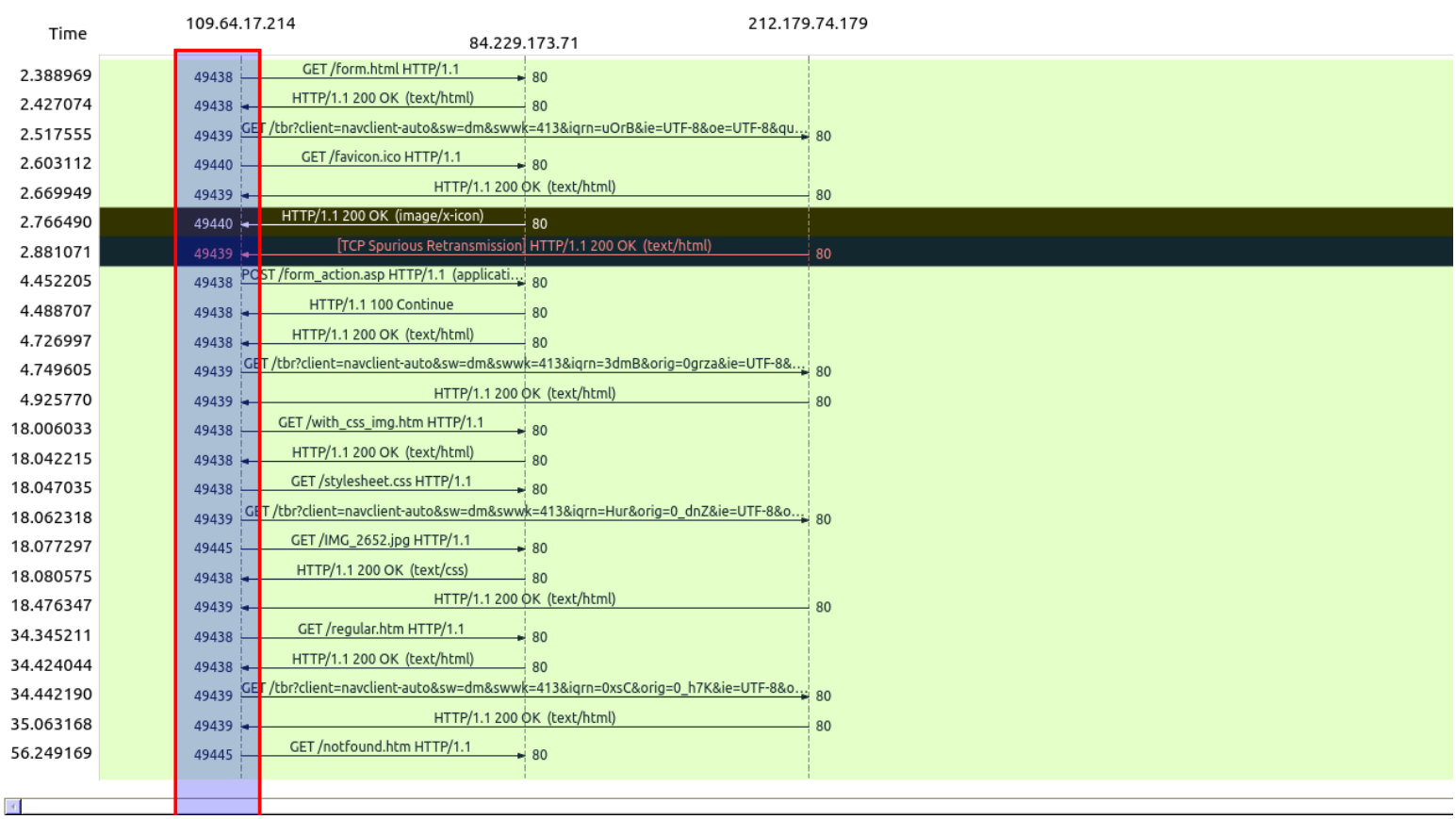
As we can see in the previous picture, the POST request has a packet number equal to 66, the response of this request is 100 continue, with a packet number 67.

The GET/Notfound.html request has a packet number equal to 283, the response of this request is 404 Not found, with a packet number 67.

The GET request (the last of the trace) has a packet number equal to 291, the response of this request is 200 ok, with a packet number 295.

Note that all the packets numbers have been found like in the question 4.

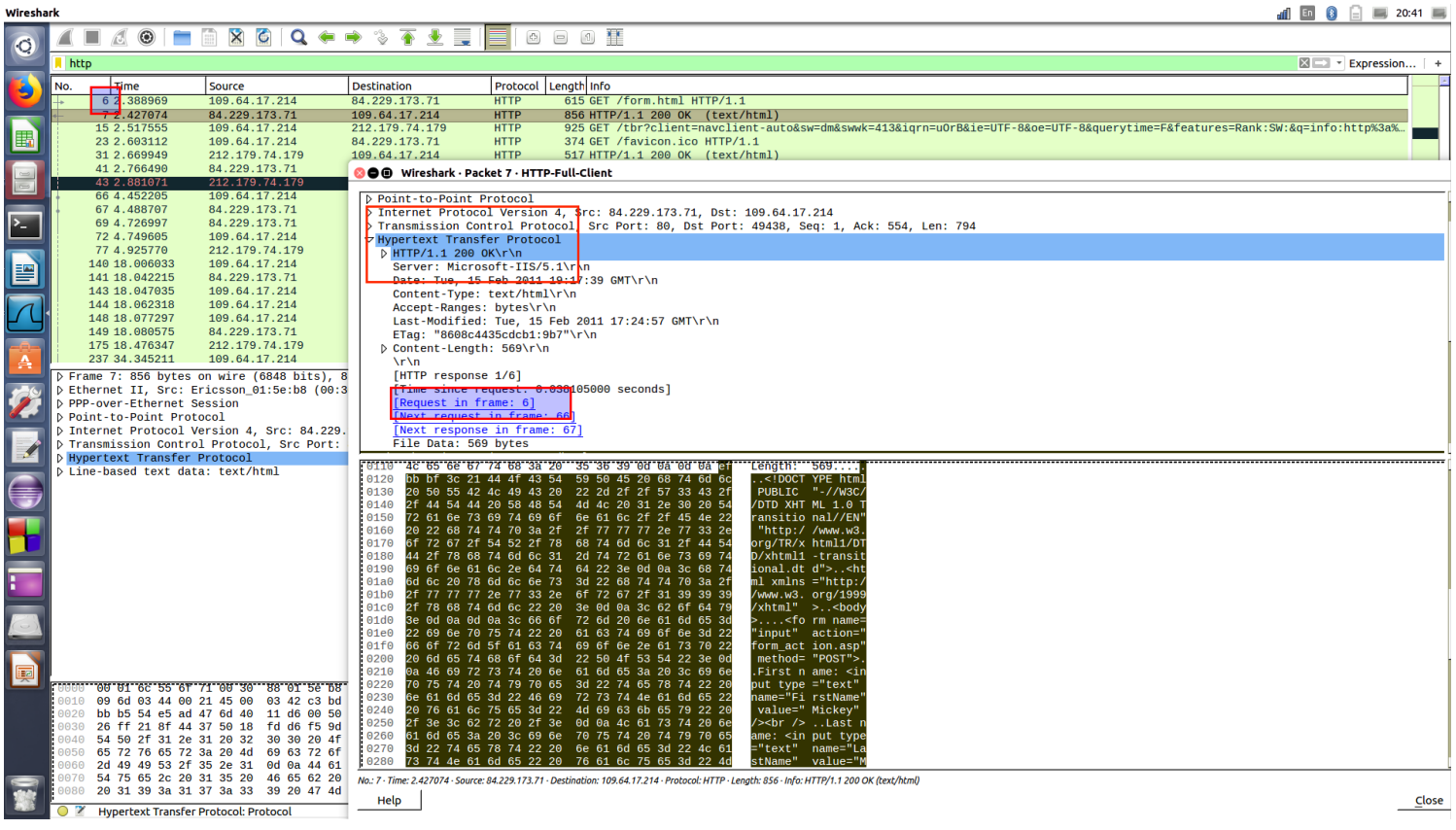
To know which request precede the answer, we look at the number of source port, which is the second number from the left side.



Screen-shot from wireshark. The rectangle focus on the DST port.

If the source port of a request is the same that the source port of an answer, so they are connected. And of course, we're looking for the clothier time.

Another way to check this is by looking the information like in the picture :



Screen-shot from wireshark.

We denote 67 different answers (without care about the unofficial code). All the answers are determine by a number, (for example here : 200 ok.)Which belongs to 6 types of answer, the 100's, 200's, 300's, 400's, 500's, and the unofficial answer.

Question 8 -

There are six types of answer :

- 1xx specify a provisional response.

- 2xx indicate that the server has received client's request, understands said request, accepts it and has processed it successfully.

- 3xx indicate that the client is required to take additional steps to complete a request.

- 4xx are meant for situations where the client has erred.

- 5xx are responses to requests that servers fail to fulfill.

- Unofficial code are still utilized by third-party services to provide restful or semantic error responses.

Question 9 -

The HTTP belong the the TCP/IP , differently said, the Internet protocol suite.

In the TCP/IP, we denote a lot of layers.

Every layers resolve some problems about the data transmission and give to the other layers some informations.

.

The highest layers are clothier from the users and handle more abstract data by using the informations from the lowest layers. The lowest layers must configure the data in the goal that the data may bu usefull by the highest layers.

