**Analysing Application layer protocols – HTTP, DNS**

**Tracing DNS and HTTP with Wireshark**

Now that we are familiar with nslookup and ipconfig, we’re ready to get down to some

serious business. Let’s first capture the DNS packets that are generated by ordinary Web surfing

activity.

• Use ipconfig to empty the DNS cache in your host.

* *ipconfig* is also very useful for managing the DNS information stored in your host. We have learned that a host can cache DNS records it recently obtained. To see these cached records, after the prompt provide the following command: ipconfig /displaydns
* Each entry shows the remaining Time to Live (TTL) in seconds. To clear the cache, enter

ipconfig /flushdns

* Flushing the DNS cache clears all entries and reloads the entries from the hosts file.

• Open your browser and empty your browser cache. (With Internet Explorer, go to Tools menu and select Internet Options; then in the General tab select Delete Files.)

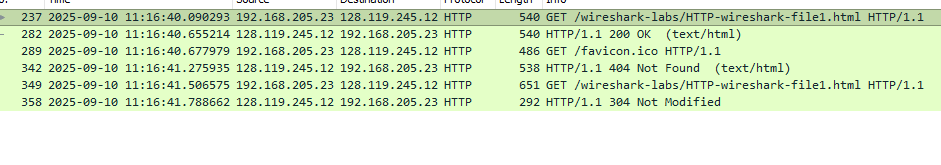
• Most of the web browsers perform object caching and thus perform a conditional GET when retrieving an HTTP object. Before performing the steps below, make sure your browser’s cache is empty. (To do this under Netscape 7.0, select *Edit->Preferences->Advanced->Cache* and clear the memory and disk cache. For Firefox, select *Tools->Clear Private Data*, or for Internet Explorer, select *Tools->Internet Options->Delete File;* these actions will remove cached files from your browser’s cache.)

Now do the following activity:

* Start up your web browser, and make sure your browser’s cache is cleared, as discussed above.
* Start up the Wireshark packet sniffer
* Enter the following to your browser  
  <http://gaia.cs.umass.edu/wireshark-labs/HTTP-wireshark-file1.html>  
  Your browser should display the very simple, one-line HTML file.
* Quickly enter the same URL into your browser again (or simply select the refresh button on your browser) to check the Conditional GET concept
* Stop Wireshark packet capture, and enter “http” in the display-filter-specification window, so that only captured HTTP messages will be displayed later in the packet-listing window.
* nslookup queries will generate DNS request and receive DNS response from local DNS server, which can be checked using ipconfig /all

You can also enter “ip.addr == your\_IP\_address” into the filter, where you obtain your\_IP\_address (the IP address for the computer on which you are running Wireshark) with ipconfig. This filter removes all packets that neither originate nor are destined to your host.

Explain the working of the DNS and HTTP protocol [DNS/HTTP Request and DNS/HTTP Response message] briefly with typed answers and answer highlighted screenshots for the above capture



* 1. Is your browser running HTTP version 1.0, 1.1, or 2? What version of HTTP is the server running?

Server running in 1.1 as well as browser running in 1.1





* 1. What languages (if any) does your browser indicate that it can accept to the server?

Yes



* 1. What is the IP address of your computer? What is the IP address of the gaia.cs.umass.edu server?



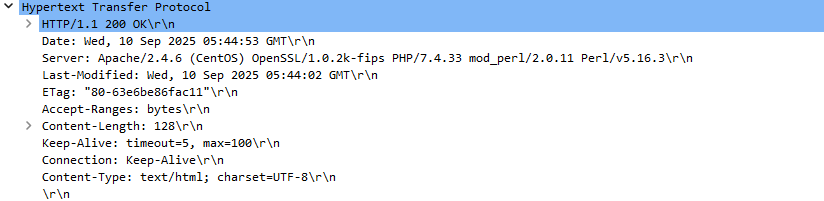
IP address of computer: 192.168.205.23

IP address of gaia: 128.119.245.12

* 1. What is the status code returned from the server to your browser?

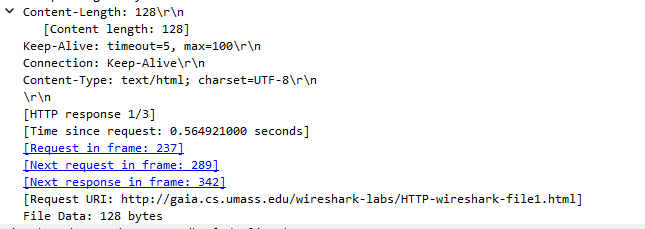


* 1. When was the HTML file that you are retrieving last modified at the server?



