CHAPTER - 9

NETWORK SERVERS

SERVERS (HTTP, DHCP, SMTP, DNS, PROXY, FTP):

1. HTTP:

HTTP is a request/response standard between a client and a server. A client is the end-user, the server is the web site. The client making an HTTP request using a web browser, spider, or other end-user tool – is referred to as the user agent. The responding server – which stores or creates resources such as HTML files and images is called the origin server. In between the user agent and origin server may be several intermediaries, such as proxies, gateways, and tunnels. HTTP is not constrained to using TCP/IP and its supporting layers, although this is its most popular application on the Internet. Indeed HTTP can be "implemented on top of any other protocol on the Internet, or on other networks. HTTP only presumes a reliable transport; any protocol that provides such guarantees can be used."

Typically, an HTTP client initiates a request. It establishes a Transmission Control Protocol (TCP) connection to a particular port on a host (port 80 by default ;). An HTTP server listening on that port waits for the client to send a request message. Upon receiving the request, the server sends back a status line, such as "HTTP/1.1 200 OK", and a message of its own, the body of which is perhaps the requested file, an error message, or some other information.

The reason that HTTP uses TCP and not UDP is because much data must be sent for a webpage, and TCP provides transmission control, presents the data in order, and provides error correction. See the difference between TCP and UDP.

Resources to be accessed by HTTP are identified using Uniform Resource Identifiers (URIs) (or, more specifically, Uniform Resource Locators (URLs)) using the http: or https URI schemes.

2. HTTPS:

(Hypertext Transfer Protocol over Secure Socket Layer) is a URI scheme used to indicate a secure HTTP connection. It is syntactically identical to the http:// scheme normally used for accessing resources using HTTP. Using an https: URL indicates that HTTP is to be used, but with a different default TCP port (443) and an additional encryption/authentication layer between the HTTP and TCP. This system was designed by Netscape Communications Corporation to provide authentication and encrypted communication and is widely used on the World Wide Web for security-sensitive communication such as payment transactions and corporate logos.

3. DHCP:

The Dynamic Host Configuration Protocol (DHCP) automates the assignment of IP addresses, subnet masks, default gateway, and other IP parameters.

When a DHCP configured client (be it a computer or any other network aware device) connects to a network, the DHCP client sends a broadcast query requesting necessary information from a DHCP server. The DHCP server manages a pool of IP addresses and information about client configuration parameters such as the default gateway, the domain name, the DNS servers, other servers such as time servers, and so forth. Upon receipt of a valid request the server will assign the computer an IP address, a lease (the length of time for which the allocation is valid), and other TCP/IP configuration parameters, such as the subnet mask and the default gateway. The query is typically initiated immediately after booting and must be completed before the client can initiate IP-based communication with other hosts.

DHCP provides three modes for allocating IP addresses. The best-known mode is dynamic, in which the client is provided a "lease" on an IP address for a period of time. Depending on the stability of the network, this could range from hours (a wireless network at an airport) to months (for desktops in a wired lab). At any time before the lease expires, the DHCP client can request renewal of the lease on the current IP address. A properly-functioning client will use the renewal mechanism to maintain the same IP address throughout its connection to a single network, otherwise it may risk losing its lease while still connected, thus disrupting network connectivity while it renegotiates with the server for its original or a new IP address.

The two other modes for allocation of IP addresses are automatic (also known as DHCP Reservation), in which the address is permanently assigned to a client, and manual, in which the address is selected by the client (manually by the user or any other means) and the DHCP protocol messages are used to inform the server that the address has been allocated.

The automatic and manual methods are generally used when finer-grained control over IP address is required (typical of tight firewall setups), although typically a firewall will allow access to the range of IP addresses that can be dynamically allocated by the DHCP server.

Depending on implementation, the DHCP server has three methods of allocating IP-addresses Dynamic Allocation: A network administrator assigns a range of IP addresses to DHCP, and each client computer on the LAN has its IP software configured to request an IP address from the DHCP server during network initialization. The request-and-grant process uses a lease concept with a controllable time period, allowing the DHCP server to reclaim (and then reallocate) IP addresses that are not renewed (dynamic re-use of IP addresses).