So we are now able to understand why does HTTP belong to another protocol. Like we explained, HTTP belong to a layer : the application layer. And the HTTP protocol needs other layers to work fine, and the other protocol needs the HTTP protocol to work fine.

What the TCP/IP add to the HTTP protocol ?

In the TCP/IP we denote for layers : application (which contains HTTP), transport, Internet, link.

The most important things for the application layer is the choice of the protocol into the transport layers. The HTTP protocol use the TCP transport.

Question 10 -

Screen-shot from Wireshark of the beginning of the discussion.

The picture shows only the beginning of the discussion, because we though that it wasn't appropriate to provide screen-shot of the entire conversation.

Here are the details :

- First, the client send a request message (GET message) to form.html
- Then, the server answers with sent it back. We can recognize the http header with the message "200 ok", and then the doctype html.
- The client ask another time, with the help of a POST message, the server form_action.asp. There are a lot of information details into the post message, like the languages and the accepted format…
- Then, the answer of the server is "100 continue", follow by all the http header, which means that the request has been received and that it will be consider.
- The client so send the rest, with a get request, to get the the css file and the picture.
- Then the server send it back with a 200 ok as header.
- To continue, the client send another request for the server regular.html.
- The server sent it back : the header with all the html doctype needed.
- Here the discussion over.

We can remark in the Wireshark packet with the filter http that, after a new request from the client, the server couldn't submit the answer. It's maybe why here is the end of the discussion.

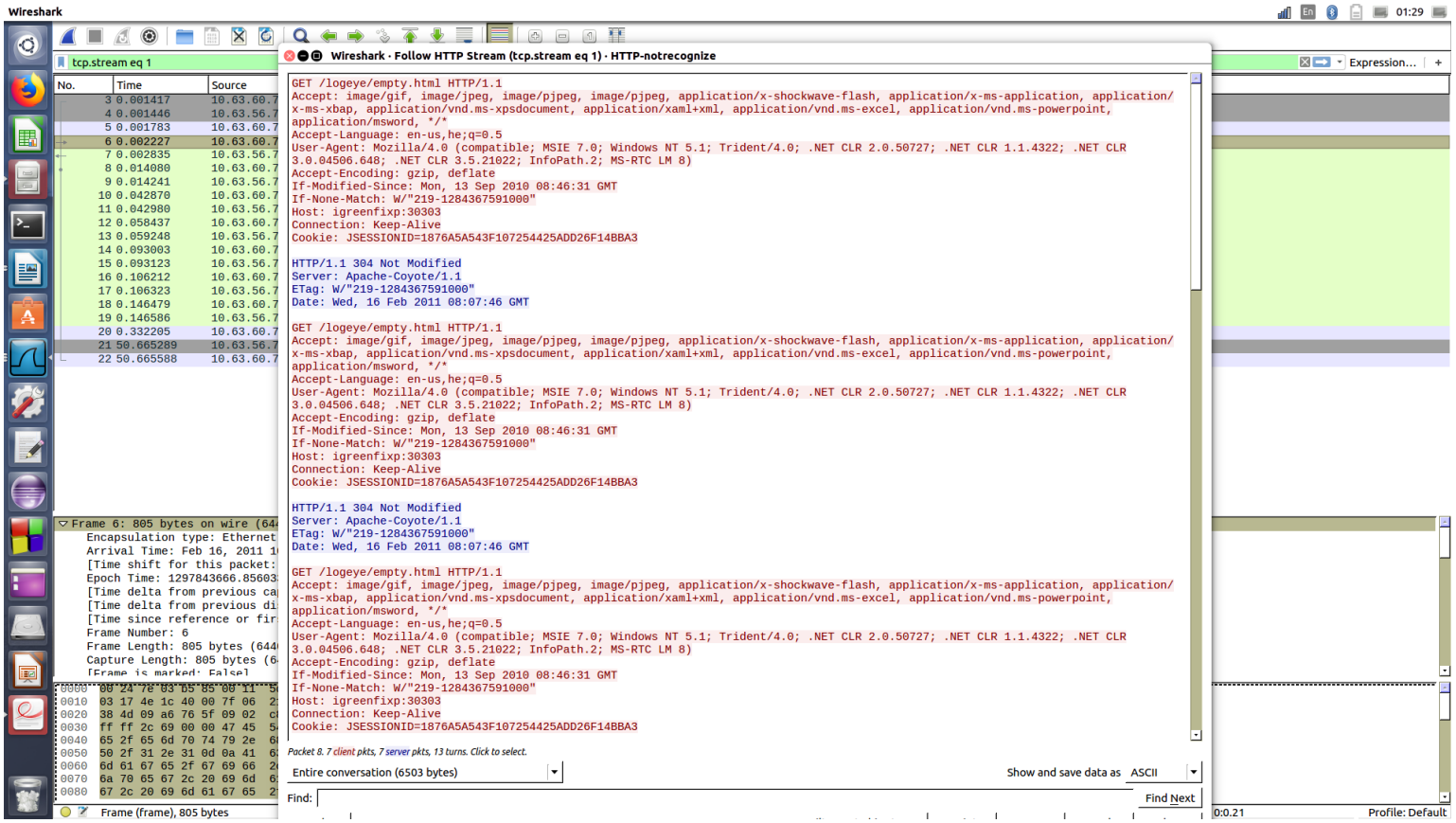
.

Question 11 -

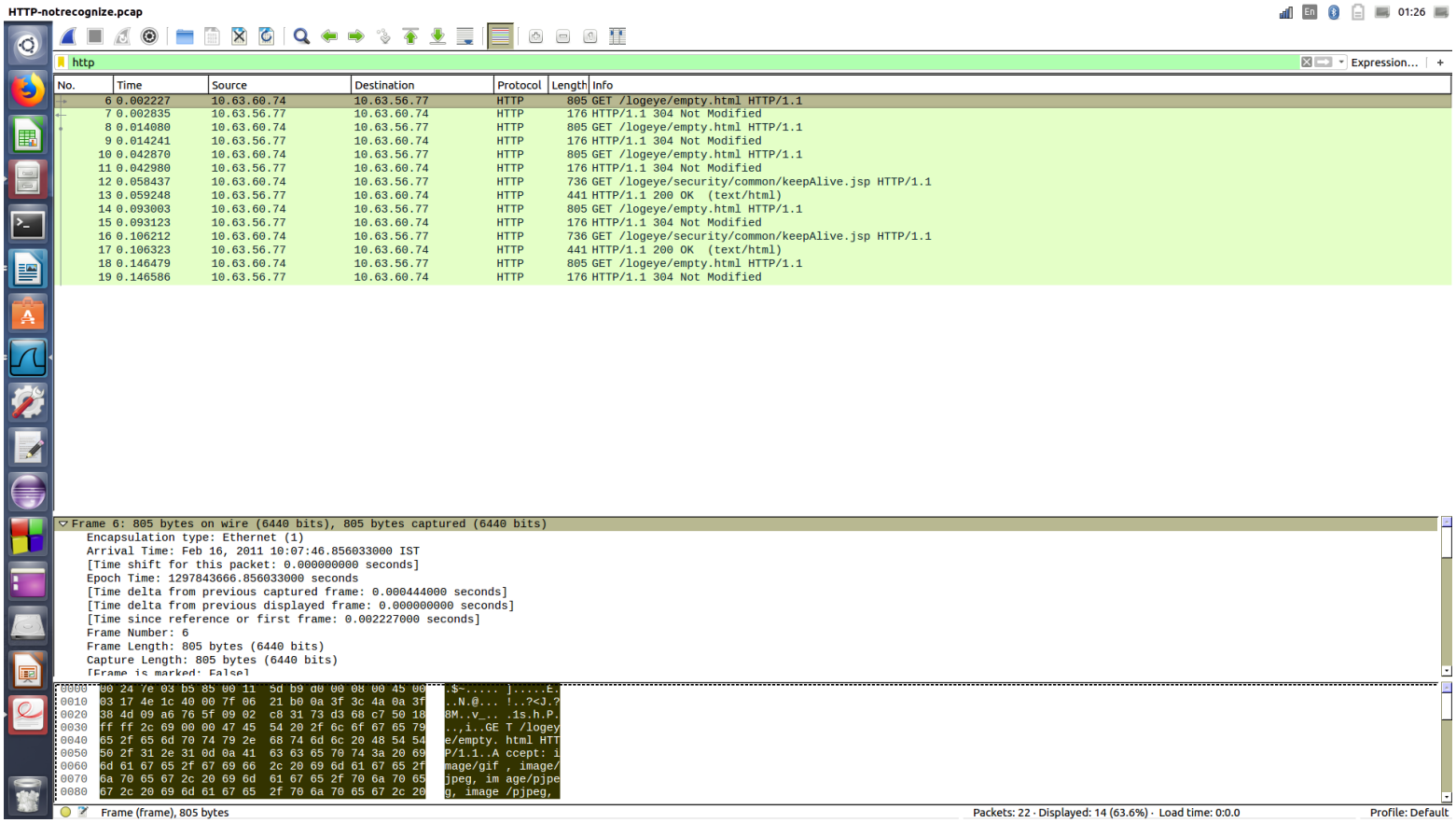
First of all, the request was a POST method, then, it's mean that it's susceptible to have side effect.

Then it does : The client has been asked to send another request because the server send the page back, and a http header with options for more data.

