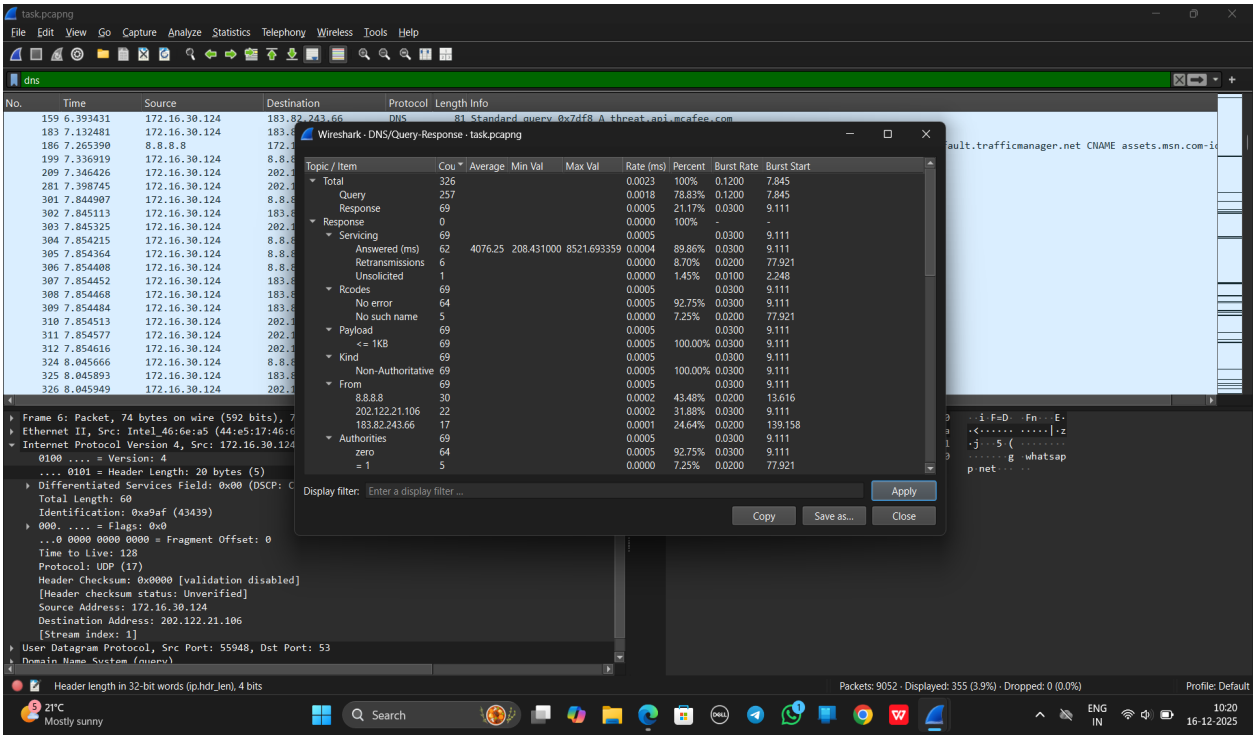
Wireshark Investigation of Application Layer Protocols

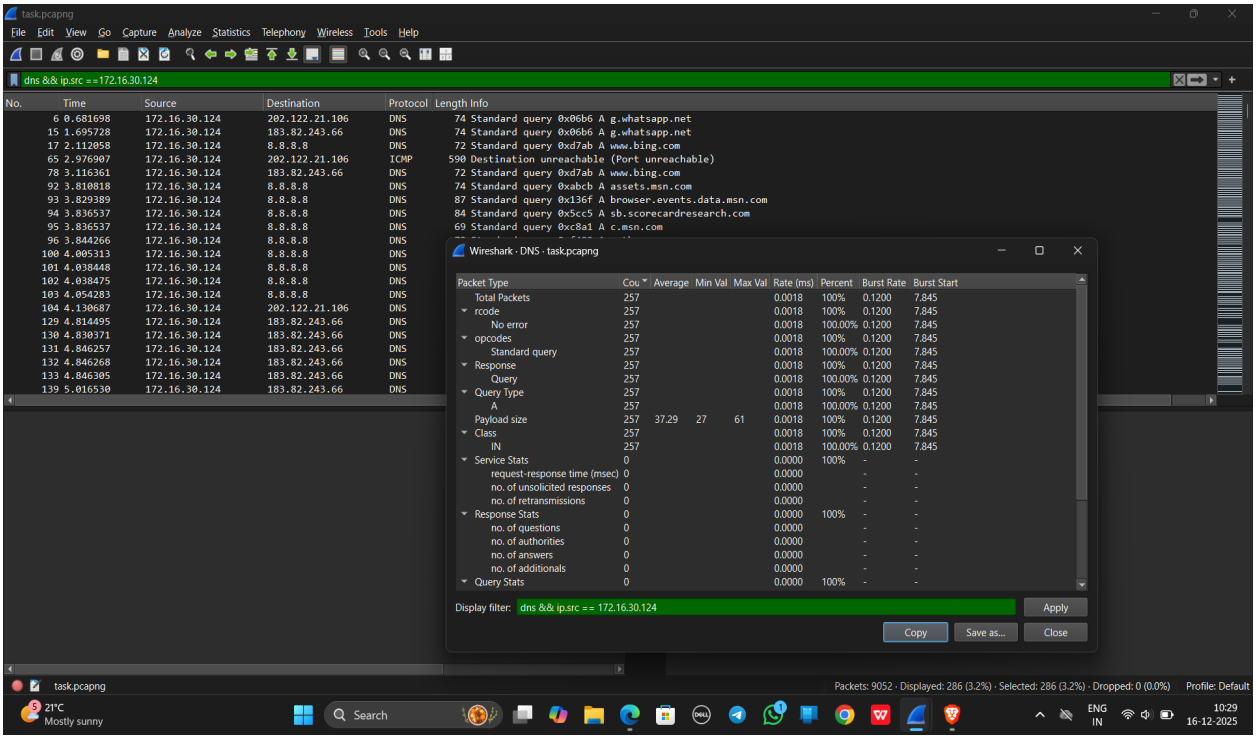
This report presents a detailed Wireshark-based analysis of DNS, TCP, HTTP/1.1, and HTTP/2 protocols while accessing http://testphp.vulnweb.com/login.php. Each screenshot below directly corresponds to a specific task or sub-question and serves as primary evidence for the analysis.

