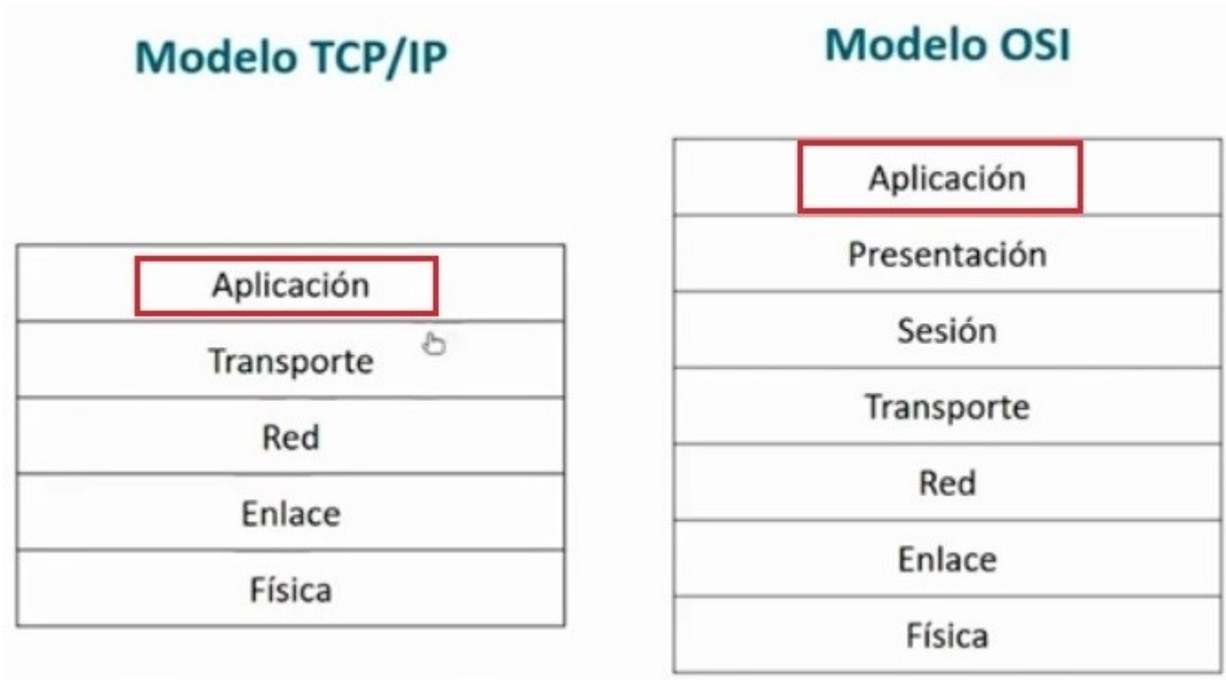
2. Modelos de referencia. Arquitectura OSI, TCP/IP.

