HTTP - Overview

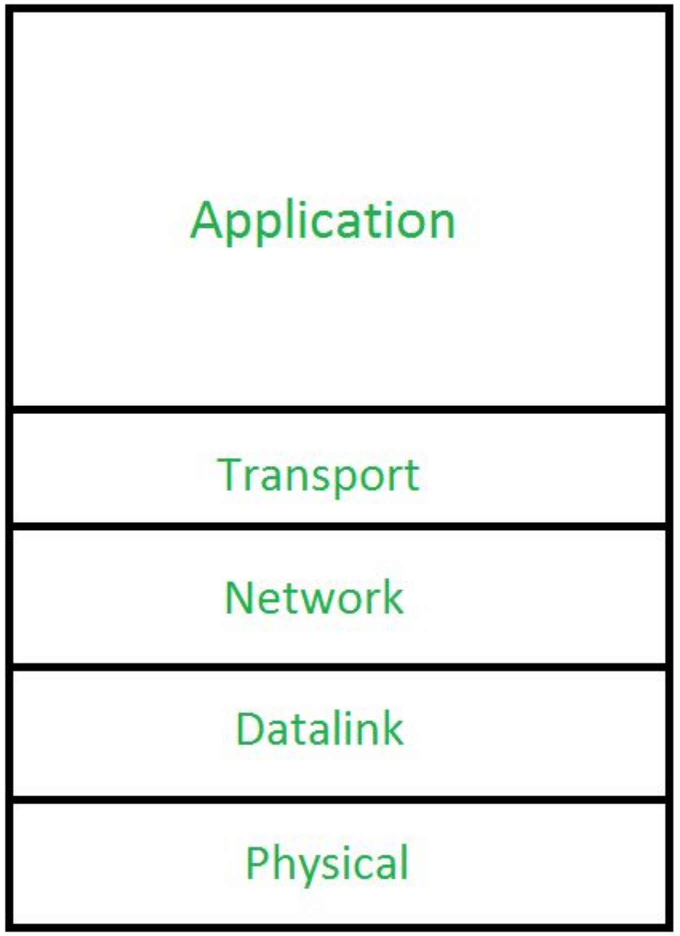
Overview of TCP/IP

- TCP/IP short for Transmission Control Protocol/ Internet Protocol, is a communication protocols suite means a set of rules and procedures which are used for interconnecting various network devices over the internet by defining how the data should be transmitted, routed, broken into packets, addressed, and received at the destination.

How does TCP/IP works?

- The functionality of TCP/IP is divided into five layers -1)Physical Layer, Data Link Layer, Network Layer, Transport Layer, Application Layer.