Task 1: DNS, TCP, and HTTP Analysis

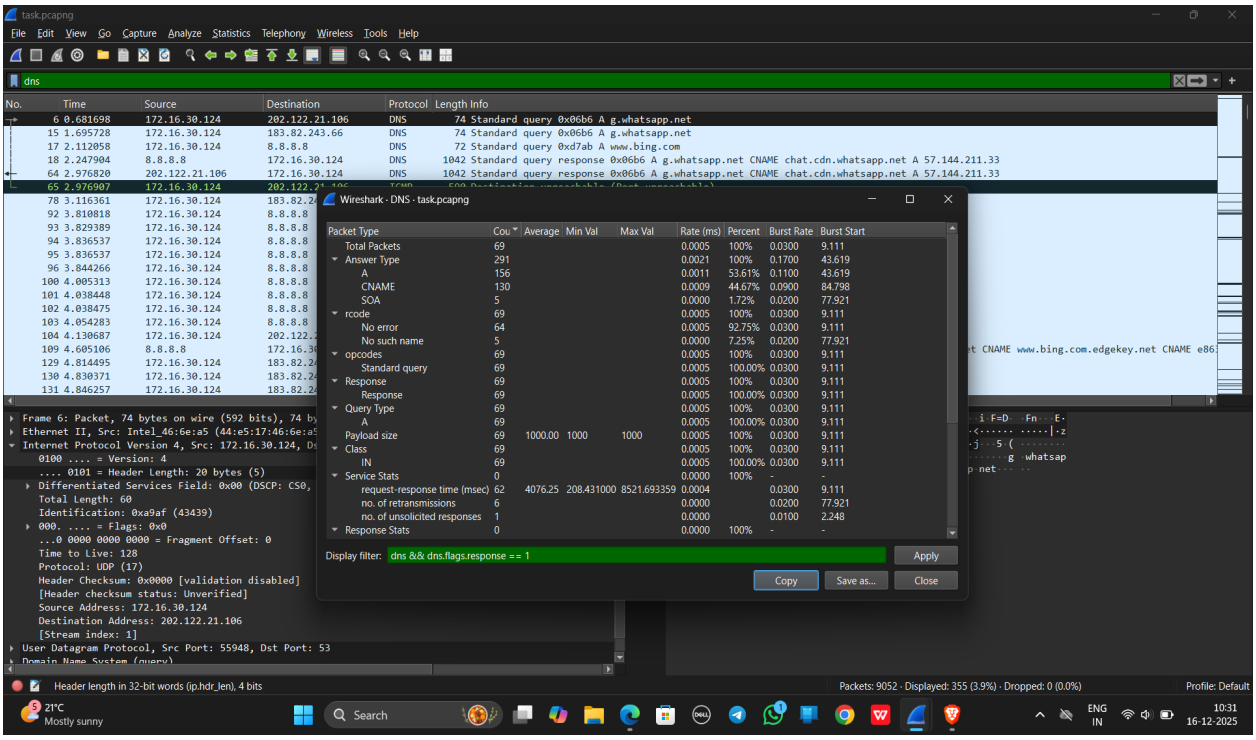
DNS Resolution Process



DNS query packets generated by the browser while resolving testphp.vulnweb.com.

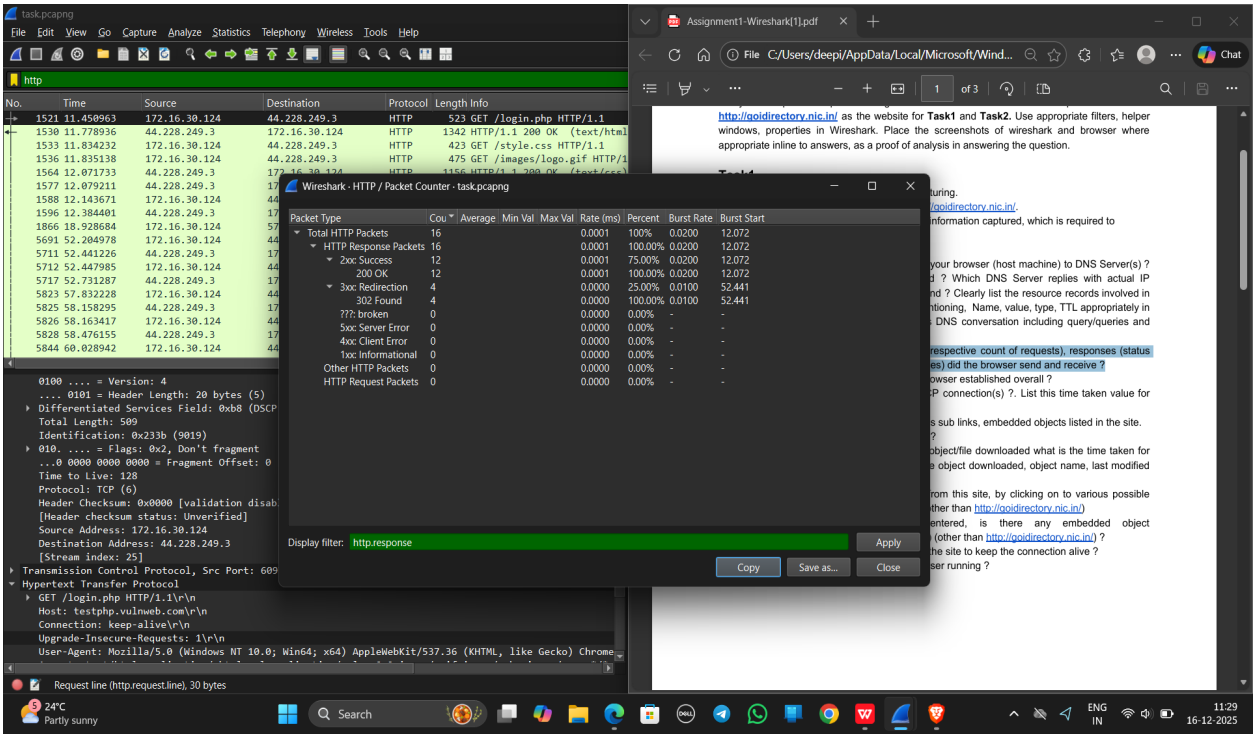


DNS response packets showing resolved IP address records returned by DNS servers.

