



Universidad Politécnica de Tlaxcala Región Poniente

Ingeniería en Sistemas Computacionales

Materia: Administración de Base de Datos

Docente: Ing. Vanesa Tenopala Zavala

Alumno: 22SIC008 Isaac Brandon Martínez Ramírez

Tema: Practicas Sistemas Distribuidos

Ciclo escolar: mayo-agosto 2024

Fecha: 11 de agosto de 2024

Reporte Práctica 1 ChatSockets

Descripción del Programa

El programa "ChatSockets" simula la comunicación entre dos computadoras utilizando sockets en una red local. La implementación permite enviar y recibir mensajes de manera bidireccional entre un cliente y un servidor, demostrando los principios básicos de la comunicación en red utilizando la arquitectura cliente-servidor. El programa se desarrolla en Java y emplea las clases Socket y ServerSocket para establecer y gestionar la conexión.

1. Objetivos

- **Establecer Comunicación:** Crear un chat que permita la comunicación entre dos computadoras utilizando sockets.
- **Enviar y Recibir Mensajes:** Permitir el envío y recepción de mensajes de texto entre un cliente y un servidor.
- **Demostrar la Arquitectura Cliente-Servidor:** Implementar un modelo donde el servidor está siempre a la espera de conexiones y el cliente inicia la comunicación.

2. Componentes del Programa

2.1. Clases

- **ChatServer:** Clase que actúa como servidor, escucha conexiones entrantes y permite la comunicación con el cliente.
- **ChatClient:** Clase que actúa como cliente, se conecta al servidor y envía/recibe mensajes.

3. Funcionamiento del Programa

- **Inicialización:**
 - Al ejecutar el programa, la clase ChatServer se inicia y espera conexiones en un puerto específico.
- **Establecimiento de Conexión:**
 - El ChatClient se conecta al servidor utilizando la dirección IP y el puerto especificado.

- **Comunicación Bidireccional:**

- Una vez establecida la conexión, el cliente puede enviar mensajes al servidor, que a su vez los recibe y puede responder.

- **Interacción del Usuario:**

- El usuario ingresa mensajes a través del cliente, los cuales son transmitidos al servidor y se muestra la respuesta en la consola de ambos lados.

4. Ejecución de el programa

- Primero, ejecuta la clase ChatServer. Para ello, haz clic derecho sobre la clase ChatServer en el panel del proyecto y selecciona Run File.
- Luego, ejecuta la clase ChatClient de la misma manera.
- Ahora deberías poder enviar mensajes desde el cliente al servidor y recibir respuestas.

5. Resultado

El programa permite la comunicación efectiva entre un cliente y un servidor mediante el intercambio de mensajes de texto. Cada mensaje enviado por el cliente se refleja en la consola del servidor, y viceversa, demostrando la funcionalidad básica de un sistema de chat.



```
Output
ChatSockets (run) x ChatSockets (run) #2 x
run:
Servidor listo para aceptar conexiones...
Cliente conectado.
hola
Cliente: hola
Cliente: como estas?

Output
ChatSockets (run) x ChatSockets (run) #2 x
run:
hola
Servidor: Servidor: hola
como estas?
Servidor: Servidor: como estas?
bien y tu
Servidor: Servidor: bien y tu
Ingenieria en sistemas computacionales
Servidor: Servidor: Ingenieria en sistemas computacionales
```

6. Conclusión

El programa “ChatSockets” es una implementación básica pero efectiva para demostrar cómo se puede establecer y gestionar una comunicación en red utilizando la arquitectura cliente-servidor en Java. Este ejercicio refuerza la comprensión de los sockets y proporciona una base sólida para desarrollar aplicaciones de red más complejas.

Práctica 2: Configuración de un Entorno Cliente-Servidor con VirtualBox

Objetivo

Crear un entorno de red entre dos máquinas virtuales (una actuando como cliente y la otra como servidor), utilizando VirtualBox. Configurar con MySQL en la máquina servidor y permitir la conexión desde la máquina cliente.

