GOBIERNO DE ESPAÑA
VICEPRESIDENCIA TERCERA DEL GOBIERNO
MINISTERIO DE ASUNTOS ECONÓMICOS Y TRANSFORMACIÓN DIGITAL
SECRETARÍA DE ESTADO DE DIGITALIZACIÓN E INTELIGENCIA ARTIFICIAL

Webinar: Basic use of Wireshark

Practical exercise

017 TU AYUDA EN CIBERSEGURIDAD
incibe_
INSTITUTO NACIONAL DE CIBERSEGURIDAD

INDEX

1. SMTP ANALYSIS 3
2. HTTP ANALYSIS 7

FIGURE INDEX

Figure 1 SMTP Command Exchange 3
Figure 2 SMTP analysis from Wireshark 4
Figure 3 In-depth SMTP 5
Figure 4 Version of the mail client used 6
Figure 5 HTTP Request 7
Figure 6 HTTP Message 8
Figure 7 HTTP analysis from Wireshark 8
Figure 8 HTTP Filter 8
Figure 9 Expanded information on the protocol stack 9

1. SMTP ANALYSIS

The SMTP protocol (Simple Mail Transfer Protocol), is a widely used protocol associated with an RFC dating from 1982. This protocol is used for sending e-mail. Most malicious attacks are related to attachments that are sent or links embedded in the body of the message.

As you can see in the image below, the SMTP protocol is a line-oriented protocol where messages are exchanged between the server (in red) and the client (in blue).

The line-oriented protocol is a protocol where a carriage return and a new line are used for each command exchanged.

