How does the communication between a browser and a web server take place?

Asked 15 years, 1 month ago Modified 5 years, 10 months ago

Viewed 108k times

41

Can anyone explain how the communication takes place between the browser and web server? I want to learn how

- GET, POST verbs (among others)
- cookies
- sessions
- query strings

work behind the scene.

http browser protocols

Share

Improve this question

Follow

edited Mar 9, 2011 at 22:07

David Victor

827 ● 9 ● 31

asked Nov 19, 2009 at 16:27

Madhan

2,639 ● 7 ● 27 ● 22

4 There are entire textbooks and college courses devoted to this subject. IF you are looking for a concise but in-depth answer, you are searching in vain. – Matthew Jones Nov 19, 2009 at 16:29

1 If you want a specific answer, you need to ask a specific question. Honestly, the appropriate answer to your question is: "(probably) http(s) over tcp". – William Pursell Nov 19, 2009 at 16:51

What answer would be considered non-technical? – Amarghosh Nov 19, 2009 at 17:00

8 Answers

Sorted by: Highest score (default)

69

Hyper Text Transfer Protocol (HTTP) is a protocol used for transferring web pages (like the one you're reading right now). A protocol is really nothing but a standard way of doing things. If you were to meet the President of the United States, or the king of a country, there would be specific procedures that you'd have to follow. You couldn't just walk up and say "hey dude". There would be a specific way to walk, to talk, a standard greeting, and a standard way to end the conversation. Protocols in the TCP/IP stack serve the same purpose.

The TCP/IP stack has four layers: Application, Transport, Internet, and Network. At each layer there are different protocols that are used to standardize the flow of information, and each one is a computer program (running on your computer) that's used to format the

information into a packet as it's moving down the TCP/IP stack. A packet is a combination of the Application Layer data, the Transport Layer header (TCP or UDP), and the IP layer header (the Network Layer takes the packet and turns it into a frame).

The Application Layer

...consists of all applications that use the network to transfer data. It does not care about how the data gets between two points and it knows very little about the status of the network. Applications pass data to the next layer in the TCP/IP stack and then continue to perform other functions until a reply is received. The Application Layer uses host names (like www.dalantech.com) for addressing. Examples of application layer protocols: Hyper Text Transfer Protocol (HTTP -web browsing), Simple Mail Transfer Protocol (SMTP -electronic mail), Domain Name Services (DNS -resolving a host name to an IP address), to name just a few.

The main purpose of the Application Layer is to provide a common command language and syntax between applications that are running on different operating systems -kind of like an interpreter. The data that is sent by an application that uses the network is formatted to conform to one of several set standards. The receiving computer can understand the data that is being sent even if it is running a different operating system than the

sender due to the standards that all network applications conform to.

The Transport Layer

...is responsible for assigning source and destination port numbers to applications. Port numbers are used by the Transport Layer for addressing and they range from 1 to 65,535. Port numbers from 0 to 1023 are called "well known ports". The numbers below 256 are reserved for public (standard) services that run at the Application Layer. Here are a few: 25 for SMTP, 53 for DNS (udp for domain resolution and tcp for zone transfers) , and 80 for HTTP. The port numbers from 256 to 1023 are assigned by the IANA to companies for the applications that they sell.

Port numbers from 1024 to 65,535 are used for client side applications -the web browser you are using to read this page, for example. Windows will only assign port numbers up to 5000 -more than enough port numbers for a Windows based PC. Each application has a unique port number assigned to it by the transport layer so that as data is received by the Transport Layer it knows which application to give the data to. An example is when you have more than one browser window running. Each window is a separate instance of the program that you use to surf the web, and each one has a different port number assigned to it so you can go to www.dalantech.com in one browser window and this site

does not load into another browser window. Applications like FireFox that use tabbed windows simply have a unique port number assigned to each tab

The Internet Layer

...is the "glue" that holds networking together. It permits the sending, receiving, and routing of data.

The Network Layer

...consists of your Network Interface Card (NIC) and the cable connected to it. It is the physical medium that is used to transmit and receive data. The Network Layer uses Media Access Control (MAC) addresses, discussed earlier, for addressing. The MAC address is fixed at the time an interface was manufactured and cannot be changed. There are a few exceptions, like DSL routers that allow you to clone the MAC address of the NIC in your PC.

