The fundamental knowledge of System Design — (1) — Networking

Today, I will share the fundamental knowledge of system design.

Please share and clap if you like this article.

1. **Client-Server Model**
   It is a software system architecture, through which the advantages of the hardware environment at both ends can be fully utilized, and tasks can be reasonably allocated to the client-side and the server-side to achieve this, which reduces the communication overhead of the system. At present, most application software systems have a two-tier structure in the form of Client/Server. Since the current software application system is developing towards distributed Web applications, both Web and Client/Server applications can perform the same business processing and apply different modules. Shared logic components; therefore, internal and external users can access new and existing application systems, and new application systems can be expanded through the logic of existing application systems.

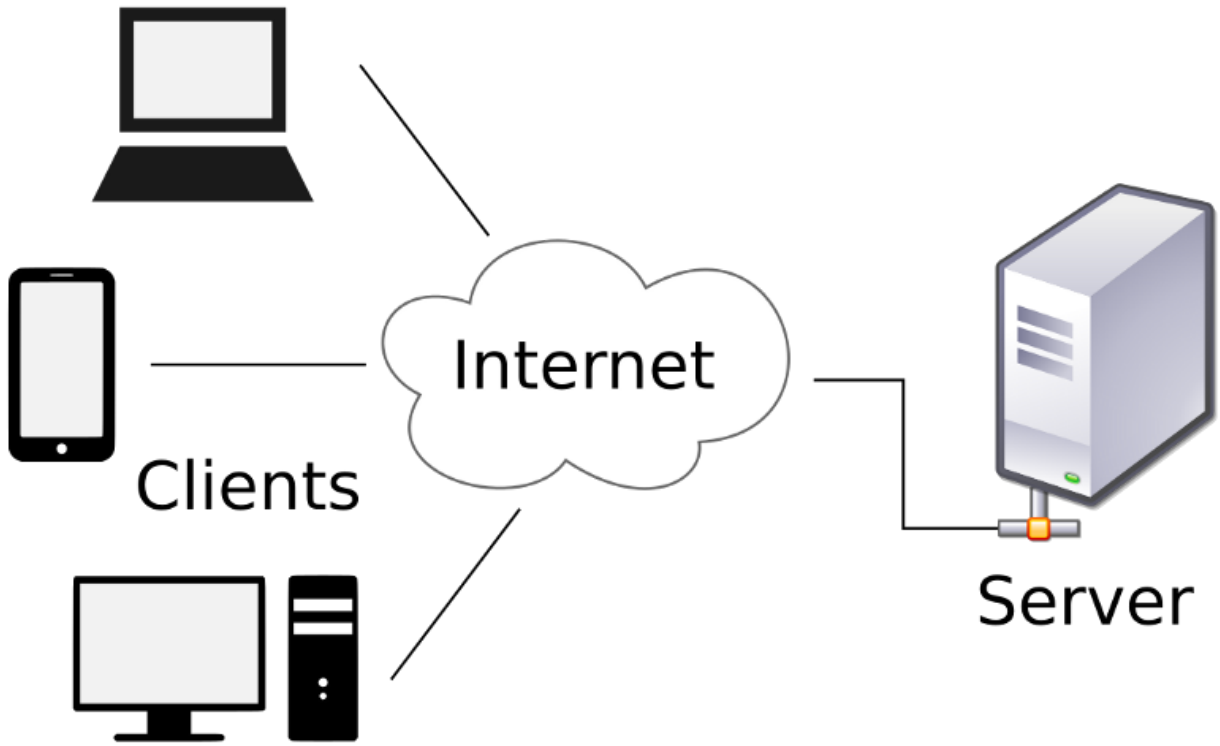
A complete HTTP request process

When you press enter http://www.google.com. After the browser receives this message, the browser recognizes the URL you want to visit according to its own algorithm and displays the search page for you. So what process did these go through?

