Informe Tarea 3.1
Sistemas Operativos y Redes

Integrantes:

- Iñaki Errázuriz.
- Johnny Donoso A.

Profesor:

Claudio Álvarez G.

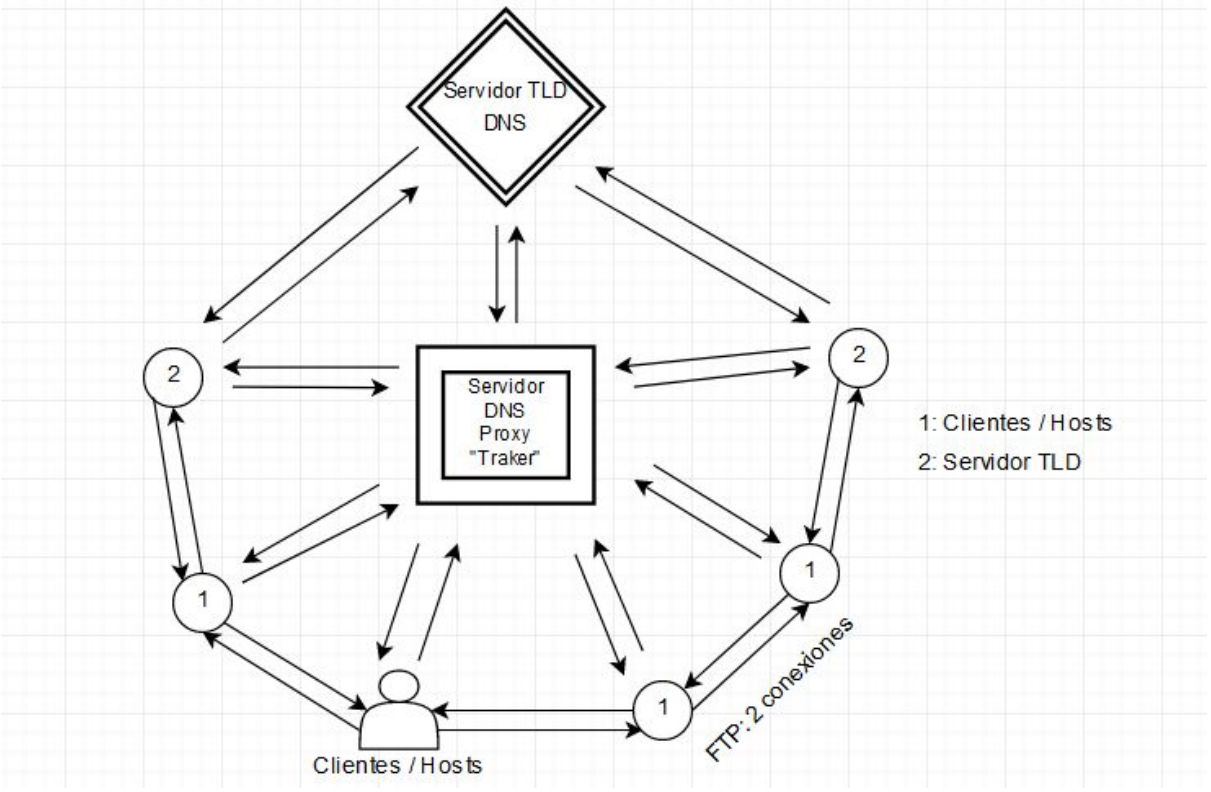
Facultad de Ingeniería
Universidad de los Andes
2018

Introducción

Existen distintos tipos de protocolo cuando se habla de comunicación de hosts mediante la red, cada una de ellas es específica en base a la funcionalidad que tiene. Para esta parte de la tarea mostraremos y explicaremos mediante este informe el protocolo que implementaremos y que posteriormente vamos a crear en base a los requisitos del enunciado.