Question 12 -



Screen-shot from wireshark of the pcap file "notrecognizeit", to display the discussion.



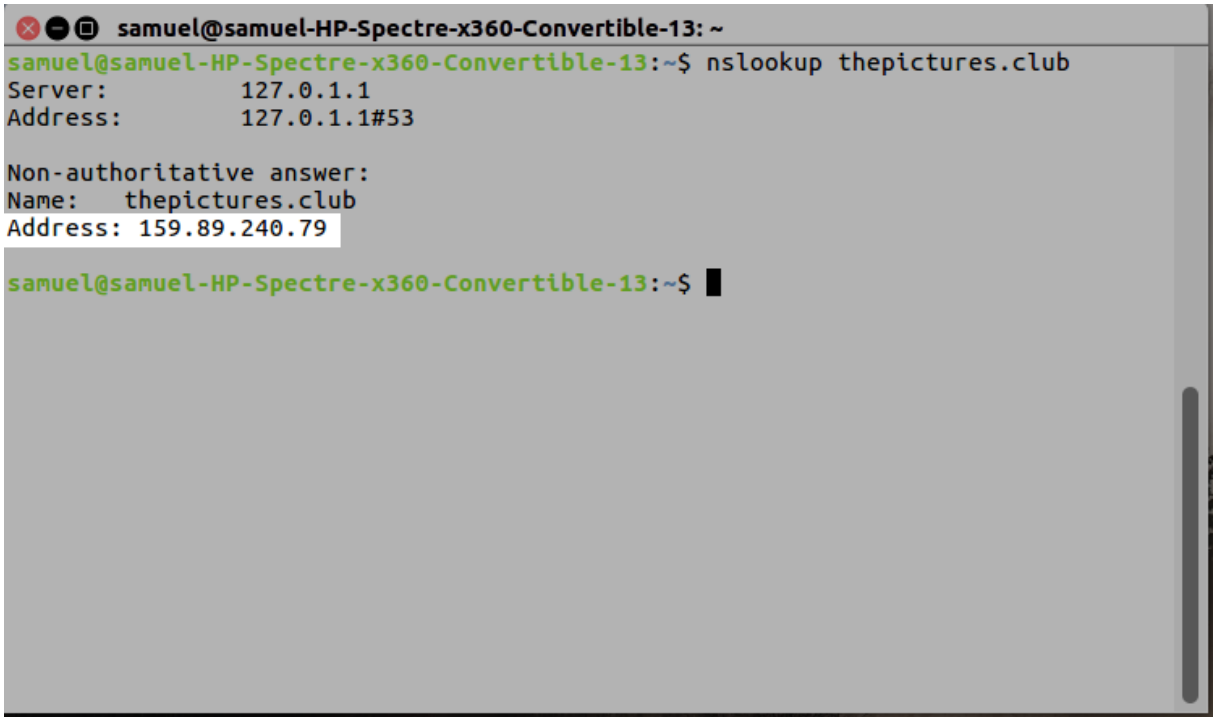
Screen-shot from wireshark of the pcap file “notrecognizeit”

It’s seem that Wireshark recognize the discussion.

.

Question 13 -

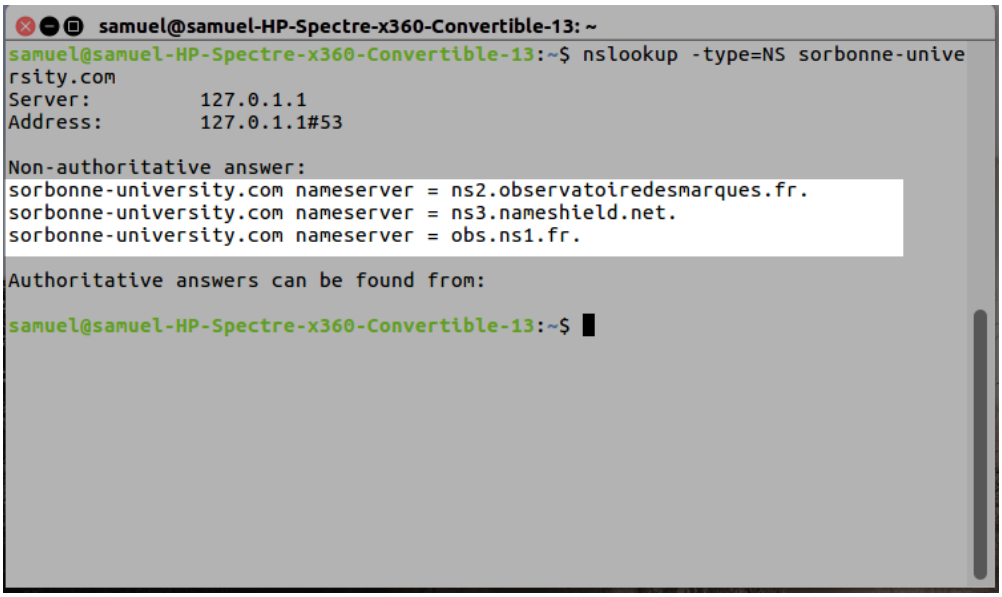
13.1 -



Screen-shot of the terminal who display the address of an Asiatic server using *nslookup*.

As we can read in the picture above, the ip address of the server is : “159.89.240.79”.

13.2 -



Screen-shot who display the authoritative server of “la sorbonne” university using nslookup.

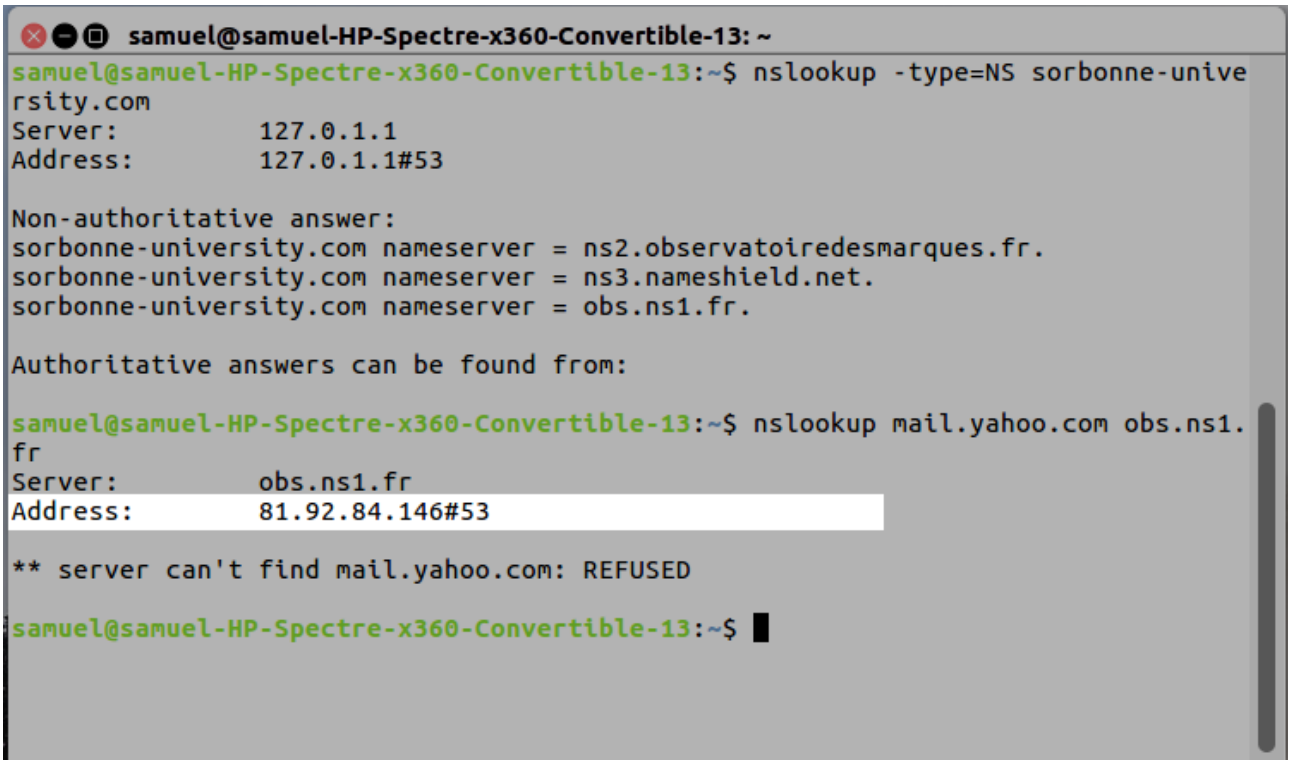
.

The name of the authoritative DNS server for La Sorbonne university are :

- ns2.observatoiredesmarques.fr
- ns3.nameshield.net
- obs.ns1.fr

as we can see in the picture above.