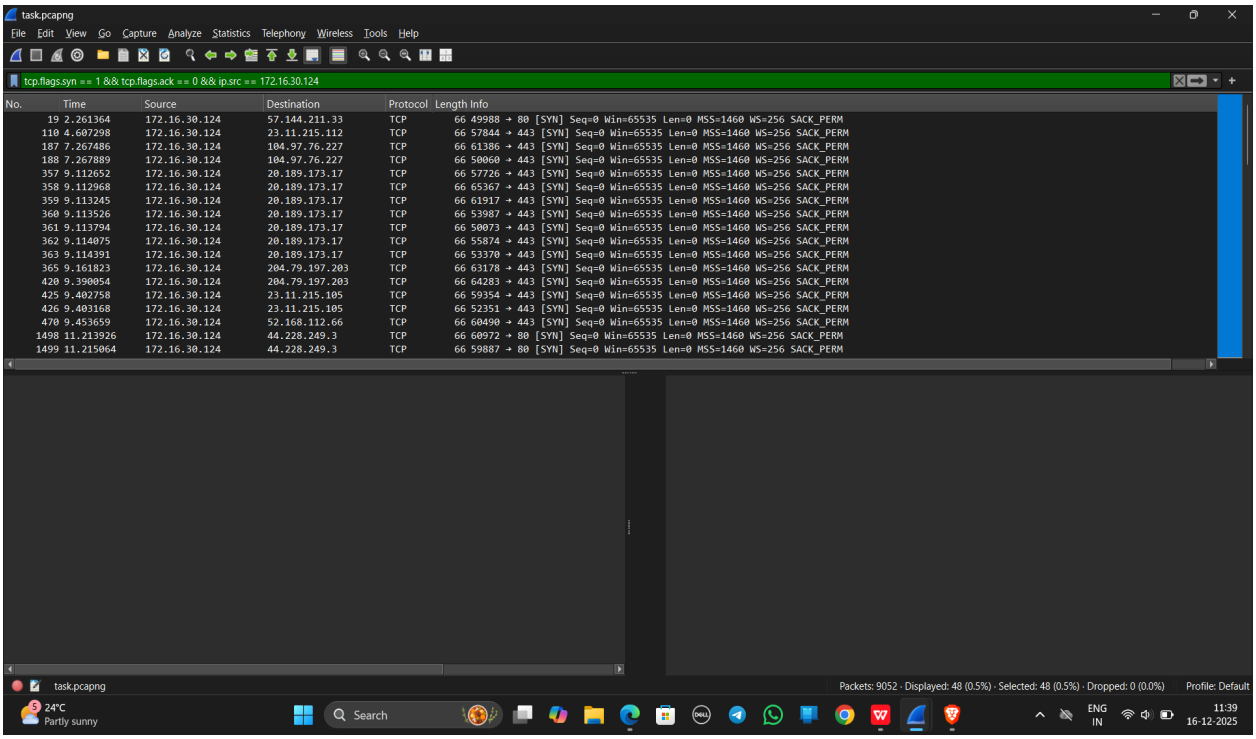


Expanded DNS answer section displaying resource record type, value, and TTL.

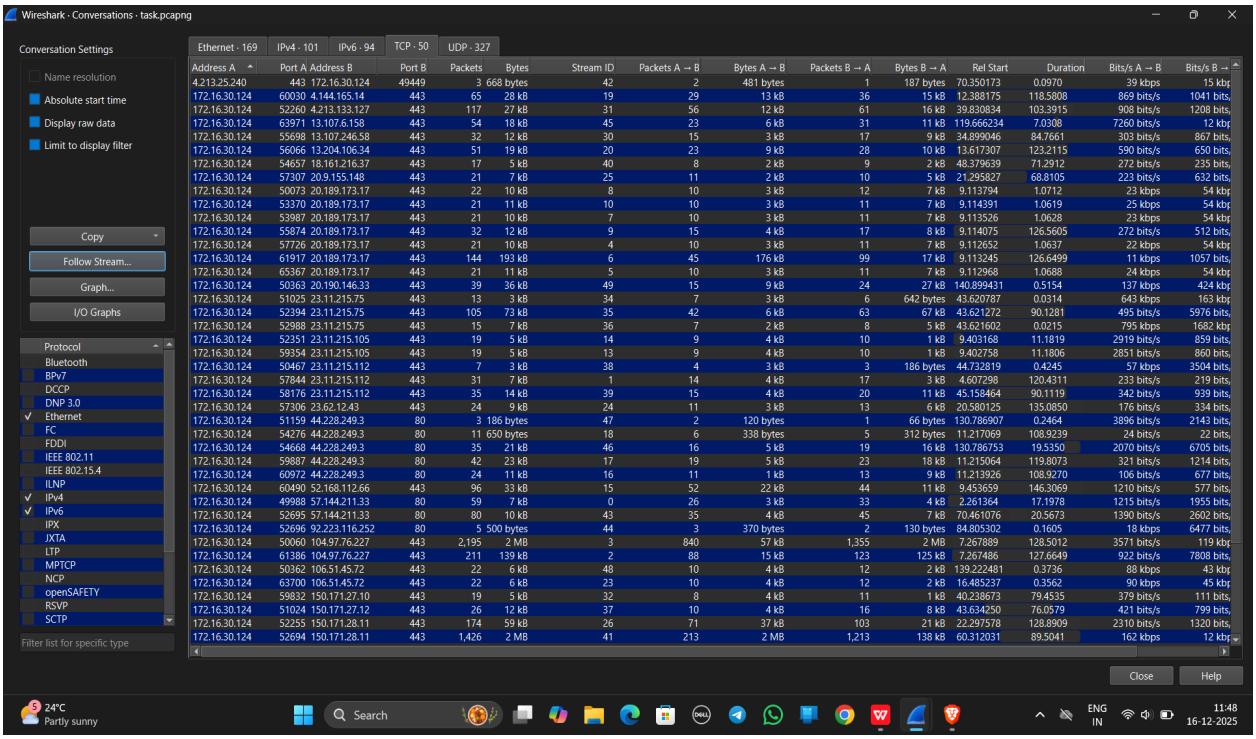
HTTP Requests and Responses



HTTP GET requests sent by the browser for login.php and embedded objects.

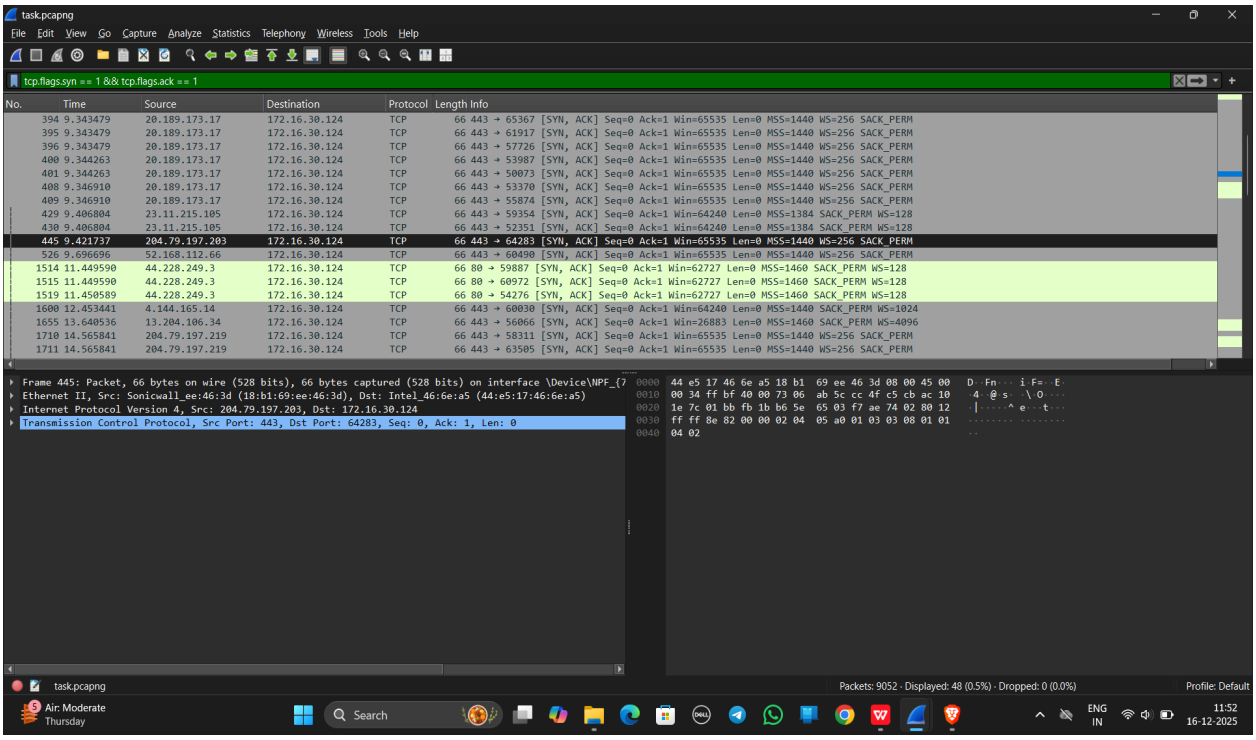


HTTP response packets indicating successful and redirected responses from the server.