Como la aplicación especificada en el archivo requería que los Host terminales funcionaran tanto como buscadores de archivo y también como servidores FTP, es que decidimos basar nuestro protocolo en un espacio en una Arquitectura Híbrida, que ocupa el paradigma Cliente/Servidor y P2P.

A continuación un esquema general de nuestro protocolo:



1. Tipos de hosts que interactúan en el protocolo.

1.- **Peers**: Estos Clientes/Servidores son nuestros hosts terminales de red. Como nuestro protocolo está basado en una arquitectura Híbrida(algo así como lo que ocupa Skype) esto puede ser posible. Ellos son los que inician la búsqueda de archivos y reciben como respuesta, según al servidor que le pregunten(DNS / Host Servidor terminal de red) una dirección IP(si el host cliente se comunica con el Servidor DNS) o una respuesta positiva para iniciar una transferencia de archivos bajo el protocolo FTP:2 conexiones, si es que el Cliente ya recibió la respuesta de los Proxys DNS y ya tiene la dirección IP en donde se encuentra el archivo alojado(Servidor terminal de red).

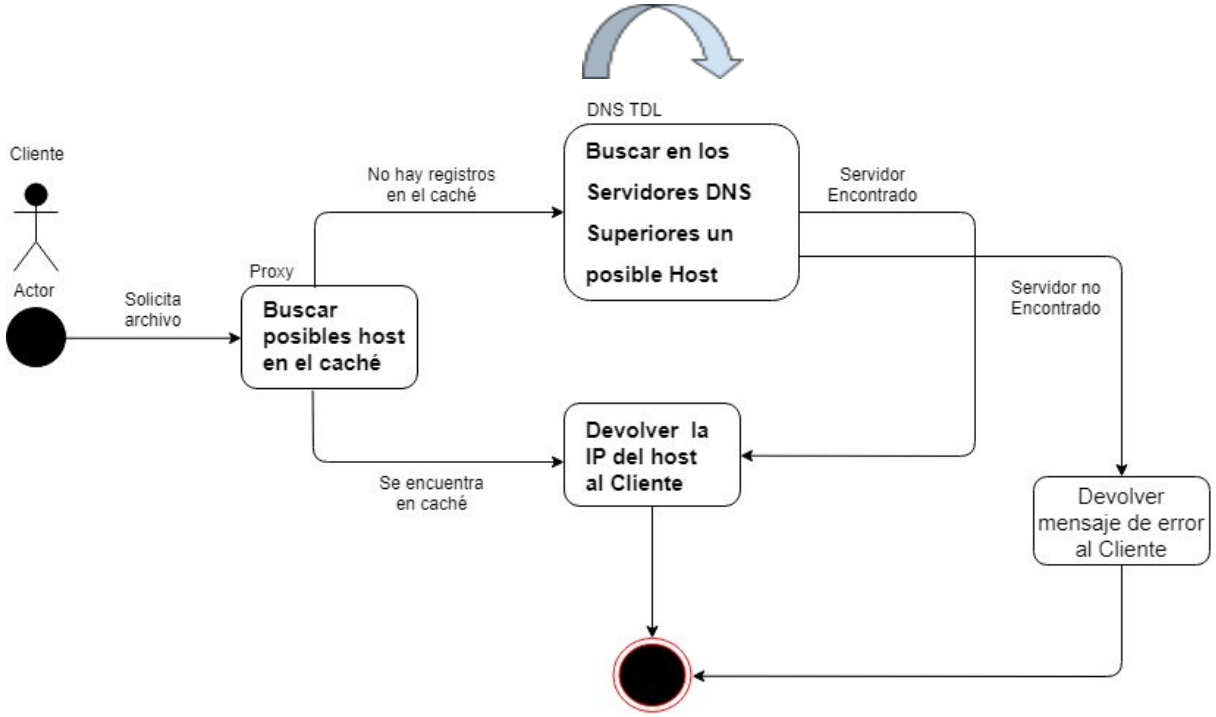
2.- **Proxys/Trackers**: Estos “trackers” serán del tipo DNS autoritario. Un host de tipo DNS nos sirve para resolver todos los requerimientos de el enunciado ya que puede manejar caché de direcciones bajo nombres lógicos(como lo es el string con el nombre del archivo) y además, si no lo mantiene en el caché(que mantiene los nombres de archivo y sus ip’s disponibles en un stack FIFO), puede iniciar la búsqueda jerárquica que manejan todos los servidores DNS y que pueden conectarse con los DNS mayores como los servidores TLD(asumiendo que nuestros proxys/trackers serán propios de nuestra appi hosteados por un TLD de dominio cl,etc).