Pasos a Seguir

1. Instalación de VirtualBox

- **Descarga e instalación:**
 - Visita el sitio oficial de VirtualBox y descarga la versión más reciente para tu sistema operativo.
 - Sigue las instrucciones de instalación según tu sistema operativo (Windows, macOS, Linux).

2. Selección de Sistemas Operativos para las Máquinas Virtuales

- **Recomendación:**
 - **Ubuntu Server (Servidor):** Fácil de usar para configuraciones de servidor, con amplio soporte para MySQL y buen rendimiento en entornos virtualizados.
 - **Ubuntu Desktop (Cliente):** Familiar y fácil de manejar, con una interfaz gráfica que facilita la gestión y el acceso al servidor.

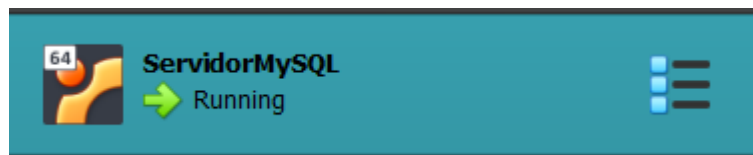
3. Creación de las Máquinas Virtuales

Máquina Virtual del Servidor (Ubuntu Server)

1. **Crear la Máquina Virtual:**
 - Abre VirtualBox y haz clic en New.
 - Asigna un nombre, por ejemplo, "ServidorMySQL".
 - Selecciona "Linux" como tipo y "Ubuntu (64-bit)" como versión.
 - Asigna al menos 1GB de RAM y 10GB de disco duro virtual (puedes ajustar según las capacidades de tu PC).

2. Instalación del Sistema Operativo:

- Descarga la imagen ISO de [Ubuntu Server](#).
- En VirtualBox, selecciona la máquina "ServidorMySQL", haz clic en Settings > Storage, y carga la ISO en el controlador de CD.
- Inicia la máquina virtual y sigue las instrucciones para instalar Ubuntu Server.
- Durante la instalación, configura el servidor SSH, que permitirá el acceso remoto.



3. Configuración de Red:

- Configura la red de la máquina en modo "Host-Only Adapter" para que las máquinas virtuales se vean entre sí sin necesidad de acceso a internet externo.
- Verifica la IP asignada a la máquina usando el comando `ip` a después de la instalación.

4. Instalación de MySQL:

- Una vez que la máquina esté instalada, actualiza los paquetes:

sudo apt-get update

sudo apt-get upgrade

- Instala MySQL:

sudo apt-get install mysql-server

```
ServidorMySQL [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help

Ubuntu 24.04 LTS server tty6
server login: server
Password:
Welcome to Ubuntu 24.04 LTS (GNU/Linux 6.8.0-40-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:    https://landscape.canonical.com
 * Support:       https://ubuntu.com/pro

System information as of dom 11 ago 2024 22:22:11 UTC

System load:  0.18           Processes:           135
Usage of /:   37.8% of 11.21GB Users logged in:       1
Memory usage: 11%           IPv4 address for enp0s3: 10.0.2.15
Swap usage:   0%

El mantenimiento de seguridad expandido para Applications está desactivado
Se pueden aplicar 44 actualizaciones de forma inmediata.
Para ver estas actualizaciones adicionales, ejecute: apt list --upgradable

Active ESM Apps para recibir futuras actualizaciones de seguridad adicionales.
Vea https://ubuntu.com/esm o ejecute «sudo pro status»

server@server:~$ sudo apt install mysql-server
```

- Configura MySQL para aceptar conexiones remotas editando el archivo de configuración mysqld.cnf:

sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf

- Cambia la línea bind-address = 127.0.0.1 a bind-address = 0.0.0.0 para permitir conexiones desde cualquier IP.
- Reinicia el servicio MySQL:

sudo systemctl restart mysql

- Crea un usuario MySQL y dale permisos para conectarse desde la máquina cliente:

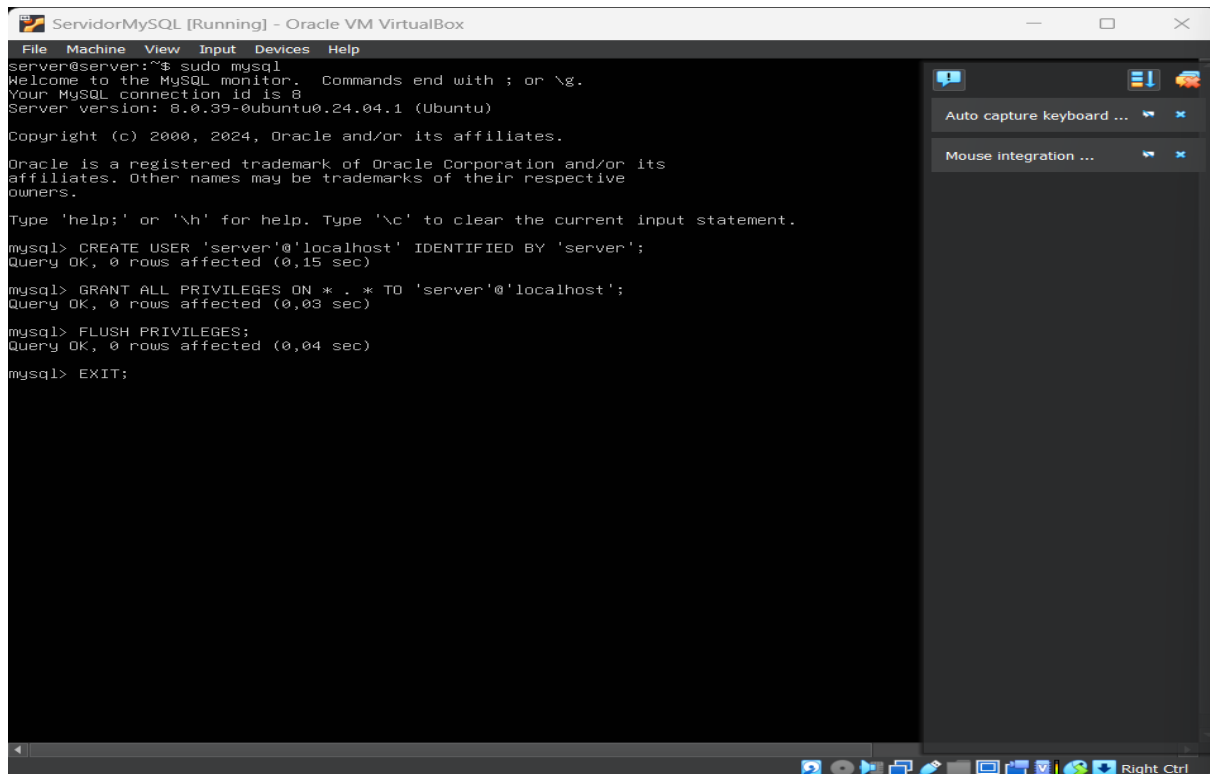
sudo mysql

CREATE USER 'usuario'@'localhost' IDENTIFIED BY 'contraseña';

*GRANT ALL PRIVILEGES ON *.* TO 'usuario'@'localhost';*

FLUSH PRIVILEGES;

EXIT;



```
server@server:~$ sudo mysql
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 8
Server version: 8.0.39-0ubuntu0.24.04.1 (Ubuntu)

Copyright (c) 2000, 2024, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> CREATE USER 'server'@'localhost' IDENTIFIED BY 'server';
Query OK, 0 rows affected (0,15 sec)

mysql> GRANT ALL PRIVILEGES ON * . * TO 'server'@'localhost';
Query OK, 0 rows affected (0,03 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0,04 sec)

mysql> EXIT;
```

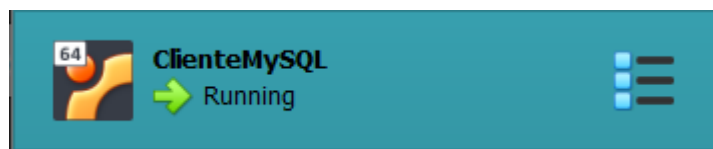
Máquina Virtual del Cliente (Ubuntu Desktop)

1. Crear la Máquina Virtual:

- Sigue un proceso similar al del servidor, pero nombra la máquina como "ClienteMySQL".
- Selecciona "Linux" como tipo y "Ubuntu (64-bit)" como versión.
- Asigna al menos 2GB de RAM y 15GB de disco duro virtual.