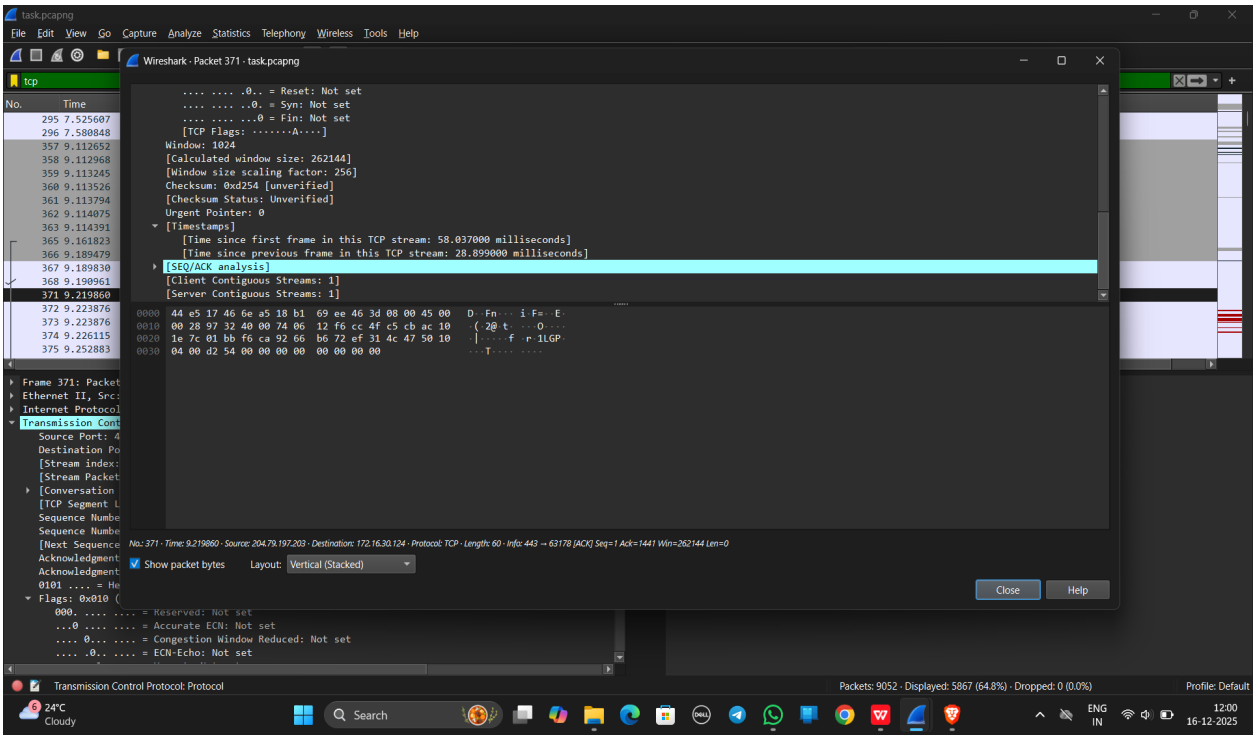


HTTP statistics window summarizing request types and response codes.

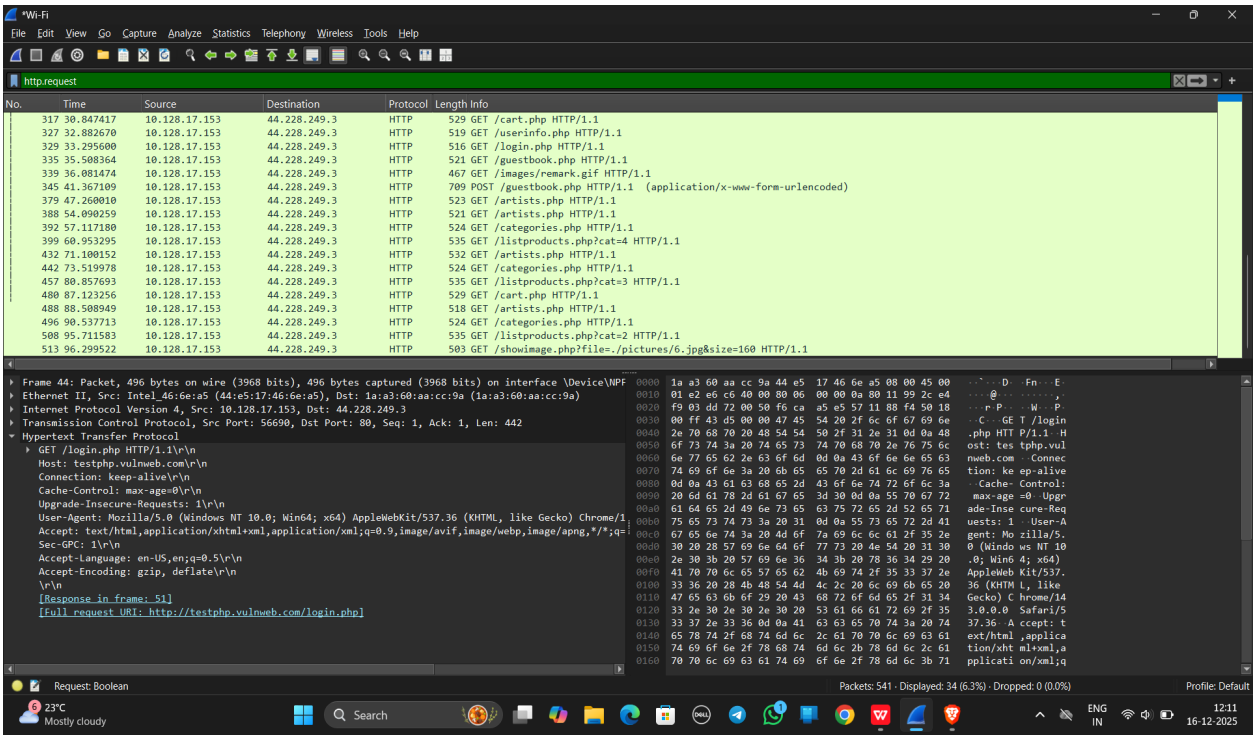
TCP Connections and Timing



TCP SYN packets initiating new connections between client and web server.

