

Universidad Nacional Autónoma de México
Facultad de Ciencias

Asignatura: Redes de computadoras
Semestre: 2024-1

Profesor: Javier León Cotonieto

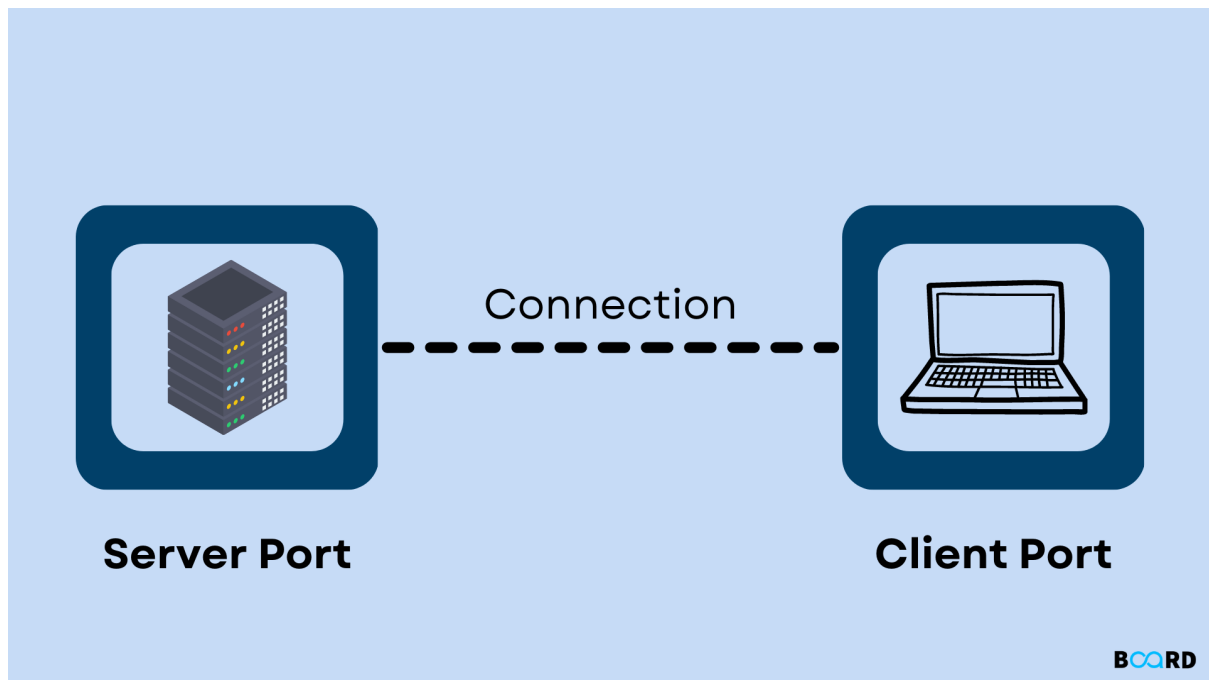
Ayudantes: Magdalena Reyes Granados
Itzel Gómez Muñoz
Sandra Plata Velázquez

Práctica 11. "Implementación de Sockets"

Equipo 5

Integrantes:

- **Almanza Torres José Luis**
- **Jimenez Reyes Abraham**
- **Martínez Pardo Esaú**



Ejercicio 1

Crear una comunicación Cliente – Servidor mediante un socket que cumplan con los siguientes requerimientos:

1) Investigue y describa 5 puertos conocidos y 6 puertos designados.

Puertos conocidos:

- Puerto 80: Es el puerto utilizado por HTTP, el protocolo de transferencia de hipertexto que se utiliza para acceder a todas las páginas web. La mayoría de los sitios web lo utilizan para transmitir todo su contenido.
- Puerto 443: Es el puerto utilizado por HTTPS, el protocolo de transferencia de hipertexto seguro. Permite la transmisión segura de datos por toda la red y se utiliza para transacciones en línea, como compras, transacciones bancarias, bolsa, entre otros.
- Puerto 21: Es el puerto utilizado por el protocolo FTP (File Transfer Protocol), que se utiliza para la transferencia de archivos entre dispositivos en una red. Su uso principal es para la transferencia de archivos grandes, como imágenes o videos entre otros archivos.
- Puerto 22: Es el puerto utilizado por el protocolo SSH (Secure Shell), que se utiliza para la conexión segura a un servidor remoto. Es utilizado por los administradores de sistemas para la gestión de servidores, entre otras aplicaciones con funciones similares.
- Puerto 25: Es el puerto utilizado por el protocolo SMTP (Simple Mail Transfer Protocol), que se utiliza para la transferencia de correo electrónico. Se utiliza para poder enviar correos electrónicos desde un cliente de correo electrónico a un servidor de correo electrónico.

