Parte 1: Taller práctico de despliegue de un aplicativo

Despliegue de un software implementado en Java en una máquina remota que funciona con Ubuntu. El aplicativo permite la conexión de un cliente desde otro host. Utilizaremos scripts en Bash, conexión SSH y el comando **nohup** para garantizar el funcionamiento continuo del software incluso después de cerrar la sesión SSH.

Paso 1: Preparar la Máquina Remota

1. Configurar una Máquina Remota Ubuntu: Asegúrate de tener acceso a una máquina remota que funcione con Ubuntu y que tenga Java instalado. Debe conectarse a una de las siguientes máquinas virtuales usando ssh. **Crear una carpeta de trabajo**, para no interferir con el trabajo de otras personas.

```
ssh usuario@direccion_ip_maquina_remota
```

Rango de ip: 192.168.131.51 - 72

usuario: vagrant

contraseña: vagrant

Se puede instalar software usando el comando **sudo apt-get install <software>**

Usar la misma contraseña

2. Configurar el Firewall (si es necesario): Si estás utilizando un firewall en la máquina remota, asegúrate de abrir el puerto que utilizará el software para la conexión del cliente (Consultar como se puede verificar en Ubuntu si hay firewall activo).

Paso 2: Transferir el Software a la Máquina Remota

1. Transferir el Software: Copia el archivo JAR o el archivo compilado de tu software Java a la máquina remota. Puedes usar SCP (Secure Copy Protocol) para transferir archivos de forma segura desde tu host local a la máquina remota.

```
scp /ruta/software.jar usuario@ip_maquina_remota:/ruta/destino
```

Paso 3: Crear un Script de Despliegue en Bash

1. Crear un Script en Bash: Crea un script en Bash para ejecutar el software en la máquina remota. El script debe incluir los comandos necesarios para iniciar el software y redirigir la salida a un archivo de registro usando **nohup**. Suponiendo que ya tiene el aplicativo compilado y ha creado el archivo *.jar, el siguiente script muestra como ejecutarlo en segundo plano. Hacer las modificaciones que considere necesarias.

```
#!/bin/bash
# Cambiar al directorio donde se encuentra el software
cd /ruta/destino
# Iniciar el software y redirigir la salida a un archivo de registro
nohup java -jar software.jar > log.txt 2>&1 &
```

También:

```
nohup java -jar /ruta/server.jar &
```

2. Dar Permiso de Ejecución al Script: Ejecuta el siguiente comando para dar permiso de ejecución al script que acabas de crear:

```
chmod +x script_despliegue.sh
```

Paso 4: Ejecutar el Script de Despliegue en la Máquina Remota

1. Conectarse a la Máquina Remota: Utiliza SSH para conectarte a la máquina remota desde tu host local.

2. Ejecutar el Script de Despliegue: Ejecuta el script de despliegue que creaste en el Paso 3 en la máquina remota:

```
./script_despliegue.sh
```

Paso 5: Consultar que hacen los comandos netsat y nc.

Verificar que el servidor esté escuchando en un terminal diferente ejecutar (cambiar el puerto de ser necesario):

```
netstat -an | grep 59090
nc localhost 59090
```

Parte 2: Taller Práctico: Conceptos Básicos de Programación Web

Este taller práctico tiene como objetivo introducir a los estudiantes en los conceptos fundamentales de la programación web, incluyendo URIs, URLs, el modelo cliente-servidor y los navegadores web. Deben realizar las consultas pertinentes

Introducción (15 minutos): Leer el documento adjunto Introducción.pdf

1. Breve introducción a la World Wide Web (WWW) y su importancia en la actualidad.

2. Diferencia entre Internet y la Web.

3. Elementos clave de la Web: hipertexto, HTML, HTTP.

Actividad 1: Explorando URIs y URLs (30 minutos) consultar y definir los temas relacionados a continuación**:**

1. Definición y estructura de una URI (Uniform Resource Identifier).

2. Tipos de URI: URL, URN, DOI.

3. Anatomía de una URL (Uniform Resource Locator): esquema, protocolo, host, puerto, ruta, consulta, fragmento.

4. Presentar algunos ejemplos de URLs: URL de un sitio web, URL de una imagen, URL de un vídeo.

5. Diferencia entre URL absoluta y URL relativa.

6. Actividad práctica: Analizar y descomponer diferentes URLs

Ejemplo de análisis de una URL:

https://www.google.com/search?client=firefox-b-d&q=programacion+web

Esquema: https (protocolo seguro)

Protocolo: HTTP (transferencia de hipertexto)

Host: www.google.com (dominio de Google)

Puerto: 80 (por defecto, no se especifica)

Ruta: /search (ruta al motor de búsqueda)

Consulta: q=programacion+web (parámetros de búsqueda)

Fragmento: No hay (no se especifica una sección específica)

Esta URL indica que se accede al motor de búsqueda de Google (https://www.google.com/search) con los parámetros de búsqueda 'q=programacion+web' (búsqueda de "programación web") y pueden aparecer otros parámetros adicionales ('oq', 'aqs', 'sourceid', 'ie') que definen la consulta y el contexto de la búsqueda.

Actividad 2: Descubriendo el modelo cliente-servidor (30 minutos) consultar y definir los temas relacionados a continuación:

1. Concepto básico del modelo cliente-servidor.

2. Funcionamiento en la Web: papel del navegador como cliente, papel del servidor web como servidor.

3. Solicitud-respuesta HTTP: solicitud del cliente, respuesta del servidor.

5. Actividad práctica: Simular una interacción cliente-servidor utilizando un navegador web y un servidor web local.

Usando su navegador web escribir la dirección: **http://direccion_ip_servidor:puerto**, donde **direccion_ip_servidor** es la dirección IP del servidor y puerto es el puerto en el que el servidor está escuchando (en este caso, 8080).

Actividad 3: Navegando la Web con diferentes navegadores (30 minutos):

1. Función principal de los navegadores web.

2. Estructura básica de un navegador web: motor de renderizado, interfaz de usuario, administración de memoria caché.

3. Diferencias y características principales de cada navegador: Chrome, Firefox, Safari, Edge, Brave
4. Comparar y contrastar la experiencia de navegación en diferentes sitios web utilizando distintos navegadores.