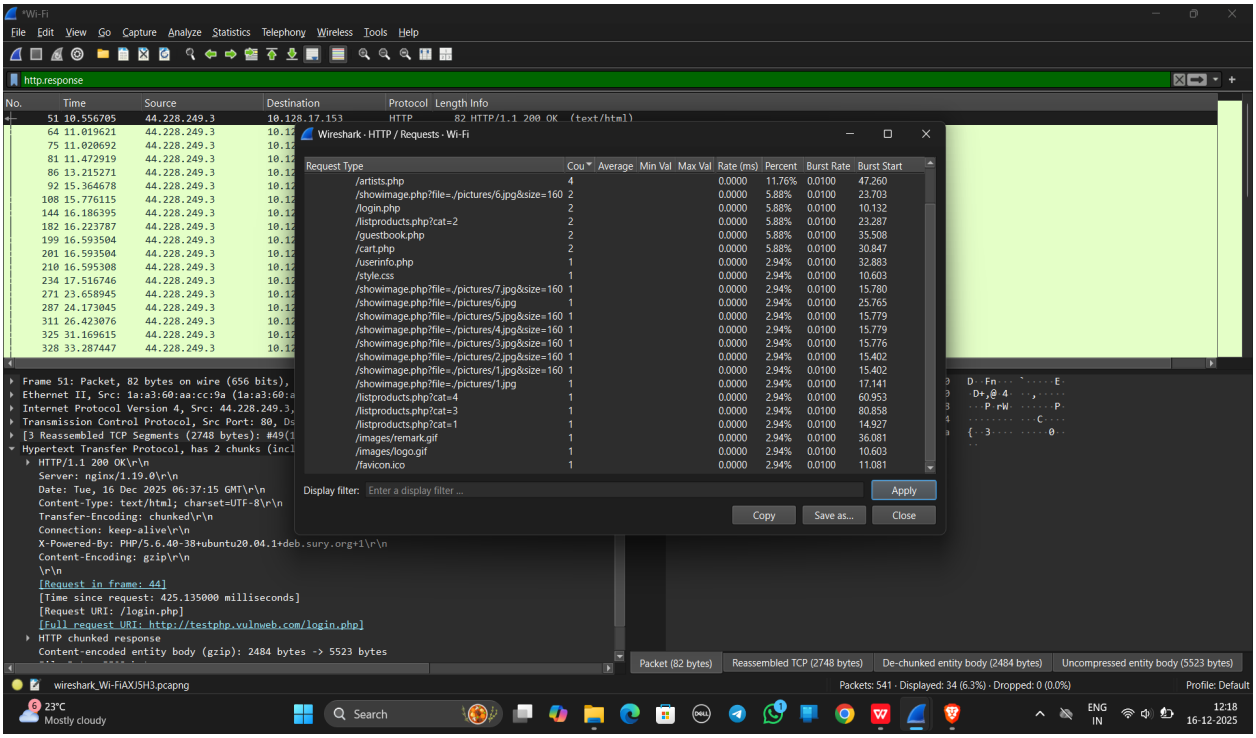


TCP three-way handshake completion confirming successful connection establishment.

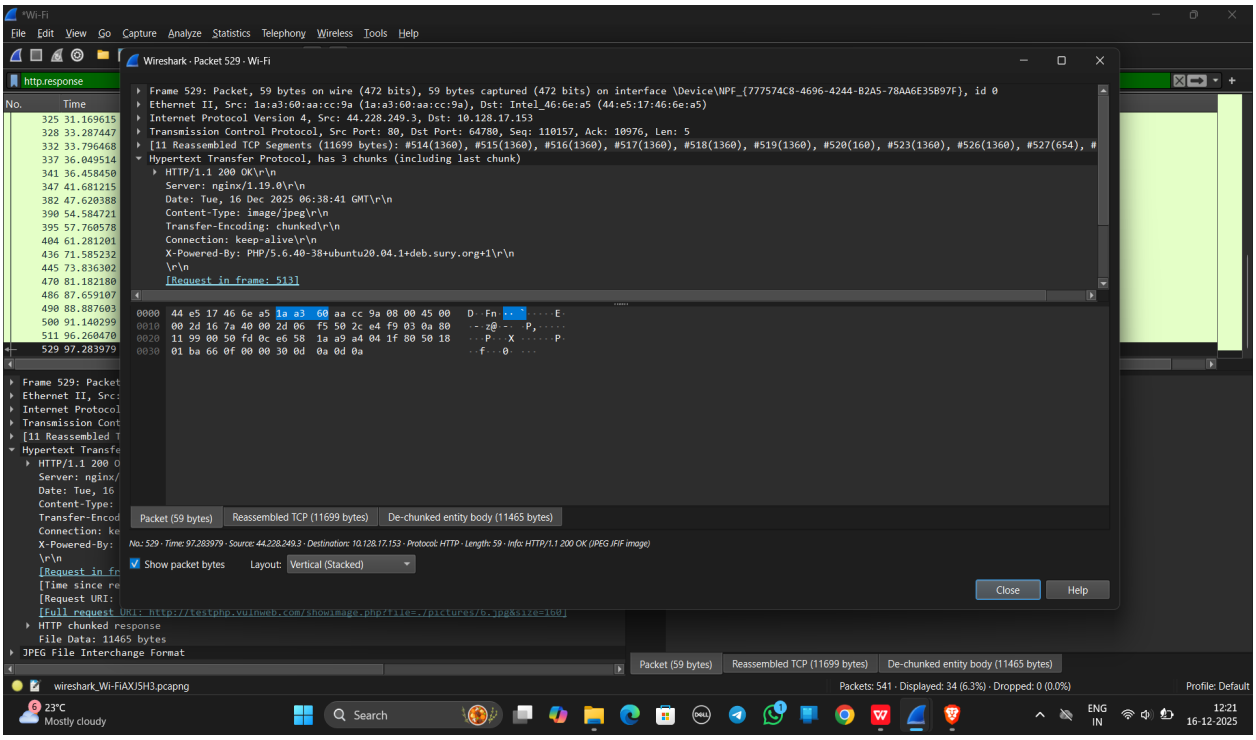


TCP stream timing details used to calculate connection establishment duration.