Puertos designados:

- Puerto 53 UDP: Es el puerto utilizado por el protocolo DNS (Domain Name System), que se utiliza para traducir nombres de dominio en direcciones IP. Es esencial para la navegación por internet hoy en día, ya que permite que los navegadores web accedan a los sitios web utilizando nombres de dominio y no direcciones IP que son más complicadas.
- Puerto 67: Los servidores del Protocolo de configuración dinámica de host usan el puerto UDP 67 para escuchar las solicitudes.
- Puerto 68: Por su parte los clientes DHCP se comunican en el puerto UDP 68.
- Puerto 110: Es el puerto utilizado por el protocolo POP3 (Post Office Protocol version 3), que se utiliza para recibir correo electrónico en un cliente de correo electrónico. En combinación con el puerto 25.
- Puerto 389 - LDAP (Lightweight Directory Access Protocol): LDAP se utiliza para acceder y mantener servicios de directorio. El puerto 389 es designado para conexiones LDAP no cifradas, mientras que el puerto 636 se utiliza para conexiones LDAP seguras mediante SSL/TLS.
- Puerto 3306: Puerto usado por las bases de datos MySQL.

2) Investigue y describa 3 aplicaciones de sockets.

- Los sockets le permiten intercambiar información entre procesos en la misma máquina o a través de una red, distribuir el trabajo a la máquina más eficiente y permitir fácilmente el acceso a datos centralizados.
- Los sockets proporcionan una interfaz suficientemente general para permitir que las aplicaciones basadas en red se construyan independientemente de los recursos de comunicación subyacentes. También apoyan la construcción de programas distribuidos contruidos sobre primitivos de comunicación.
- Los WebSockets son un protocolo de comunicación bidireccional sobre un único canal TCP, implementado utilizando sockets. Se utilizan en aplicaciones web para habilitar la comunicación en tiempo real entre el servidor y el cliente. Esto es particularmente útil en aplicaciones como chats en línea, juegos multijugador y actualizaciones en tiempo real, donde la baja latencia es crucial.
- Los sockets son ampliamente utilizados en aplicaciones de transferencia de archivos, como FTP. El FTP utiliza dos canales de comunicación: uno para comandos y otro para la transferencia de datos. Los sockets permiten la conexión entre el cliente y el servidor para enviar comandos y archivos de manera eficiente.

3) Socket programado en Python.