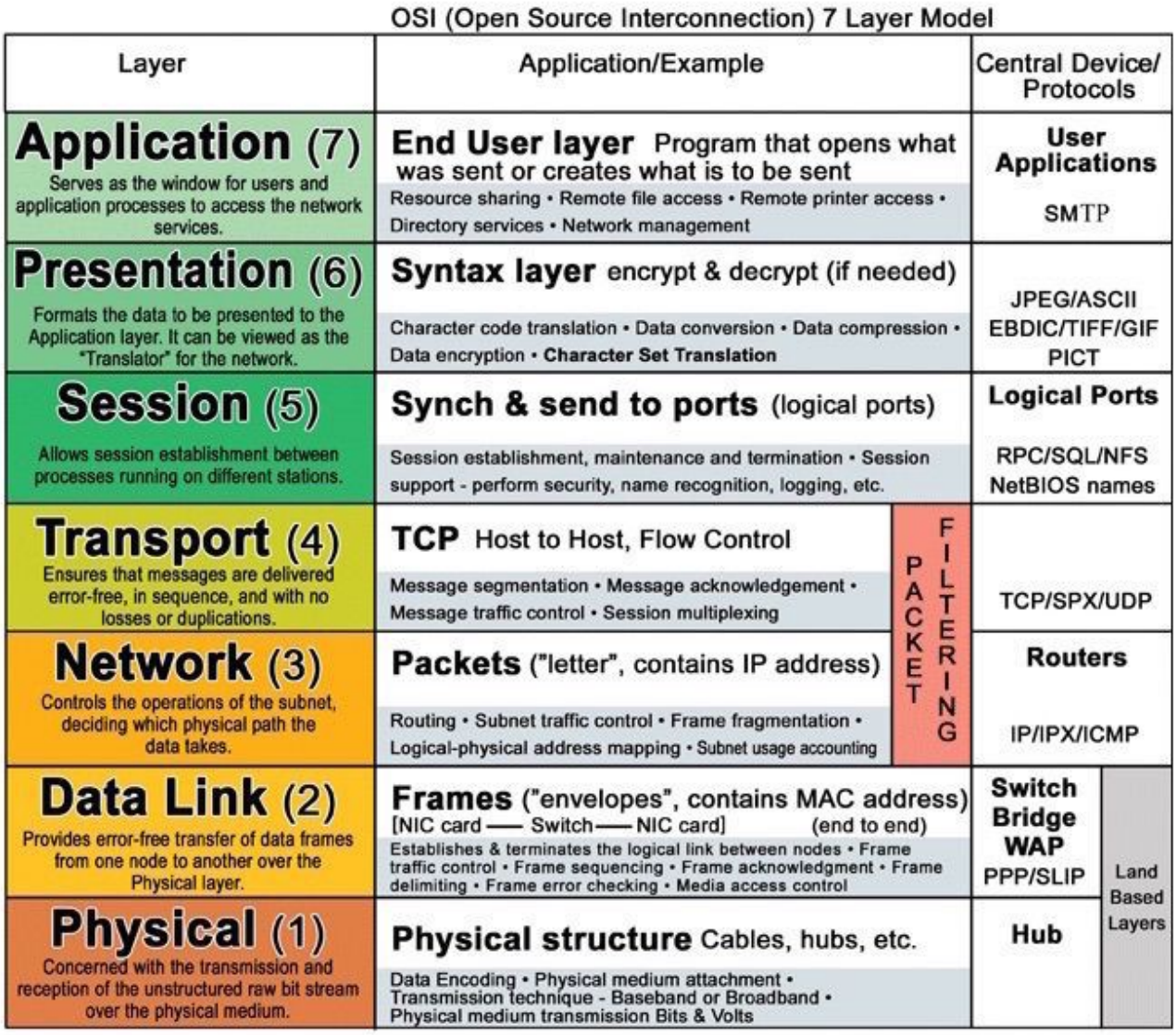
The general process is as follows:

- 1) The browser queries DNS to obtain the IP address corresponding to the domain name; the specific process includes the browser searching its own DNS cache, searching the DNS cache of the operating system, reading the local host file, and querying the local DNS server.
- 2) After the browser obtains the IP address corresponding to the domain name, the browser requests the server to establish a link and initiates a three-way handshake;

- 3) After the TCP/IP link is established, the browser sends an HTTP request to the server;
- 4) The server receives this request, maps it to a specific request processor for processing according to the path parameter, and returns the processing result and the corresponding view to the browser;
- 5) The browser parses and renders the view. If it encounters references to static resources such as js files, CSS files, and pictures, repeat the above steps and request these resources from the servers;
- 6) The browser renders the page according to the requested resources and data, and finally displays a complete page to the user.