2. Interacciones posibles en el protocolo. De preferencia, ilustrar con diagramas formales, por ejemplo, UML de secuencia y/o de estado.

Nuestro protocolo implementa dos protocolos muy ocupados en Internet. El protocolo DNS (para búsqueda de servidores que alojen cierto archivo) y el protocolo FTP (para descarga de archivos).

Primero vamos a ver las interacciones de búsqueda de archivos (basado en protocolo DNS):

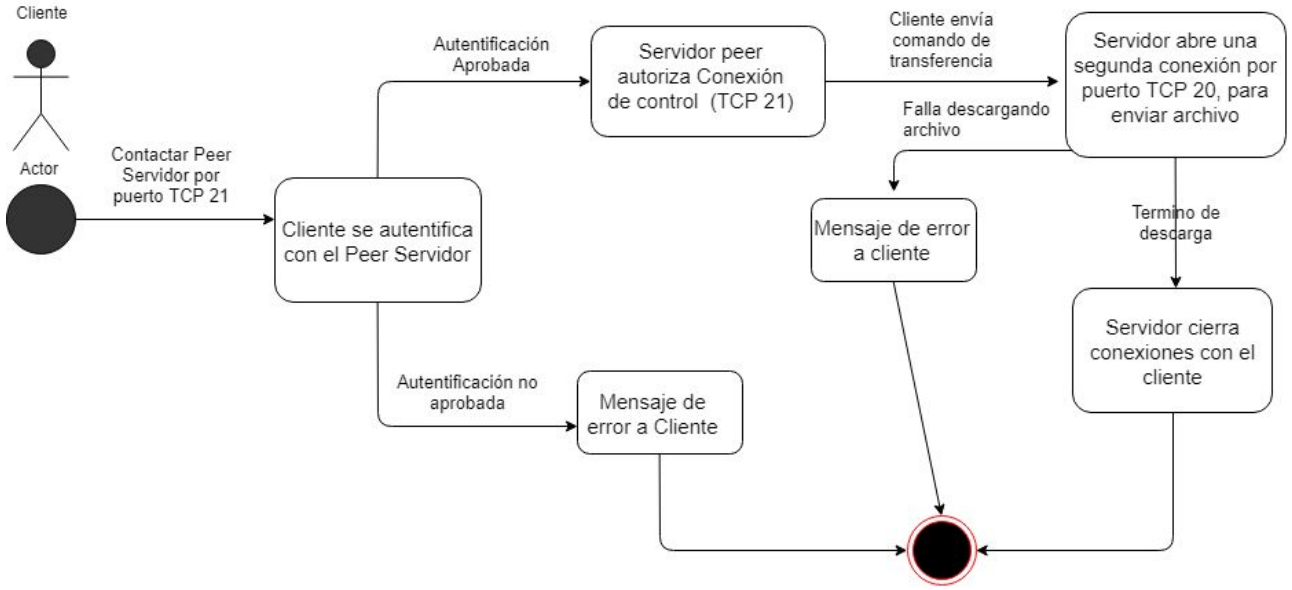
**Host = Peer Servidor.



