CS425 MP2 Report

Xilun Jin and Yuchen Liang

Introduction

In this MP, we implemented a distributed group membership system using SWIM, where every machine keeps a full membership list, and keep pinging other machine to get update.

Design

Message format:

- request for joining: `vm_num:timestamp`
- send membership_list: `current_vm_number|vm_number1+timestamp1,incarnation_number,status|vm_number2+timestamp2,incarnation_number,status|...`

New join:

We have an introducer(which is first vm1 in our case) to listen for the join request. The request format is `vm_num:timestamp`, which is just the id for the process. Then introducer will add this new process to its membership list and send its membership to the new-joined process

Ping and ACK:

Every machine ping to the next four alive machines in its membership list. This can avoid 3 machines fail simultaneously and cause failure unable to be detected. And ACK from pinged machine will piggyback the its membership list. To distingush the new change of a machine, we have a incarnation number for every machine in the membership list. This number only increment when a machine join, leave, or refuse a failure detection on itself. So when update the membership list, we compare the incarnation number and take the record with higher incarnation number, or if same, take the failure one.

Logging:

We logg each time a change is made to the local membership list and each time a failure is detected or communicated from one machine to another

Debug with MP1 code

We can use MP1 code easily grab the log files to see when a machine join, left or fail and which machine pinged or acked other machine. We can use count to find out if all machine received the update.

Analysis:

To measure the bandwidth, we used tcpdump and the codes from https://superuser.com/questions/356907/how-to-get-real-time-network-statistics-in-linux-with-kb-mb-bytes-format-and-for

- Background bandwidth usage for 4 machines (in Bps)

Time Interval\Machine#	**1**	**2**	**3**	**4**
T1	1898.98	1774.68	1899.00	1898.97
T2	1773.08	1898.97	1872.41	1925.47
T3	1898.95	1898.99	1898.90	1890.48
T4	1903.25	1925.41	1898.96	1925.49

- Average bandwidth usage for a new node joins a 4 machine group (in Bps)

Machine	**1**	**2**	**3**	**4**	**Avg**
	1907.59	1903.24	1914.95	1898.94	1906.18

- Average bandwidth usage for a node leaves a 4 machine group (in Bps)

Machine	**1**	**2**	**3**	**4**	**Avg**
	1864.13	1685.27	1898.90	1803.31	1812.53

- Average bandwidth usage for a node crashes a 4 machine group (in Bps)

Machine	**1**	**2**	**3**	**4**	**Avg**
	1929.76	1726.34	1388.28	1898.99	1735.25

- False positive rate with message loss

message loss rate\Machine #	**2**	**4**
3%	0.024, 0.018,0.015,0.023,0.029	0.053,0.045,0.052,0.033,0.043
10%	0.082,0.060, 0.072,0.058, 0.085	0.102, 0.093, 0.084, 0.092, 0.078
30%	0.162, 0.143,0.137, 0.158, 0. 145	0.352, 0.284, 0.269,0.318, 0.238
