1. application architecture - designed by the application developer and dictates how the application is structured over the various end systems.
2. client-server - an always-on host, called the server, which services requests from many other hosts, called clients.
3. data center - housing many hosts, is often used to create a powerful virtual server.
4. P2P - direct communication between pairs of intermittently connected hosts, called peers.
5. self-scalability - property of a system to handle a growing amount of work by adding resources.
6. ISP friendly – file distribution that ISP’s agree with.
7. P2P security -
8. P2P incentives
9. Messages – how two different end systems communicate with each other across the computer network.
10. API - interface between the application layer and the transport layer within a host.
11. IP address – how a host is identified on the internet.
12. port number - a way to identify a specific process to which an Internet or other network message is to be forwarded when it arrives at a server.
13. reliable data transfer - guaranteed data delivery.
14. loss-tolerant applications - multimedia applications that can tolerate some amount of data loss.
15. bandwidth-sensitive apps - Applications that have throughput requirements.
16. elastic applications - can make use of as much, or as little, throughput as happens to be available.
17. TCP connection - After the handshaking phase, a TCP connection is said to exist between the sockets of the two processes.
18. UDP - lightweight transport protocol, providing minimal services. UDP is connectionless, so there is no handshaking before the two processes start to communicate. UDP provides an unreliable data transfer service.
19. application-layer protocol - defines how an application’s processes, running on different end systems, pass messages to each other.
20. HTTP - the Web’s application-layer protocol. implemented in two programs: a client program and a server program.
21. web page – documents that consists of objects.
22. HTML file – a webpage coded in html that can be displayed in a web browser.
23. web browser - implement the client side of HTTP.
24. web server - implement the server side of HTTP.
25. stateless protocol - maintains no information about the clients.
26. non-persistent connection - each TCP connection is closed after the server sends the object—the connection does not persist for other objects.
27. persistent connection - the server leaves the TCP connection open after sending a response.
28. RTT - the time it takes for a small packet to travel from client to server and then back to the client.
29. request message – a message sent by an http client to an http server.
30. request line - The first line of an HTTP request message. The request line has three fields: the method field, the URL field, and the HTTP version field.
31. header lines – lines after the request line that let the client and server pass additional info.
32. response message – a response returned by the http server to a http client.
33. status line - has three fields: the protocol version field, a status code, and a corresponding status message.
34. header line
35. entity body - the meat of the message—it contains the requested object itself.
36. Cookies - allow sites to keep track of users.
37. set-cookie – a header which contains the identification number.
38. web cache - also called a proxy server—is a network entity that satisfies HTTP requests on the behalf of an origin Web server.
39. proxy server – web cache.
40. conditional GET - a mechanism that allows a cache to verify that its objects are up to date.
41. if-modified-since - a HTTP header that is sent to a server as a conditional request. If the content has not changed the server responds with a 304 status code. If the content has changed the server responds with a 200 status code.
42. control connection - the client sends commands regarding session state changes.
43. data connection -
44. out-of-band - data transferred through a stream that is independent from the main in-band data stream.
45. in-band - involves managing devices through the common protocols such as telnet or SSH, using the network itself as a media.
46. State – an instance of a network.
47. USER – username, Used to send the user identification to the server
48. PASS – password, Used to send the user password to the server
49. LIST – Used to ask the server to send back a list of files
50. RETR - filename: Used to retrieve (that is, get) a file
51. STOR - filename: Used to store a file
52. user agents - software that is acting on behalf of a user.
53. mail servers – house user mailboxes
54. SMTP - e-mail’s principal application-layer protocol. provides for the transfer of e-mail messages).
55. Mailbox – a location in one of the mail servers.
56. message queue -where messages are held.
57. SMTP handshake - SMTP clients and servers introduce themselves before transferring information. the SMTP client indicates the email address of the sender (the person who generated the message) and the e-mail address of the recipient.
58. HELO -­ SMTP command sent by an e-mail client when connecting to an e-mail server.
59. MAIL FROM – specify sender
60. RCPT TO – specify recipients.
61. DATA – specify the body
62. QUIT – terminates conversation with server
63. mail message formats
64. From - The email address of the sender.
65. To - The email address of the recipient.
66. Subject - A short summary of what the message is about
67. POP3 - A mail access protocol used to transfer mail from the recipient's mail server to the recipient's user agent.
68. IMAP - A server that will associate each message with a folder
69. port number -
70. socket - One endpoint of a two-way communication link between two programs running on the network.
71. UDP - connectionless and sends independent packets of data from one end system to the
72. other, without any guarantees about delivery
73. socket() - The first parameter indicates the address family. The second parameter indicates the type of socket
74. sendto() - attaches the destination address (serverName, serverPort) to the message and sends the resulting packet into the process's socket
75. recvfrom() - extract the clientside (source) port number from the segment it receives from the client
76. close() - closes the socket and terminates the process
77. bind() - assigns) the port number 12000 to the server's socket.
78. connect() - initiates the TCP connection between the client and server.
79. send() - sends the string sentence through the client's socket and into the
80. TCP connection - a full-duplex connection in that two processes can send messages to each other over the connection at the same time.
81. recv() - receives characters from the server
82. listen() - server listens for TCP connection requests from the client.
83. accept() - accepts a TCP/UDP connection.