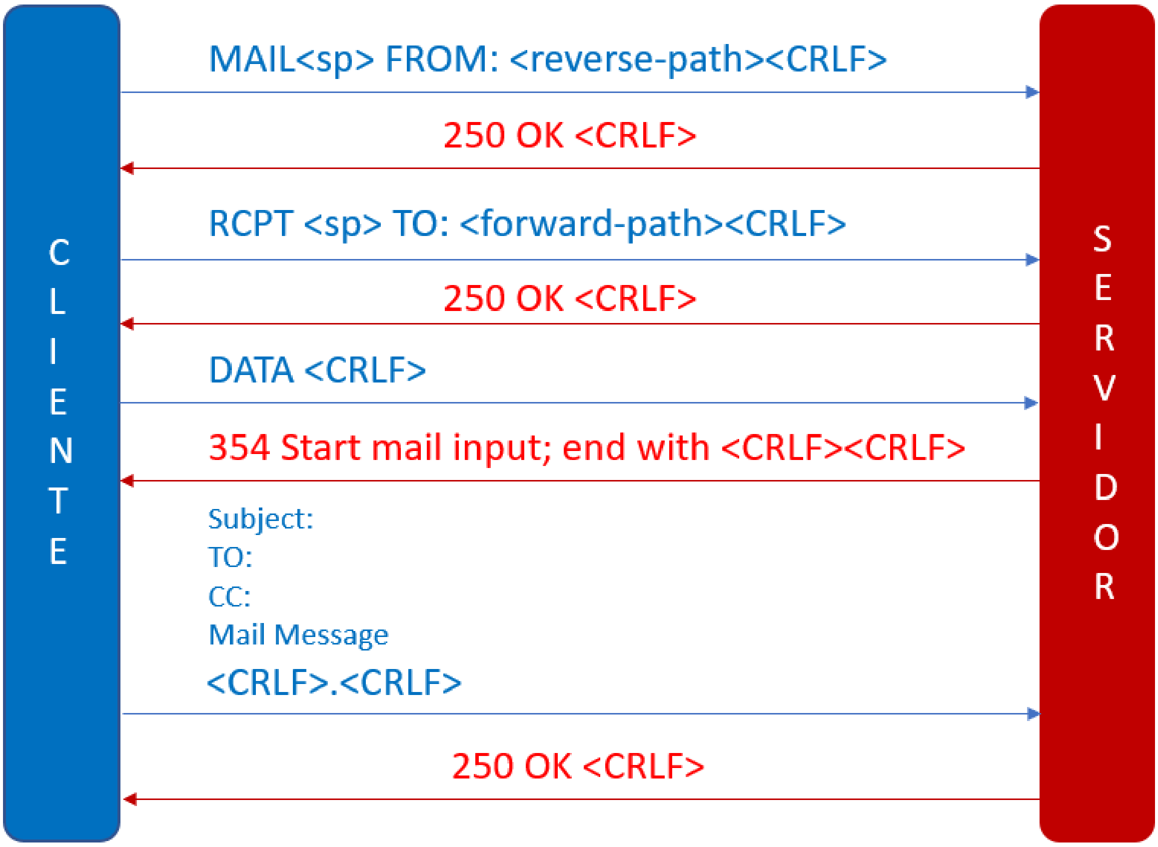

Figure 1 SMTP Command Exchange

An SMTP transaction starts with the MAIL command from the client to the server, along with the FROM parameter where possible failures in sending the mail are indicated. SMTP ends with a carriage return and line feed (CRLF). If the MAIL command is accepted by the server, the server will respond with the code "250 OK", and this means that everything is correct.

The next step is to send the RCPT command to identify one or more recipient email addresses. If the SMTP server accepts this, it will respond with "250 OK" Otherwise, it will respond with a "550 No such user here", indicating that it has not been able to find the recipient.

The mail client sends the DATA command to indicate that the next command will be the message. The server responds with a "354 Start mail input, end with <CRLF>.<CRLF>", where the server tells the client how to end the message, in our case a line containing "." If everything is correct the server will return a 250 OK.

Let's analyze a screenshot that contains an SMTP communication:

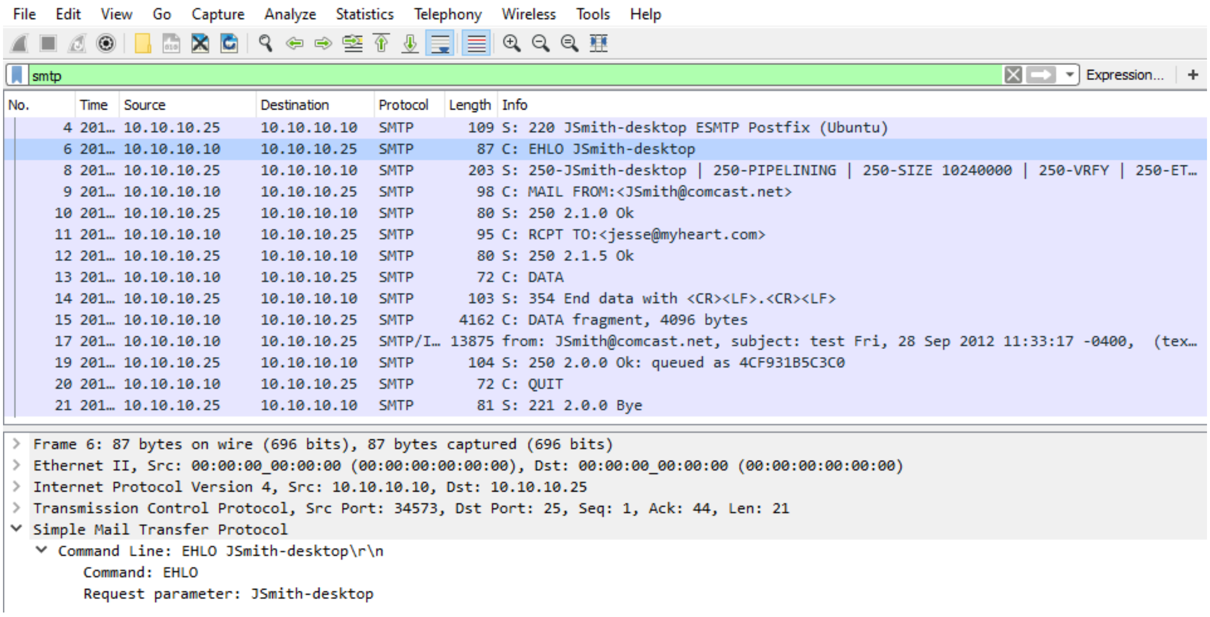

Figure 2 SMTP analysis from Wireshark

As we see in the previous capture, the communication starts previously with a new HELO or EHLO command.

