A Day in the Life of a Web Page Request

Introduction

Take facebook as an example. To access this website, you first need to enter facebook.com in the browser . This is just the domain name of the website. The browser does not know where to access the corresponding resources. At this time, you need to use the DNS protocol pair.

DNS

The domain name is resolved, and the domain name is bound to the corresponding server IP address at the domain name registrar, so DNS can obtain the IP address bound to it through the domain name, such as 69.63.176.13, The IP address is equivalent to the "house number" of the Internet world.

Prepare Data Request

After knowing the target IP address, the browser starts to package this request. Here, it is divided into HTTP request and HTTPS request according to whether the transmission data is encrypted or not. The HTTP protocol and HTTPS protocol are used respectively. No matter which protocol is used, it must be encapsulated. Request headers and request parameters so that the server can return the corresponding response:

Message	Format
HTTP header	GET,URL,HTTP1.0/HTTP1.1 body format, body length
	requested data.

The layer where DNS, HTTP, and HTTPS are located is the application layer. After the application layer is encapsulated, the browser passes the application layer packets to the next layer to complete. This process is implemented through socket programming.

Make connection with TCP and UDP

UDP is a connectionless protocol, and the other is the connection-oriented TCP protocol. UDP can communicate without establishing a connection, but it is unreliable and may cause packet loss. A three-way handshake is required to establish a connection to ensure that the data packet reaches the destination, but there is additional overhead, and the performance and speed are not as good as UDP. Which protocol to use depends on specific needs.

For HTTP/HTTPS requests, they are all reliable connections based on the TCP protocol. The TCP protocol has two ports, one is the port that the browser listens to (listening for server responses), and the other is the port that the server listens for (for listening to client requests, for HTTP request, usually port 80, for HTTPS request, usually port 443). The operating system will judge according to the port, which process to forward the obtained packet to:

Message	Format
TCP header	Browser Port: xyz server port 443
HTTP header	GET,URL,HTTP1.0/HTTP1.1 body format, body length
	Requested data.

Getting Client and Server IP's

At this stage we have a protocol whose name is the IP protocol. At this layer, an IP header is added to the packet passed by the transport layer, which contains the source IP address (Information such as the machine where the browser is located), the target IP address (the machine where the server is located):

Message	Format
IP header	Client IP: 192.168.1.1 Server IP: 69.63.176.13
TCP header	Browser Port: xyz server port 443
HTTP header	GET,URL,HTTP1.0/HTTP1.1 body format, body length
	Requested data.

After getting the IP

After the operating system knows the IP address of the target machine, it starts to find the target machine based on it. If it is a machine in the local area network, it can be judged directly by the IP address. If it is a machine outside the local area network, it needs to go to the outside through the gateway World lookup. Obviously, the target IP here is not on the local LAN.

Getting MAC address

When the operating system starts, it will configure the IP address through the DHCP protocol and the IP address of the default gateway: 192.168.1.1. The operating system will obtain the MAC address of the gateway through the IP address through the ARP protocol, and add the MAC address of the local gateway and computer to the MAC header:

Message	Format
MAC header	Local client MAC address local gateway MAC address
IP header	Client IP: 192.168.1.1 Server IP: 114.215.241.29
TCP header	Browser Port: xyz server port: abc
HTTP header	GET,URL,HTTP1.0/HTTP1.1 body format, body length
	Requested data.

In this way, the operating system passes the IP packet to the next layer-the link layer, and then sends it out via the network card (there is a physical layer connection between the client machine and the gateway).When using a network card (NIC), the MAC address will be burned into the ROM, and the MAC address of any network card is unique in the world. The MAC address is equivalent to our ID number. Once we are born, it is hard-coded and will not change for a lifetime, and it is globally unique. The IP address can be dynamically allocated and cannot be guaranteed to be globally unique. It can only be guaranteed to be unique within the LAN, which is equivalent to your residential address, which may change over time.

Receiving message by the gateWay:

After the gateway receives the packet, it will judge the next step based on its own knowledge. The gateway is often a router. There is a routing table for how to get to a certain target IP address. Network request packets often need to go through multiple gateways to reach the final target machine.

Suppose that the network packet finally reaches the gateway where the target server is located after passing through multiple gateways. Through the ARP protocol, the target server returns a MAC address according to the target IP address, indicating that the target server is here, and then the network packet is in the LAN where the server is located through this MAC address Find the target machine within.

Note:When the server receives the request, contrary to the client sending the request, the network flow is bottom-up

Step1:

The target server finds that the MAC address of the network request packet is matched, removes the MAC header, and passes the packet to the upper network layer. When it finds that the IP is also matched, it removes the IP header, and then hands it to the transport layer.

Step 2:

In the transport layer, for every packet received, a reply packet is received. This reply is not a response to this request, only that the reply packet has been received. This reply will go back and forth along the packet's path. If after a period of time (timeout period), the client still does not receive a reply from the server, it will resend the packet until it receives a reply. Similarly, this retransmission does not re-initiate the above client request, but the transport layer will retry the same request repeatedly. For the user, there is only one request.

Step 3:

Back to the target server, when the network packet reaches the transport layer, there is a server listening port number in the TCP header. Through this port number, you can find the port that the facebook website is listening to, that is, the port 443 configured in Nginx. After the port is matched , Remove the TCP header, hand the network packet to the application layer, and start processing the HTTP/HTTPS request.

Step 4:

If it is a front-end resource, it will respond directly through Nginx. If it is a PHP dynamic request, Nginx will forward the request to the PHP-FPM process running in the background for processing. Of course, if Nginx does load balancing, and the back-end service is a distributed system or provides microservices (involving RPC remote calls), there is more complex processing logic, which we will talk about later.

Step 5:

When the background service processing is completed, it will return an HTTPS response packet to inform the user that the request is successful, and return the response content. Similarly, this network response packet is the same as the request packet, which is packaged from top to bottom and goes along the path. Layers of "gates" (gateways), return to the client that initiated the request, and then go through bottom-up processing, and finally display the facebook Login/Homepage on the client browser.