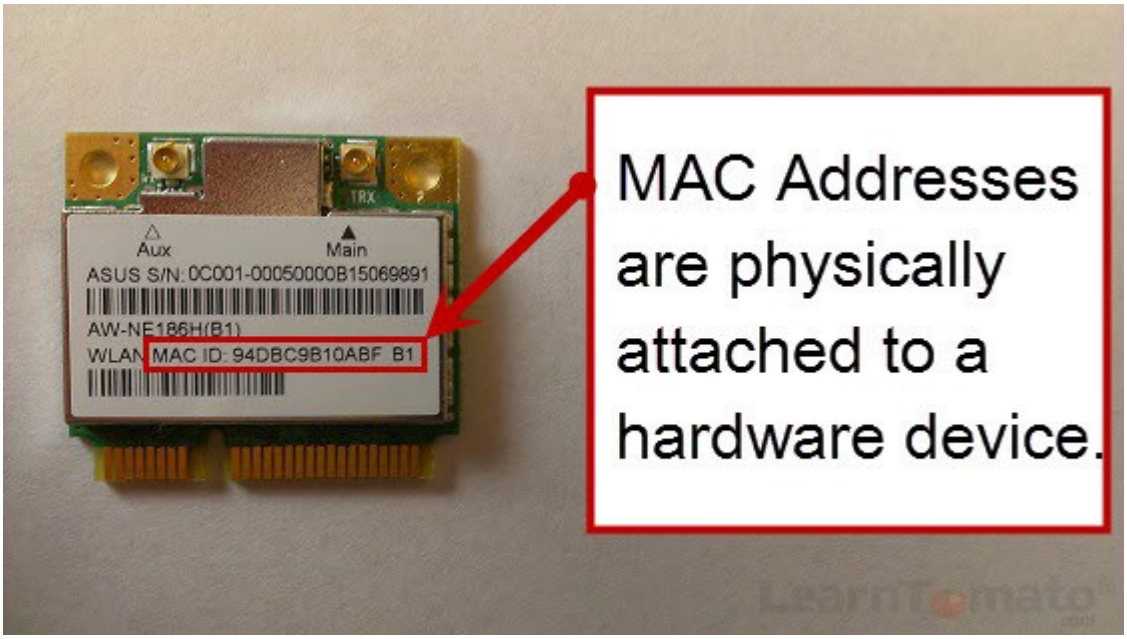
2. Network Protocols



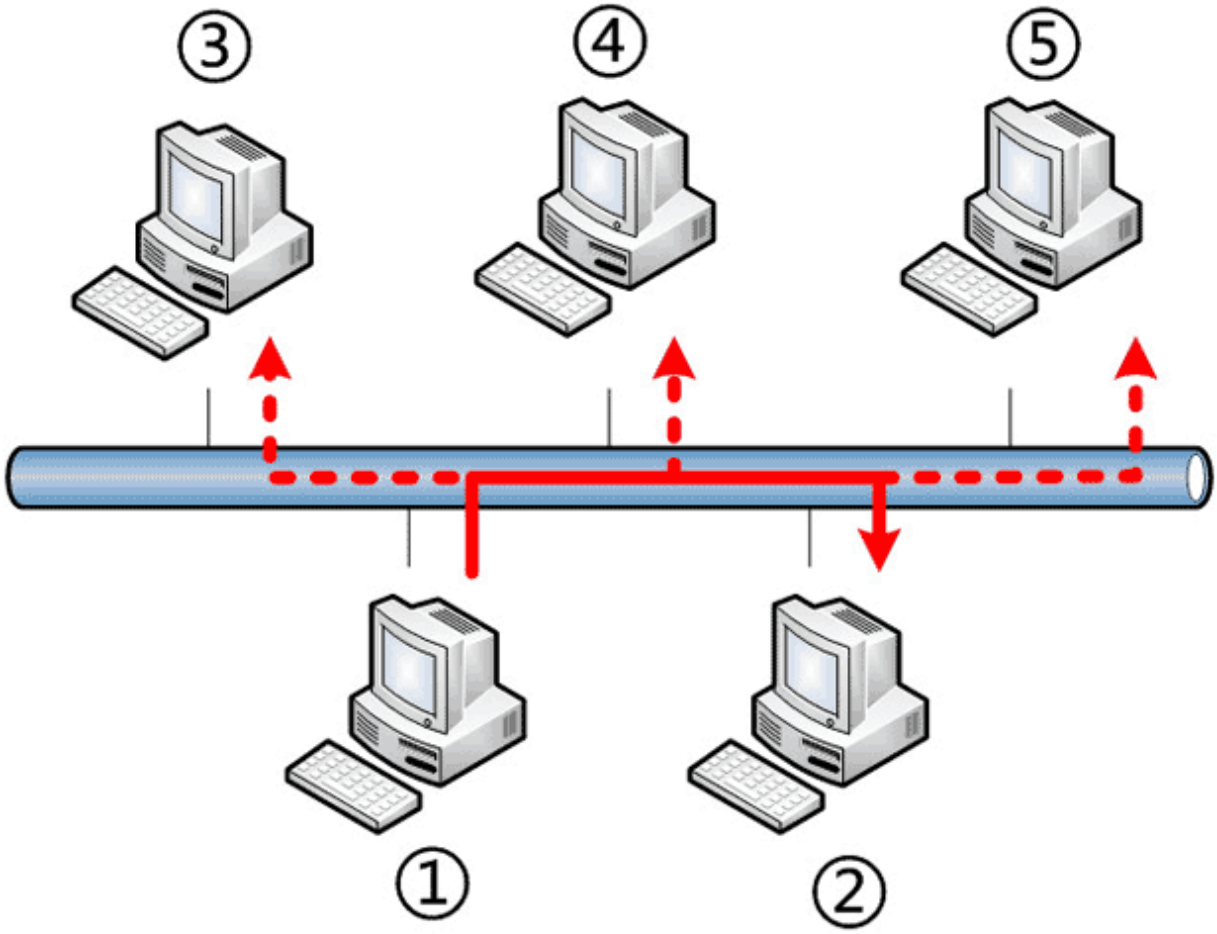
- The end-user uses only the application layer and does not know the other layer at all. To understand the internet, we must start from the bottom layer and understand the functions of each layer.
- Each layer is to complete a function. In order to achieve these functions, everyone needs to abide by common rules, called protocols. Many protocols are called " Internet Protocol Suite", as the core of the Internet.
- **The physical layer**: the computer needs to be networked first. It is optical cables, cables twisted pairs, radio waves, etc. So, the physical layer means to connect computers. It mainly specifies some electrical characteristics of the network and is responsible for transmitting electrical signals of 0 and 1. The simple 0 and 1 have no

meaning, and the interpretation method must be specified: how many electrical signals are counted as a group? What is the meaning of each signal bit?