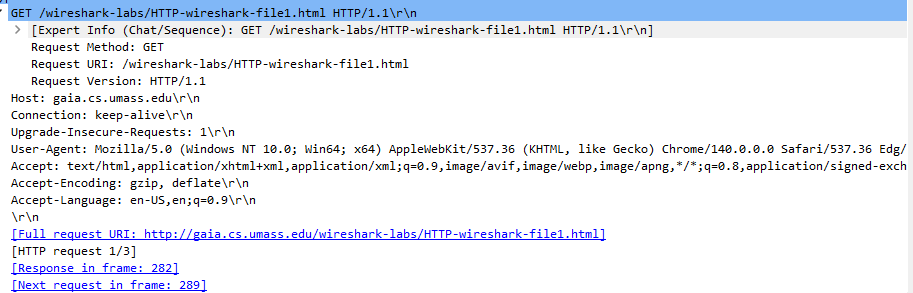
Last modified: wed, 10th September 2025

* 1. How many bytes of content are being returned to your browser in the first HTTP Response for first GET to gaia.cs.umass.edu server?

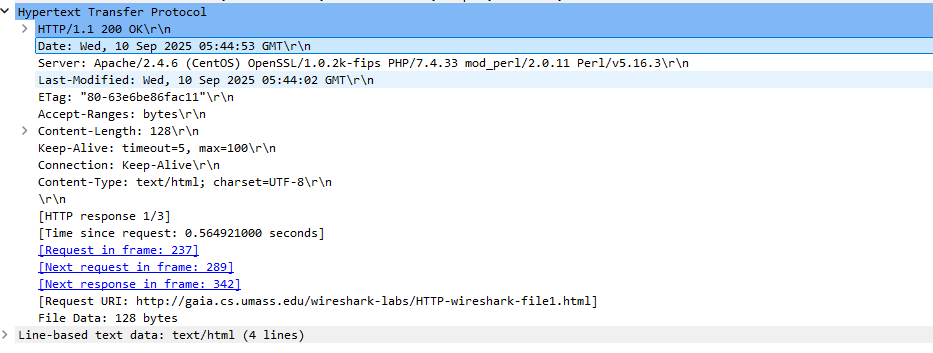


* 1. Inspect the contents of the first HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE” line in the HTTP GET?

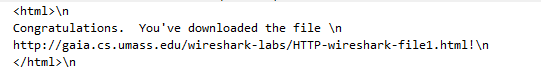
There is no IF MODIFIED SINCE present in the HTTP GET



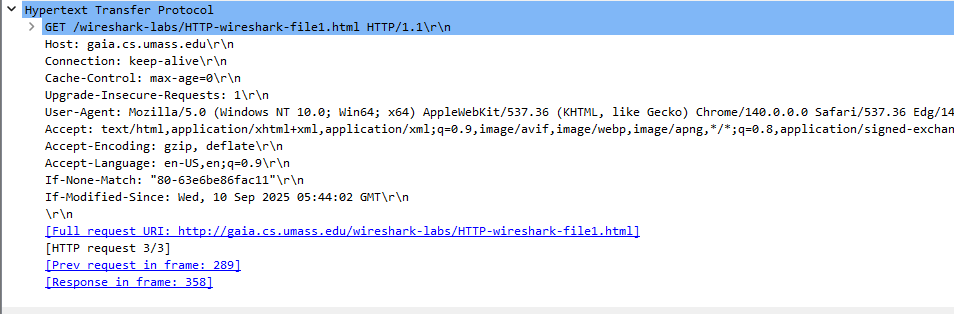
* 1. Inspect the contents of the server response. Did the server explicitly return the contents of the file? How can you tell?



The server explicitly returned the contents of the file. Line-based text data: is returned

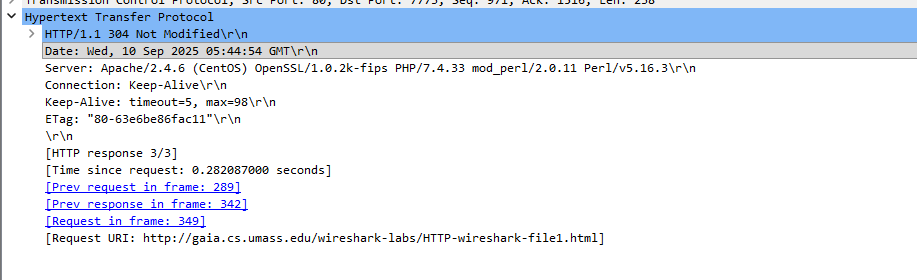


* 1. Now inspect the contents of the second HTTP GET request from your browser to the server. Do you see an “IF-MODIFIED-SINCE:” line in the HTTP GET[[1]](#footnote-1)? If so, what information follows the “IF-MODIFIED-SINCE:” header?



