Seminarios Sistemas y Aplicaciones Distribuidas (SAD)

Miembros:
Ismael Pérez Martín
Jesús Enrique Vélez Gutiérrez
Carlos Herrera Piedra

2019 - 2020

RECURSOS

A continuación se detallan una serie de recursos o información necesarios para la evaluación y comprensión de esta memoria.

Repositorio

El código desarrollado durante todas las sesiones de seminario se ha gestionado mediante el sistema de control de versiones Git.
Este repositorio se ha almacenado en el sistema de almacenamiento en la nube para repositorios GitHub, y se encuentra disponible en el siguiente enlace:
https://github.com/jesusinri/SAD

El número del último commit realizado antes de la entrega es:
d6b7a014aebc500e3ecf619ed79504abd1f8b09c

El enlace al repositorio en dicho commit es el siguiente:
https://github.com/jesusinri/SAD/tree/d6b7a014aebc500e3ecf619ed79504abd1f8b09c

Es importante mencionar que no todos los seminarios se encuentran en el repositorio, ya que algunos de ellos al ser de contenido teórico se han añadido en esta memoria, mientras que los mayoritariamente de programación están detallados y realizados en el repositorio.

Organización del repositorio

El repositorio está organizado en carpetas, por lo cual para poder acceder a cada uno de los seminarios basta con:

- Acceder a la URL del repositorio
- Acceder a la carpeta Seminarios
- Acceder a la carpeta del seminario correspondiente

Cada uno de los seminarios cuenta con un README.md el cual presenta información importante del laboratorio así como los pasos necesarios para su instalación y/o ejecución.
En los casos que es necesario se detalla el procedimiento seguido para completar las actividades asignadas a dicho seminario.

SEMINARIOS REALIZADOS

Seminario 2 - Módulo Carrito de la compra

Este primer seminario plantea la funcionalidad de un carrito de la compra. Para ello se ha creado un módulo que abre la conexión con la bases de datos y permite agregar y borrar productos. En un principio se planteó usar moongose, un módulo de Node que facilita mucho el acceso de MongoDB, pero se decidió hacer sin él para entender al 100% el funcionamiento sin ninguna abstracción.

Seminario 3 - Servicio REST carrito de la compra

En este seminario se convierte el carrito de la compra anterior en un servicio Rest. Para ello se ha usado el módulo de express, que facilita mucho el desarrollo de APIs. Una de las ventajas, es que se le pueden pasar identificadores en las urls y tratar esa información para buscar en una base de datos o realizar cualquier tipo de operación. Además de esto se ha creado un módulo para el registro y borrado de servicios.

Seminario 4 - Express y Websockets

En el seminario de express y websockets se han realizado una serie de actividades planteadas en el boletín para mejorar la funcionalidad de un chat de texto.
En concreto, las funcionalidades desarrolladas han sido:

- Funcionalidades obligatorias
    - Broadcast de un mensaje a todos los usuarios cuando un usuario se conecta/desconecta
    - Añadir soporte para nombres de usuario
    - No enviar el mensaje al usuario emisor, sino añadirlo directamente en la lista de mensajes
- Funcionalidades opcionales:
    - Mostrar que usuario está escribiendo
    - Mostrar la lista de usuarios conectados

La realización de este seminario está explicada con mayor detalle en el README.md correspondiente al seminario en el repositorio del proyecto: https://github.com/jesusinri/SAD/tree/master/Seminarios/S4

Seminario 5 - Promesas

En este seminario se han realizado las actividades correspondientes para, partiendo de una base de código basada en callbacks, desarrollar el código análogo usando promesas.
En concreto, las funcionalidades desarrolladas han sido:

- Cambiar, en la medida de lo posible, todas las funciones de un login para usar promesas en lugar de callbacks. Cambiar también todas las funciones que se considere para usar promesas.
- Realizar un módulo con mongodb para la persistencia del carrito de la compra desarrollado anteriormente y gestionar el control de stock al usar el carrito. Realizar esta actividad usando promesas en todo lo posible en lugar de callbacks.

La realización de este seminario está explicada con mayor detalle en el README.md correspondiente al seminario en el repositorio del proyecto: https://github.com/jesusinri/SAD/tree/master/Seminarios/S5

Seminario 6 y 7 - ZeroMQ

En este seminario se ha desarrollado, mediante sockets ZeroMQ, un sistema de balanceado de carga y redirección de peticiones para el carrito desarrollado anteriormente.
Tras analizar la estructura requerida, se llegó a la conclusión de que la arquitectura más indicada para la realización del seminario era una arquitectura router-router, la cual nos permite realizar ese balanceo de carga manteniendo una lista de clientes, trabajadores y trabajos en tiempo real.
La funcionalidad desarrollada en este seminario es la actividad obligatoria, junto con la adaptación del carrito para usarlo mediante peticiones REST.