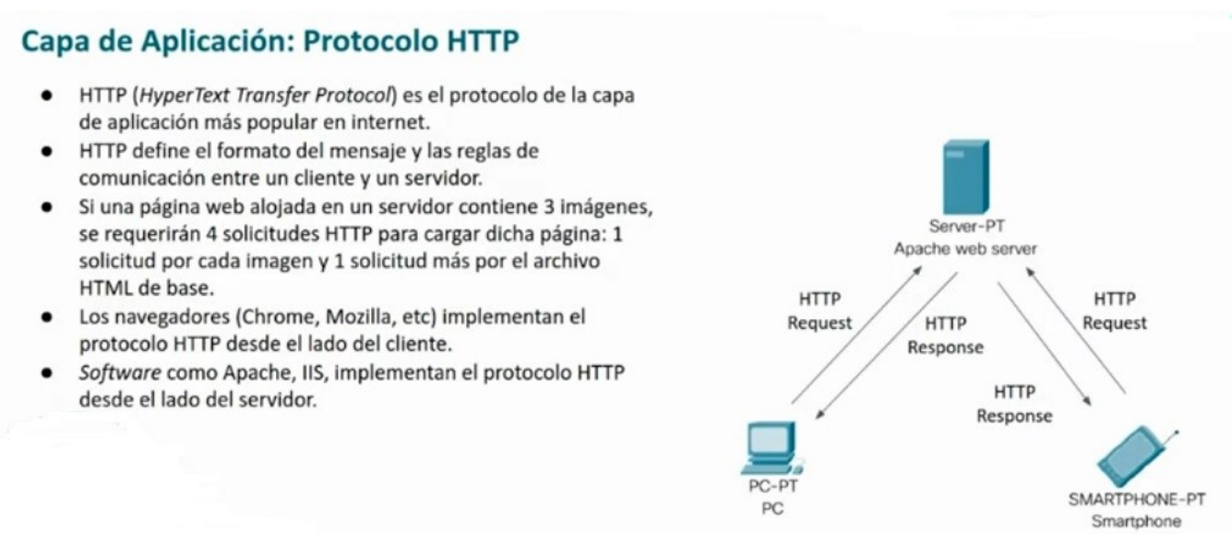
2.2. Protocolos HTTP y DNS

Ahora revisaremos ahora **los protocolos HTTP y DNS**. Estos dos protocolos son muy importantes dentro del modelo TCP IP o dentro del modelo OSI, dependiendo de cuál estemos mirando y son protocolos de la **capa de aplicación**, es decir la primera capa de vista de arriba hacia abajo de cada uno de estos modelos.



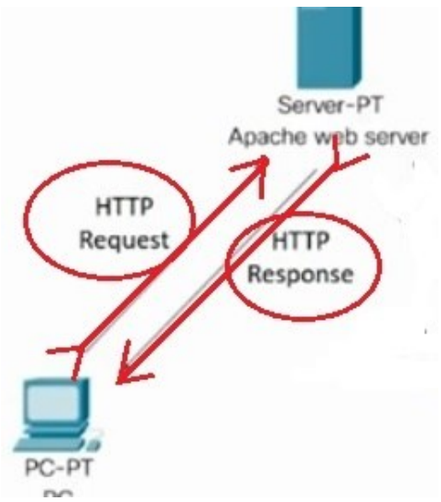
Protocolo HTTP (Capa de aplicación)

Primero vamos a hablar del **protocolo HTTP**. El protocolo HTTP como lo hemos mencionado previamente es uno de los protocolos más utilizados en internet, es el protocolo que utilizamos hoy en día para comunicarnos con

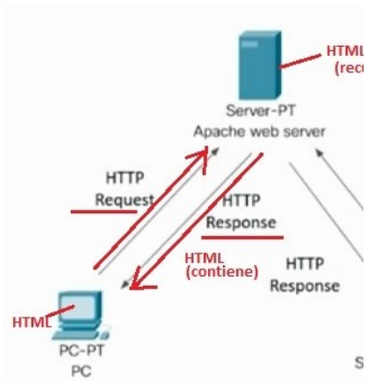


servicios web, servidores web y poder descargar contenido. **HTTP significa a hypertext transfer protocol o protocolo de transferencia de hipertexto** y básicamente está compuesto por 2 formatos de mensajes:

Uno que son mensajes de tipo **HTTP request**, esos mensajes que van desde un equipo de origen hacia un equipo de destino, hacia un servidor. Y mensajes de tipo **HTTP response** que son equipos pues que van en sentido contrario que atienden el respectivo request hecho por el cliente.