- This is the function of the **data link layer**, which determines the grouping of 0 and 1.
- **Ethernet Layer/ Network Layer**: Ethernet stipulates a group of electrical signals from a data packet, is called a " frame". Each frame is divided into 2 parts: Head and Data. "Header" contains some descriptive items of the data packet, such as sender, receiver, data type, etc.; "data" is the specific content of the data packet. The length of the "header" is fixed at 18 bytes. The length of "data" is 46 bytes at the shortest and 1500 bytes at the longest. Therefore, the shortest "frame" is 64 bytes and the longest is 1518 bytes. If the data is very long, it must be divided into multiple frames for transmission. The "header" of an Ethernet packet contains information about the sender and receiver. So, how are the sender and receiver identified? Ethernet stipulates that all devices connected to the network must have a "network card" interface. Data packets must be transmitted from one network card to another network card. The address of the network card is the sending address and receiving address of the data packet, which is called the **MAC address.** When each network card leaves the factory, it has a unique MAC address in the world, with a length of 48 binary digits, usually represented by 12 hexadecimal numbers. The first 6 hexadecimal numbers are the manufacturer's serial number, and the last 6 are the serial number of the manufacturer's network card. With the MAC address, you can locate the network card and the path of the data packet.



- Broadcast: How does one network card know the MAC address of another network card? The answer is that there is an ARP protocol that can solve this problem. You only need to know that Ethernet data packets must know the MAC address of the receiver before they can be sent. Second, even with the MAC address, how can the system accurately deliver the data packet to the receiver? The answer is that Ethernet uses a very “ primitive” method. Instead of sending the data packet to the receiver accurately, it sends it to all computers in the network, allowing each computer to determine whether it is the receiver.



- In the above figure, computer №1 sends a data packet to computer №2, and computers №3, №4, and №5 in the same subnet will all receive this packet. They read the “header” of the packet, find the receiver’s MAC address, and compare it with their own MAC address. If the two are the same, they accept the packet for further processing, otherwise, they discard the packet. This way of sending is called “broadcasting”. With the definition of the data packet, the MAC address of the network card, and the sending method of the broadcast, the “link layer” can transmit data between multiple computers.

- The Ethernet protocol relies on the MAC address to send data. In theory, the Shanghai network card can find the Los Angeles network card by relying solely on the MAC address, which is technically achievable. However, this has a major disadvantage. Ethernet uses broadcast to send data packets, and all members have one “packet” by hand, which is not only inefficient but also limited to the sub-network where the sender is located. In other words, if the two computers are not in the same subnet, the

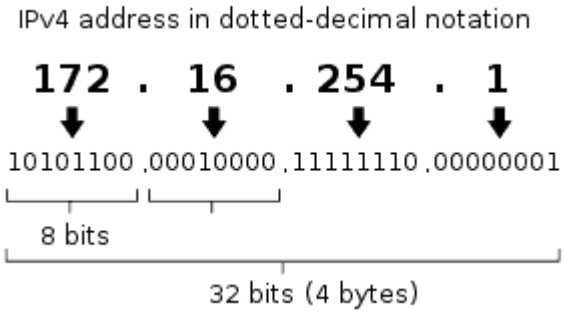
broadcast will not pass. This design is reasonable, otherwise, every computer on the Internet will receive all packets, which will cause disaster. The Internet is a giant network composed of countless sub-networks. It is like imagining that the computers in Shanghai and Los Angeles will be on the same sub-network. This is almost impossible.

- Therefore, a method must be found to be able to distinguish which MAC addresses belong to the same subnet and which are not. If it is the same subnet, it will be sent by broadcast, otherwise, it will be sent by "routing". (The meaning of "routing" refers to how to distribute data packets to different sub-networks. This is a big topic, which is not covered in this article.) Unfortunately, the MAC address itself cannot do this. It is only related to the manufacturer and has nothing to do with the network. **This led to the birth of the "Ethernet/ network layer". Its function is to introduce a new set of addresses so that we can distinguish whether different computers belong to the same subnet. This set of addresses is called "network address", or "web address" for short.**

- Therefore, after the emergence of the "network layer", each computer has two types of addresses, one is the MAC address, and the other is the network address. There is no connection between the two types of addresses. The MAC address is bound to the network card, and the network address is assigned by the administrator. They are just randomly combined.

- The network address helps us determine the sub-network where the computer is located, and the MAC address sends the data packet to the target network card in the sub-network. Therefore, it can be logically inferred that the network address must be processed first, and then the MAC address.

- IP Protocol / TCP: This specifies the network address. The address it defines is called an IP address. Currently, the fourth edition of the IP protocol, referred to as IPv4, is widely adopted. This version stipulates that the network address consists of 32 binary bits. Traditionally, we use decimal numbers divided into four segments to represent the IP address, from 0.0.0.0 to 255.255.255.255.



- Every computer on the Internet is assigned an IP address. This address is divided into two parts, the first part represents the network, and the second part represents the host. For example, the IP address 172.16.254.1, which is a 32-bit address, assumes that its network part is the first 24 bits (172.16.254), then the host part is the last 8 bits (the last 1). Computers on the same subnet must have the same network part of their IP address, which means that 172.16.254.2 should be on the same subnet as 172.16.254.1.

- However, the problem is that we cannot judge the network part from the IP address alone. Let's take 172.16.254.1 as an example. Whether its network part is the first 24 bits, the first 16 bits, or even the first 28 bits, you can't tell from the IP address. So, how can we judge whether two computers belong to the same subnet from the IP address? This requires another parameter "subnet mask" (subnet mask). The so-called "subnet mask" is a parameter that indicates the characteristics of the subnet. It is equivalent to the IP address in the form and is also a 32-bit binary number. Its network part is all 1s and the host part is all 0s. For example, the IP address 172.16.254.1, if it is known that the network part is the first 24 digits and the host part is the last 8 digits, then the subnet mask is 11111111.11111111.11111111. 00000000, which is 255.255.255.0 in decimal. Knowing the "subnet mask", we can determine whether any two IP addresses are on the same subnet. The method is to perform an AND operation on the two IP addresses and the subnet mask respectively (both digits are 1, the result of the operation is 1, otherwise it is 0), and then compare whether the results are the same. If so, it means that they are in the same subnet. In the network, otherwise, it is not.