- **Automatic Allocation:** The DHCP server permanently assigns a free IP address to a requesting client from the range defined by the administrator.
- **Manual Allocation:** The DHCP server allocates an IP address based on a table with MAC address IP address pairs manually filled in by the server administrator. Only requesting clients with a MAC address listed in this table will be allocated an IP address.

Some DHCP server software can manage hosts by more than one of the above methods. For example, the known hosts on the network can be assigned an IP address based on their MAC address (manual allocation) whereas "guest" computers (such as laptops via Wi-Fi) are allocated a temporary address out of a pool compatible with the network to which they're attached (dynamic allocation).

4. DNS:

The Domain Name System (DNS) associates various sorts of information with domain names; most importantly, it serves as the "phone book" for the Internet by translating human-readable computer hostnames, e.g. http://www.example.com, into the IP addresses, e.g. 208.77.188.166, that networking equipment needs to deliver information. It also stores other information such as the list of mail exchange servers that accept email for a given domain. In providing a worldwide keyword-based redirection service, the Domain Name System is an essential component of contemporary Internet use.

The most basic task of DNS is to translate hostnames to IP addresses. In very simple terms, it can be compared to a phone book. DNS also has other important uses.

Above all, DNS makes it possible to assign Internet names to organizations (or concerns they represent), independently of the physical routing hierarchy represented by the numerical IP address. Because of this, hyperlinks and Internet contact information can remain the same, whatever the current IP routing arrangements may be, and can take a human-readable form (such as "example.com"), which is easier to remember than the IP address 208.77.188.166. People take advantage of this when they recite meaningful URLs and e-mail addresses without caring how the machine will actually locate them.

The Domain Name System distributes the responsibility for assigning domain names and mapping them to IP networks by allowing an authoritative server for each domain to keep track of its own changes, avoiding the need for a central registrar to be continually consulted and updated.

Parts of a Domain Name:

A domain name usually consists of two or more parts (technically labels), separated by dots. For example example.com.

The rightmost label conveys the top-level domain (for example, the address http://www.example.com has the top-level domain com).

Each label to the left specifies a subdivision, or subdomain of the domain above it. Note: "subdomain" expresses relative dependence, not absolute dependence. For example: example.com comprises a subdomain of the com domain, and http://www.example.com comprises a subdomain of the domain example.com. In theory, this subdivision can go down to 127 levels deep. Each label can contain up to 63 characters. The whole domain name does not exceed a total length of 255 characters. In practice, some domain registries may have shorter limits.

A hostname refers to a domain name that has one or more associated IP addresses; ie: the http://www.example.com and example.com domains are both hostnames, however, the com domain is not.

DNS servers:

The Domain Name System consists of a hierarchical set of DNS servers. Each domain or subdomain has one or more authoritative DNS servers that publish information about that domain and the name servers of any domains "beneath" it. The hierarchy of authoritative DNS servers matches the hierarchy of domains. At the top of the hierarchy stand the root name servers: the servers to query when looking up (resolving) a top-level domain name (TLD).

DNS resolvers

A resolver looks up the resource record information associated with nodes. A resolver knows how to communicate with name servers by sending DNS queries and heeding DNS responses.

A DNS query may be either a recursive query or a non-recursive query:

A non-recursive query is one where the DNS server may provide a partial answer to the query (or give an error). DNS servers must support non-recursive queries.

A recursive query is one where the DNS server will fully answer the query (or give an error). DNS servers are not required to support recursive queries.

The resolver (or another DNS server acting recursively on behalf of the resolver) negotiates use of recursive service using bits in the query headers.

Resolving usually entails iterating through several name servers to find the needed information. However, some resolvers function simplistically and can only communicate with a single name server. These simple resolvers rely on a recursive query to a recursive name server to perform the work of finding information for them.