- a) Datos de identificación de las personas que elaboraron el código.
- b) Crear un socket que genere comunicación por TCP (STREAM).
- c) Que se pueda conectar a cualquier dispositivo en la red (INET).
- d) El puerto efímero por ocupar es libre.
- e) Sea capaz de enviar - recibir mensajes.

```

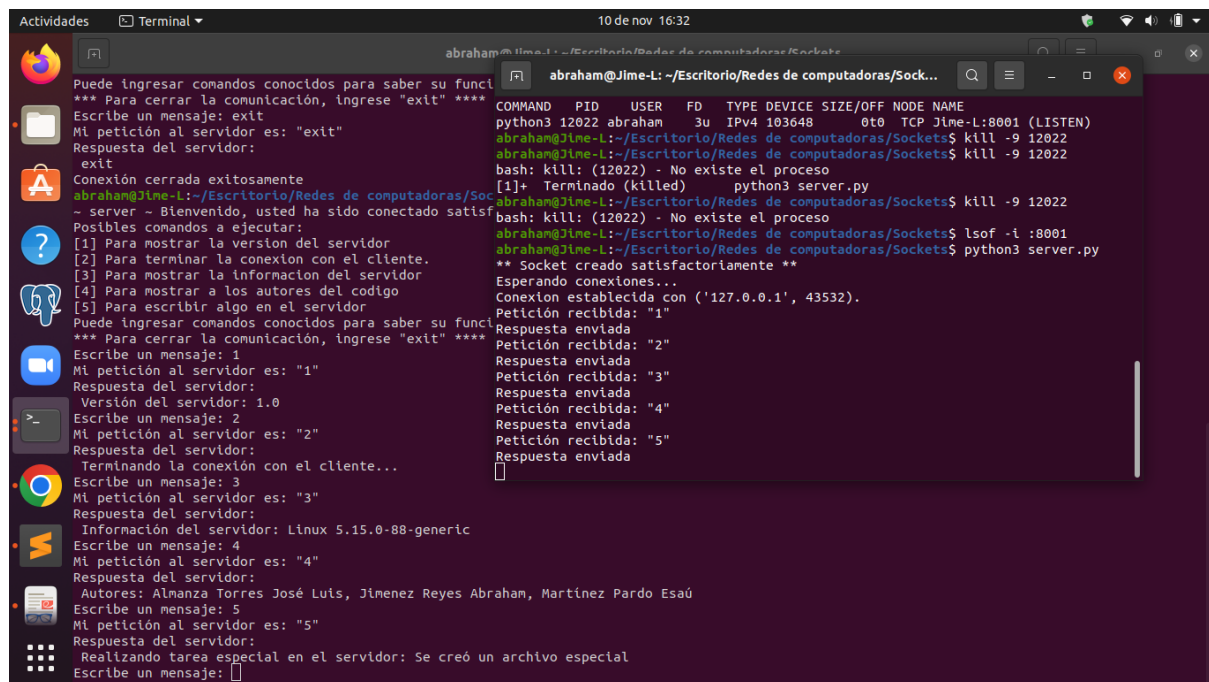
abraham@Jlme-L: ~/Escritorio/Redes de computadoras/Sockets
** Comunicación cerrada con éxito **
^Z
[1]+  Detenido                  python3 server.py
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 server.py
** Socket creado satisfactoriamente **
Traceback (most recent call last):
  File "server.py", line 23, in <module>
    server_socket.bind((hostname, port))
OSError: [Errno 98] Address already in use
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ lsof -i :8001
COMMAND  PID  USER  FD  TYPE  DEVICE  SIZE/OFF  NODE  NAME
python3 12022 abraham 3u  IPv4 103648      0t0  TCP Jlme-L:8001 (LISTEN)
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ kill -9 12022
bash: kill: (12022) - No existe el proceso
[1]+  Terminado (killed)      python3 server.py
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ kill -9 12022
bash: kill: (12022) - No existe el proceso
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ lsof -i :8001
COMMAND  PID  USER  FD  TYPE  DEVICE  SIZE/OFF  NODE  NAME
python3 12022 abraham 3u  IPv4 103648      0t0  TCP Jlme-L:8001 (LISTEN)
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 server.py
** Socket creado satisfactoriamente **
Esperando conexiones...
Conexion establecida con ('127.0.0.1', 43532).
Mi petición al servidor es: "hola"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: holla
Mi petición al servidor es: "holla"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: exit
Mi petición al servidor es: "exit"
Respuesta del servidor:
exit
Conexion cerrada exitosamente
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 clienteUno.py
~ server ~ Bienvenido, usted ha sido conectado satisfactoriamente al servidor :D
Posibles comandos a ejecutar:
[1] Para mostrar la version del servidor
[2] Para terminar la conexion con el cliente.
[3] Para mostrar la informacion del servidor
[4] Para mostrar a los autores del codigo
[5] Para escribir algo en el servidor
Puede ingresar comandos conocidos para saber su función, como ls, mkdir, cd, sudo.
*** Para cerrar la comunicación, ingrese "exit" ****
Escribe un mensaje:

```

4) Servidor

- a) Capaz de conectar varios clientes (al menos 3).
- b) Conexión cifrada.
- c) Enviar - recibir mensajes.
- d) Enviar un mensaje de bienvenida cada que se conecte un cliente nuevo.
- e) Crear al menos 5 funciones nuevas (comando).