- For example, the subnet masks of the known IP addresses 172.16.254.1 and 172.16.254.233 are both 255.255.255.0, are they in the same subnet? Both and the subnet mask are AND operated separately, and the result is 172.16.254.0, so they are in the same subnet. To sum up, there are two main functions of the IP protocol. One is to

assign an IP address to each computer, and the other is to determine which addresses are on the same subnet.

- **IP packet:** The data sent according to the IP protocol is called an IP packet. It is not difficult to imagine that it must include IP address information. But as mentioned earlier, the Ethernet data packet only contains the MAC address, and there is no field for the IP address. So do I need to modify the data definition and add another field? The answer is no, we can put the IP data packet directly into the "data" part of the Ethernet data packet, so there is no need to modify the Ethernet specifications at all. This is the benefit of the hierarchical structure of the Internet: changes in the upper layer do not involve the structure of the lower layer at all. Specifically, IP data packets are also divided into two parts: "header" and "data". The "header" part mainly includes information such as version, length, and IP address, while the "data" part is the specific content of the IP data packet. After it is put into the Ethernet data packet, the Ethernet data packet becomes the following.

- The length of the "header" part of an IP packet is 20 to 60 bytes, and the total length of the entire packet is a maximum of 65,535 bytes. Therefore, in theory, the "data" part of an IP data packet can be up to 65,515 bytes long. As mentioned earlier, the "data" part of an Ethernet packet is only 1500 bytes long. Therefore, if the IP data packet exceeds 1500 bytes, it needs to be divided into several Ethernet data packets and sent separately.

- **ARP protocol:** Because IP data packets are sent in Ethernet data packets, we must know two addresses at the same time, one is the other party's MAC address, and the other is the other party's IP address. Normally, the other party's IP address is known (explained later), but we don't know its MAC address. Therefore, we need a mechanism to get the MAC to address from the IP address. This can be divided into two situations. In the first case, if the two hosts are not in the same subnet, there is no way to get the

MAC address of the other party. The only way is to send the data packet to the "gateway" where the two subnets connect and let the gateway handle it. In the second case, if two hosts are on the same subnet, then we can use the ARP protocol to get each other's MAC addresses. The ARP protocol also sends out a data packet (included in the Ethernet data packet), which contains the IP address of the host it wants to query. In the field of the other party's MAC address, fill in FF:FF:FF:FF:FF: FF, Which means this is a "broadcast" address. Each host in its subnet will receive this data packet, take out the IP address from it, and compare it with its own IP address. If the two are the same, both reply and report their MAC address to the other party, otherwise the packet will be discarded. In short, with the ARP protocol, we can get the MAC address of the host in the same subnet and can send the data packet to any host.

- **the transport layer:** With the MAC address and IP address, we can already establish communication between any two hosts on the Internet. The next problem is that there are many programs on the same host that need to use the Internet. For example, you browse the web while chatting online with your friends. When a data packet is sent from the Internet, how do you know whether it represents the content of a web page or the content of an online chat? In other words, we also need a parameter to indicate which program (process) this data packet is used. This parameter is called "port", which is actually the number of each program that uses the network card. Each data packet is sent to a specific port of the host, so different programs can get the data they need. "Port" is an integer between 0 and 65535, exactly 16 binary bits. Ports from 0 to 1023 are occupied by the system, and users can only use ports greater than 1023. Whether it is browsing the web or chatting online, the application will randomly select a port and then contact the corresponding port of the server.

- **The function of the "transport layer" is to establish "port-to-port" communication. In contrast, the function of the "network layer" is to establish "host-to-host" communication. As long as the host and port are determined, we can achieve communication between programs.** Therefore, the Unix system calls the host + port a "socket". With it, web application development can be carried out.

- **UDP Protocol/ Session layer**: we must add port information to the data packet, which requires a new protocol. The simplest implementation is called UDP protocol, its format is almost in front of the data, plus the port number. UDP data packet is also

composed of two parts: “header” and “data”. The “header” part mainly defines the sending port and the receiving port, and the “data” part is the specific content. Then, put the entire UDP data packet into the “data” part of the IP data packet. The UDP data packet is very simple, the “header” part has only 8 bytes in total, and the total length does not exceed 65,535 bytes, which fits into an IP data packet. As mentioned earlier, the IP data packet is placed in the Ethernet data packet, so the entire Ethernet data packet now becomes the following :

- **TGP protocol/ Presentation Layer**: The advantage of the UDP protocol is that it is relatively simple and easy to implement, but the disadvantage is that it has poor reliability. Once a data packet is sent, it is impossible to know whether the other party has received it. In order to solve this problem and improve network reliability, the TCP protocol was born. This protocol is very complicated, but it can be approximated as a UDP protocol with a confirmation mechanism, which requires confirmation every time a data packet is sent. If a data packet is missing, the confirmation cannot be received, and the sender knows that it is necessary to resend the data packet. Therefore, the TCP protocol can ensure that data will not be lost. Its disadvantages are that the process is complex, difficult to implement, and consumes more resources. TCP data packets, like UDP data packets, are embedded in the “data” part of IP data packets. There is no limit on the length of TCP data packets, and theoretically, they can be infinitely long. However, in order to ensure the efficiency of the network, usually, the length of TCP data packets will not exceed the length of IP data packets to ensure that a single TCP data packet does not need to be split.