5. FTP:

In computing, the File Transfer Protocol (FTP) (Port 21) is a network protocol used to transfer data from one computer to another through a network, such as over the Internet.

FTP is a file transfer protocol for exchanging files over any TCP/IP based network to manipulate files on another computer on that network regardless of which operating systems are involved (if the computers permit FTP access). There are many existing FTP client and server programs. FTP servers can be set up anywhere between game servers, voice servers, internet hosts, and other physical servers.

6. SMTP:

SMTP is a relatively simple, text-based protocol, in which one or more recipients of a message are specified (and in most cases verified to exist) along with the message text and possibly other encoded objects. The message is then transferred to a remote server using a procedure of queries and responses between the client and server. Either an end-user's email client, a.k.a. MUA (Mail User Agent), or a relaying server's MTA (Mail Transport Agents) can act as an SMTP client.

An email client knows the outgoing mail SMTP server from its configuration. A relaying server typically determines which SMTP server to connect to by looking up the MX (Mail eXchange) DNS record for each recipient's domain name (the part of the email address to the right of the at (@) sign). Conformant MTAs (not all) fall back to a simple A record in the case of no MX. Some current mail transfer agents will also use SRV records, a more general form of MX, though these are not widely adopted. (Relaying servers can also be configured to use a smart host.)

The SMTP client initiates a TCP connection to server's port 25 (unless overridden by configuration). It is quite easy to test an SMTP server using the telnet program (see below).

SMTP is a "push" protocol that does not allow one to "pull" messages from a remote server on demand. To do this a mail client must use POP3 or IMAP. Another SMTP server can trigger a delivery in SMTP using ETRN.

7. Proxy Server:

In computer networks, a proxy server is a server (a computer system or an application program) which services the requests of its clients by forwarding requests to other servers. A client connects to the proxy server, requesting some service, such as a file, connection, web page, or other resource, available from a different server. The proxy server provides the resource by connecting to the specified server and requesting the service on behalf of the client. A proxy server may optionally alter the client's request or the server's response, and sometimes it may serve the request without contacting the specified server. In this case, it would 'cache' the first request to the remote server, so it could save the information for later, and make everything as fast as possible.

A proxy server that passes all requests and replies unmodified is usually called a gateway or sometimes tunneling proxy.

A proxy server can be placed in the user's local computer or at specific key points between the user and the destination servers or the Internet.

Caching Proxy Server:

A proxy server can service requests without contacting the specified server, by retrieving content saved from a previous request, made by the same client or even other clients. This is called caching.

Web Proxy:

A proxy that focuses on WWW traffic is called a "web proxy". The most common use of a web proxy is to serve as a web cache.

Content Filtering Web Proxy:

A content filtering web proxy server provides administrative control over the content that may be relayed through the proxy. It is commonly used in commercial and non-commercial organizations (especially schools) to ensure that Internet usage conforms to acceptable use policy.

8. Client - Server:

Client server is a computing architecture which separates a client from a server, and is almost always implemented over a computer network. A client-server application is a distributed system that constitutes of both client and server software. A client is a software or process that may initiate a communication session, while a server can not initiate sessions, but is waiting for a requests from a client. Client and server may also aim at the host computer hardware connected to a network that are residing the client and server software respectively.

Client/server describes the relationship between two computer programs in which one program, the client, makes a service request from another program, the server, which fulfills the request. Although the client/server idea can be used by programs within a single computer, it is a more important idea in a network. In a network, the client/server model provides a convenient way to interconnect programs that are distributed efficiently across different locations. Computer transactions using the client/server model are very common. Most Internet applications, such as email, web access and database access, are based on the client/server model. For example, a web browser is a client program at the user computer that may access information at any web server in the world. To check your bank account from your computer, a web browser client program in your computer forwards your request to a web server program at the bank. That program may in turn forward the request to its own database client program that sends a request to a database server at another bank computer to retrieve your account balance. The balance is returned back to the bank database client, which in turn serves it back to the web browser client in your personal computer, which displays the information for you.

