Submitted by: chagar2, ntalasu2

REPORT - CS425 - MP2 - Group 59

Design

Algorithm

This program implements a heartbeat-based membership protocol with failure detection. The N processes in the system are arranged in a pseudo-ring topology which is updated every time a process joins/leaves/exits the system. The system consists of four components running simultaneously/continuously – client, server, failure detector and heartbeat handler.
One chosen server acts as the introducer node which is responsible for allowing new processes to join the membership system. No other process is allowed to join the system when the introducer is down. When the introducer is back up, all existing processes automatically rejoin the system.
Failure detector detects a process as failed if it does not receive a heartbeat in the last 1 second and removes the process from the membership list if heartbeat is not received for 2 seconds.
Whenever a failure is detected, all membership lists are updated within 2-3 seconds, thus maintaining time-bound completeness.
Each system sends its heartbeat to 1 predecessor and 2 successor nodes every 500 milliseconds. Since it is guaranteed that there cannot be more than 3 simultaneous failures, the system ensures completeness.
The client thread expects various inputs from the user to join, leave, exit system, and print useful information such as nodeId, membershipList, etc.
All the messages use UDP protocol, and the message payload is represented in a JSON format that is converted to byte array during transmission.

Scaling

Every time a process joins/fails/leaves, the membership list is automatically updated in the system and also each process maximum sends 3 messages as each process has only 3 neighbors (1 predecessor and 2 successors) regardless of the size 'N' of the system.
Each message only contains the message type (JOIN, LEAVE, FAILED) and minimal process information.

Marshalled message format

A JSON of the following format:

```
{
    messageType: JOIN/LEAVE/HEARTBEAT
    id: IpAddress_timestamp,
     count: heartbeat_count
}
```

Usefulness of MP1

The distributed logging implemented in MP1 was very useful while debugging MP2. The GrepLogger from MP1 particularly helped in identifying corner-case failures that were difficult to track on the console, especially since a huge number of messages were being transmitted across the network, and it was difficult to track some specific messages which had a low frequency of occurrence.

Measurements

- Background bandwidth usage when network has 6 machines = 6.78 kB per node per second
- When a 7th process **joins** the system: An additional 562 bytes usage for each node plus some minor overhead for the introducer. This overhead is due to the introducer handling the join for the new process
- When the 7th process **leaves** the system: The leaving node sends 279 bytes to indicate to its 3 neighbors that it is leaving the system.
- **Exit**: For the failed process, there won't be any message transmission. The bandwidth for the remaining 5 processes is 5.74 kB per node per second.

As one can see from the below graphs, the trend is the False Positive Rate (FPR) Mean, Standard Deviation and Confidence Interval increase with increase in the packet loss frequency. This is expected as greater the packets loss, greater is the chance of a failure getting detected. The FPR Mean for N=2 machines is higher than the FPR mean for N=6 machines. This is expected because, for N=2, the mean FPR is close to the packet loss frequency due to there being only 2 processes which are sending heartbeats to each other. For N=6, mean FPR is slightly lower as all processes do not depend on heartbeats to identify a particular failure. This is instead reflected via the membership lists. The FPR for standard deviation is greater for N=6. This is expected as the randomness of packet loss is higher when the number of processes are higher.