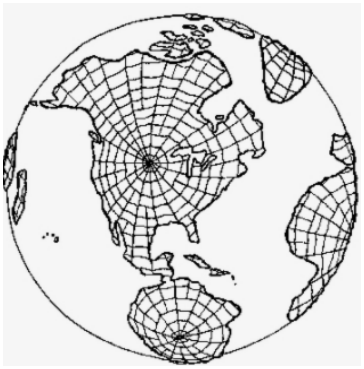
# INTERNET

## Some resumes about the structure of the internet what is

A Wire to Communicate With other systems
We are not connected directly to the internet
the computer Pass info for **servers**
which turn to connect in all over the world
on a **backbone** connection.

Is a USA research used after the wars to communicate data between universities.

At the start that does not have much use but when evolves and applied by patterns make the internet how we know. some of these patterns are:



### WWW

A Protocol that specifies internet content & communication

Made with some security systems like SSL and DNS facilities

### DNS

used to facilitate user life when needing to record some access.
A Domain can have a structure like this


label1.label2.company.com.br

Defined by the Department of register on California
need to be registered by a registrant company like hostinger
every computer and every internet provider has a DNS list.

To see the DNS list of your computer use:

**ipconfig/displaydns**

This list is updated when you access a new site.
for the internet provider and, this list is updated automatically passed on a process managed by DNS Registrants

| II | IV |
| --- | --- |
| The computer check DNS to know if you know | if not ask for the internet provider the DNS and try again |

| I | III |
| --- | --- |
| You type an address | If know access that |

Also in the WWW and DNS, we have the **URL**

is a pattern used to add some other Infos about the site
every info about page that you are, used security and some parameters are on that;

and all of that need to be

## HOSTING

Is a server used to display the info of your website.
This will have some administrator panels with lookup tools and services of email and caching of the data.
the most popular host server is Amazon AWS
but you have the option of another web hosting or make your server.
The most popular administrative panel is Cpanel.

## HTTP

Is the base of the WWW and is auto managed by web browsers
all HTTP communication made a **request** and receive from the server a **response**
HTTP is used to standardize the communications and facilitate that, cause every request needs to pass for several servers, modems, and computers.
it will have

### Head
with types and info about the expected response and origin of a request.

### Body
That can be optional and send custom data to communicate with the server

### Method
defines what must be done, like if you want to remove some data the HTTP method wants to be DELETE, we have these methods:

GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE AND PATCH

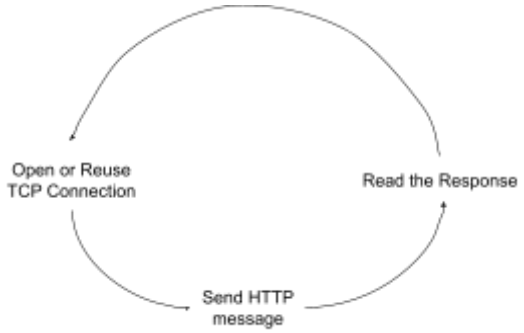
Every response sends a code that informs you about the server conditions:
100 - INFORMATIONAL - used for example: to check if can upload some file
200 - SUCCESS
300 - REDIRECT
400 - USER FAULT
500 - SERVER FAULT

The HTTP Request is the method that makes the DDOS attack possible, cause all new requests, made a new connection to a server, that also makes the server more slowly

## HTTP Versions

for over the years, HTTP has had some improvements
for example, at the star: every request made a new TCP connection
because this and some other problems was done the HTTP 1.1

### HTTP 1.1




Supports connections multi-request between pipelines
Enables unique server responses for less internet consumption.
have some problems that will furthermore be resolved by HTTP 2

TCP connections had a three-way handshake that's mean lossless data connection and works like this:

Computer send a request with a number **3** thats check the send (it's a random number)

Server receive this request, open the connection and send **4** (number + 1) validating the connection. this also send the data and a random number **8** to check the deliver

The computer receive it and send to connection 9 (number + 1) to validate that it arrived

After the things are served. close the connection

## HTTP 2.0

Made to make the connection faster

grateful by the optic fiber made it, even more, faster

binary protocol - every request now is binary, and can send every type of file and data

stream - can send a stream of data

multiplexing - in addition, to keep the connection open. now can send multiple data and receive many files at once

server push - can send packages that it knows will be used before it will be used, maximize website speed

request priority - by multiplexing can use a priority header to send a data faster than another.

security - enable new methods of authentication like AUTH0 and SSL integration.

some of these features didn't work well

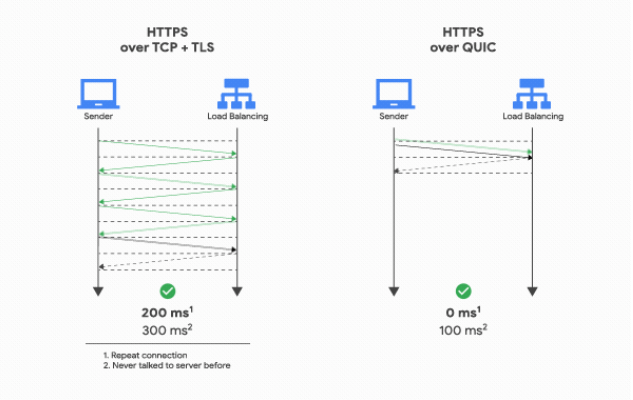
## HTTP 3.0

have been created to fix the problems of HTTP 2.0

has defined to enables QUIC connection

this resolves some TCP problems of multiplexing

When a TCP connection loses some data it will be restarted on QUIC only the data listed will be resent.




QUIC also does not need to check the connection, it just connects.

this new protocol can be used by all servers that already uses TCP

but because it's made by Google and it may take a lot of effort to be applied

QUIC is developed thinking on the new internet