i) Comandos que hagan funciones específicas en el servidor: terminar conexión con el cliente, mostrar quién lo elaboró, funciones especiales, etcétera.



```
abraham@jlime-l: ~/Escritorio/Redes de computadoras/Socket...
Puede ingresar comandos conocidos para saber su funci
*** Para cerrar la comunicación, ingrese "exit" ****
Escribe un mensaje: exit
Mi petición al servidor es: "exit"
Respuesta del servidor:
exit
Conexión cerrada exitosamente
abraham@jlime-l:~/Escritorio/Redes de computadoras/Socket...
~ server ~ Bienvenido, usted ha sido conectado satisf
Posibles comandos a ejecutar:
[1] Para mostrar la versión del servidor
[2] Para terminar la conexión con el cliente.
[3] Para mostrar la información del servidor
[4] Para mostrar a los autores del código
[5] Para escribir algo en el servidor
Puede ingresar comandos conocidos para saber su funci
*** Para cerrar la comunicación, ingrese "exit" ****
Escribe un mensaje: 1
Mi petición al servidor es: "1"
Respuesta del servidor:
Versión del servidor: 1.0
Escribe un mensaje: 2
Mi petición al servidor es: "2"
Respuesta del servidor:
Terminando la conexión con el cliente...
Escribe un mensaje: 3
Mi petición al servidor es: "3"
Respuesta del servidor:
Información del servidor: Linux 5.15.0-88-generic
Escribe un mensaje: 4
Mi petición al servidor es: "4"
Respuesta del servidor:
Autores: Almanza Torres José Luis, Jiménez Reyes Abraham, Martínez Pardo Esaú
Escribe un mensaje: 5
Mi petición al servidor es: "5"
Respuesta del servidor:
Realizando tarea especial en el servidor: Se creó un archivo especial
Escribe un mensaje:

COMMAND PID USER FD TYPE DEVICE SIZE/OFF NODE NAME
python3 12022 abraham 3u IPv4 103648 0t0 TCP jlime-l:8001 (LISTEN)
abraham@jlime-l:~/Escritorio/Redes de computadoras/Socket...$ kill -9 12022
bash: kill: (12022) - No existe el proceso
[1]+ Terminado (killed) python3 server.py
abraham@jlime-l:~/Escritorio/Redes de computadoras/Socket...$ kill -9 12022
bash: kill: (12022) - No existe el proceso
abraham@jlime-l:~/Escritorio/Redes de computadoras/Socket...$ lsof -i :8001
abraham@jlime-l:~/Escritorio/Redes de computadoras/Socket...$ python3 server.py
** Socket creado satisfactoriamente **
Esperando conexiones...
Conexión establecida con ('127.0.0.1', 43532).
Petición recibida: "1"
Respuesta enviada
Petición recibida: "2"
Respuesta enviada
Petición recibida: "3"
Respuesta enviada
Petición recibida: "4"
Respuesta enviada
Petición recibida: "5"
Respuesta enviada
```

5) Cliente

a) Establecer comunicación con el servidor.

b) Ingresar un comando conocido (ls, mkdir, cd, sudo, etcétera) y que el servidor despliegue un texto o una imagen que contenga el comando, una breve descripción y un ejemplo (al menos 6).

c) En caso de que el comando esté mal escrito o no sea alguno de los 6, se genera un error.

```
Actividades Terminal 10 de nov 16:37
abraham@Jime-L: ~/Escritorio/Redes de computadoras/Sockets

ERROR: Comando no encontrado.
Escribe un mensaje: 9
Mi petición al servidor es: "9"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 0
Mi petición al servidor es: "0"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 9
Mi petición al servidor es: "9"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 8
Mi petición al servidor es: "8"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 9
Mi petición al servidor es: "9"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: jj
Mi petición al servidor es: "jj"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: nyn
Mi petición al servidor es: "nyn"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: ny
Mi petición al servidor es: "ny"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: fy
Mi petición al servidor es: "fy"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 
```