2. Instalación del Sistema Operativo:

- Descarga la imagen ISO de [Ubuntu Desktop](#).
- Carga la ISO en la máquina virtual y sigue las instrucciones para instalar Ubuntu Desktop.



3. Configuración de Red:

- Configura la red en modo "Host-Only Adapter" para que pueda conectarse con la máquina servidor.

4. Instalación del Cliente MySQL:

- Una vez que Ubuntu Desktop esté instalado, abre una terminal y actualiza los paquetes:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

- Instala el cliente MySQL:

```
sudo apt-get install mysql-client
```

Conexión entre Cliente y Servidor

1. Conectar el Cliente al Servidor:

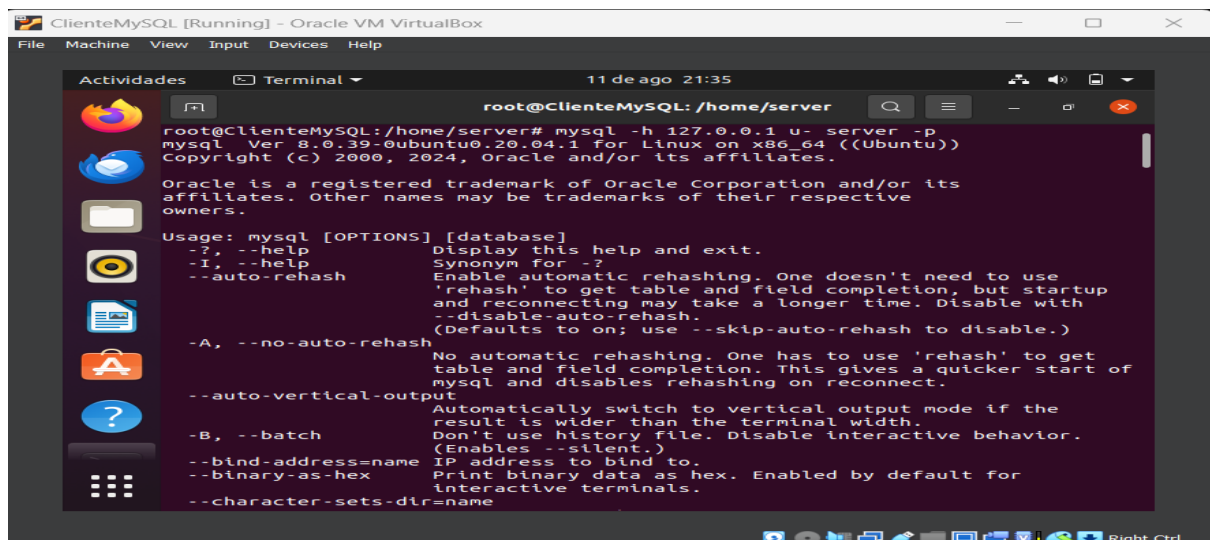
- Desde la terminal de la máquina "ClienteMySQL", conéctate al servidor MySQL usando el siguiente comando:

```
mysql -h [IP_SERVIDOR] -u usuario -p
```

- Sustituye [IP_SERVIDOR] en este caso 127.0.0.1 por la dirección IP del servidor y proporciona la contraseña cuando se te pida.

2. Prueba de Conexión:

- Una vez conectado, intenta crear una base de datos y realizar algunas operaciones básicas para confirmar que la configuración funciona correctamente.



Conclusión

Utilizar Ubuntu Server para el servidor y Ubuntu Desktop para el cliente es una opción sencilla y bien documentada para esta práctica. Siguiendo estos pasos, se puede configurar un entorno cliente-servidor funcional, con MySQL gestionando la base de datos y permitiendo la comunicación entre ambas máquinas virtuales a través de VirtualBox.