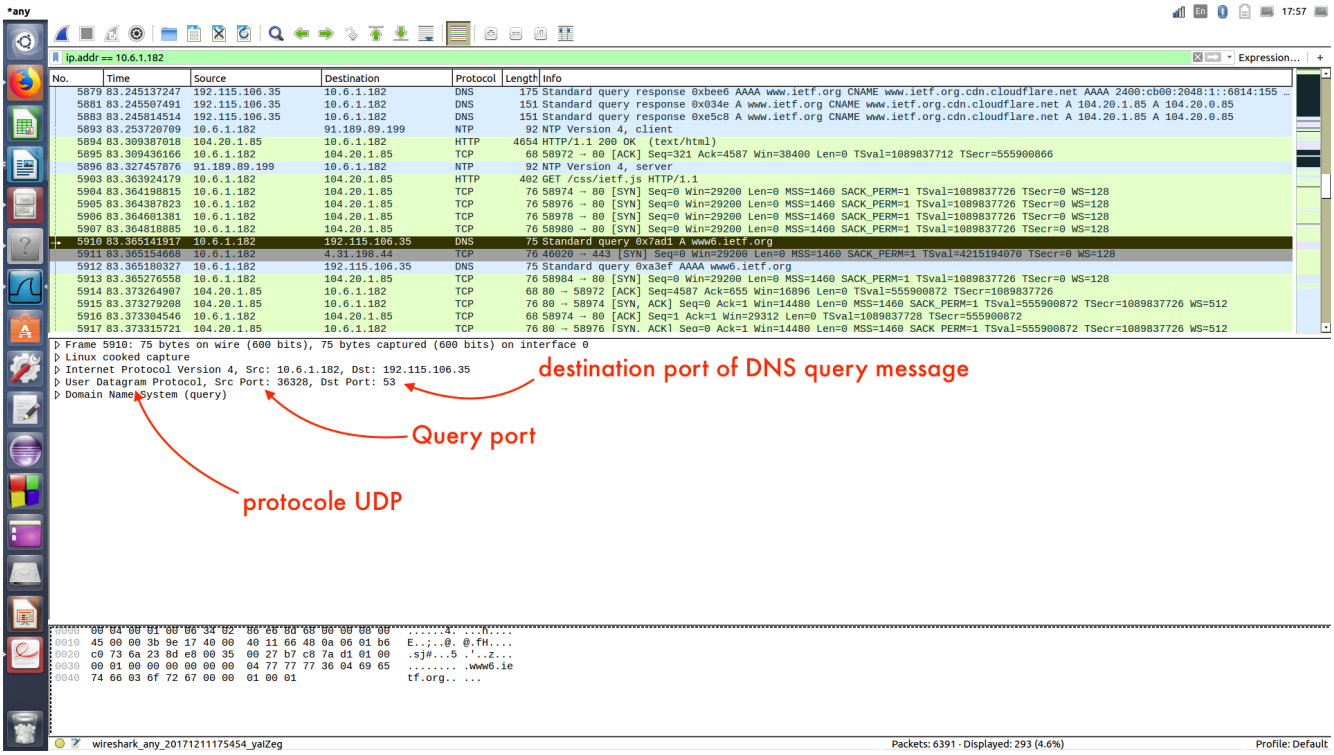
13.3 -

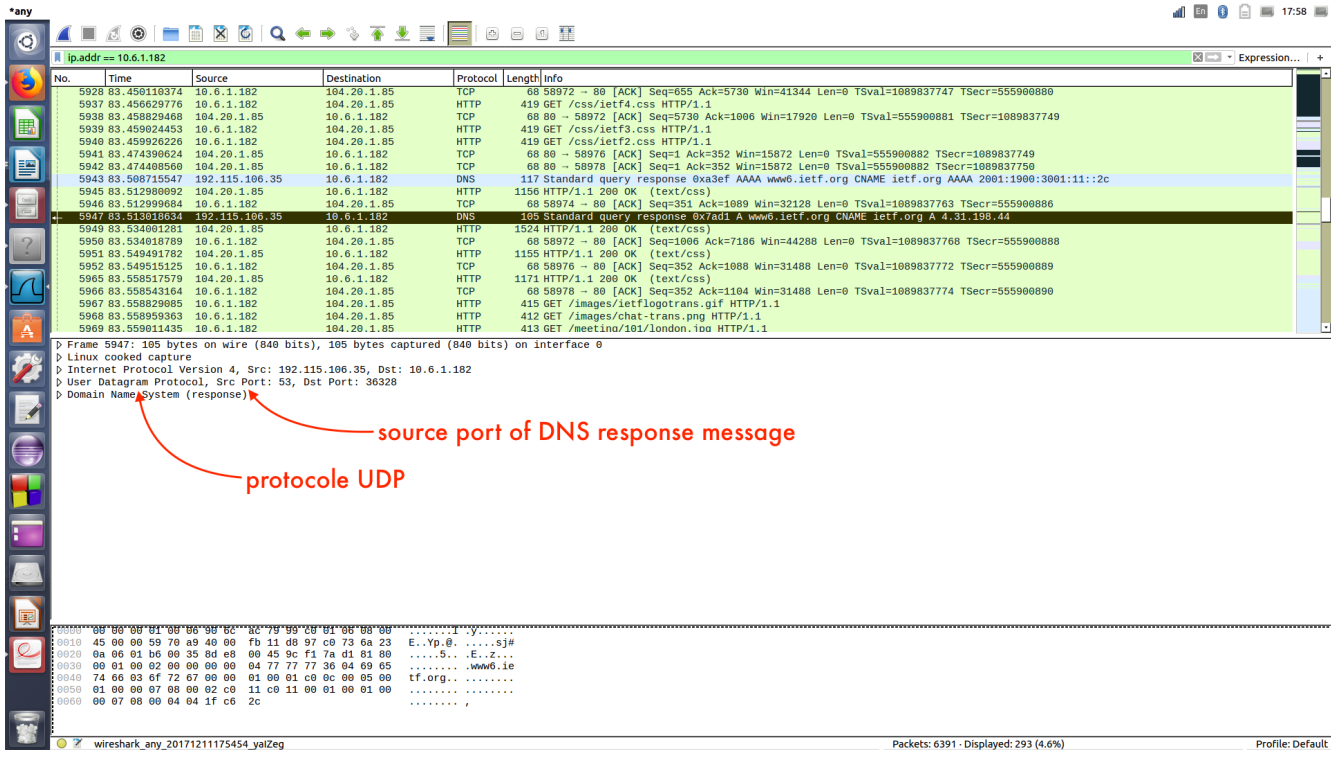


Screen-shot who display the authoritative server of “la sorbonne” university using nslookup.

The ip address is : 81.92.84.146.

13.4 -





Screen-shots who from wireshark (up Query, down Response)

As we can see in the pictures above, the DNS query and response messages are sent over UDP (User Datagram Protocol).

.

13.5 -

By looking the same screen-shot from the question 4, we notice that :

source port of the DNS query message is 36328,

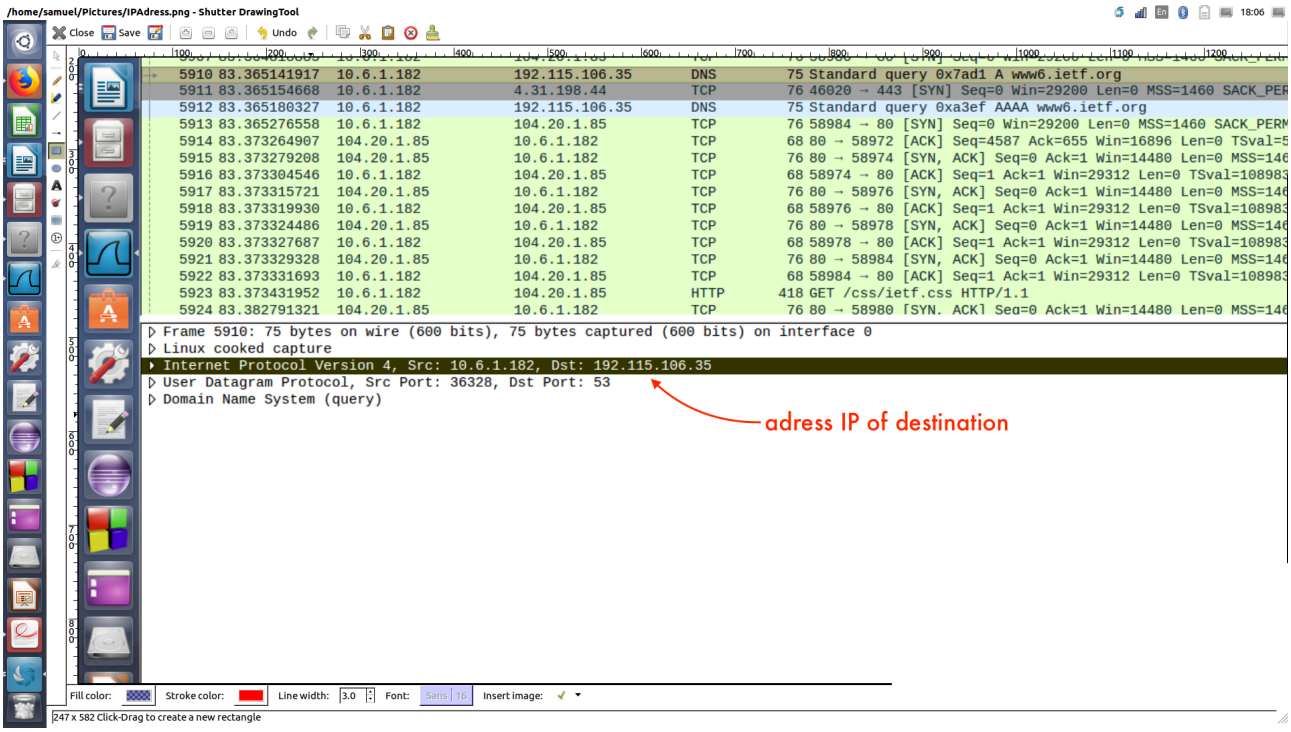
source port of the DNS query response is 53,