For more info:

- Protocols
- TCP/IP

edited Mar 7, 2011 at 16:13

Greg

16.7k ● 9 ● 54 ● 100

answered Nov 19, 2009 at 16:45

OMG Ponies

332k ● 85 ● 534 ● 507

65535 = 2 ^ 16 - 1 is the last port. – Display Name Mar 3, 2011 at 4:16

1 Nice description. I think there is a problem with your layer names though. What you call Network Layer is called Link layer in TCP/IP (= combination of what OSI calls Data Link Layer and Physical Layer). This is software (eg, device driver) and hardware (eg, NIC + cabling). What you call Internet Layer is usually called Network layer and is the IP part of TCP/IP. – Andrew W. Phillips Oct 3, 2015 at 13:27

The *for more info* links are dead. – kleinfreund Jul 11, 2017 at 7:49

18

Your browser first resolves the servername via DNS to an IP. Then it opens a TCP connection to the webserver and tries to communicate via HTTP. Usually that is on TCP-port 80 but you can specify a different one (`http://server:portnumber`).

HTTP looks like this:

Once it is connected, it sends the request, which looks like:

```
GET /site HTTP/1.0
Header1: bla
Header2: blub
{emptyline}
```

E.g., a header might be `Authorization` or `Range`. See here for more.

Then the server responds like this:

```
200 OK
Header3: foo
Header4: bar

content following here...
```

E.g., a header might be `Date` or `Content-Type`. See here for more.

Look at Wikipedia for HTTP for some more information about this protocol.

Share Improve this answer Follow

edited Jun 22, 2013 at 13:14

answered Nov 19, 2009 at 16:32

Albert

67.9k ● 67 ● 251 ● 400

1 This is the answer I liked. Thank you. How can I read these responses? – Edward Mar 19, 2013 at 16:37

+1 Can you explain what is header1,header2 ..etc. I know its a very old post but it want to know it :). – Vivek Sadh Jun 22, 2013 at 12:54

16

+25

The links for specifications of each aspect of the question is as follows:

- **GET, POST verbs (among others)** - The HTTP Specification exhaustively discusses all aspects of HTTP communication (the protocol for communication between the web server and the browser). It explains the Request message and Response message protocols.
- **Cookies** - are set by attaching a `Set-Cookie` HTTP Header to the HTTP response.
- **QueryStrings** - are the part of the URL in the HTTP request that follow the first occurrence of a "?" character. The linked specification is for section 3.4 of the URI specification.
- **Sessions** - HTTP is a synchronous, stateless protocol. Sessions, or the illusion of state, can be created by (1) using cookies to store state data as plain text on the client's computer, (2) passing data-values in the URL and querystring of the request, (3) submitting POST requests with a collection of values that may indicate state and, (4) storing state information by a server-side persistence mechanism that is retrieved by a session-key (the session key is resolved from either the cookie, URL/Querystring or POST value collection.

An explanation of HTTP can go on for days, but I have attempted to provide a concise yet conceptually complete

answer, and include the appropriate links for further reading of official specifications.

Share Improve this answer

Follow

edited Oct 7, 2021 at 5:49

Community Bot

1 ● 1

answered Mar 7, 2011 at 16:08

smartcaveman

42.2k ● 31 ● 132 ● 217

8

Your browser is sitting on top of TCP/IP, as the web is based on standards, usually port 80, what happens is when you enter an address, such as google.com, your computer where the browser is running on, creates packets of data, encapsulated at each layer accordingly to the OSI standards, (think of envelopes of different sizes, packed into each envelope of next size), OSI defines 7 layers, in one of the envelopes contains the source address and destination address(that is the website) encoded in binary.

As it reaches the 1st layer, in OSI terms, it gets transmitted across the media transmitter (such as cable, DSL).

If you are connected via ISP, the layered pack of envelopes gets transmitted to the ISP, the ISP's network system, peeks through the layered pack of envelopes by decoding in reverse order to find out the address, then the ISP checks their Domain Name System database to

find out if they have a route to that address (cached in memory, if it does, it forwards it across the internet network - again layered pack of envelopes).

If it doesn't, the ISP interrogates the top level DNS server to say 'Hey, get me the route for the address as supplied by you, ie. the browser', the top level DNS server then passes the route to the ISP which is then stored in the ISP's server memory.