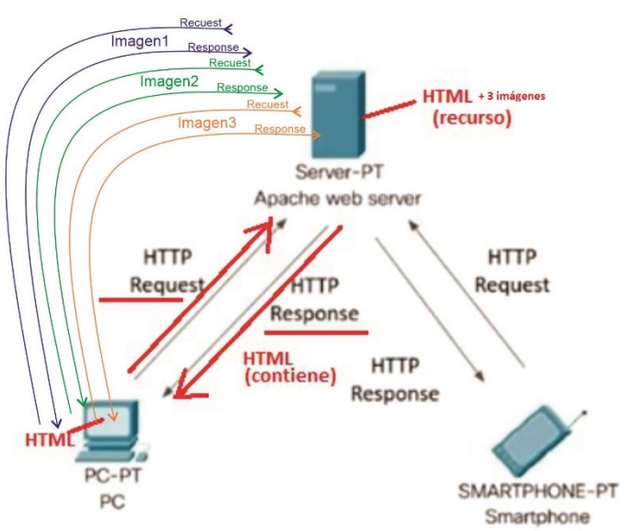
Hay un estándar que define cuál es el formato de estos mensajes, cómo tiene que escribirse los HTTP request, como tiene que escribirse también su HTTP response y que debería esperarse después de enviar una comunicación, cuál debería ser como la forma de recibir la comunicación y esas reglas digamos que existen, ese protocolo que existe de comunicación utilizando HTTP.



Hay algo muy importante a considerar y es que cuando se hace una solicitud, yo puedo tener Un HTML (un recurso) dentro de este web server y cuando hago una solicitud hacia ese HTML, envié un mensaje HTTP request y voy a obtener un mensaje HTTP response que va a contener ese HTML, ese HTML va a quedar cargado en mi navegador de Google Chrome por ejemplo (en la PC).

Pero qué sucede si es HTML tiene a su vez por ejemplo tres imágenes, dentro del HTML hay tres fotos por ejemplo, yo tengo que primero hacer la petición para traer el HTML, pero para poder cargar cada una estas imágenes también tengo que hacer peticiones adicionales, es decir, tendría que hacer una nueva petición para poder traer un nuevo HTTP request para poder traer la imagen número uno con respectiva respuesta, una petición para traer la imagen número dos y finalmente una tercera petición para traer la imagen número tres. Entonces no necesariamente para traer un recurso una página web por ejemplo de un web server solamente se necesita una petición, dependiendo si el HTML tiene objetos que necesito ir a traer, pues voy a tener que hacer nuevas peticiones HTTP request para poder traer todos esos recursos que están componiendo el recurso HTML que acabo de obtener.

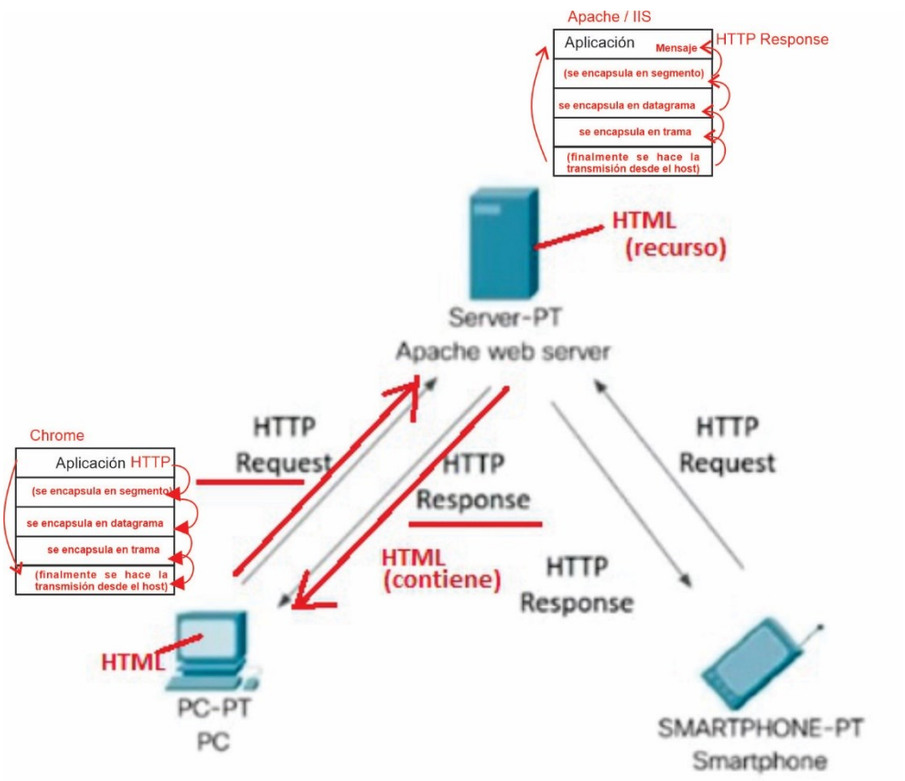
Ahora, hay aplicaciones dentro del PC que hacen uso de ese protocolo HTTP por ejemplo Google Chrome. Google Chrome es la aplicación que hace uso de ese protocolo HTTP para poder hacer las solicitudes hacia un servidor de destino y en el caso por ejemplo acá de ese servidor de destino estaría por ejemplo Apache web server (o podría estar por ejemplo Internet Information server que es otro web server) y ellos hacen uso también de ese protocolo HTTP para poder recibir las peticiones y para también enviar las respectivas respuestas.