and reciprocally,

destination port of the DNS query message is 53,

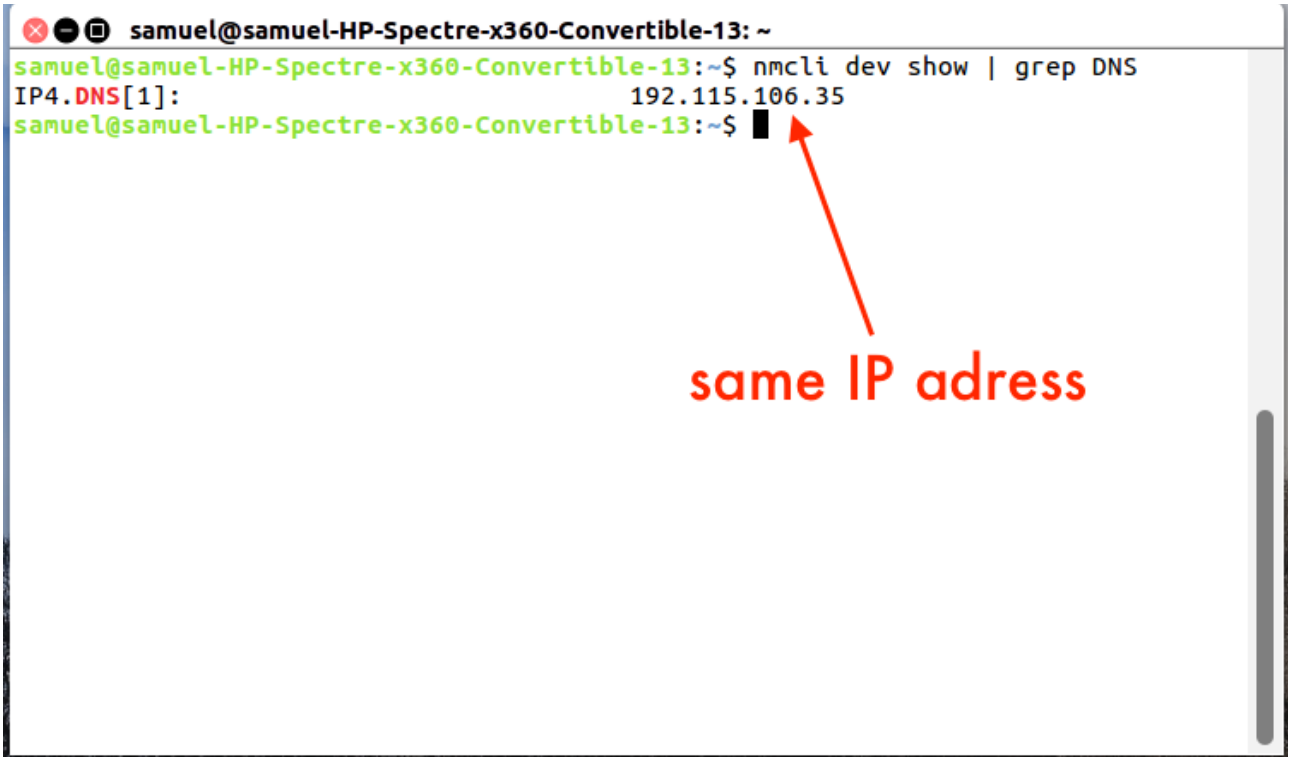
destination port of the DNS query response is 36328,

13.6 -



Screen-shot who from wireshark.

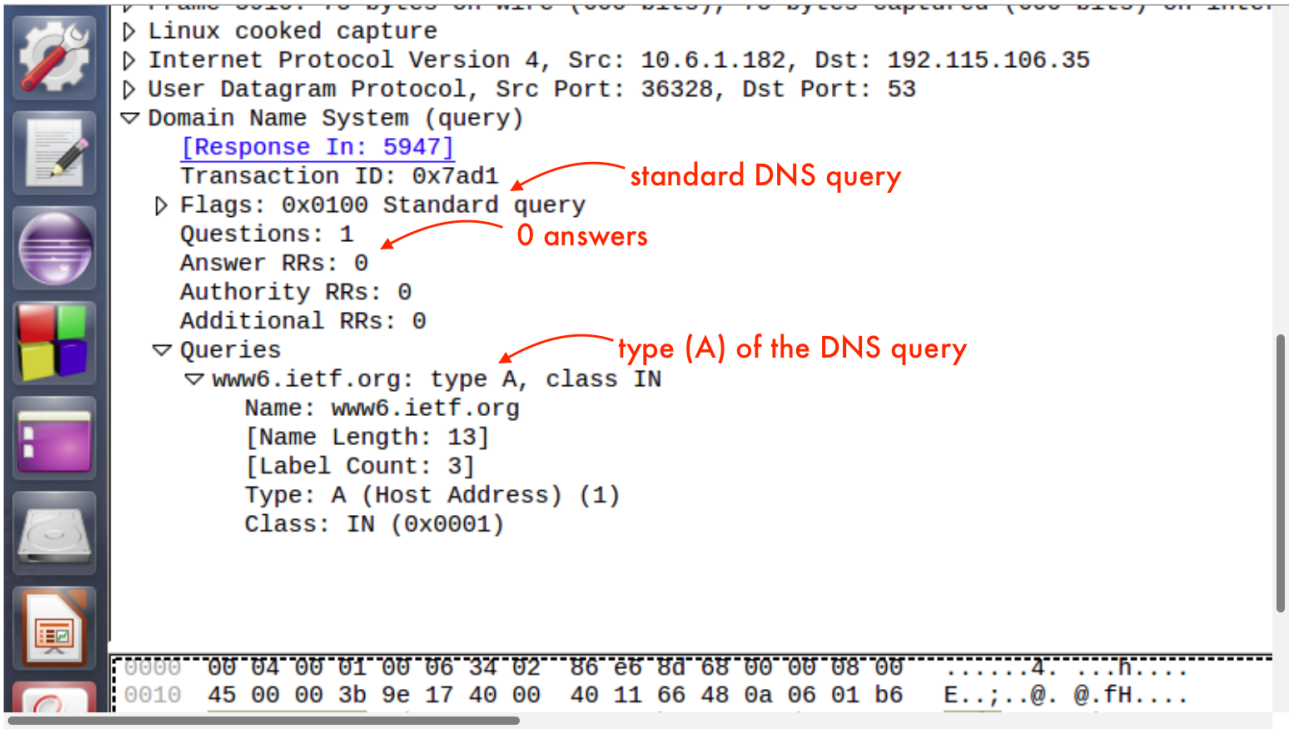
The DNS query message send to the IP address 192.115.106.35, as we can see above.



Screen-shot from the terminal and the IP address of your local DNS server.

The two address are the same : 192.115.106.35.

13.7 -

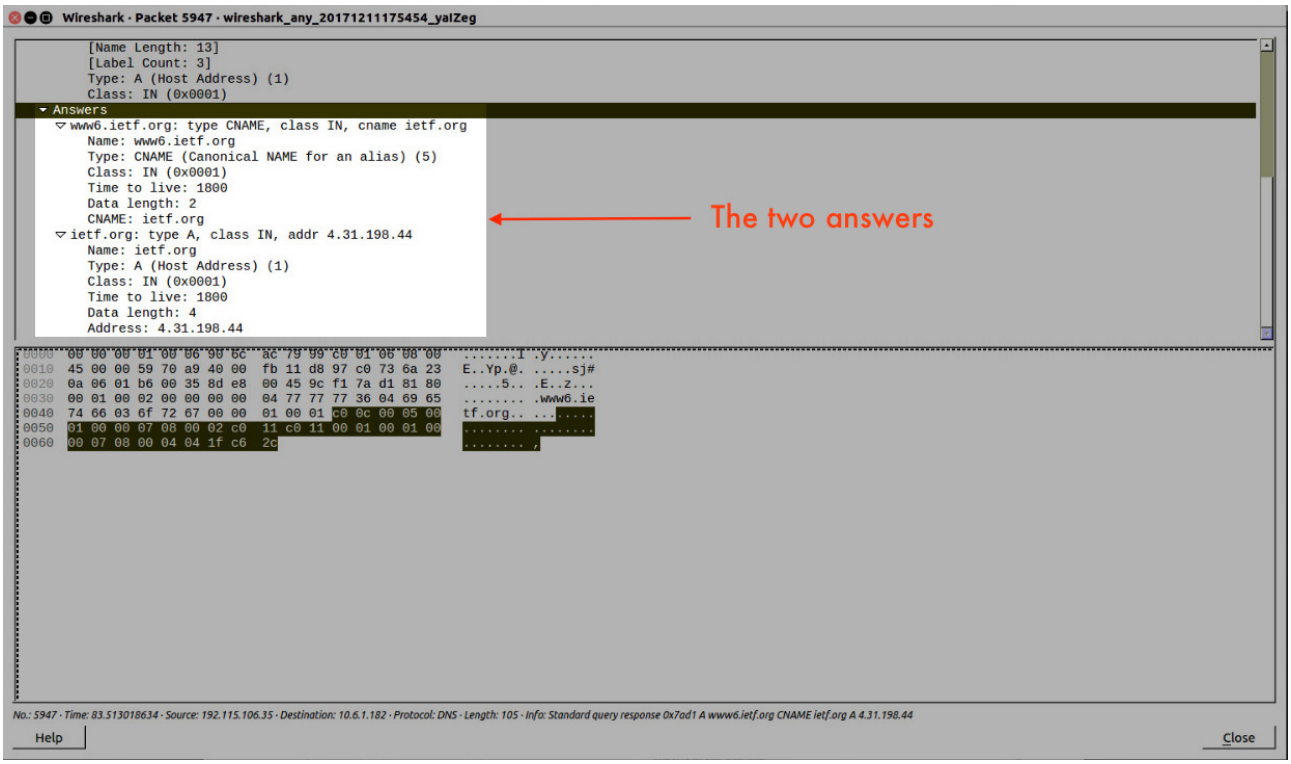


Screen-shot who from Wireshark who display the type of DNS query.