- **Application layer:** The application program receives the “transport layer” data, and then it must be interpreted. Since the Internet is an open architecture, data sources are diverse, and the format must be specified in advance, otherwise, it will not be interpretable at all. For example, the TCP protocol can transfer data for various programs, such as Email, WWW, FTP, and so on. Then, there must be different

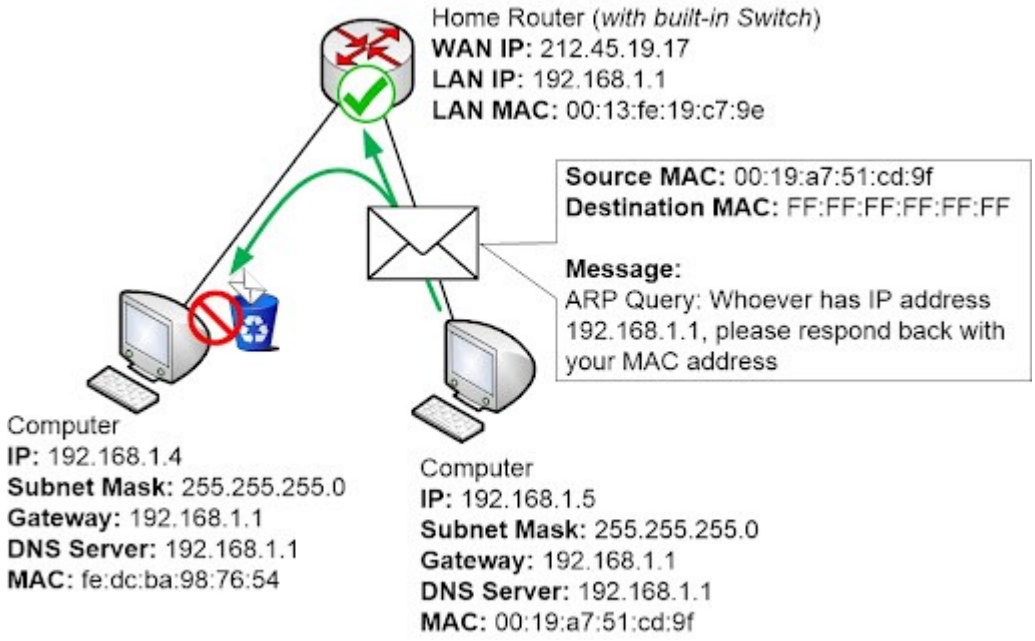
protocols that specify the format of e-mail, web pages, and FTP data, and these application protocols constitute the “application layer.” This is the highest level, directly facing the user. Its data is placed in the “data” part of the TCP packet.

Now, we want to switch to the user’s perspective to see how users interact with these agreements. We already know that network communication is the exchange of data packets. Computer A sends a data packet to computer B, and the latter receives it and replies with a data packet, thus realizing the communication between the two computers.

To send this packet, you have to know 2 addresses:
1. The other party’s MAC address

2. The IP address of the other party

- With these two addresses, the data packet can be accurately delivered to the receiver. However, as mentioned earlier, MAC addresses have limitations. If two computers are not in the same subnet, they cannot know each other’s MAC address and must be forwarded through a gateway.



- In the above picture, computer №1 wants to send a data packet to computer №4. It first judges whether the №4 computer is on the same subnet, and it turns out that it is not

(the judgment method will be described later), so it sends this data packet to gateway A. Through the routing protocol, gateway A finds that computer №4 is located in subnet B, and sends the data packet to gateway B, and gateway B forwards it to computer №4.

- Computer №1 sends the data packet to gateway A, and it must know the MAC address of gateway A. Therefore, the destination address of the data packet is actually divided into two situations:

- The other party's MAC address, the other party's IP address: Not the same subnet

- Before sending a data packet, the computer must determine whether the other party is in the same subnet and then select the corresponding MAC address. Next, let's take a look at how this process is completed in actual use.

- **Static IP address**: When you bought a new computer, plug in the internet cable, and turn it on. Will the computer be able to access the internet at this time? Usually, you have to make some internet settings. Sometimes, the ISP/ the administrator will tell you the following 4 parameters, so that the computer can connect to the Internet:
  1) The local IP address
  2) Subnet Mask
  3) Gateway IP address
  4) DNS IP address

- These four parameters are indispensable, and I will explain why you need to know them to get online. Because they are given, the computer will be assigned the same IP address every time it is turned on, so this situation is called "static IP address Internet access". However, such a setting is very professional, and ordinary users are daunted, and if the IP address of one computer remains unchanged, other computers cannot use this address, which is not flexible enough. For these two reasons, most users use "dynamic IP addresses to surf the Internet."