El cliente al solicitar el archivo, necesita traducir ese nombre de archivo a una Dirección IP(dirección de el servidor donde se encuentra el archivo). Para ello, lo mejor que podemos ocupar, son los proxys DNS. Estos Proxys DNS(Second Level Domain) manejan un caché, un registro. Al llegar esta petición de el Cliente, el servidor DNS busca si el lo tiene en sus registros, si lo encuentra, lo manda directamente al cliente, para que el cliente pueda comenzar la descarga, ya conociendo el IP del servidor a contactar para iniciar el protocolo de transferencia de archivos. Si el Proxy más cercano al cliente no lo encuentra, este re-dirige la consulta a servidores DNS superiores(Top Level Domains) hasta encontrar el Servidor. Si lo encuentra, le devuelve la IP al servidor DNS correspondiente en la escala jerárquica, hasta llegar al Tracker y este lo devuelva al cliente. Y si no lo encuentra, se debe mandar un mensaje de error al cliente y así finalizar la interacción.

Ahora veamos las interacciones para la descarga de archivos:

Como el peer cliente ya posee la dirección IP del peer servidor, ahora sólo basta establecer la conexión entre ellos para comenzar la descarga de el archivo. Para ello ocuparemos algo muy parecido al protocolo FTP: 2 conexiones visto en clases.



Explicación de la interacción:

1.- Cliente contacta al Servidor

2.- Servidor ve si autentifica o no al Cliente para que realice la descarga.

3.- Si la conexión se aprueba, el Servidor autoriza una conexión de Control con el cliente(que se asegura de revisar la llegada de paquetes de archivo). Si la conexión no se aprueba, se manda un mensaje de error al cliente, y se termina el proceso.

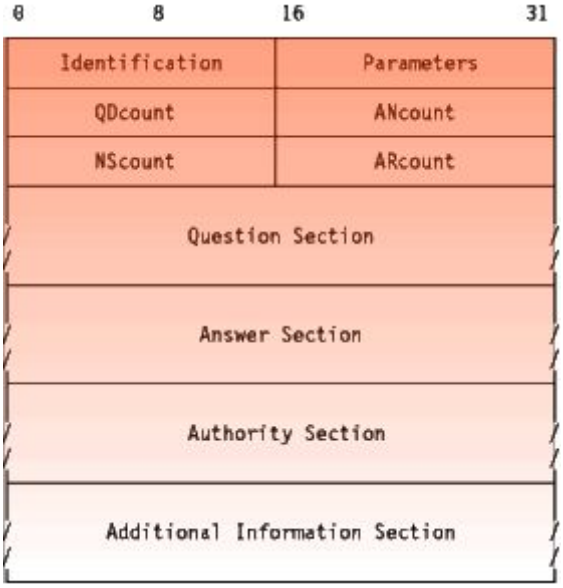
4.- Después de haber hecho la conexión de control, el cliente envía un comando de transferencia, que abre un puerto TCP 20 (para el envío de archivo) y comienza la descarga.

5.- Al término de la descarga, estas conexiones entre Peer Cliente y Peer Servidor se cierran, y se termina el proceso correspondiente al Protocolo de descarga.Si la descarga falla por algún motivo, se le envía un mensaje de error al Cliente y se termina la secuencia.

3. Tipos y formatos detallados de mensaje intercambiados en las interacciones entre los hosts contemplados en el protocolo.

En esta primera parte de el punto 3 analizaremos los mensajes intercambiados en la primera interacción mostrada en el punto 2, los mensajes de Cliente a los Servidores DNS y sus respuestas.
Esta interacción ocupa mensajes designados y diseñados específicamente para las conexiones DNS, llamados mensajes DNS.