The type of DNS query is type A standard query.

The message query doesn’t contains any answer.

13.8 -



Screen-shot who from wireshark who display the answers.

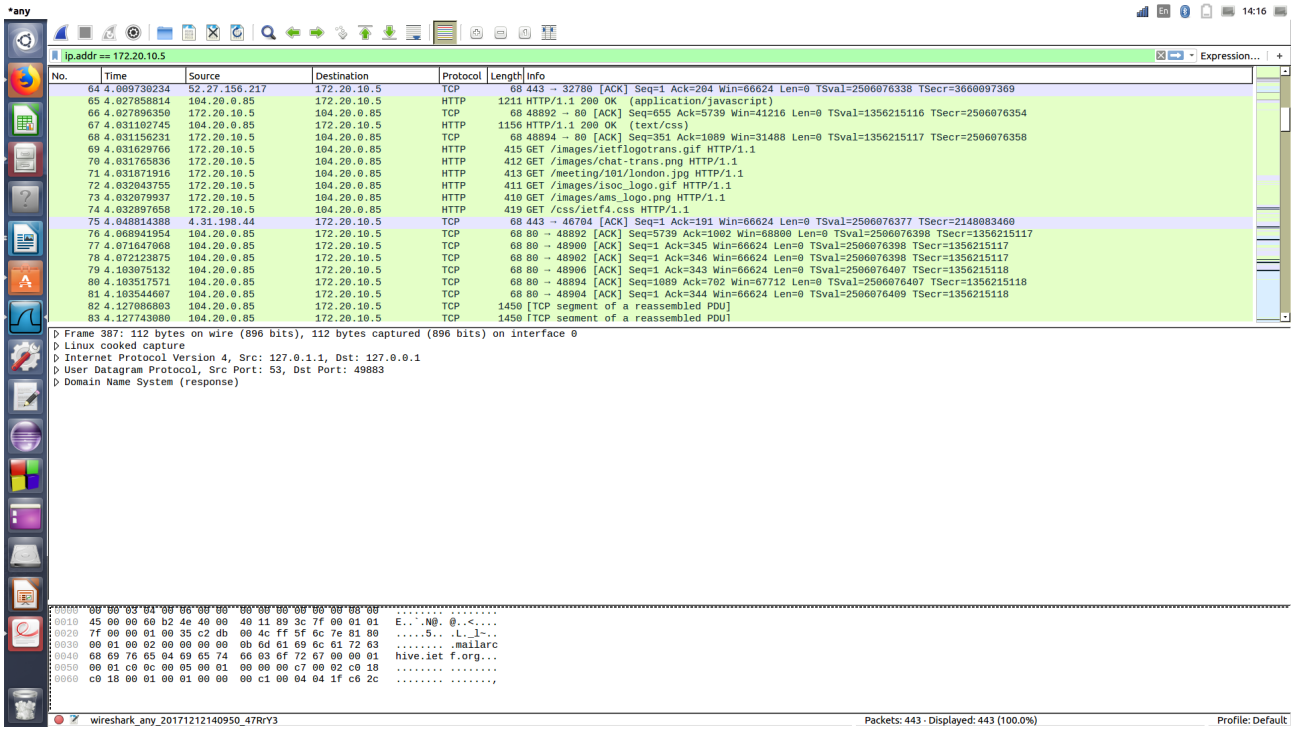
We notice two answers which contains information about the host, the type of the address, the class, the time to live, the data length, the name, and the IP address (4.31.198.44).

13.9 -

Yes it does.

The destination of the SYN packet is 4.31.198.44, is the same address that was provided in the DNS response message as the type “A” address of the webpage.

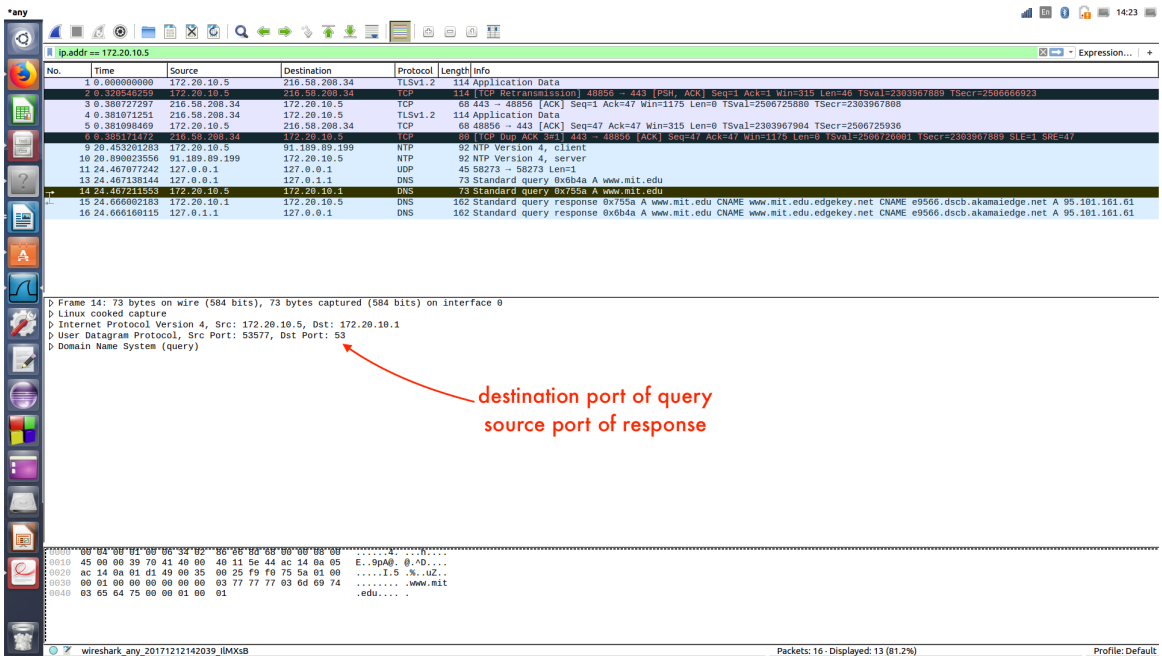
13.10 -



Screen-shot who from wireshark.

As we can see, there is no new query before retrieving image.

13.11 -



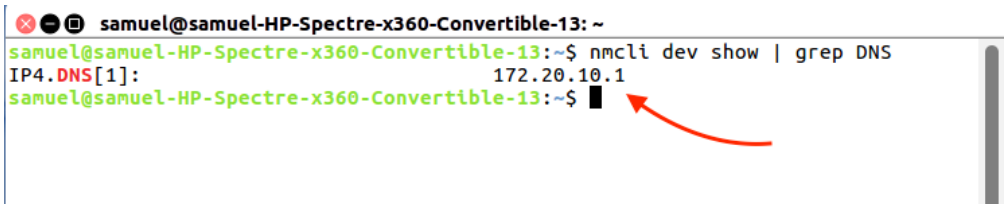
Screen-shot who from Wireshark.

.