Entonces, en la capa de aplicación no está como tal el software. En la capa aplicación

únicamente está el protocolo HTTP que a su vez es utilizado por software, por ejemplo Google Chrome o por ejemplo Apache para poder hacer el envío de los mensajes o para poder hacer el envío de las respuestas también ante los mensajes de solicitud y los mensajes de respuesta.

Por supuesto aquí también hay una encapsulación que se da hacia abajo, el HTTP request que se está enviando por aquí se encapsula como segmento, como datagrama, como trama, se envía por el medio de transmisión cuando llega acá al destino se recibe por la capa física, se pasa de señales eléctricas por ejemplo a bits, se pasa a tramas, a datagramas, a segmentos y finalmente mensajes y este mensaje finalmente ya lo recibe el Apache, recibe la solicitud que se le está haciendo y a su vez el apache puedes envía una respuesta con el HTML por ejemplo, que contiene el portal accedido, lo encapsula una vez, dos veces, tres veces, lo envía por el medio físico hacia el equipo de destino, el equipo destino de nuevo hace el proceso de desencapsulación que ya conocemos y finalmente pues aquí el HTTP response, que sería lo que obtendría acá cierto, el HTTP response que obtendría el equipo cliente se lo pasaríamos a al Google Chrome, y este se encargaría ya de mostrarlo al usuario final o lo muestra de una manera legible para un humano.