Cerramos la sesion

```
Actividades Terminal 10 de nov 16:39
abraham@Jime-L: ~/Escritorio/Redes de computadoras/Sockets

Escribe un mensaje: 0
Mi petición al servidor es: "0"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 9
Mi petición al servidor es: "9"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 8
Mi petición al servidor es: "8"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: 9
Mi petición al servidor es: "9"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: jj
Mi petición al servidor es: "jj"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: nyn
Mi petición al servidor es: "nyn"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: ny
Mi petición al servidor es: "ny"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: fy
Mi petición al servidor es: "fy"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: exit
Mi petición al servidor es: "exit"
Respuesta del servidor:
exit
Conexión cerrada exitosamente
abraham@Jime-L: ~/Escritorio/Redes de computadoras/Sockets$

Respuesta enviada
Petición recibida: "9"
Respuesta enviada
Petición recibida: "8"
Respuesta enviada
Petición recibida: "9"
Respuesta enviada
Petición recibida: "jj"
Respuesta enviada
Petición recibida: "nyn"
Respuesta enviada
Petición recibida: "ny"
Respuesta enviada
Petición recibida: "fy"
Respuesta enviada
exit
ei
f
Petición recibida: "exit"
Respuesta enviada
~ Cerrando comunicación con ('127.0.0.1', 43532)... ~
** Comunicación cerrada con éxito **
```

Con 3 dispositivos

```
Abraham@Jlme-L: ~/Escritorio/Redes de computadoras/Sockets
ERROR: Comando no encontrado.
Escribe un mensaje: jj
Mi petición al servidor es: "jj"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: nyn
Mi petición al servidor es: "nyn"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: ny
Mi petición al servidor es: "ny"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: fy
Mi petición al servidor es: "fy"
Respuesta del servidor:
ERROR: Comando no encontrado.
Escribe un mensaje: exit
Mi petición al servidor es: "exit"
Respuesta del servidor:
exit
Conexión cerrada exitosamente
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 clienteUno.py
Traceback (most recent call last):
  File "clienteUno.py", line 22, in
    s.connect((host, port))
ConnectionRefusedError: [Errno 11] Connection refused
abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 server.py
~ server ~ Bienvenido, usted ha sido conectado satisfactoriamente al servidor :D
Posibles comandos a ejecutar:
[1] Para mostrar la versión del servidor
[2] Para terminar la conexión con el cliente
[3] Para mostrar la información del cliente
[4] Para mostrar a los autores del servidor
[5] Para escribir algo en el servidor
Puede ingresar comandos conocidos para:
*** Para cerrar la comunicación, ingrese:
Escribe un mensaje:

abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 clienteTres.py
~ server ~ Bienvenido, usted ha sido conectado satisfactoriamente al servidor :D
Posibles comandos a ejecutar:
[1] Para mostrar la versión del servidor
[2] Para terminar la conexión con el cliente.
```

