Protocolo de control de transmisión - Capa de tranporte

Introducción

Recuerden que en la sesión anterior estuvimos discutiendo acerca de cómo la capa de tranporte proporciona un tipo de direccionamiento (virtual) que permite que la información recibida en un extremo pueda ser entregada a la aplicación correcta. Para realizar esta práctica de laboratorio, vamos a introducir un término/concepto fundamental: el socket. Este es un concepto abstracto, pero de forma simple un socket es un punto final virtual donde se lleva a cabo la comunicación interproceso. De forma visual, el socket se representa como la unión de la dirección IP más el número de puerto que utiliza el proceso. Por ejemplo, para un proceso que utiliza *HTTP* en un dispositivo con dirección *a.b.c.d* entonces la representación en forma de socket sería **{a.b.c.d:80}**. Existen dos procesos que (como fin) intercambian información, al primer proceso que inicia la comunicación lo etiquetamos como cliente y al segundo como el servidor. La distinción entre el cliente y servidor es importante porque cada uno utiliza la interfaz de socket de forma diferente al ciertos pasos en la comunicación. En esta guía nos vamos enfocar primeramente en el cliente el cual, como ya sabemos, su función ya sabemos es iniciar una comunicación con el servidor que espera de forma pasiva para ser contactado. Para trabajar con sockets debemos utilizar la interfaz de programas de aplicación con el mismo nombre. Esta contiene las funciones principales para establecer las comunicaciones, tal como se muestra en la siguiente imagen:

Como se puede observar, el servidor ejerce mayor trabajo en cuanto a las funciones que maneja y debe preparar para recibir inicios de conexión.

Descripción

La comunicación de cliente TCP usual requiere de los siguientes pasos:

1. Crear un socket utilizando la función *socket()*
2. Establecer una conexión utilizando *connect()*
3. Establecer comunicaciones utilizando *send()* and *recv()*
4. Finalizar la comunicación utilizando *close()*

Creacion de un socket en el cliente

Importando el modulo socket y confirmando que esta cargado

```
from socket import*
import socket
# confirmar que el modulo esta cargado
```

```
import sys
'socket' in sys.modules
```

Una de las funciones principales del modulo *socket* es la creación de sockets. Para lograrlo, una vez que hemos cargado el modulo para crear un socket (asociar una direccion IP con un puerto) lo que debemos hacer es llamar la clase **socket()**:

```
nuevo_socket= socket.socket(AF_INET, SOCK_STREAM, 0)
```

Para crear un socket necesitamos pasar argumentos como la familia de socket que vamos a utilizar, el tipo de socket (el protocolo de tranporte a utilizar), y la variación del protocolo dentro de la familia y tipo de socket. Por lo general, este se deja en 0. En caso de AF_INET (*Address Format Internet*), estamos diciendo que vamos a utilizar los sockets de internet como se le conoce (o sockets Berkeley/BSD). Por lo anterior es que la creación del socket pudo haberse definido de la siguiente manera:

```
nuevo_socket= socket.socket(family=AF_INET, type=SOCK_STREAM, proto=0)
```

*Pregunta 1: Como habrán visto, en la creación del socket para definir el tipo de socket que utilizamos es SOCK_STREAM para decir que es TCP. STREAM se puede traducir como **flujo**, ¿por qué se le conoce a TCP como un protocolo de flujo?*

Cuando el socket (como objeto) ha sido creado, necesitamos establecer la conexión con el servicio. Para ello, hacemos una llamada al método *connect()* que inicia el establecimiento de la conexión por medio del reconocimiento de tres vías. En este caso vamos crear una conexión a un servidor público.

```
servidor_destino= 'www.google.com'
puerto_destino= 80
nuevo_socket.connect(servidor_destino, puerto_destino)
```

Como podrán recordar, el *reconocimiento de tres vias* es importante porque nos permite saber que ambos lados (cliente, servidor) son alcanzables en la red.

Creación de un socket cliente local

Lo que hemos visto anteriormente lo vamos a poner junto para que se puede ejecutar en forma de script.

```
#importamos el modulo socket
import socket

#Definimos las variables que vamos a pasarle a la clase socket
```

```
destino = 'www.linux.org'
puerto_destino = 80
bufer = 4096
designacion_socket = (destino, puerto_destino)


if __name__ == '__main__':
    socket_cliente = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    socket_cliente.connect(designacion_socket)

    while True:
        datos = 'GET / HTTP/1.0\r\n\r\n'
        if not datos:
            break
        socket_cliente.send(datos.encode('utf-8'))
        datos = socket_cliente.recv(bufer)
        if not datos:
            break
        print(datos.decode('utf-8'))
    socket_cliente.close()
```

*Ejercicio 1: Explique el resultado obtenido desde el cliente TCP ¿Qué diferencia en el resultado obtiene si cambia la variable **destino** con los valores '**www.unan.edu.ni**' o '**www.yahoo.com**'?*