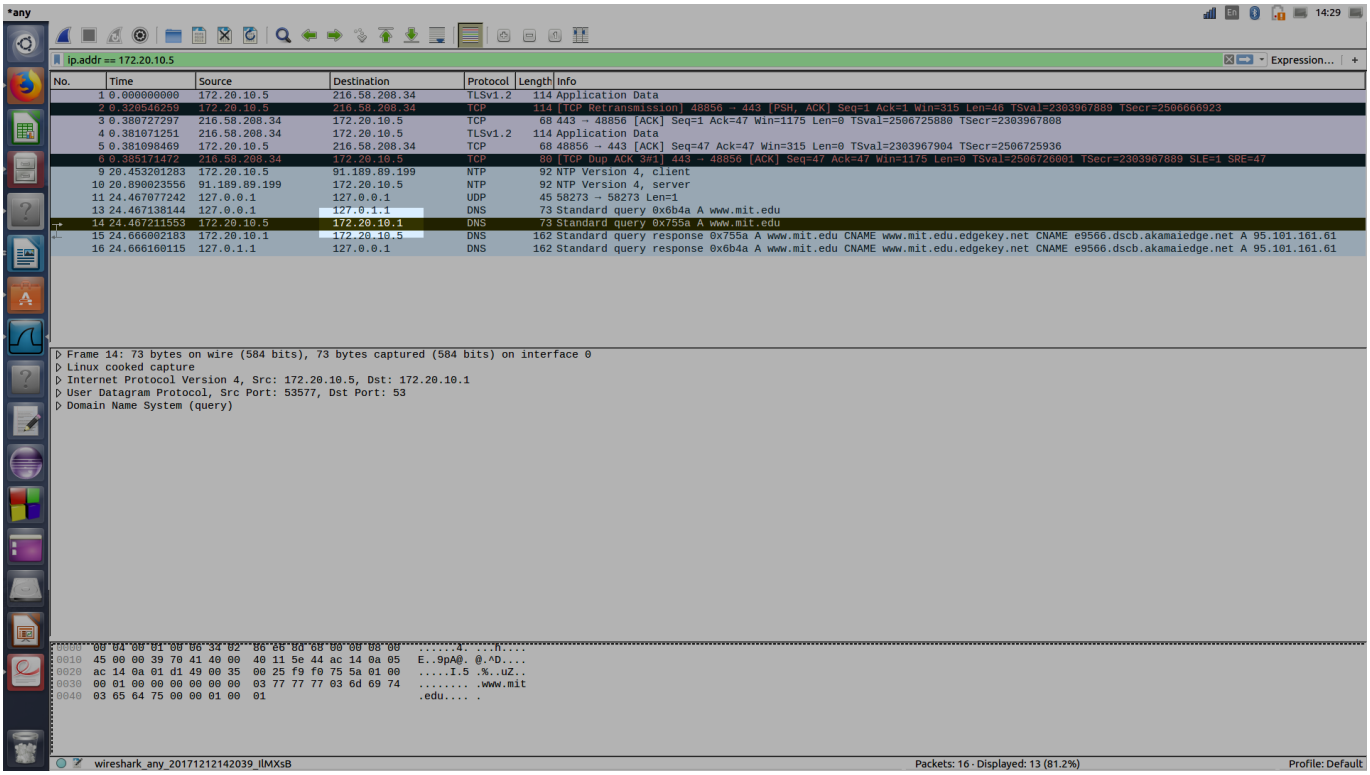
The destination port for the DNS query message is 53,

The source port for the response query is 53.

13.12 -



Screen-shot who from the terminal.

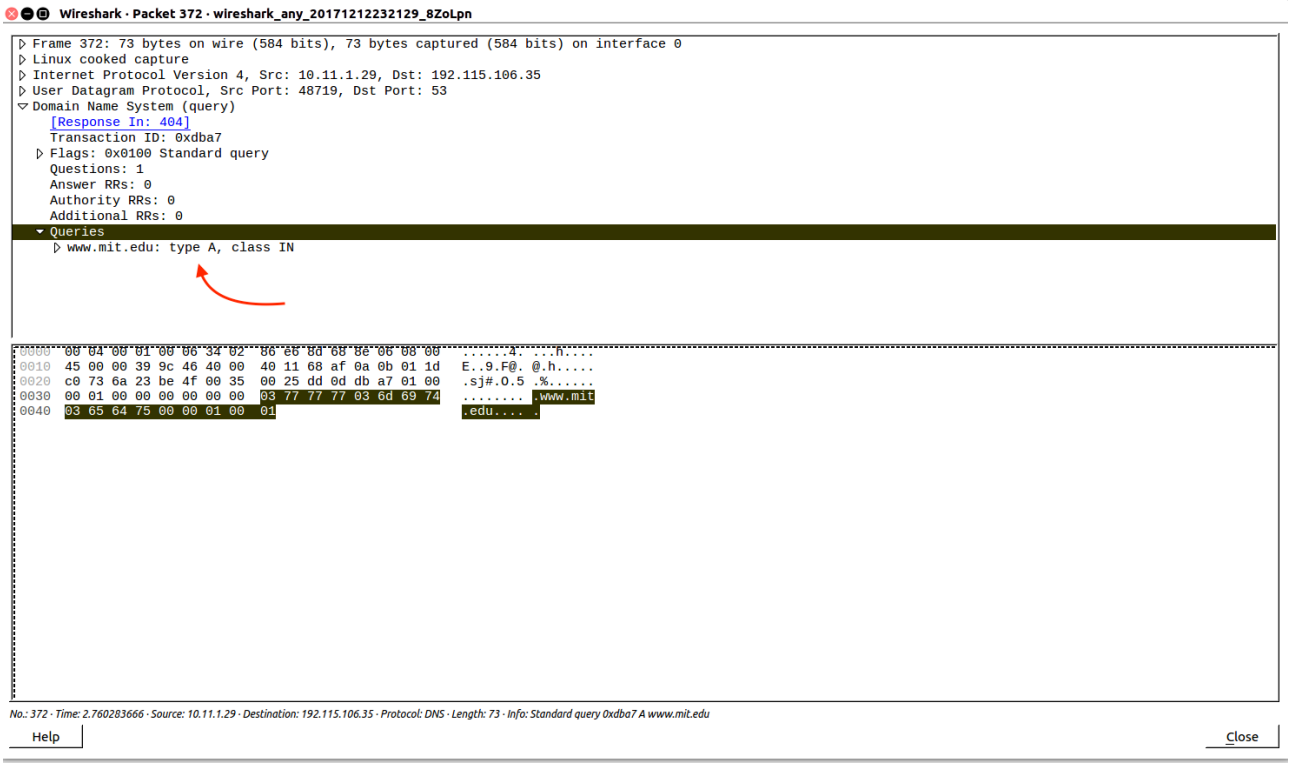


Screen-shot who from wireshark.

The IP of the DNS server is 172.20.10.1 same as my local DNS server.

.

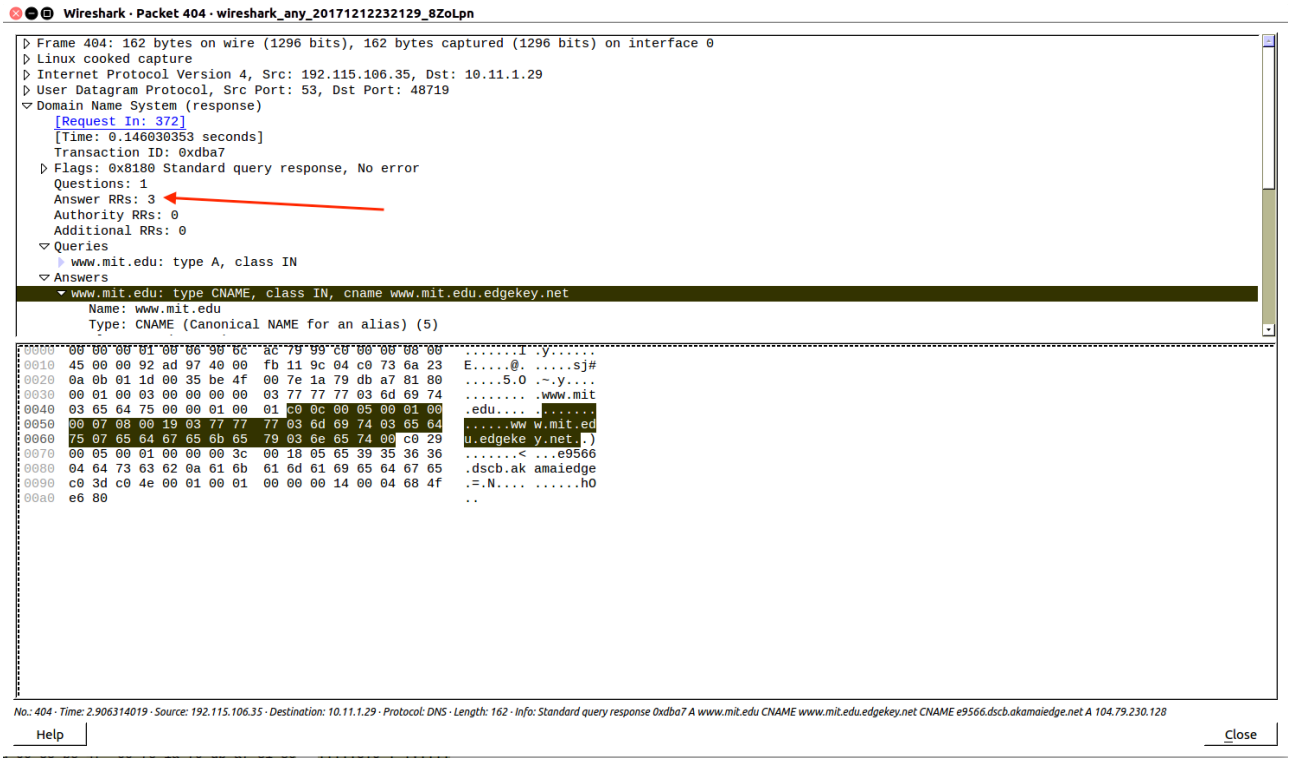
13.13 -



Screen-shot who from wireshark, to display the type of the query message.

