ECE750-T37 Assignment #1 Report G13

0. Abstract

We deployed a Java-based benchmark application Acme Air, and then monitored application metrics through IBM Cloud. The metrics from JMX Monitoring were collected through a driver program. Then we added different loads to the server such as 20, 100, 200 and 400 threads to the server. Finally we recorded and analyzed the data to summarize the impact of different loads on the server.

1. Implementation of Driver Program

We first initiated our management of the IBM Cloud Monitoring service by configuring essential IBM Cloud credentials. Following this setup, we created a simple function named "fetch_data_from_ibm_cloud," designed to accept parameters such as the Sysdig client, metric, and aggregation method. Within the main function, we defined three distinct metrics, each employing a different aggregation method: average (avg), summation (sum), maximum (max).

For each of these metrics, we invoked the "fetch_data_from_ibm_cloud" function to retrieve data from a remote server and save it in JSON format within the "datasets" directory. To ensure the periodic retrieval of metrics from the cloud, we introduced an infinite loop within the main function. After each data collection cycle, the program sleeps for an hour before repeating the process again. This continuous cycle persists as long as the program is running.

2. Presentation and Analysis of Plots

2.1 Java Runtime

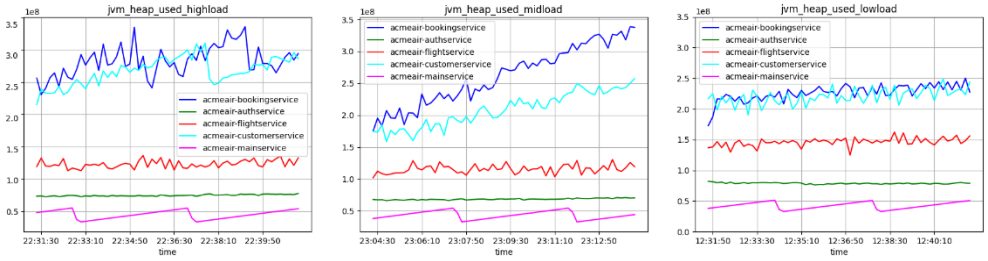

Fig. 1. Jvm.heap.used of high workload, medium workload and low workload

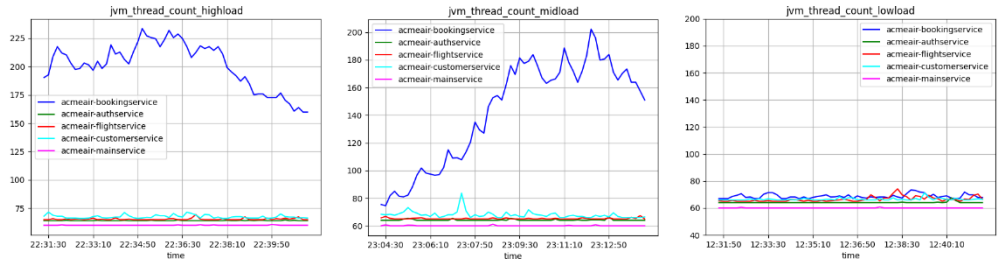

Fig. 2. Jvm.thread.count of high workload, medium workload and low workload

As heap is a shared area that is utilized during the runtime of Java applications, more resources need to be allocated to the bookingservice with the number of requests increases, resulting in fewer resources for other services (Fig.1.).

At high loads, the number of threads in the bookingservice reaches more than 200 and decreases as the load situation decreases (Fig.2.), while the number of threads in the

other services remains relatively stable.