The client/server model has become one of the central ideas of network computing. Most business applications being written today use the client/server model. So does the Internet's main application protocols, such as HTTP, SMTP, Telnet, DNS, etc. In marketing, the term has been used to distinguish distributed computing by smaller dispersed computers from the "monolithic" centralized computing of mainframe computers. But this distinction has largely disappeared as mainframes and their applications have also turned to the client/server model and become part of network computing.

Each instance of the client software can send data requests to one or more connected servers. In turn, the servers can accept these requests, process them, and return the requested information to the client. Although this concept can be applied for a variety of reasons to many different kinds of applications, the architecture remains fundamentally the same.

The most basic type of client-server architecture employs only two types of hosts: clients and servers. This type of architecture is sometimes referred to as two-tier. It allows devices to share files and resources.

These days, clients are most often web browsers, although that has not always been the case. Servers typically include web servers, database servers and mail servers. Online gaming is usually client-server too. In the specific case of MMORPG, the servers are typically operated by the company selling the game; for other games one of the players will act as the host by setting his game in server mode.

The interaction between client and server is often described using sequence diagrams. Sequence diagrams are standardized in the Unified Modeling Language.

When both the client- and server-software are running on the same computer, this is called a single seat setup.

Characteristics of a client:

- Request sender is known as client
- Initiates requests
- Waits for and receives replies.
- Usually connects to a small number of servers at one time
- Typically interacts directly with end-users using a graphical user interface

Characteristics of a Server:

- Receiver of request which is sent by client is known as server
- Passive (slave)
- Waits for requests from clients
- Upon receipt of requests, processes them and then serves replies
- Usually accepts connections from a large number of clients
- Typically does not interact directly with end-users

Comparison to Peer-to-Peer Architecture:

Another type of network architecture is known as peer-to-peer, because each host or instance of the program can simultaneously act as both a client and a server, and because each has equivalent responsibilities and status. Peer-to-peer architectures are often abbreviated using the acronym P2P.Both client-server and P2P architectures are in wide usage today.

Comparison to Client-Queue-Client Architecture:

While classic Client-Server architecture requires one of communication endpoints to act as a server, which is much harder to implement, Client-Queue-Client allows all endpoints to be simple clients, while the server consists of some external software, which also acts as passive queue (one software instance passes its query to another instance to queue, e.g. database, and then this other instance pulls it from database, makes a response, passes it to database etc.). This architecture allows greatly simplified software implementation. Peer-to-Peer architecture was originally based on Client-Queue-Client concept.

Advantages:

In most cases, a client-server architecture enables the roles and responsibilities of a computing system to be distributed among several independent computers that are known to each other only through a network. This creates an additional advantage to this architecture: greater ease of maintenance. For example, it is possible to replace, repair, upgrade, or even relocate a server while its clients remain both unaware and unaffected by that change. This independence from change is also referred to as encapsulation.

All the data is stored on the servers, which generally have far greater security controls than most clients. Servers can better control access and resources, to guarantee that only those clients with the appropriate permissions may access and change data.

Since data storage is centralized, updates to those data are far easier to administer than would be possible under a P2P paradigm. Under a P2P architecture, data updates may need to be distributed and applied to each “peer” in the network, which is both time-consuming and error-prone, as there can be thousands or even millions of peers.

Many mature client-server technologies are already available which were designed to ensure security, ‘friendliness’ of the user interface, and ease of use.

It functions with multiple different clients of different capabilities.

Disadvantages:

Traffic congestion on the network has been an issue since the inception of the client-server paradigm. As the number of simultaneous client requests to a given server increases, the server can become severely overloaded. Contrast that to a P2P network, where its bandwidth actually increases as more nodes are added, since the P2P network’s overall bandwidth can be roughly computed as the sum of the bandwidths of every node in that network.

The client-server paradigm lacks the robustness of a good P2P network. Under client-server, should a critical server fail, clients’ requests cannot be fulfilled. In P2P networks, resources are usually distributed among many nodes. Even if one or more nodes depart and abandon a downloading file, for example, the remaining nodes should still have the data needed to complete the download.

