Proyecto de Curso

Entrega Final

Introducción a los Sistemas Distribuidos

Ingeniería de Sistemas

Estudiante:

Juan Sebastian Galeano Gonzalez

Docente:

Mariela Josefina Curiel Huérfano

3 de abril de 2024.

Especificaciones de Hardware

A continuación, se presentan las especificaciones de los tres computadores en los que se creo el proyecto, teniendo en cuenta que son máquinas virtuales prestadas por la universidad y dependen en su mayoría de los recursos asignados por la misma universidad:

Hardware:

Processor	Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz 2.59 GHz (3 processors)
Installed RAM	16,0 GB
System type	64-bit operating system, x64-based processor

A continuación, se detallan las especificaciones de los softwares usados tanto para el desarrollo del proyecto, como para las pruebas realizadas:

Java:	Java		22
Maven:	Maven	versión	3.9.6
ZeroMQ:	Jeromq	versión	0.6.0

Git: git versión 2.45.1

Experimentos realizados

Para la prueba de funcionalidad y rendimiento del sistema se tienen en cuenta los siguientes escenarios:

Escenario 1	**Escenario 2**	**Escenario 3**
Las probabilidades generadas por los sensores de humedad y temperatura se encuentran en los siguientes rangos: 0,5 valores correctos 0,4 valores fuera del rango 0,1 errores	Las medidas de temperatura, desde los sensores, se envían cada 4 segundos.	Se mantienen las periodicidades del escenario 2 y probabilidades del escenario 1, pero se produce una falla en el proxy del nivel fog.

Teniendo en cuenta los escenarios anteriormente mostrados se establecen las siguientes variables de medición por cada uno de los escenarios:

1. Cantidad de mensajes enviados por minuto, se tienen en cuenta dos variables para estas medidas:
    a. Cantidad de mensajes enviados en una misma capa por minuto.
    b. Cantidad de mensajes enviados en todo el sistema por minuto.
2. Tiempos de comunicación, se tienen en cuenta dos variables para esta medida:
    a. Tiempo promedio de comunicación entre dos capas, en este caso Cloud y proxy.
    b. Desviación estándar del promedio anteriormente calculado.
3. Cantidad de alertas generadas por cada indicador (Temperatura, humedad y humo) por minuto.

Para la obtención de estas variables se usarán las siguientes tácticas:

1. Cantidad de mensajes envidos en una misma capa por minuto: Usando la redirección de la salida estándar de java se guarda la información de los mensajes enviados en cada capa, para luego ser contada por cada capa.
2. Cantidad de mensajes en el sistema: Usando el proyecto que contiene la identificación de cada mensaje, además, con la base de datos creada en la capa de Cloud se cuenta el número de mensajes recibidos.

3. Tiempo promedio de comunicación entre Cloud y proxy: En este caso, se usa la clase de system.nanoTime, desde el momento de envío del mensaje y el momento de obtención de la respuesta, se guardan las diferencias entre él envió y la respuesta. Después de un minuto se calcula el promedio de tiempo de comunicación entre las dos capas.

4. Usando técnicas de estadística se calcula la desviación estándar del promedio anterior.

5. Usando la misma táctica de la redirección de salida estándar por comando, se obtienen todas las alarmas generadas por los sistemas de calidad.

Interpretación de los resultados.

Los resultados necesarios para la interpretación se encuentran en los **anexos uno y dos**, ya que son información no relevante para el presente documento.

En cuanto a la cantidad de mensajes enviados, se observa un aumento gradual en la capa edge, con 639 mensajes en el escenario 1, 858 en el escenario 2, y 978 en el escenario 3. Este incremento podría deberse a una mayor frecuencia de eventos o a un aumento en los datos procesados por los sensores. En la capa fog, los mensajes aumentan del primer al segundo escenario (2353 a 2552) y disminuyen en el tercero (2312), posiblemente debido a la falla del proxy que afectó la eficiencia del procesamiento. La capa cloud muestra una cantidad relativamente baja de mensajes en comparación con otras capas, con un ligero aumento en el segundo escenario (23, 33 y 22 mensajes respectivamente), indicando mayor actividad o generación de datos. En general, el sistema completo muestra un incremento en los mensajes en el segundo escenario (3015 a 3443) y una disminución en el tercero (2135), alineándose con la falla del proxy y su impacto.