2.2 Application Specific

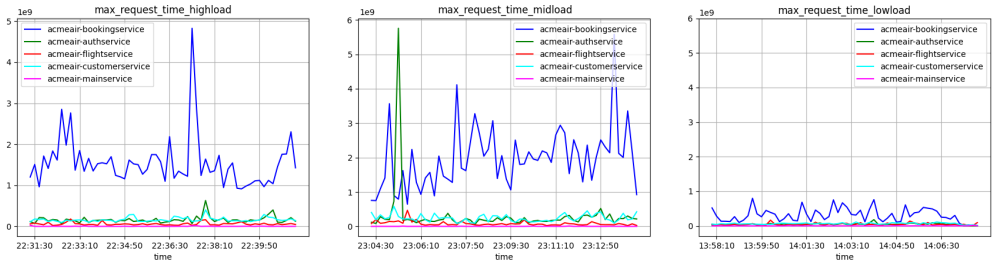

Fig. 3. Maximum response times of high workload, medium workload and low workload

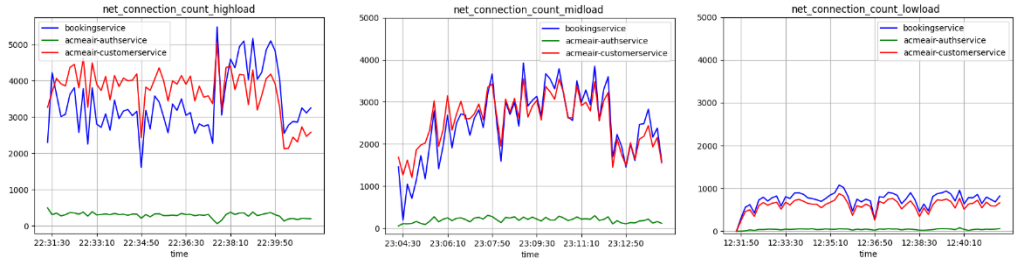

Fig. 4. Transactions over time of high workload, medium workload and low workload

The higher the load, the higher max request time for the bookingservice (Fig.3.), which is due to the fact that more accesses are made to the sever. When the server experiences a high workload with numerous client requests, it is likely to establish more outgoing connections to accommodate those requests (Fig.4.).

2.1 System Resources

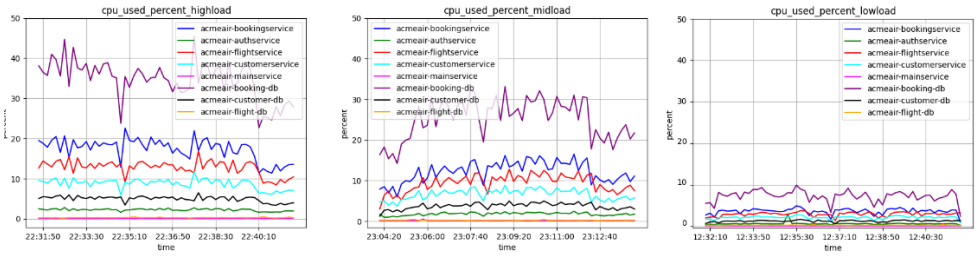

Fig. 5. CPU used percentage of high workload, medium workload and low workload

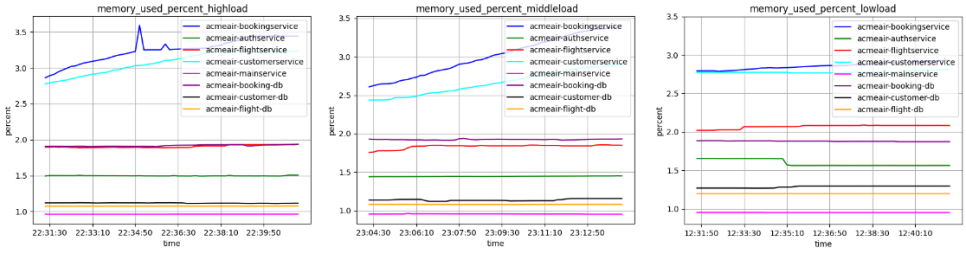

Fig. 6. Memory used percentage of high workload, medium workload and low workload

CPU utilization increases (Fig.5.) because as the load increases more data needs to be processed. The booking database takes up the highest amount of CPU usage as accessing calls for more resources. In terms of memory footprint, bookingservice and customerservice remain high (Fig.5.), and overall increase with higher workloads.