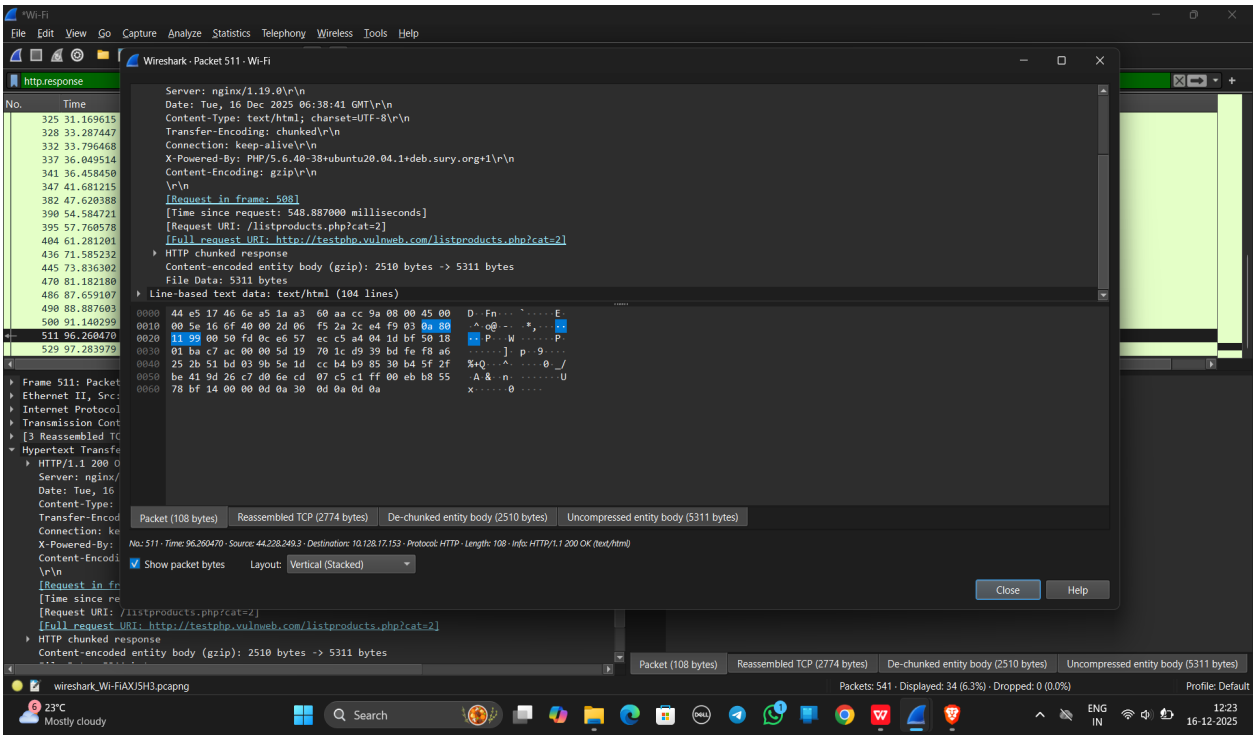
Downloaded Objects and Keep-Alive



List of HTTP object requests generated while browsing sub-links on the website.

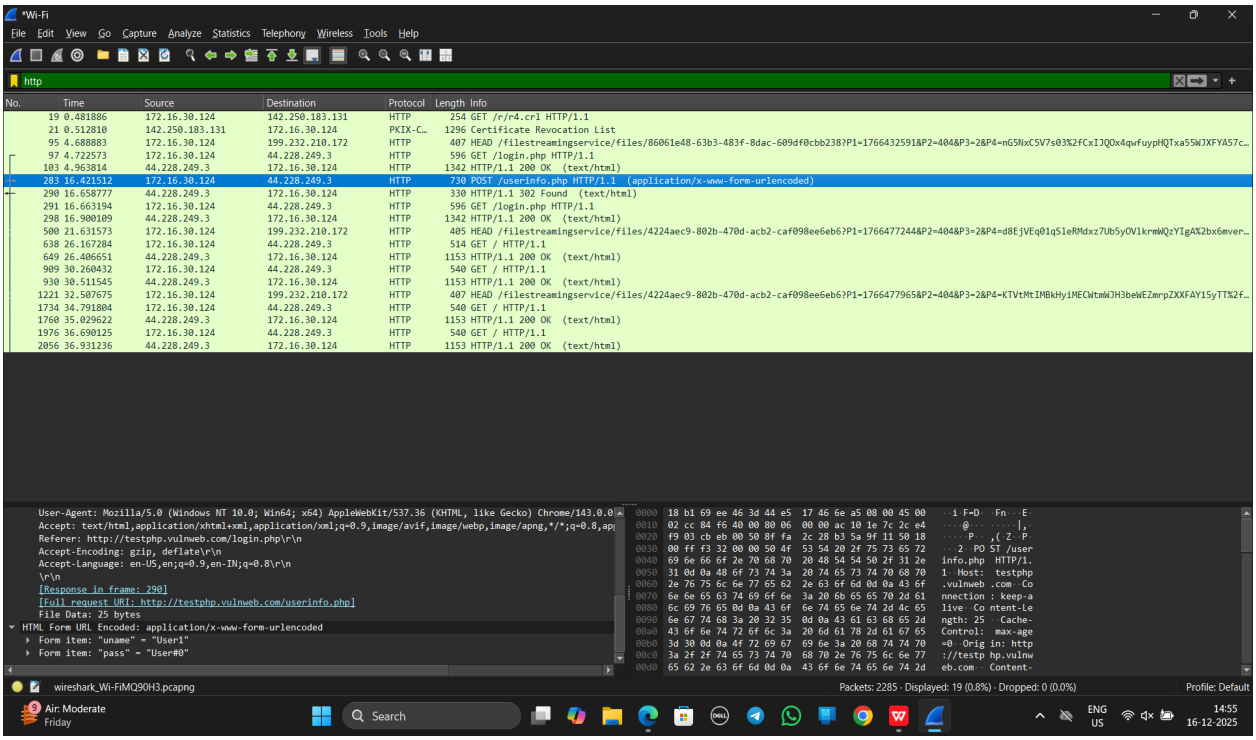


HTTP response headers showing Content-Length and Last-Modified fields for downloaded objects.