La realización de este seminario y la respuesta a sus preguntas está explicada con mayor detalle en el README.md correspondiente al seminario en el repositorio del proyecto: https://github.com/jesusinri/SAD/tree/master/Seminarios/S6-7

Seminario 8 - Consistencia

Para este seminario se plantean una serie de actividades relacionadas con las consistencia en los sistemas distribuidos.
Estas actividades han sido realizadas a papel y se pueden consultar en el repositorio:
https://github.com/jesusinri/SAD/tree/master/Seminarios/S8

Seminario 9 - Seguridad en sistemas distribuidos

El famoso proveedor cloud AWS, sufrió el pasado 23 de octubre de 2019 un ataque a sus sistemas. El incidente duró aproximadamente 8 horas y no se conoce un informe oficial detallado sobre problema. Sin embargo, Amazon sí que reconoció el fallo con el siguiente mensaje:

"*Between 10:30 AM and 6:30 PM PDT, we experienced intermittent errors with resolution of some AWS DNS names. Beginning at 5:16 PM, a very small number of specific DNS names experienced a higher error rate. These issues have been resolved*"

El ataque usado fue un DDoS, una forma muy común de hacer caer los sistemas informáticos por parte de los piratas de Internet, consiste en dirigir una cantidad enorme de tráfico hacia los servidores, de tal forma que saturan el sistema y dejan de funcionar correctamente. Normalmente estos ataques se realizan haciendo uso de ordenadores "zombies" que han sido infectados mediante algún malware.

Debido a esto, algunos usuarios tuvieron problemas para utilizar los servicios de S3 (Almacenamiento de objetos), RDS (Bases de datos relacionales) y ELB (Elastic Load Balancing), todos ellos dependientes de consultas de DNS externas de su servicio Route 53.

Al parecer incluso las grandes compañías tienen problemas con ataques de este tipo, aunque disponen de mecanismos para reducirlos como es el Amazon DDoS Shield que se encarga de monitorizar la red y de detectar peticiones maliciosas. Debido a esto, algunos usuarios fueron identificados como “atacantes” y se les denegó el servicio por algunos momentos.

Más información:

1. https://www.techradar.com/news/aws-hit-by-major-ddos-attack
2. https://www.gorillastack.com/news/aws-shield-ddos-protection/
3. https://www.theregister.co.uk/2019/10/28/amazon_ddos_attack/

Seminario 10 - Blockchain

Primero se cambiaron los métodos en el archivo dm.js a promesas, para volverlos asincronos, tambien se creo un script runnerblockchain, el cual creó el nodo y la cadena de bloques en el host local.

Se creó el archivo Blockchain.js, el cual exporta un módulo con los métodos a usar en el foro, como :

CrearRoom: Crea un Stream en usando Multichain

SubscribeRoom: se subscribe el archivo para detectar los cambios o items publicados

PublichsMessage: publica un Item sobre el archivo creado

UnsubscribeRoom: Cierra la subscripción al archivo.

Posteriormente se cambiaron los métodos de **addPrivateMessage y addPublicMessage,** para que cuando creará una nueva conversación privada, se creará un nuevo Stream en blockchain el cual representa la base de datos, y en donde se publican los mensajes de la conversación.

Debido a que windows detecta multichain como virus y el firewall por defecto restringe el proceso de comunicación del nodo, no se pudo hacer pruebas del funcionamiento en este ambiente.

Seminario 11 - Despliegue de servicios

Primero se creó el dockerfile para el despliegue de un foro y su servidor correspondiente, posteriormente se configuró el archivo docker-compose.yml en donde se definieron tres servicios de foro y se hizo la configuración de los puertos, de esta manera se despliegan varios servicios del foro en distintos contenedores.

```
services:
  forone:
    build:
      context: ./forum/
      args:
        otherhost: "tcp://forotwo:1016,tcp://forotree:1016"
    ports:
    - "1000:1000"
    - "1011:1011"
    - "1016:1016"
  forotwo:
    build:
      context: ./forum/
      args:
        otherhost: "tcp://forone:1016,tcp://forotree:1016"
    ports:
    - "1001:1000"
    - "1012:1011"
    - "1017:1016"
  forotree:
    build:
      context: ./forum/
      args:
```

```
      otherhost: "tcp://forone:1016,tcp://forotwo:1016"
  ports:
  - "1003:1000"
  - "1013:1011"
  - "1018:1016"
```