- **Dynamic IP address:** The so-called "dynamic IP address" means that after the computer is turned on, it will automatically be assigned an IP address without a manual setting. The protocol it uses is called the DHCP protocol. This agreement stipulates that in each sub-network, there is a computer responsible for managing all the IP addresses of the network, which is called a "DHCP server". When a new computer joins the network, it must send a "DHCP request" packet to the "DHCP server" to apply for an IP

address and related network parameters. As mentioned earlier, if two computers are on the same subnet, they must know the MAC address and IP address of each other before they can send data packets. However, the newly added computer does not know these two addresses, how can it send data packets? The DHCP protocol makes some clever provisions.

- **DHCP protocol :**
  (1) At the top of the "Ethernet header", set the MAC address of the sender (this machine) and the MAC address of the receiver (DHCP server). The former is the MAC address of the local network card, the latter does not know at this time, just fill in a broadcast address: FF-FF-FF-FF-FF-FF.
  (2) In the "IP header" at the back, set the IP address of the sender and the IP address of the receiver. At this time, the machine does not know about both. Therefore, the sender's IP address is set to 0.0.0.0, and the receiver's IP address is set to 255.255.255.255.
  (3) In the last "UDP header", set the sender's port and receiver's port. This part is stipulated by the DHCP protocol, the sender is port 68, and the receiver is port 67.

- After the data packet is constructed, it can be sent out. Ethernet is a broadcast transmission, and every computer on the same subnet has received this packet. Because the receiver's MAC address is FF-FF-FF-FF-FF-FF, it is not clear to who it is sent, so every computer that receives this packet must also analyze the IP address of the packet to determine it. Not for yourself. When you see that the sender's IP address is 0.0.0.0 and the receiver is 255.255.255.255, the DHCP server knows that "this packet is sent to me", and other computers can discard this packet.

- Next, the DHCP server reads the data content of this packet, assigns an IP address, and sends back a "DHCP response" packet. The structure of this response packet is similar. The MAC address of the Ethernet header is the network card addresses of both parties, and the IP address of the IP header is the IP address of the DHCP server (sender) and 255.255.255.255 (receiver), UDP header The ports are 67 (sender) and 68 (receiver). The IP address assigned to the requester and the specific parameters of the network is included in the Data section.

- The newly added computer receives this response packet, so it knows its own IP address, subnet mask, gateway address, DNS server, and other parameters.

Visit a webpage

- Local Parameters: We assume that after the steps in the previous section, the user has set his own network parameters

- *This machine's IP address: is 192.168.1.100*
  Subnet mask: 255.255.255.0
  Gateway's IP address: 192.168.1.1
  DNS's IP address: 8.8.8.8

- **DNS protocol:** We know that to send a data packet, you must know the other party's IP address. However, now, we only know the website www.google.com, not its IP address. The DNS protocol can help us to convert this URL into an IP address. The known DNS server is 8.8.8.8, so we send a DNS packet (port 53) to this address. Then, the DNS server responded, telling us that Google's IP address is 172.194.72.105. Thus, we know the other party's IP address.

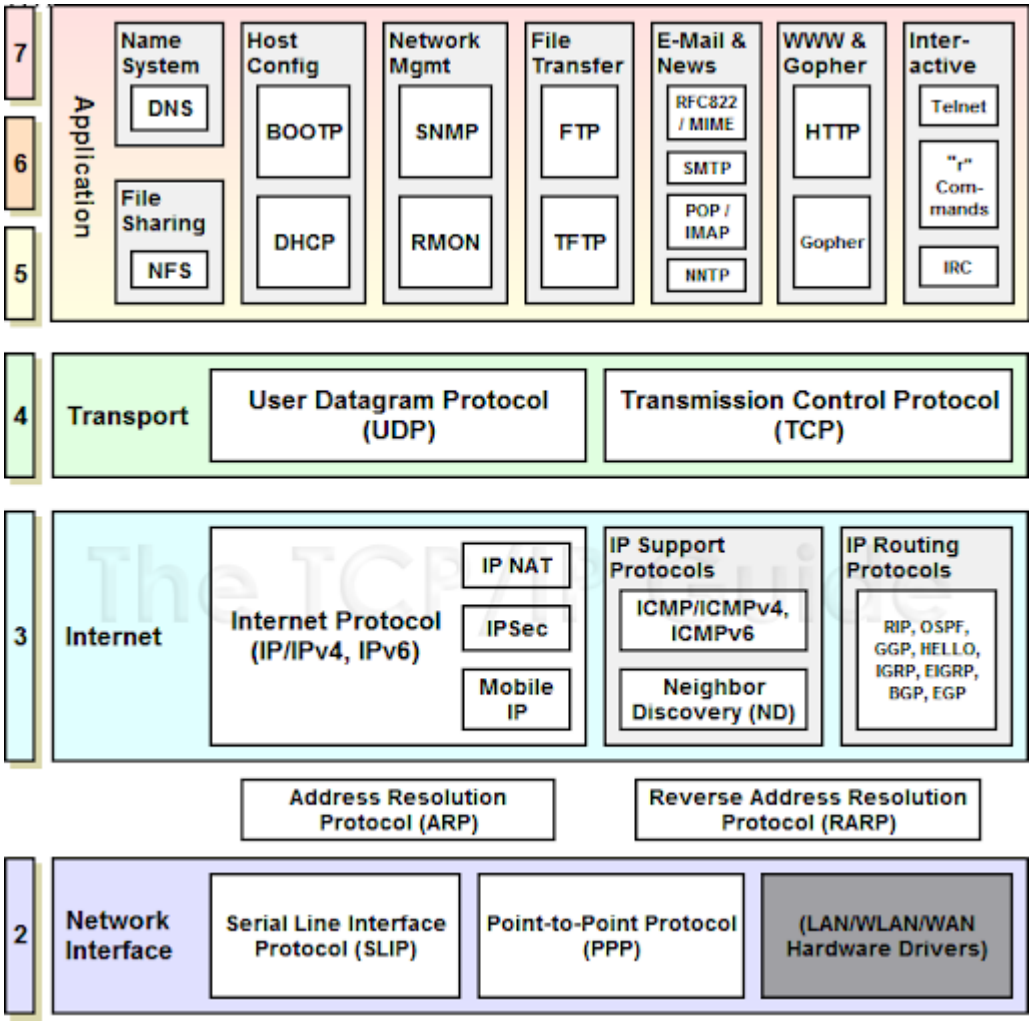
- **Subnet mask:** Next, we have to determine whether this IP address is in the same subnet, which requires the subnet mask. The known subnet mask is 255.255.255.0. This machine uses it to perform a binary AND operation on its IP address 192.168.1.100 (both digits are 1, the result is 1, otherwise it is 0), and the calculation result is 192.168 .1.0; Then an AND operation is also performed on Google's IP address 172.194.72.105, and the result is 172.194.72.0. The two results are not equal, so the conclusion is that Google and this machine are not in the same subnet. Therefore, if we want to send a data packet to Google, it must be forwarded through the gateway 192.168.1.1, that is to say, the MAC address of the receiver will be the MAC address of the gateway.