Estos siguen el siguiente formato:



Para hacer una consulta al servidor DNS, este responderá al nombre de el archivo, ya que cuando se registra un archivo nuevo al servidor DNS, para poder ser descargado por otros usuarios, este se guarda con un alias(gracias a CNAME(perteneciente a los códigos posibles de los mensajes DNS)) así, los nombres de dominio quedarán guardado bajo el nombre del archivo a buscar y podrán ser consultados por los demás Peers clientes.
El servidor DNS al momento de guardar el registro de un nuevo archivo, debe ejecutar la siguiente orden:

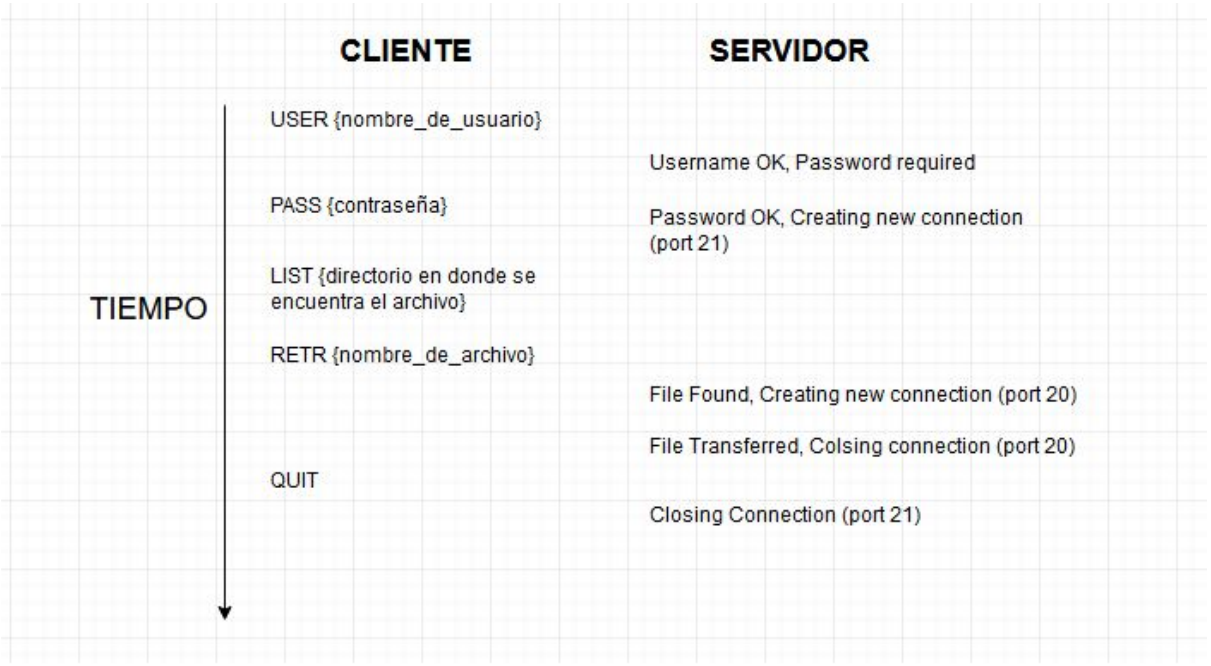
→ $ dig <nombre_server_original> CNAME <nombre_Archivo>

Para buscar un archivo entonces, el tipo de mensaje será

→ $ dig @ip_cliente CNAME <nombre_archivo>

Para la segunda parte, una vez que el cliente obtenga la dirección IP del peer servidor comenzará la interacción entre estos 2 para poder bajar el archivo.

Este va a ser el protocolo FTP el cual iniciara, como se vio antes, con una conversación de verificación entre estos 2. Los mensajes a mostrar, y la “conversación” que tendran sera esta:



Aquí se puede ver una comunicación exitosa entre el cliente y servidor para la descarga de un archivo.

El cliente necesita ingresar bien sus datos para autorizar la conexión, de lo contrario aparecerá:

- **User information invalid, can't create connection**

Si al momento de buscar el archivo usando LIST no lo encuentra, arrojara este mensaje (y se cerrará la coneccion):

- **File could not be found, the connection will be closed**

Al momento de la descarga, si falla en descargarse el archivo, arrojara este mensaje (y se cerrará la coneccion):

- **File could not be successfully downloaded, the connection will be closed**

Todo lo mencionado anteriormente es teórico, y para la siguiente entrega se planea tener este protocolo completamente funcional con todas las características dichas en este informe.