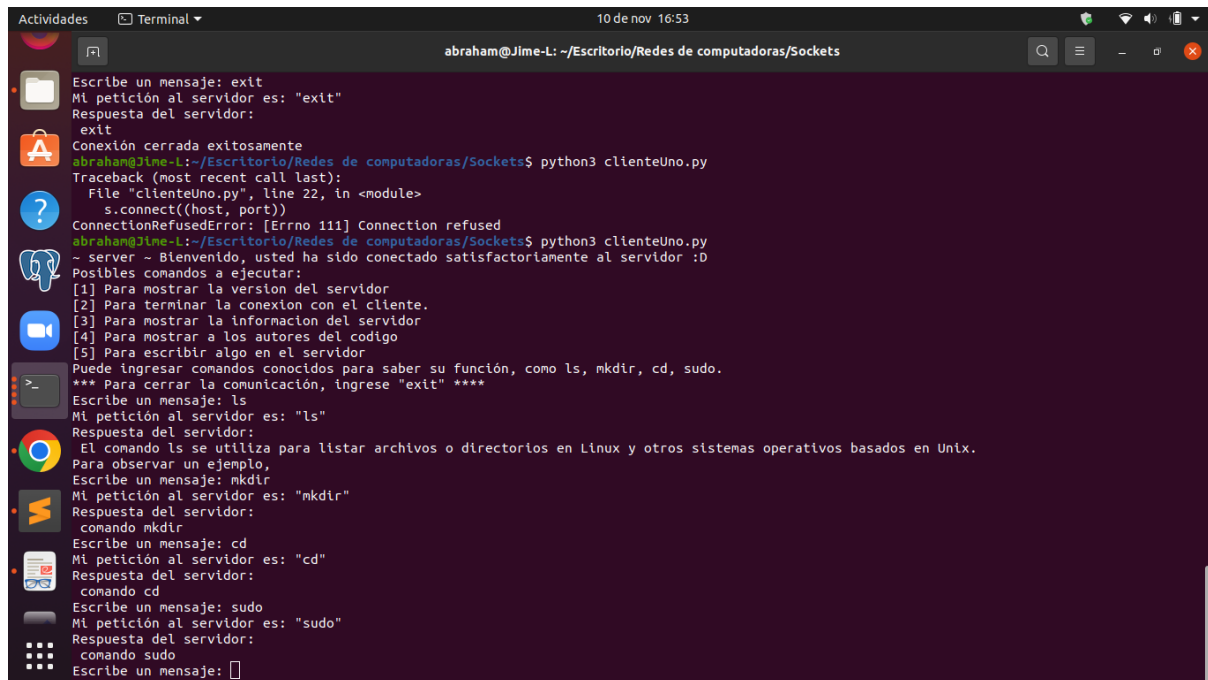
Cerramos la sesion con 3 dispositivos

```
Abraham@Jlme-L: ~/Escritorio/Redes de computadoras/Sockets
Petición recibida: "exit"
Respuesta enviada:
~ Cerrando comunicación con ('127.0.0.1', 43532)... ~
** Comunicación cerrada con éxito **
^CTraceback (most recent call last):
  File "server.py", line 100, in <module>
    client_connection, client_address = server_socket.accept()
  File "/usr/lib/python3.8/socket.py", line 292, in accept
    fd, addr = self._accept()
KeyboardInterrupt

abraham@Jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 server.py
** Socket creado satisfactoriamente **
Esperando conexiones...
Conexión establecida con ('127.0.0.1', 37692).
Conexión establecida con ('127.0.0.1', 55634).
Conexión establecida con ('127.0.0.1', 58862).
Petición recibida: "ls"
Respuesta enviada:
Petición recibida: "mkdir"
Respuesta enviada:
Petición recibida: "cd"
Respuesta enviada:
Petición recibida: "sudo"
Respuesta enviada:
Petición recibida: "exit"
Respuesta enviada:
~ Cerrando comunicación con ('127.0.0.1', 37692)... ~
** Comunicación cerrada con éxito **
Petición recibida: "exit"
Respuesta enviada:
~ Cerrando comunicación con ('127.0.0.1', 58862)... ~
** Comunicación cerrada con éxito **
Petición recibida: "exit"
Respuesta enviada:
~ Cerrando comunicación con ('127.0.0.1', 55634)... ~
** Comunicación cerrada con éxito **

```

Comandos ya conocidos



```
Actividades Terminal 10 de nov 16:53
abraham@jlme-L: ~/Escritorio/Redes de computadoras/Sockets

Escribe un mensaje: exit
Mi petición al servidor es: "exit"
Respuesta del servidor:
exit
Conexión cerrada exitosamente
abraham@jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 clienteUno.py
Traceback (most recent call last):
  File "clienteUno.py", line 22, in <module>
    s.connect((host, port))
ConnectionRefusedError: [Errno 111] Connection refused
abraham@jlme-L:~/Escritorio/Redes de computadoras/Sockets$ python3 clienteUno.py
~ server ~ Bienvenido, usted ha sido conectado satisfactoriamente al servidor :D
Posibles comandos a ejecutar:
[1] Para mostrar la version del servidor
[2] Para terminar la conexion con el cliente.
[3] Para mostrar la informacion del servidor
[4] Para mostrar a los autores del codigo
[5] Para escribir algo en el servidor
Puede ingresar comandos conocidos para saber su función, como ls, mkdir, cd, sudo.
*** Para cerrar la comunicación, ingrese "exit" ****
Escribe un mensaje: ls
Mi petición al servidor es: "ls"
Respuesta del servidor:
El comando ls se utiliza para listar archivos o directorios en Linux y otros sistemas operativos basados en Unix.
Para observar un ejemplo,
Escribe un mensaje: mkdir
Mi petición al servidor es: "mkdir"
Respuesta del servidor:
comando mkdir
Escribe un mensaje: cd
Mi petición al servidor es: "cd"
Respuesta del servidor:
comando cd
Escribe un mensaje: sudo
Mi petición al servidor es: "sudo"
Respuesta del servidor:
comando sudo
Escribe un mensaje: 
```

NOTAS

El reporte por entregar debe contener las siguientes evidencias.