- **Application layer protocol:** The HTTP protocol is used to browse the web, and its entire data packet structure is like this:

- The content of the HTTP part is similar to the following:
  GET / HTTP/1.1
  Host: www.google.com
  Connection: keep-alive
  User-Agent: Mozilla/5.0 (Windows NT 6.1) .....
  Accept: text/html,application/xhtml+xml,application/xml ;q=0.9,/*;q=0.8*
  Accept-Encoding: gzip,deflate,sdch

Accept-Language: zh-CN,zh;q=0.8
Accept-Charset: GBK,utf-8;q=0.7,; q=0.3*
Cookie: … …

- We assume that the length of this part is 4960 bytes, and it will be embedded in the TCP packet.
- **TCP protocol:** TCP data packets need to set the port. The HTTP port of the receiver (Google) is 80 by default, and the port of the sender (the machine) is a randomly generated integer between 1024–65535, which is assumed to be 51775. The header length of a TCP data packet is 20 bytes, plus the embedded HTTP data packet, the total length becomes 4980 bytes.
- **IP protocol:** Then, the TCP data packet is embedded in the IP data packet. IP data packets need to set the IP addresses of both parties, which are known, the sender is 192.168.1.100 (local machine), and the receiver is 172.194.72.105 (Google). The header length of the IP packet is 20 bytes, plus the embedded TCP packet, the total length becomes 5000 bytes.
- **Ethernet protocol:** Finally, the IP data packet is embedded in the Ethernet data packet. Ethernet data packets need to set the MAC addresses of both parties, the sender is the MAC address of the machine's network card, and the receiver is the MAC address of the gateway 192.168.1.1 (obtained through the ARP protocol). The maximum length of the data part of the Ethernet data packet is 1500 bytes, while the current IP data packet length is 5000 bytes. Therefore, the IP data packet must be divided into four packets. Because each packet has its own IP header (20 bytes), the lengths of the IP data packets of the four packets are 1500, 1500, 1500, and 560, respectively.
- **Server-side response:** After being forwarded by multiple gateways, Google's server 172.194.72.105 received these four Ethernet data packets. According to the serial number of the IP header, Google puts together the four packets, takes out the complete TCP data packet, then reads the "HTTP request" inside, then makes the "HTTP response", and then sends it back using the TCP protocol.
- After the machine receives the HTTP response, it can display the web page and complete a network communication. This example ends here. Although it has been simplified, it roughly reflects the entire communication process of the Internet protocol.


7
6
5
Application
Name System
DNS
File Sharing
NFS
Host Config
BOOTP
DHCP
Network Mgmt
SNMP
RMON
File Transfer
FTP
TFTP
E-Mail & News
RFC822 / MIME
SMTP
POP / IMAP
NNTP
WWW & Gopher
HTTP
Gopher
Inter-active
Telnet
"r" Com-mands
IRC
4
Transport
User Datagram Protocol (UDP)
Transmission Control Protocol (TCP)
3
Internet
Internet Protocol (IP/IPv4, IPv6)
IP NAT
IPSec
Mobile IP
IP Support Protocols
ICMP/ICMPv4, ICMPv6
Neighbor Discovery (ND)
IP Routing Protocols
RIP, OSPF, GGP, HELLO, IGRP, EIGRP, BGP, EGP
Address Resolution Protocol (ARP)
Reverse Address Resolution Protocol (RARP)
2
Network Interface
Serial Line Interface Protocol (SLIP)
Point-to-Point Protocol (PPP)
(LAN/WLAN/WAN Hardware Drivers)