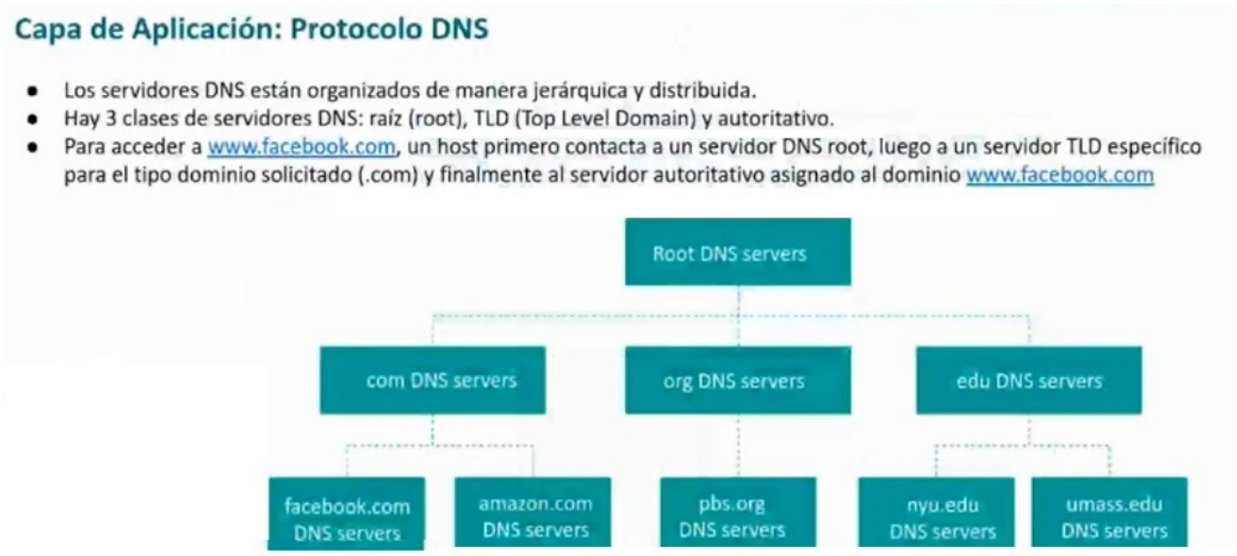
Ahora, para poder hacer una conexión entre un cliente y un servidor, es necesario primero saludar, ese saludo se conoce como el Three Way Handshake, el Three Way Handshake o saludo de tres vías, es algo que siempre va a ocurrir entre el cliente y el servidor cuando se abre justamente una comunicación entonces aquí tenemos un ejemplo de cómo ocurre cuando un equipo cliente quiere comunicarse con el equipo del servidor , lo primero que hace es por supuesto es enviar un mensaje de saludo y dice oye mira quiero comunicarme contigo, el servidor puede aceptar o rechazar esa solicitud entrante, esa comunicación entrante, en este caso pues asumamos que va a aceptar la comunicación entrante y le dice listo, estoy dispuesto a hablar contigo, ya tenemos un canal de comunicación digamos abierto. Todavía no hemos intercambiado información como tal, pero al menos en estos dos primeros intercambios ya tenemos un canal de comunicación abierto. Luego ya viene entonces el cliente, envía un tercer mensaje con la solicitud justamente, la solicitud del recurso, le dice mira finalmente quiero acceder por ejemplo a un recurso llamado index.html, ¿qué hace el Apache? el Apache recibe la solicitud y envía por supuesto HTTP response que contiene el index.html solicitado. Aquí en el número 3 estoy haciendo la solicitud, pero aquí al número 4 ahí sí ya estoy enviando la respuesta, el servidor está enviando la respuesta con el recurso solicitado. ¿Qué sucede para poder lograr esta comunicación? como lo hemos visto aquí pues se tiene que dar justamente este saludo de tres vías, se llama así porque tiene un mensaje de saludo del cliente al servidor, una confirmación del servidor al cliente y luego ya por fin en el tercer mensaje, miren que solamente hasta el tercer mensaje se envía la solicitud del recurso que quiero hacer, el HTTP request se enviaría sólo hasta el tercer mensaje, por eso se llama de tres vías porque se necesitan primero estos tres pasos para poder enviar

justamente la petición. Hay algo muy importante también y es un concepto que se llama RTT o round-trip time, ese RTT contiene el tiempo que tarda el mensaje en ir y volver, en este caso del cliente al servidor y del servidor al cliente, entonces cuando hacemos una petición al portal de cualquier tipo miren que se están dando de manera implícita dos RTTs, un RTT para poder hacer el saludo, el establecimiento de la comunicación inicial y el segundo RTT para poder ahí sí traer el recurso que me interesa a mi como usuario.

Protocolo DNS (Capa de aplicación)

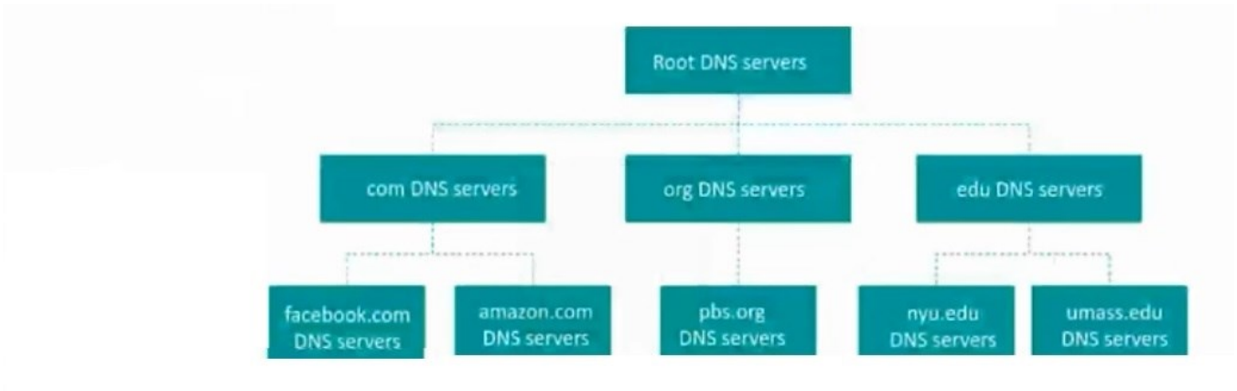
El siguiente que veremos es el **protocolo DNS** también tiene muchísima relevancia, es un protocolo que permite que cuando accedamos un dominio como Facebook podemos traducirlo a una dirección IP, por ejemplo, una dirección IP como esta, que es el identificador de ese dominio, identificador lógico de ese dominio.