✓ Capturas de pantalla

o Punto 3 en incisos a - d se pueden englobar en una sola captura.

✓ Las capturas de pantalla referentes al servidor:

o Demostrar la conexión cifrada de varios clientes.

o Mensaje de bienvenida en cada cliente.

o Explicar cada función nueva creada además de la demostración.

✓ Las capturas de pantalla referentes al cliente:

o Comunicación con el servidor

o Ingresar el comando y que el servidor arroje algo similar a lo siguiente

Conclusiones

José Luis: Lo que aprendimos en esta práctica fue la implementación de sockets en Python y entendimos la arquitectura de comunicación entre clientes y servidores, ahora sabemos que los sockets proporcionan una interfaz efectiva para el intercambio de datos. Durante la práctica, se implementaron las dinámicas de espera del servidor para recibir mensajes de los clientes, y se analizó la capacidad del cliente para enviar mensajes y explorar comandos, proporcionando información detallada sobre el servidor, además de el monitoreo del estado de estas conexiones desde el lado del servidor. Otro punto importante fue los puertos conocidos y designados, ya que conocimos la diversidad de funciones que pueden cumplir los puertos, abriendo la puerta a una variedad de aplicaciones.

Abraham: En la práctica implementamos los sockets, podemos ver cómo nos conectamos al servidor con un puerto asociado. Hay que tener en cuenta que si el puerto ya existe no podemos acceder a él y lo tenemos que cambiar. Entonces podemos ver como tienen comunicación entre redes, la implementación con python estuvo sencilla y pues no tuvimos algún problema. En el código agregamos comandos que son sencillos y su lógica no está sencilla de implementar pero colocamos algunos mensajes para saber a qué se refieren. Me sorprendió como podíamos tener 3 clientes al mismo tiempo.

Esaú:

En esta práctica vimos la implementación de sockets. Los sockets son un canal de comunicación entre un cliente y un servidor, estos intercambian datos localmente y entre redes. El servidor siempre debe estar a la espera del mensaje del cliente(s) y el cliente(s) manda el mensaje cuando él quiera establecer la comunicación. El cliente en lo que nos piden, puede ingresar un mensaje y este será enviado al servidor, también puede saber de qué tratan comandos especiales que le proporcionarán información sobre el servidor y comandos básicos de linux, mientras que en el servidor se irá mostrando el estado de estas conexiones. Además de una investigación sobre 5 puertos conocidos y 6 puertos designados, nos dimos cuenta que hay muchísimos puertos que podemos usar que tienen distintas funciones y las aplicaciones de los sockets, que lo más general sería el envío de mensajes, archivos y en aplicaciones web como un chat en línea en tiempo real.

Referencias

- IBM. (2023). Sockets | IBM. Recuperado el 9 de noviembre de 2023, <https://www.ibm.com/docs/es/aix/7.3?topic=concepts-sockets>
- IBM. (2023). Programación de sockets | IBM. Recuperado el 9 de noviembre de 2023, <https://www.ibm.com/docs/es/i/7.5?topic=communications-socket-programming>
- Espinoza, O. (2023). Principales puertos TCP y UDP y para qué sirven cada uno de ellos | RedesZone. Recuperado el 9 de noviembre de 2023, <https://www.redeszone.net/tutoriales/configuracion-puertos/puertos-tcp-udp/>
- programacionpython80889555. (2023). PROGRAMACIÓN CON SOCKETS EN PYTHON I : ESTABLECIENDO CONEXIÓN CLIENTE-SERVIDOR | programacionpython80889555. Recuperado el 9 de noviembre de 2023, <https://programacionpython80889555.wordpress.com/2022/10/03/programacion-con-sockets-en-python-i-estableciendo-conexion-cliente-sevidor/>