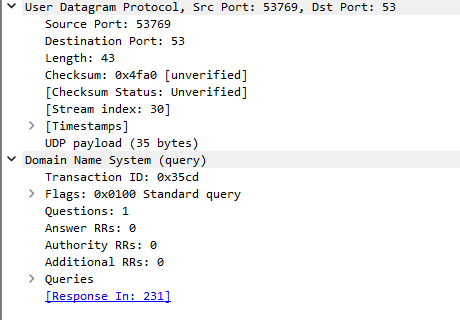
IF MODIFIED SINCE is present in the second request. Information following it is when it was last modified

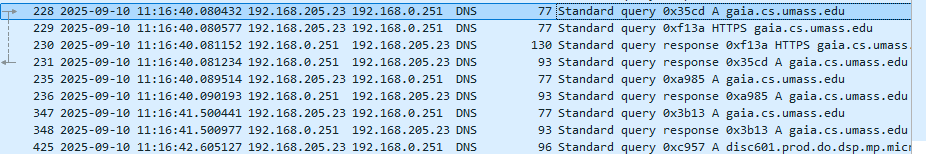
* 1. What is the HTTP status code and phrase returned from the server in response to this second HTTP GET? Did the server explicitly return the contents of the file? Explain.



The server didn’t return the contents of the file explicitly because the file isn’t modified. If the file is modified, then the contents would have been explicitly returned

* 1. Locate the DNS query and response messages. Are they sent over UDP or TCP?

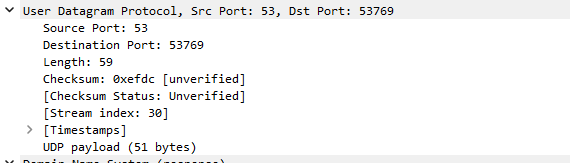




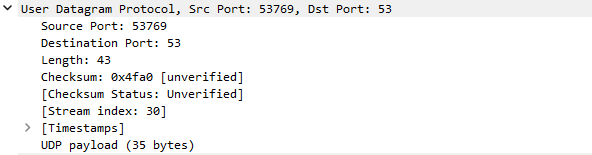
They are sent over UDP

* 1. What is the destination port for the DNS query message? What is the source port of DNS response message?

Response:



Source:



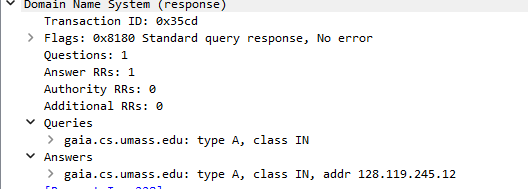
Both are 53

* 1. To what IP address is the DNS query message sent? Use ipconfig to determine the IP address of your local DNS server. Are these two IP addresses the same?



IP Address: 192.168.0.251

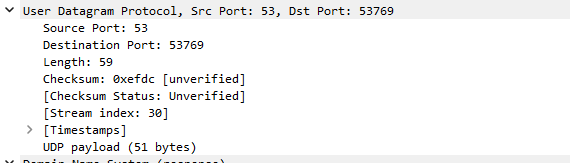
* 1. Examine the DNS response message. How many “answers” are provided? What do each of these answers contain?



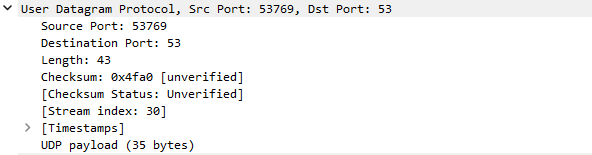
There is one question with one answer

* 1. What is the destination port for the DNS query message? What is the source port of DNS response message?

Response:



Source:



Both are 53

* 1. To what IP address is the DNS query message sent? Is this the IP address of your default local DNS server?



IP Address: 192.168.0.251



Yes, it is the IP address pf default local DNS server

* 1. Before retrieving each image/object in your web page, does your host issue new DNS queries?

No, it doesn’t

* 1. Before the HTTP how the connection is established?

Before an HTTP connection is established, the browser performs a DNS lookup to find the server’s IP address, then initiates a TCP three-way handshake by sending a SYN packet, receiving a SYN-ACK from the server and sending an ACK back to acknowledge the connection. This must be done before any HTTP request or HTTP response

1. [↑](#footnote-ref-1)