The layered pack of envelopes are transmitted and received by the website server after successful routing of the packets (think of routing as signposts for directions to get to the server), which in turn, unpacks the layered pack of envelopes, extracts the source address and says 'Aha, that is for me, right, I know the destination address (that is you, the browser), then the server packetizes the webpages into a packed layered envelopes and sends it back (usually in reverse route, but not always the case).

Your browser than receives the packetized envelopes and unpacks each of them. Then your computer descrambles the data and your browser renders the pages on the screen.

I hope this answer is sufficient enough for your understanding.

edited Feb 24, 2015 at 21:18

answered Nov 19, 2009 at 16:44

t0mm13b

34.6k ● 8 ● 84 ● 112

7

this is from Browser — Server Communication

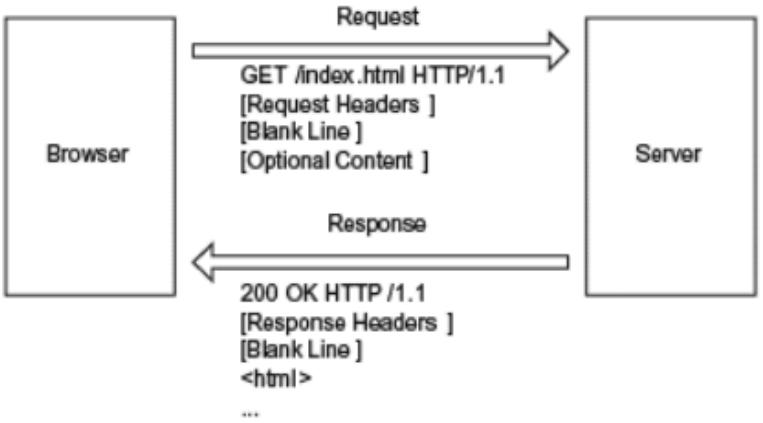

Figure 1.1 The request and response have specific formats, as specified by the HTTP protocol.

When the browser makes the request, it mentions the protocol that it is using: HTTP/1.1. When the server sends the response, it also identifies the protocol it is using: HTTP/1.1. A protocol is not a language; it is a set of rules that must be followed. For instance, one rule in HTTP is that the first line of a request will contain the type of request, the address of the page on the server and the version of the protocol that the browser is using. Another rule is that the first line of the response will contain a numeric code indicating the success of the request, a sentence describing the code and the version of the protocol that the server is using.

Protocols are used in many places, not just with computers. When the leaders of two countries meet, they must decide on a common protocol in order to communicate. Do they bow or shake hands when they meet? Do they eat with chopsticks or silverware? It is the same situation for computers, in order for the browser and server to communicate, they must decide on a common protocol.

Share Improve this answer

Follow

answered Mar 10, 2011 at 3:26

edgarmtze

25k ● 81 ● 246 ● 390

4

It depends on the web server, but if you're wondering what it looks like from the client side, just install Live Headers and Firebug for firefox. With the net tab in firebug and live headers open, it should be clear exactly how the two interact.

For a more in-depth look at the actual data going back and forth, use wireshark.

Share Improve this answer

Follow

answered Nov 19, 2009 at 16:30

Stefan Kendall

67.7k ● 69 ● 257 ● 409

3

There is a commercial product with an interesting logo which lets you see all kind of traffic between server and client named charles.

Another open source tools include: Live HttpHeaders, Wireshark or Firebug.

Share Improve this answer

Follow

answered Nov 19, 2009 at 16:37

miku

188k ● 47 ● 312 ● 317

2

Communication between a browser and a webserver takes place at so many levels that is close to impossible to answer this question. HTTP plays a role, but HTTP is meaningless without TCP which is meaningless without IP which is meaningless without a physical network on which it sent. Then, there are POST vs GET requests which are similar but enough different to warrant a special dicussion. Sometimes an HTTP request needs to be authenticated, sometimes, it needs not. Mime types should be mentioned. Then, a browser sends a different request if there is a proxy. And then also encodings play a

role. So, I guess, the most concise answer to this kind of question is: the browser asks the server for data and the server gives the requested data to the browser.

Share Improve this answer

Follow

answered Nov 19, 2009 at 16:44

René Nyffenegger
40.4k ● 34 ● 176 ● 312