Currently both are used, but when using EHLO the server responds with additional features such as PIPELINING, SIZE, HELP and ENHACEDSTATUSCODES as can be identified in the following screenshot:

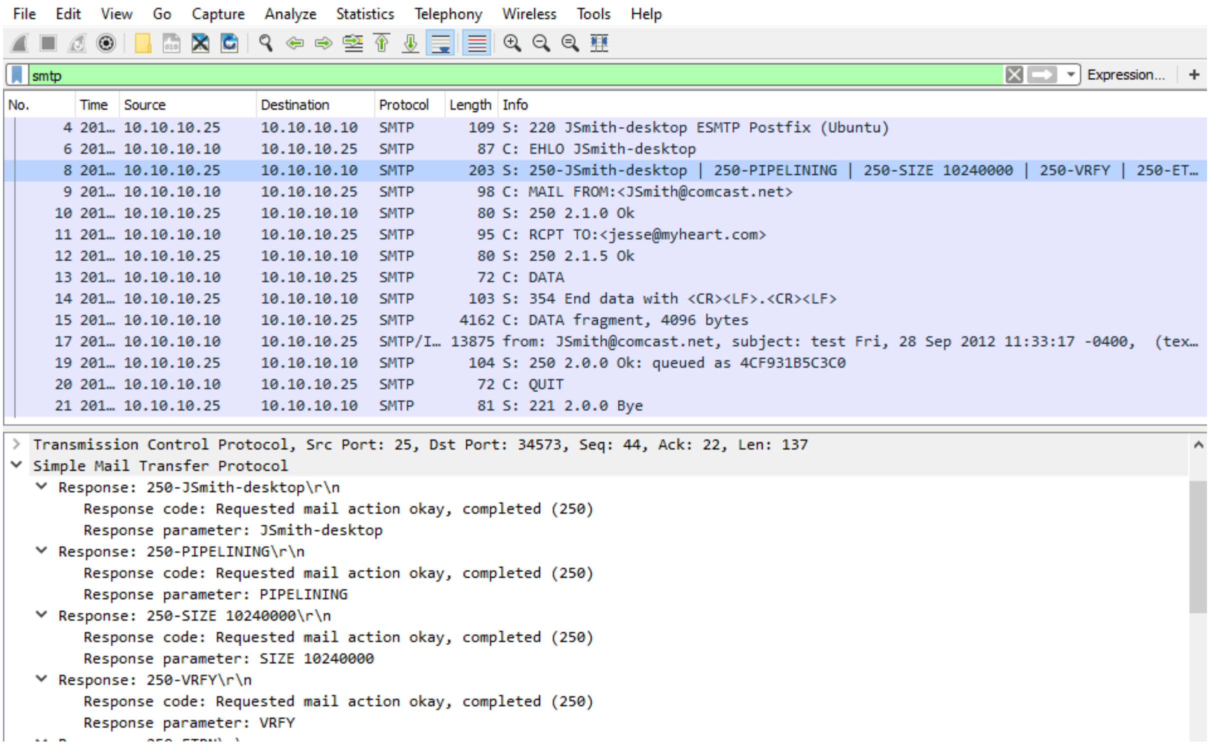

Figure 3 In-depth SMTP

The session starts with a TCP three-way handshake to the SMTP server, and the server responds with a 220 as shown in packet number 4. The client continues with the message, in packet number 6 with an EHLO identifying itself as JSmith-desktop and the server responds with a "250 OK".

Once we have explained how SMTP works, we propose the following exercise, given the capture smtp_sample.pcap. Can you tell us which is the email client that was supposedly used to send the email in question?

Knowing that the client has to send the DATA message with the information of the email, including the headers, you could obtain such information:

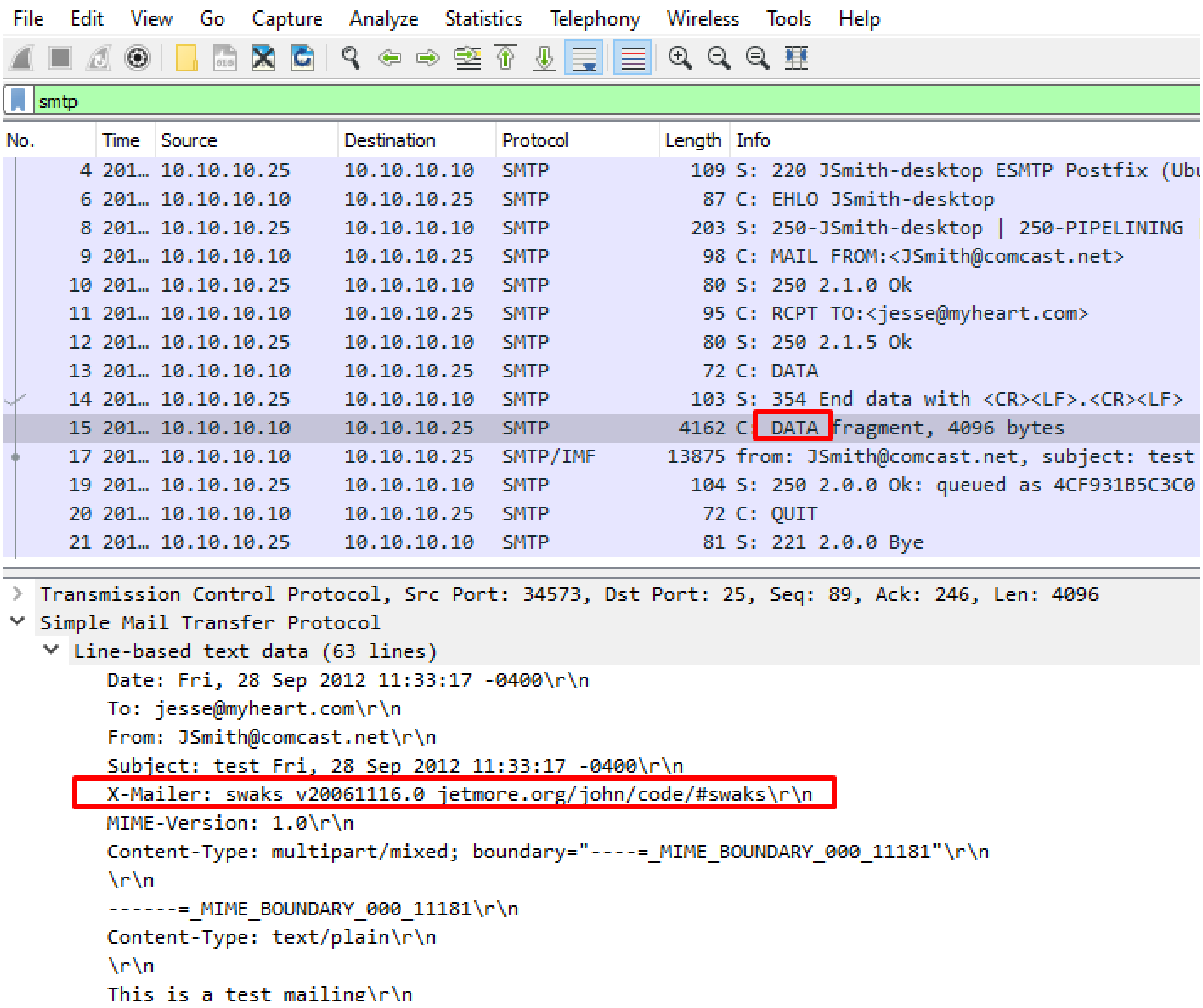

Figure 4 Version of the mail client used

The answer would be X-Mailer: swaks v20061116.0 jetmore.org/john/code/#swaks

2. HTTP ANALYSIS

"Hypertext Transfer Protocol" (HTTP) is a worldwide known protocol where initially attacks were perpetrated against servers, but today often the target is the browser to compromise the computer host.

As you can see, HTTP is a simple format protocol, but the body format of an HTTP request or response can be complicated. Servers and browsers are susceptible to many vulnerabilities and types of attacks. HTTP is a stateless protocol because the server does not maintain status between transactions in a session.

An HTTP request begins with a "Start Line" that includes a method, a URL, the HTTP version, and ends with a carriage return and new line (CRLF). There are different HTTP methods where the most used is GET.