LAYERED ARCHITECTURE



TCP/IP Protocols

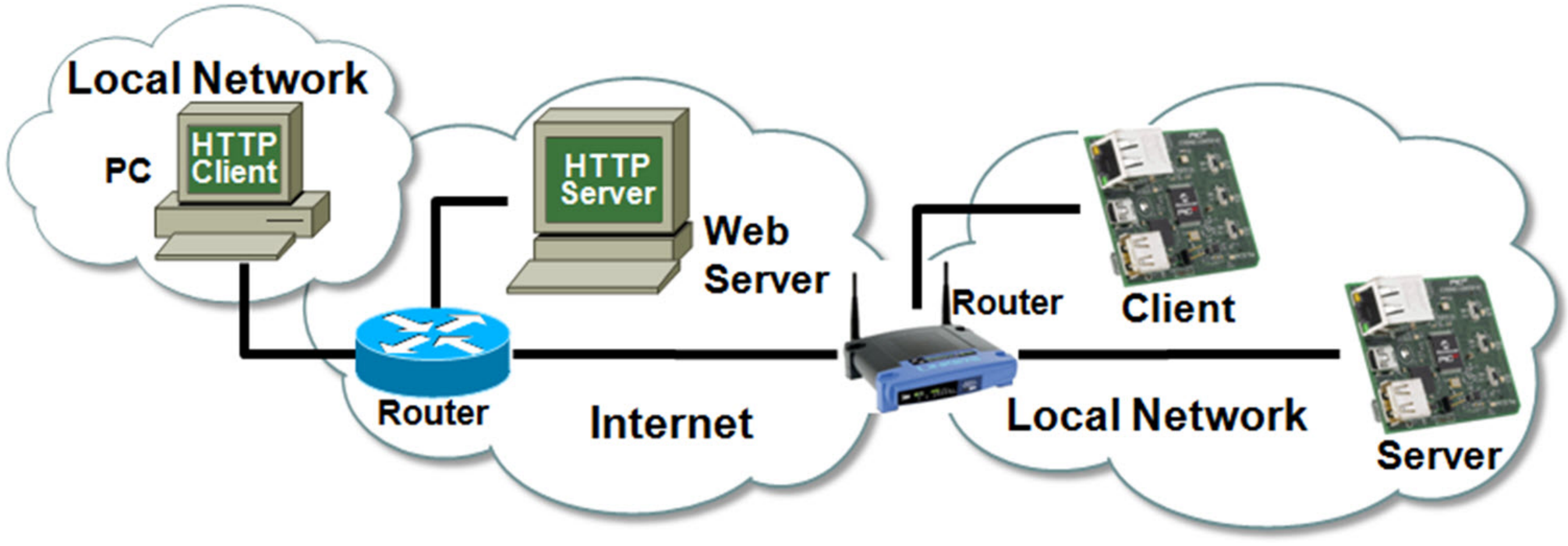
- Below is the list of commonly used TCP/IP protocols:-
- **HTTP**
- HTTP stands for **HyperText Transfer Protocol**. HTTP establishes a connection between client and server for data transmission. It is a non-secure transmission. A client sends a request to the server through a web browser to view specified information. After receiving a request, the server sends specified information to the client.
-

- **HTTPS**
- HTPPS stands for HyperText Transfer Protocol Secure. HTTPS establishes a connection between the client and the server for data transmission. It is a secure transmission. The client mainly uses this HTTPS to send private information like credit card details, online transactions, etc to the server across the internet connection.
- **FTP**
- FTP Stands for File Transfer Protocol. It uses TCP services to transfer files from one host to another. It establishes a connection between two hosts. after the connection is established, the host can send and receive data or files.

- **SMTP**
- SMTP stands for Simple Mail Transfer Protocol. It is a widely used and most important protocol used to transfer emails from sender to receiver. It is an application layer protocol as we have seen previously. It is a Push protocol that is used to send an email. After that, POP post office protocol) or IMAP (internet message access protocol) protocols retrieve emails on the receiver end.

Client-Server Model

- The **client-server** programming model is a distributed computing architecture that segregates information users (clients) from information providers (servers).
- A **client** is an application that needs something like a web page or IP address from a server. Clients may contact a server for this information at any time. Clients are information users.
- A **server** is an application that provides information or resources to clients. It needs to be always up and running, waiting for requests from clients.


Local Network
HTTP Client
PC
HTTP Server
Web Server
Router
Internet
Router
Client
Local Network
Server

- The Hypertext Transfer Protocol (HTTP) is an application-level protocol for distributed, collaborative, hypermedia information systems. This is the foundation for data communication for the World Wide Web (i.e. internet) since 1990. HTTP is a generic and **stateless protocol** which can be used for other purposes as well using extensions of its request methods, error codes, and headers.

Basic Features

- **HTTP is connectionless:** The HTTP client, i.e., a browser initiates an HTTP request and after a request is made, the client waits for the response. The server processes the request and sends a response back after which client disconnect the connection. So client and server knows about each other during current request and response only. Further requests are made on new connection like client and server are new to each other.