Components/tools of a Client Server Network:

A client/server network has three main components: workstations, servers and the network devices that connect them. Workstations are the computers that are subordinate to servers. They send requests to servers to access shared programs, files and databases, and are governed by policies defined by servers. A server "services" requests from workstations and can perform many functions as a central repository of files, programs, databases and management policies. Network devices provide the communication path for servers and workstations. They act as connectors and route data in and out of the network.

1. Workstations:

Workstations, or client computers, initially differentiate themselves by the operating systems running them. In a client/server network, Windows 2000, Windows XP, Windows Vista and Windows 7 are examples of workstation operating systems. Aside from being relatively cheaper than server operating systems, their functions and processes are essentially intended for client computers. Centralized databases, shared programs, management and security policies are not part of their operating systems. What they have are localized versions of databases, programs and policies that can be applied individually to them. Workstations also have lower technical specifications than servers in the areas of memory, hard drive space and processor speed, because they are not required to process requests or record data from multiple computers.

2. Servers:

Servers are distinguished by different sets of operating systems like Windows 2000 Server, Windows 2003 or Windows 2008. They also have higher memory and hard drive space and faster processors because they store and service multiple (and often simultaneous) requests from workstations. A server can assume many roles in a client/server network. It can be a file server, a mail server, a database server and domain controller all at the same time. A well-set-up network, however, delineates these roles to different servers to optimize performance. A server, regardless of what role it has, essentially acts as a centralized repository of network files, programs, databases and policies. It makes for easier management and backup because it is not dependent to individual user configurations, but can be universally and uniformly implemented across the network.

3. Network Devices:

Network devices connect workstations and servers. They ensure that requests to and from workstations are routed properly to the correct server. Several network devices each provide different types of network connectivity. In a simple client/server network, a hub can connect a server to multiple workstations. It acts as a repeater, passing on data from one device to another. Bridges separate network segments. This is useful for offices with several departments to distinguish which department a particular workstation belongs to. Another network device, a switch, is similar to a bridge, but can detect conflicts between network segments like same IP addresses or computer names across departments. Wide-area networks use routers to connect network segments in different locations. Routers are also used to connect networks, or route information to the Internet.

4. Other Components:

Client/server networks usually have network printers or scanners, which are shared and can be used by all computers in the network. Instead of installing them individually to each computer, they can be placed in one location that everyone can access. This saves both space and money.

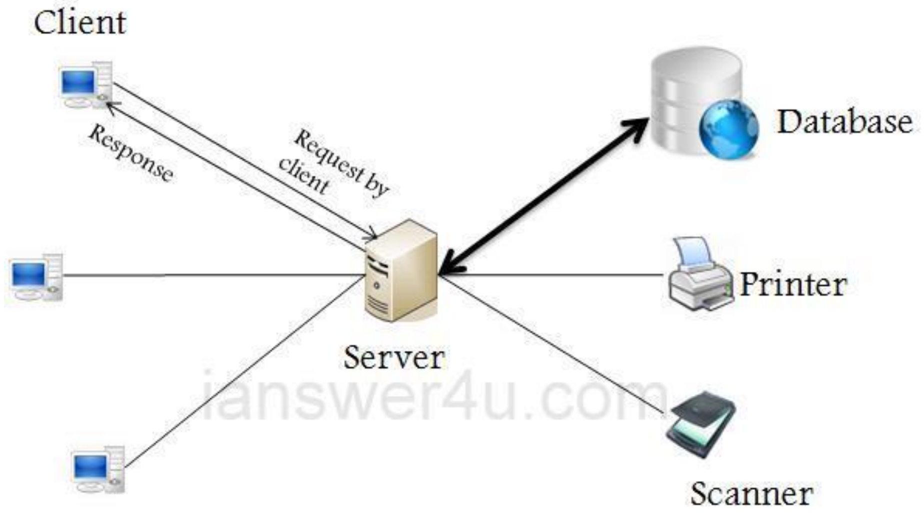

Fig: Client Server Architecture