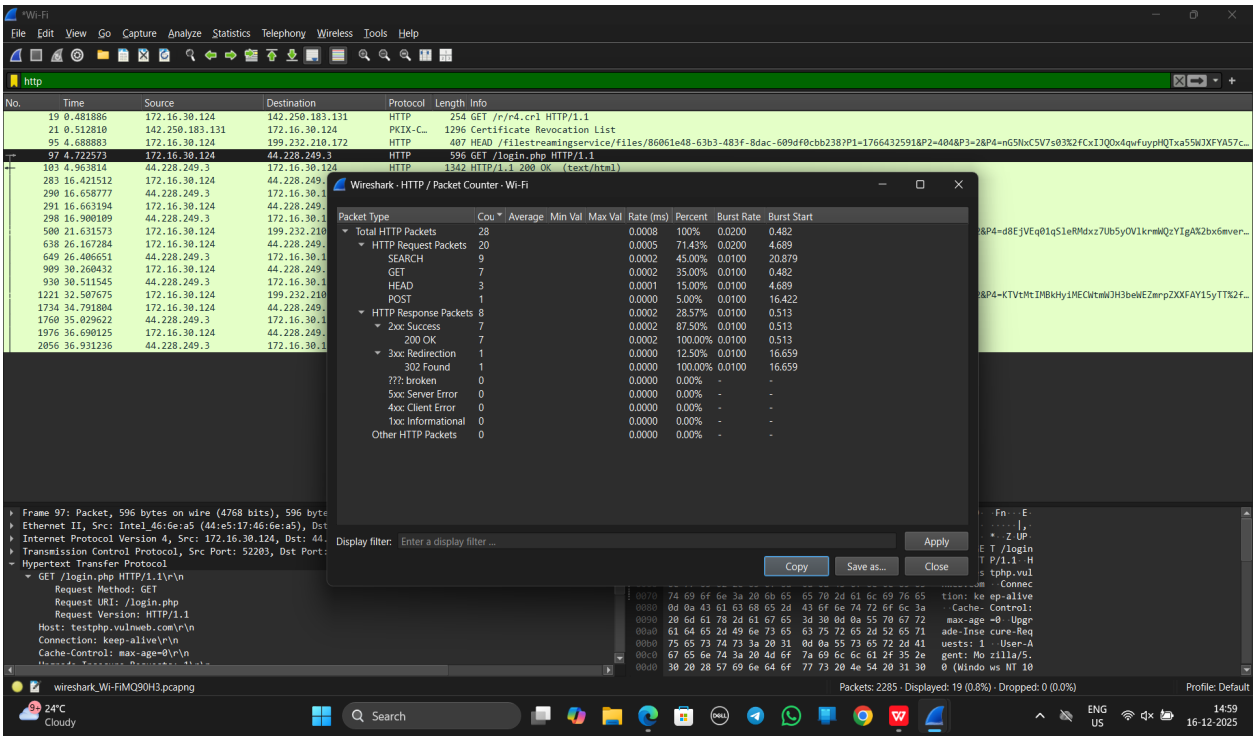


Persistent TCP connection usage demonstrating HTTP keep-alive behavior.

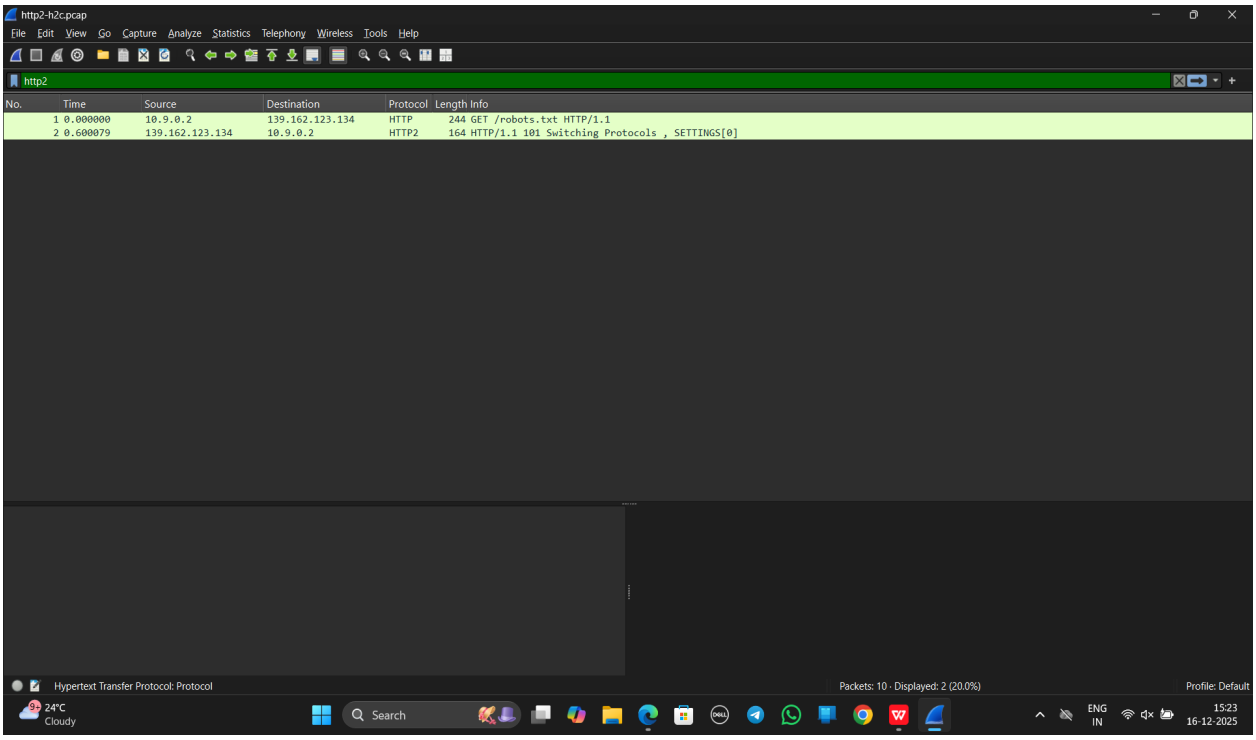
Task 2: Browser Cache and Conditional GET Analysis



Initial HTTP request after clearing browser cache, resulting in full object download.



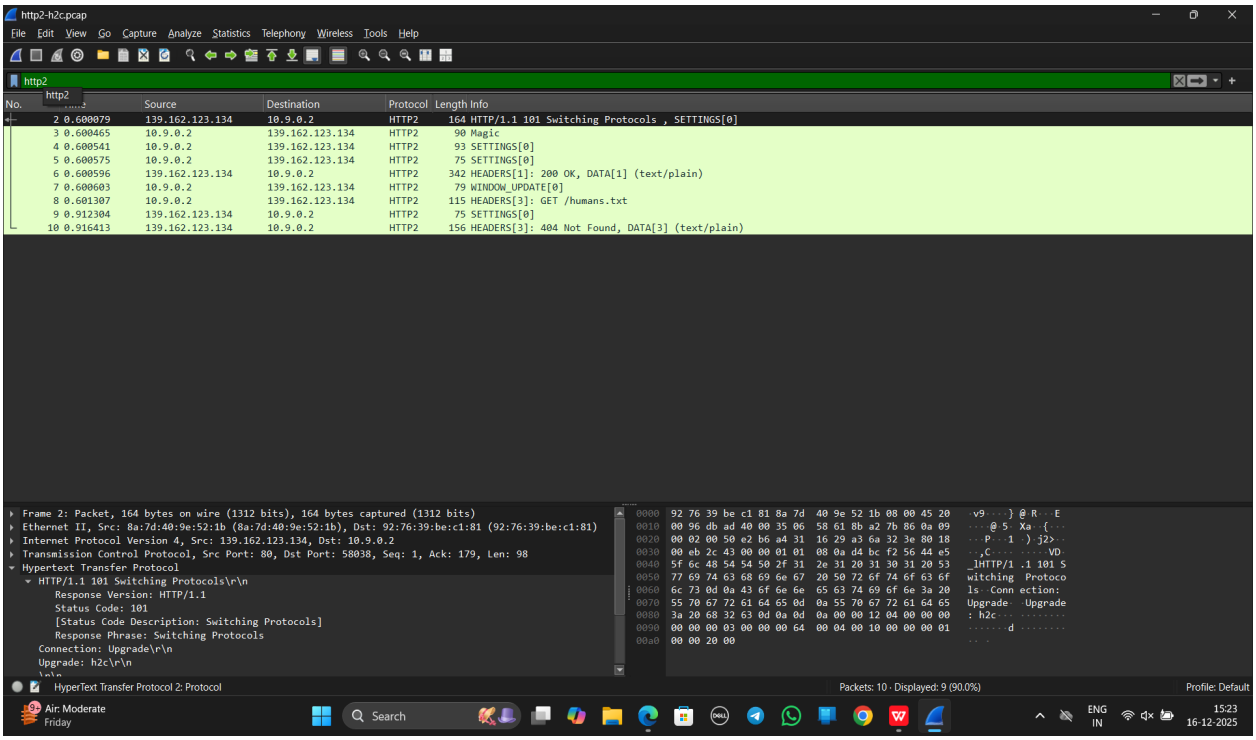
Subsequent HTTP request containing conditional headers such as If-Modified-Since.