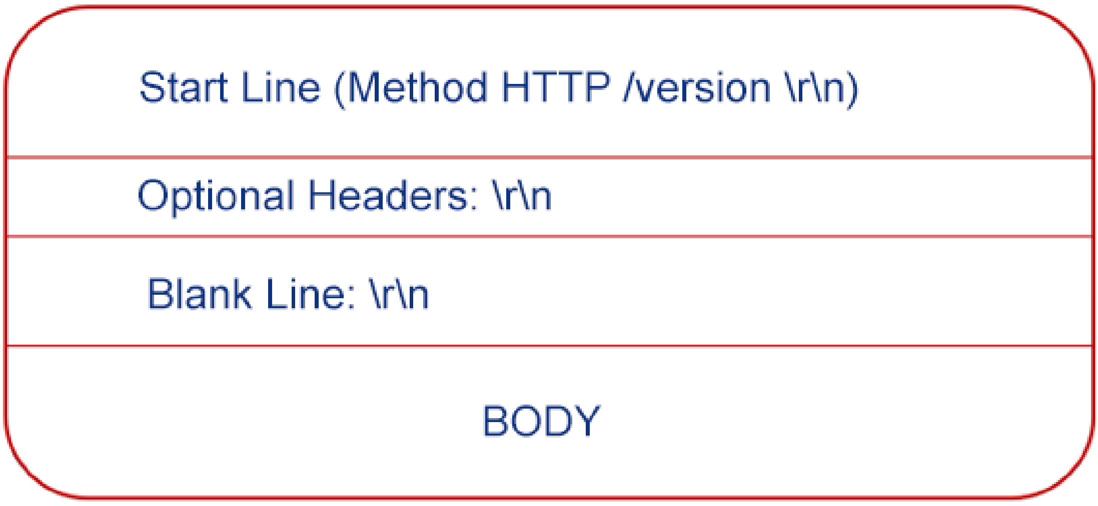

Figure 5 HTTP Request

A GET method makes requests for some type of resource or document identified by the server URL. Another type of method is POST, which sends data to a server specified by a URL. There may be optional headers from which content is accepted, languages or encodings and some also for security. After the headers we can find a blank line with a CRLF and finally the body of the message.

HTTP and SMTP are line-oriented protocols, this means that the protocol uses a new line to delimit the different elements that make up the request.

The format of the HTTP response is not very different from the request. The only noticeable difference is that the start line is the version, status code, and reason. The version field tells us the version supported by the HTTP server. The status code is 3 digits that indicate the result of the request. The first digit indicates the class of codes such as success or error. The reason field explicitly indicates what the status code indicates. Finally, the start line should end with CRLF.

Headers are optional again, but most servers include them. They must be followed by a blank line and body which is optional.

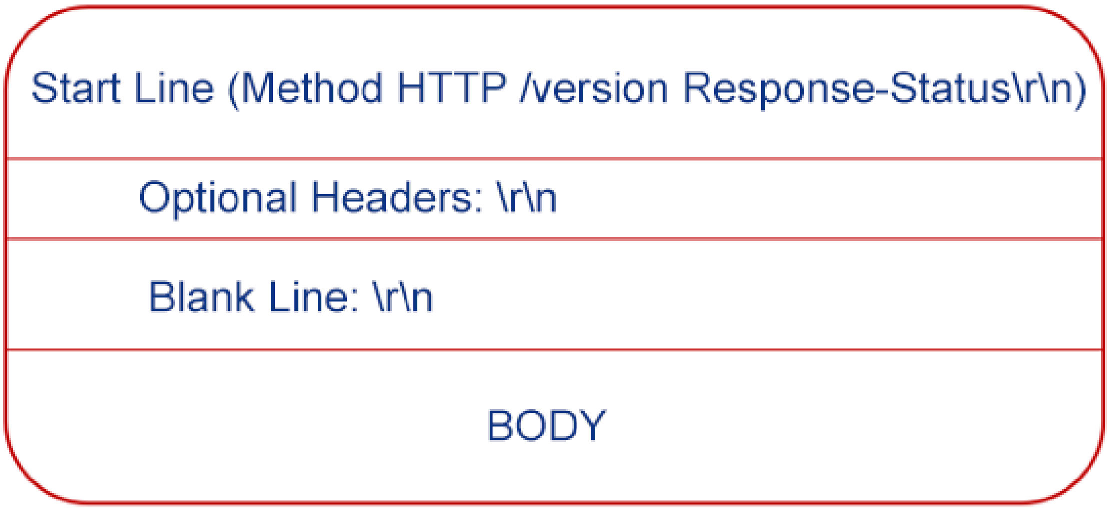

Figure 6 HTTP Message

By giving the http.cap file, could you tell us which server would be involved in the request, i.e. the type of server used?