Nosotros pues como humanos no nos resulta tan fácil memorizar direcciones IP de los dominios, lo que, si nos resulta más fácil de memorizar por ejemplo el nombre de dominio cierto Facebook, Google, YouTube, Gmail, entonces necesitamos un servidor DNS, que funciona con el protocolo DNS claramente, para poder hacer la conversión de ese nombre de dominio a la dirección IP respectiva.

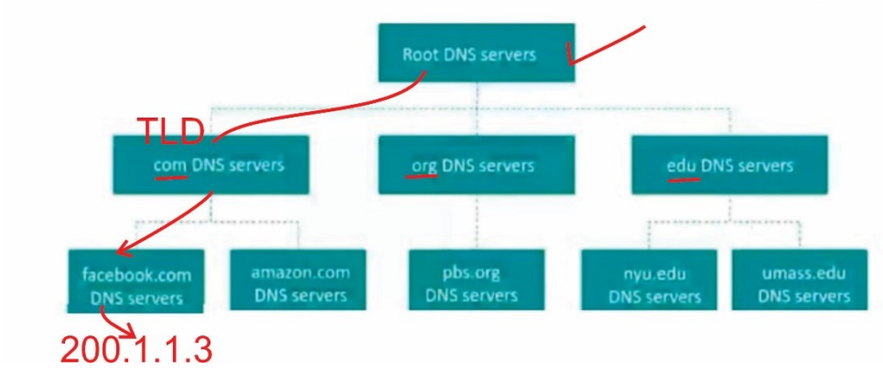


Entonces el **protocolo DNS** tiene esa utilidad, DNS significa justamente **domain name system** o **sistema de nombres de dominio**, porque es un servidor que contiene la equivalencia entre el nombre de dominio y la respectiva dirección IP. Hay algo muy importante para entender aquí y es que yo por ejemplo cuando quiero acceder a una página como Facebook, claramente en este caso yo hago la petición hacia mi servidor DNS, mi servidor DNS hace algo por detrás de tal forma que me entrega cual cuál es la dirección IP de Facebook, y me dice mira la IP de Facebook es la 200.1.10.3 por ejemplo, yo con esa dirección IP ya puedo entonces hacer mensajes dirigidos directamente hacia Facebook, porque ya conozco la dirección IP del destino hacia donde quiero comunicarme.