En los tiempos de comunicación, el promedio mejora significativamente del primer al segundo escenario (0.01124 a 0.00342 segundos) y se mantiene casi igual en el tercero (0.00355 segundos), lo que sugiere optimizaciones en la red o en el manejo de mensajes entre capas. La desviación estándar del tiempo de comunicación también disminuye notablemente en el segundo escenario (0.00463 a 0.00101 segundos), indicando una mayor consistencia, y aunque aumenta ligeramente en el tercero (0.00114 segundos), se mantiene baja.

El número de alertas generadas varía significativamente entre los escenarios. Las alertas de temperatura disminuyen de 51 en el segundo escenario a 36 en el tercero, mientras que las de humedad aumentan de 17 a 32. Las alertas de humo aumentan drásticamente en el tercer escenario (34 a 73), lo que puede indicar una mayor incidencia de eventos relacionados con humo, posiblemente como consecuencia de la falla en el proxy.

Sugerencias para mejorar la eficiencia del sistema.

En el tercer escenario, la capa fog muestra una disminución en la cantidad de mensajes a 2312 debido a la falla del proxy A, lo que sugiere que este podría ser un cuello de botella. La baja cantidad de mensajes en la capa cloud en comparación con otras capas también indica un posible cuello de botella en la transmisión o procesamiento de datos hacia la nube. Aunque los tiempos de comunicación promedio mejoraron, la ligera variación en la desviación estándar en el tercer escenario sugiere que aún hay inconsistencias que podrían optimizarse.

Para abordar estos cuellos de botella, se proponen varias mejoras. Primero, optimizar la configuración de la red y la infraestructura de comunicación entre capas para asegurar tiempos de respuesta más consistentes y bajos, utilizando técnicas de compresión y protocolos más eficientes. Implementar un balanceo de carga más robusto y mecanismos de redundancia en la capa fog, asegurando que la conmutación por falla sea más fluida y menos impactante en el procesamiento de mensajes. Fortalecer el sistema de monitoreo y generación de alertas proactivas utilizando herramientas como Prometheus y Grafana para detectar y mitigar fallas antes de que impacten significativamente el rendimiento del sistema.

Además, mejorar la capacidad de procesamiento y almacenamiento en la nube para manejar eficientemente la carga de trabajo y los picos en la generación de datos, considerando la escalabilidad automática. Utilizar herramientas de perfilado como VisualVM para identificar y optimizar las partes del código que consumen más recursos. Implementar estas mejoras asegurará que el sistema distribuido sea más robusto, eficiente y capaz de manejar la carga de trabajo esperada sin degradar el rendimiento.

Conclusiones:

El análisis de los tres escenarios del sistema distribuido reveló incrementos graduales en la cantidad de mensajes enviados desde la capa edge y variaciones significativas en la capa fog y cloud, especialmente debido a la falla del proxy en el tercer escenario. Los tiempos de comunicación mejoraron notablemente, aunque algunas inconsistencias persisten.

Para optimizar el rendimiento, es crucial mejorar la configuración de la red, implementar balanceo de carga robusto, fortalecer el monitoreo y las alertas proactivas, y aumentar la capacidad de procesamiento en la nube. Utilizar herramientas de perfilado como VisualVM ayudará a identificar y optimizar los recursos del sistema.

Estas mejoras asegurarán un sistema distribuido más eficiente, robusto y capaz de manejar la carga de trabajo esperada, minimizando los cuellos de botella y mejorando la consistencia del rendimiento.