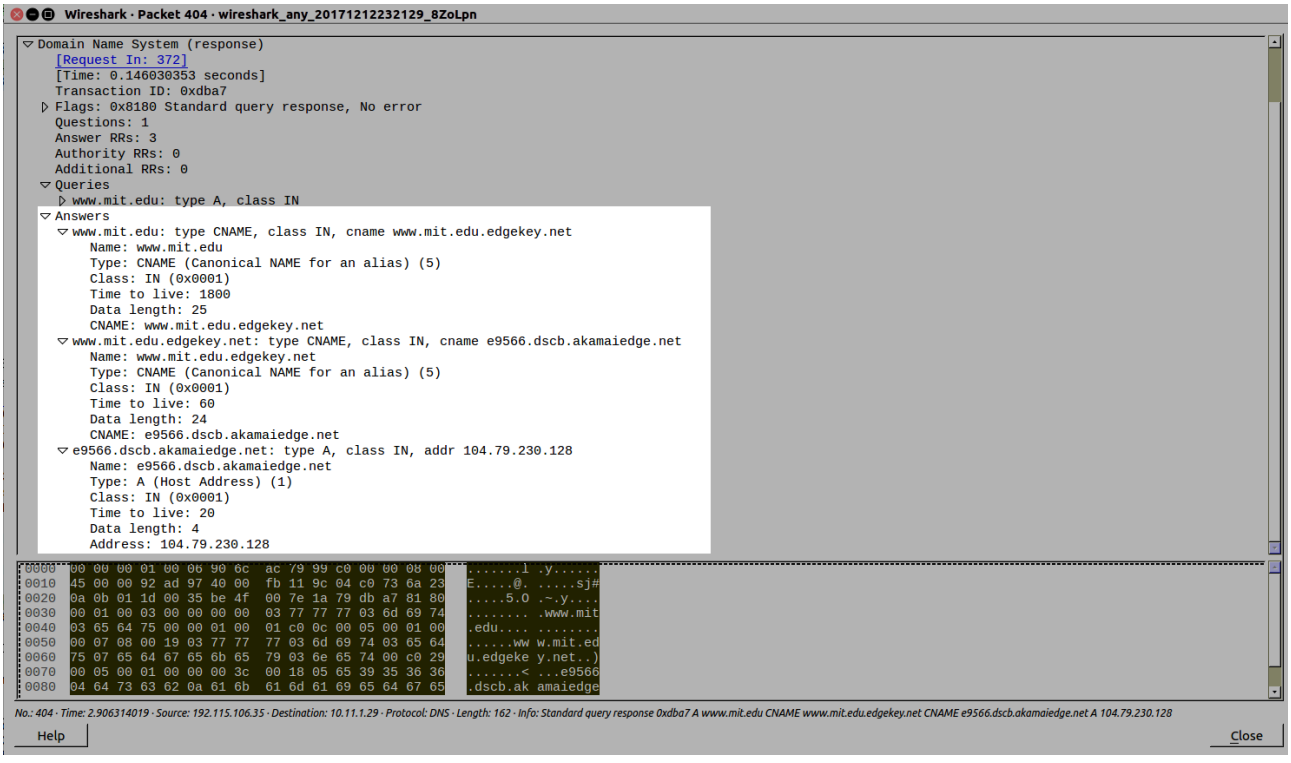
As we can see above in the picture, the message is type A, and there is no any answers.

13.14 -



Screen-shot who from wireshark, to display the numbers of answers.

There are 3 answers as mentioned in the picture above.

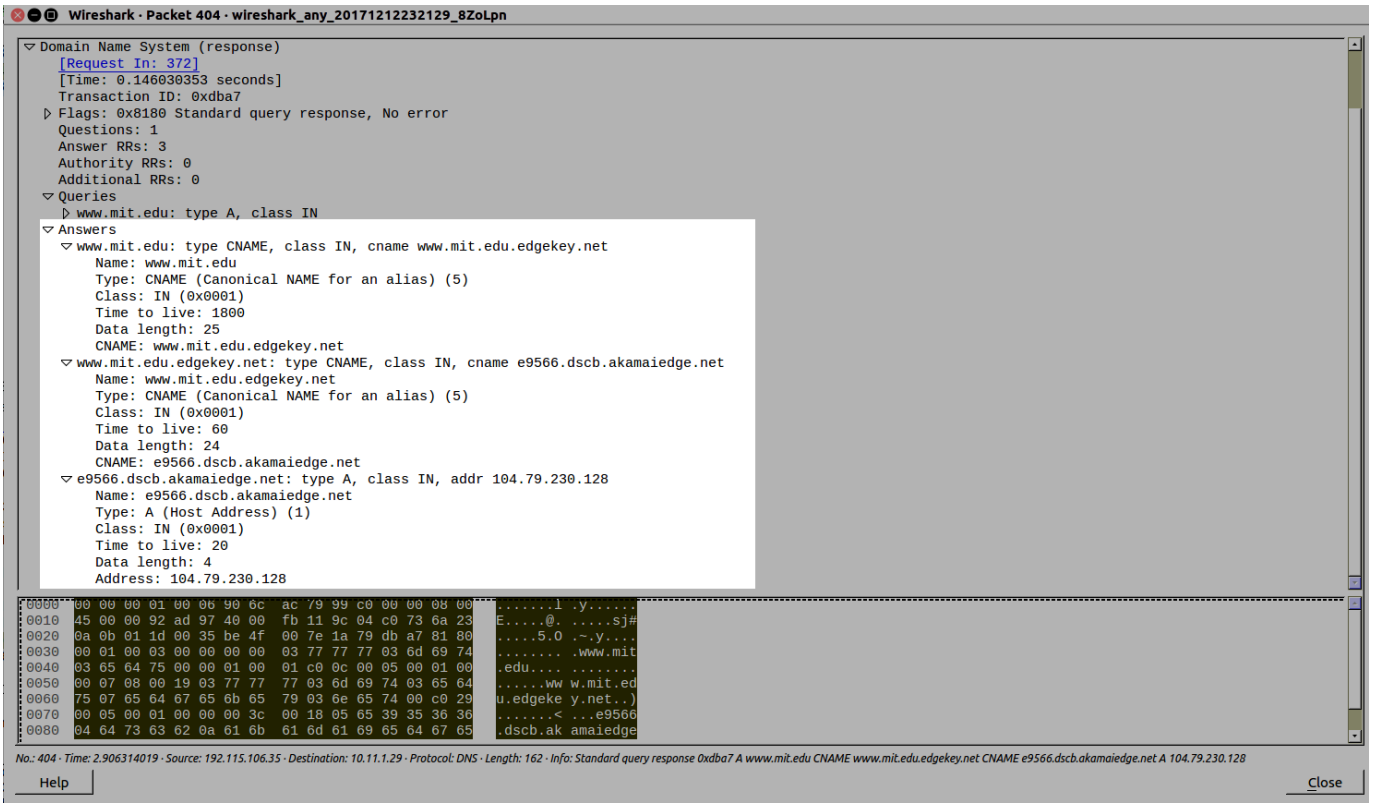


Screen-shot who from wireshark, to display the answers.

.

1. The first response message is of type CNAME that point to a domain name : www.mit.edu.edgekey.net.
2. The second response message is another type CNAME that point to another domain name : e9566.dscb.akamaiedge.net.
3. The last response message is a type A that points to the ip address : 104.79.230.128.

Question – 15
All the interesting screen-shots have been provided.


Wireshark · Packet 404 · wireshark_any_20171212232129_8ZoLpn
Domain Name System (response)
[Request In: 372]
[Time: 0.146030353 seconds]
Transaction ID: 0xdba7
Flags: 0x8180 Standard query response, No error
Questions: 1
Answer RRs: 3
Authority RRs: 0
Additional RRs: 0
Queries
www.mit.edu: type A, class IN
Answers
www.mit.edu: type CNAME, class IN, cname www.mit.edu.edgekey.net
Name: www.mit.edu
Type: CNAME (Canonical NAME for an alias) (5)
Class: IN (0x0001)
Time to live: 1800
Data length: 25
CNAME: www.mit.edu.edgekey.net
www.mit.edu.edgekey.net: type CNAME, class IN, cname e9566.dscb.akamaiedge.net
Name: www.mit.edu.edgekey.net
Type: CNAME (Canonical NAME for an alias) (5)
Class: IN (0x0001)
Time to live: 60
Data length: 24
CNAME: e9566.dscb.akamaiedge.net
e9566.dscb.akamaiedge.net: type A, class IN, addr 104.79.230.128
Name: e9566.dscb.akamaiedge.net
Type: A (Host Address) (1)
Class: IN (0x0001)
Time to live: 20
Data length: 4
Address: 104.79.230.128
0000 00 00 00 01 00 06 90 6c ac 79 99 c0 00 00 08 00 ....l .y......
0010 45 00 00 92 ad 97 40 00 fb 11 9c 04 c0 73 6a 23 E.....@. .....sj#
0020 0a 0b 01 1d 00 35 be 4f 00 7e 1a 79 db a7 81 80 .....5.O .~.y....
0030 00 01 00 03 00 00 00 00 03 77 77 77 03 6d 69 74 .... .www.mit
0040 03 65 64 75 00 00 01 00 01 c0 0c 00 05 00 01 00 .edu.... ....
0050 00 07 08 00 19 03 77 77 77 03 6d 69 74 03 65 64 ......ww w.mit.ed
0060 75 07 65 64 67 65 6b 65 79 03 6e 65 74 00 c0 29 u.edgeke y.net..)
0070 00 05 00 01 00 00 00 3c 00 18 05 65 39 35 36 36 ....< ...e9566
0080 04 64 73 63 62 0a 61 6b 61 6d 61 69 65 64 67 65 .dscb.ak amaiedge
No.: 404 · Time: 2.906314019 · Source: 192.115.106.35 · Destination: 10.11.1.29 · Protocol: DNS · Length: 162 · Info: Standard query response 0xdba7 A www.mit.edu CNAME www.mit.edu.edgekey.net CNAME e9566.dscb.akamaiedge.net A 104.79.230.128
Help
Close