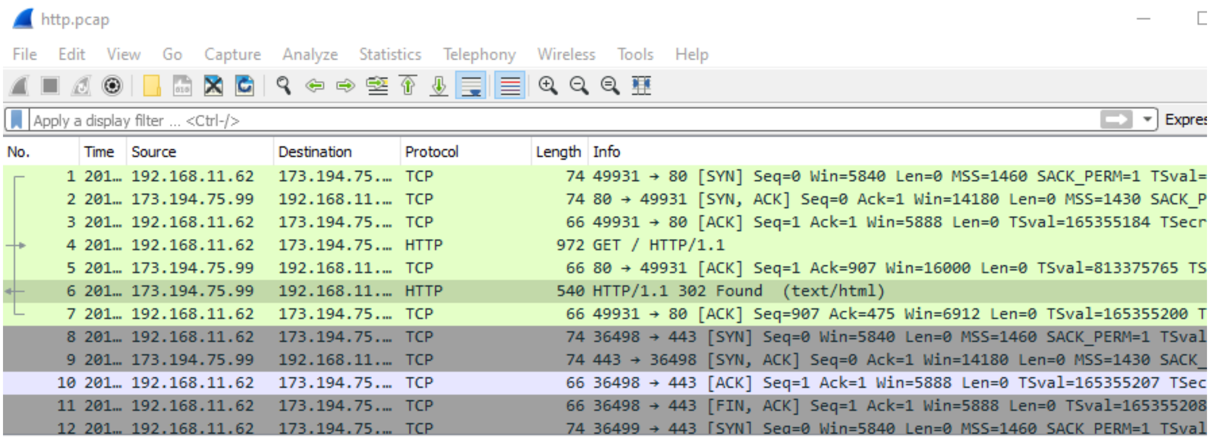

Figure 7 HTTP analysis from Wireshark

The first step would be to filter by the HTTP protocol:

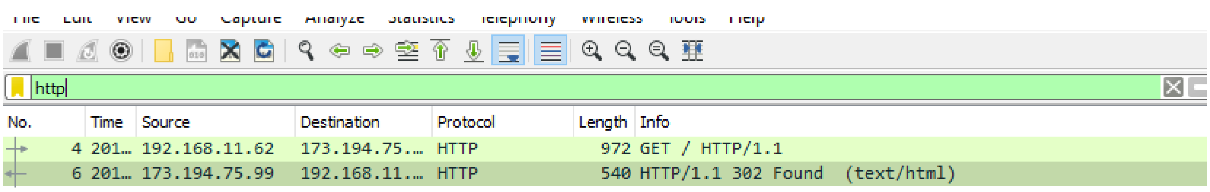

Figure 8 HTTP Filter

Once filtered we would have two HTTP packets left where we would have to analyze the server's response. When we open the number 6 packet we can see the headers of the response:

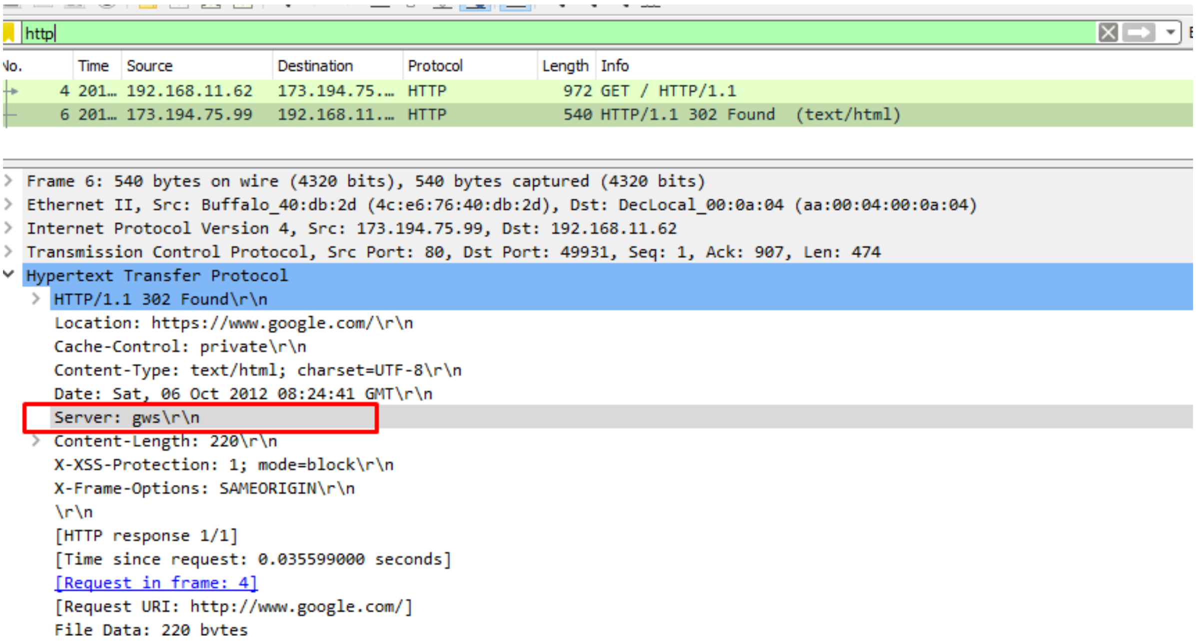

Figure 9 Expanded information on the protocol stack

We can finally see the “gws” server in the headers.