- **HTTP is media independent:** It means, any type of data can be sent by HTTP as long as both the client and the server know how to handle the data content. It is required for the client as well as the server to specify the content type using appropriate MIME-type.
- **HTTP is stateless:** As mentioned above, HTTP is connectionless and it is a direct result of HTTP being a stateless protocol. The server and client are aware of each other only during a current request. Afterwards, both of them forget about each other. Due to this nature of the protocol, neither the client nor the browser can retain information between different requests across the web pages.

HTTP Version

- HTTP uses a **<major>.<minor>**
- Example
- HTTP/1.0 or HTTP/1.1

Uniform Resource Identifiers (URI)

- URI = "http:" "//" host [":" port] [abs_path ["?" query]]

- http://abc.com:80/smith/home.html

HTTP - Messages

- HTTP is based on the client-server architecture model and a stateless request/response protocol that operates by exchanging messages across a reliable TCP/IP connection.
- Two parts- message header and message body.

Two HTTP Request Methods: GET and POST

- Two commonly used methods for a request-response between a client and server are: GET and POST.
- **GET** - Requests data from a specified resource
- **POST** - Submits data to be processed to a specified resource
- HEAD-Same as GET but returns only HTTP headers and no document body
- PUT-Uploads a representation of the specified URI
- DELETE-Deletes the specified resource
- OPTIONS-Returns the HTTP methods that the server supports
- CONNECT-Converts the request connection to a transparent TCP/IP tunnel

Request Model

- Request-Line
- The Request-Line begins with a method token, followed by the Request-URI and the protocol version, and ending with CRLF. The elements are separated by space SP characters.
- Request-Line = Method SP Request-URI SP HTTP-Version CRLF Let's discuss each of the parts mentioned in the Request-Line.
- Request Method
- The request **method** indicates the method to be performed on the resource identified by the given **Request-URI**. The method is case-sensitive and should always be mentioned in uppercase. The following table lists all the supported methods in HTTP/1.1.

HTTP - Requests

- Examples of Request Message
- Now let's put it all together to form an HTTP request to fetch **hello.htm** page from the web server running on tutorialspoint.com
- [illegible]

```
GET /hello.htm HTTP/1.1
User-Agent: Mozilla/4.0 (compatible; MSIE5.01; Windows NT)
Host: www.tutorialspoint.com
Accept-Language: en-us
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
```

HTTP Responses

- Status Code
- The Status-Code element is a 3-digit integer where first digit of the Status-Code defines the class of response and the last two digits do not have any categorization role. There are 5 values for the first digit:
- **1xx: Informational** It means the request was received and the process is continuing.
- **2xx: Success** It means the action was successfully received, understood, and accepted.
- **3xx: Redirection** It means further action must be taken in order to complete the request.
- **4xx: Client Error** It means the request contains incorrect syntax or cannot be fulfilled.
- **5xx: Server Error** It means the server failed to fulfill an apparently valid request.

HTTP - Responses

```
HTTP/1.1 200 OK
Date: Mon, 27 Jul 2009 12:28:53 GMT
Server: Apache/2.2.14 (Win32)
Last-Modified: Wed, 22 Jul 2009 19:15:56 GMT
Content-Length: 88
Content-Type: text/html
Connection: Closed
```

```
<html>
<body>
<h1>Hello, World!</h1>
</body>
</html>
```

The following example shows an HTTP response message displaying error condition when the web server could not find the requested page:

```
HTTP/1.1 404 Not Found
Date: Sun, 18 Oct 2012 10:36:20 GMT
Server: Apache/2.2.14 (Win32)
Content-Length: 230
Connection: Closed
Content-Type: text/html; charset=iso-8859-1
```

```
<!DOCTYPE HTML PUBLIC "-//IETF//DTD HTML 2.0//EN">
<html>
<head>
<title>404 Not Found</title>
</head>
<body>
<h1>Not Found</h1>
<p>The requested URL /t.html was not found on this
server.</p>
</body>
</html>
```

Please go through this

- https://www.tutorialspoint.com/http/http_responses.htm