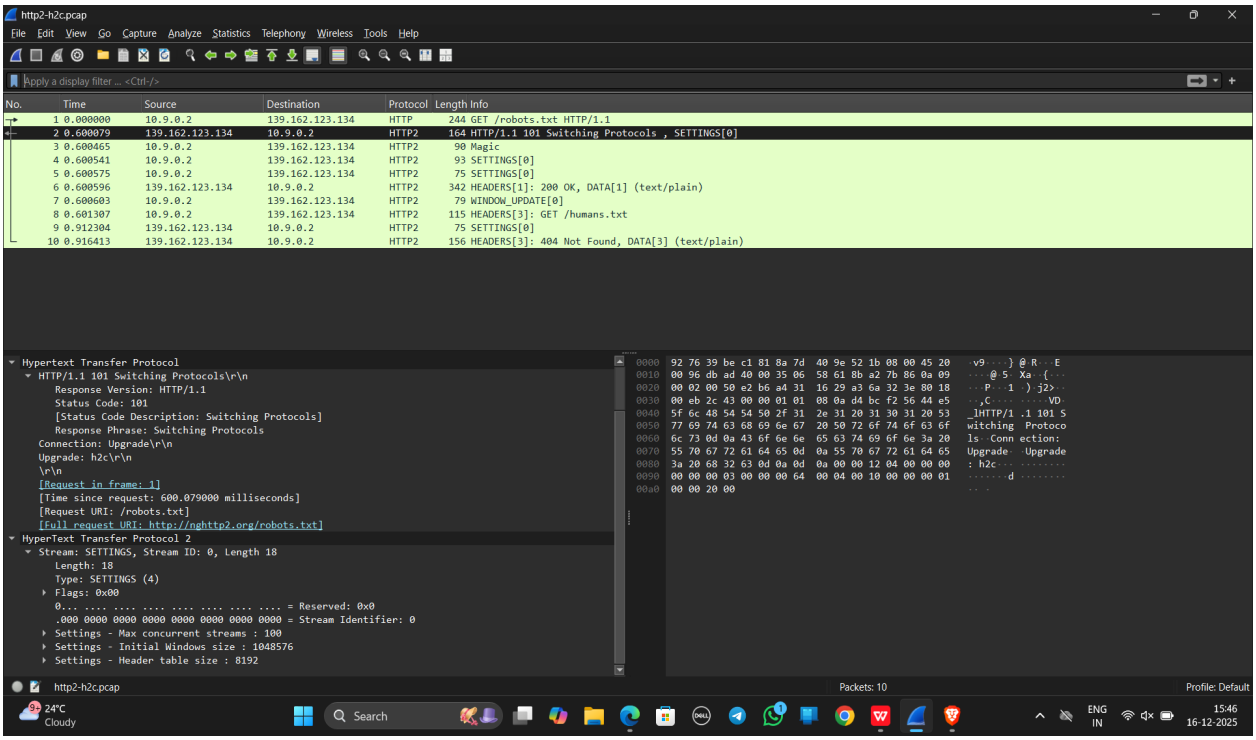
Server response indicating cache validation behavior during repeated access.

The difference between first and subsequent requests is governed by HTTP cache-control mechanisms. Conditional GET requests allow the server to respond with reduced data or status codes such as 304 Not Modified, improving performance.

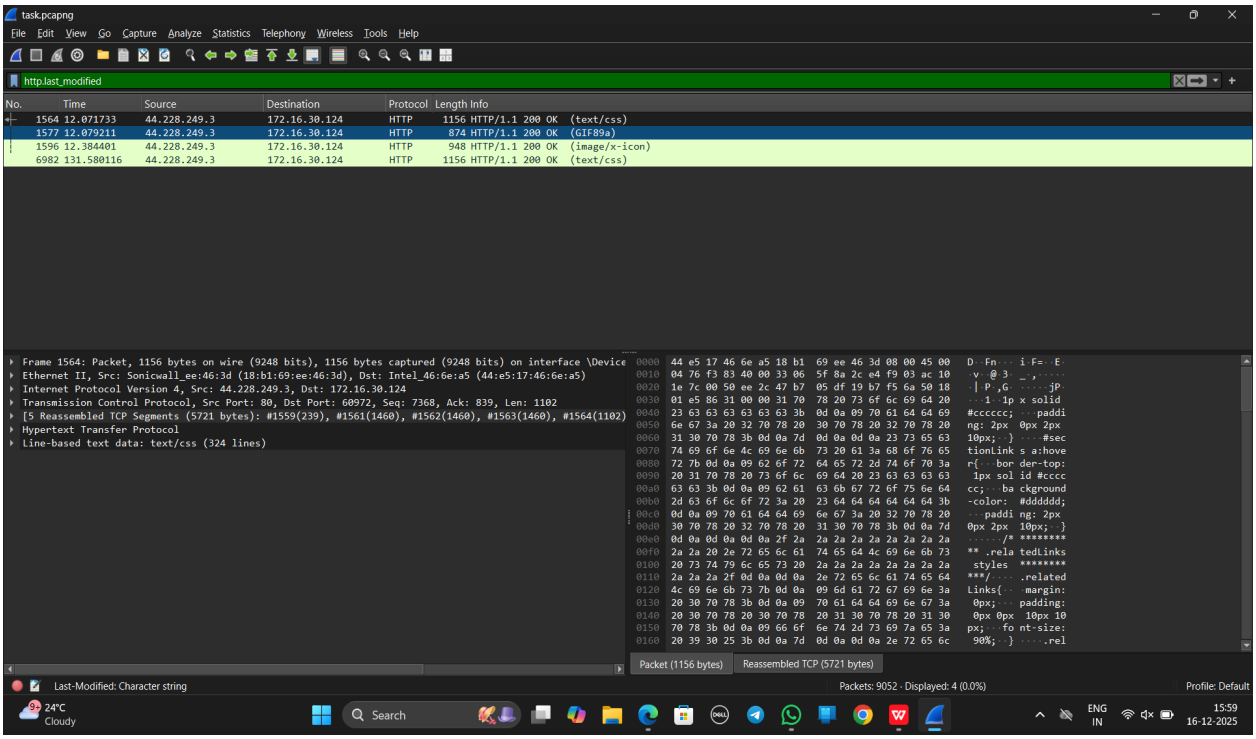
Task 3: HTTP/2 Protocol Analysis



HTTP/1.1 to HTTP/2 protocol upgrade using 101 Switching Protocols (h2c).



HTTP/2 SETTINGS and HEADERS frames exchanged between client and server.



HTTP/2 GET request and response demonstrating multiplexed data transfer.

HTTP/2 differs from HTTP/1.1 by using binary framing, header compression, and multiplexed streams, resulting in improved efficiency and reduced latency.