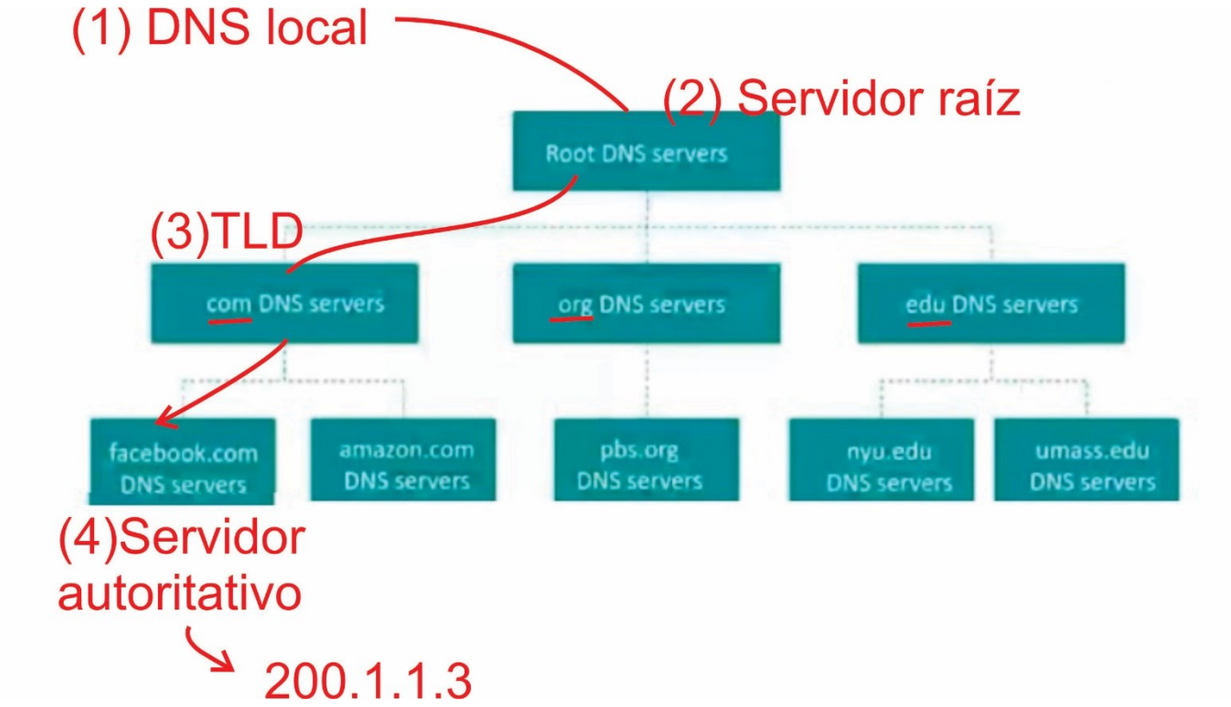
Pero lo que el servidor DNS hace por detrás para poder lograr darme esa dirección IP está vinculado a esa estructura jerárquica que tenemos aquí representada en este árbol.



Tengo primero un servidor root DNS, al cual se le hace la primera petición. El servidor root DNS lo que hace es identificar si se trata de un dominio tipo .edu o educativo, tipo .org u organizacional, tipo .com o comercio y dependiendo del tipo de dominio que yo esté intentado conectar, él me va a decir cuál es el servidor a cuál yo debería contactar. Me dice mira, para facebook.com me dice contacté a ese servidor de tipo TLD o Top Level Domain, que de él te va a dar la respuesta. Entonces después de que se contacta a este servidor TLD ese TLD a su vez tiene también distribuidos los servidores que se encargan de cada tipo de dominio, entonces él puede si me mira para contactarte para resolver el nombre de dominio Facebook a una dirección IP contacta a su vez a estos servidores DNS que son los encargados de resolver los dominios de facebook.com. Finalmente, éste ya va a indicarnos cual la dirección IP que nos interesa para el dominio facebook.com.



Entonces siempre se contacta primero, bueno primero el servidor DNS local, el servidor DNS local se encarga de contactar primero al servidor raíz, el servidor raíz nos redirige a un servidor de tipo TLD, el servidor de tipo TLD nos redirige hacia un servidor de tipo autoritativo que se llaman estos de acá, los de la última línea y ese servidor de tipo autoritativo, si nos va a dar la dirección IP que necesitamos asociada al dominio facebook.com que era el que nos interesaba. Es muy importante tener claro esta jerarquía porque las peticiones de tipo DNS no se resuelven de manera directa, se tiene que contactar a estos servidores de manera jerárquica


(1) DNS local
(2) Servidor raíz
Root DNS servers
(3)TLD
com DNS servers
org DNS servers
edu DNS servers
facebook.com DNS servers
amazon.com DNS servers
pbs.org DNS servers
nyu.edu DNS servers
umass.edu DNS servers
(4)Servidor autoritativo
200.1.1.3