

LevelUpDB

A Cloud-Native version of LevelDB

Benita Kathleen Britto
Krishna Ganerwal
Reetuparna Mukherjee
Shreyansh Sharma

Design





Specifications



[LEVELDB](#): CLOUD KEY-VALUE
DATA STORAGE



[RAFT](#): TO REACH CONSENSUS



[GRPC C++](#): FOR RPC



TUNABLE CONSISTENCY:
STRONG AND EVENTUAL



LOAD BALANCER: TO
DISTRIBUTE LOAD AMONG
PARTICIPATING NODES AND
MAINTAIN SYSTEM STATE



DYNAMIC RESOURCE
ALLOCATOR: TO SUPPORT ON
DEMAND ADDITION OR
REMOVAL OF NODES



Assumptions

No network
partition

Put and Get APIs
support only
string data type

Batch
Update/Creation
not supported

Put API for both
KV Creation and
Update

Load Balancer will
always be up

Sufficient memory
for commit logs
on each node

Cannot tolerate
Byzantine Faults

Does not account
for security

No disk failures
and other forms
of data corruption



APIs



Get()



Put()

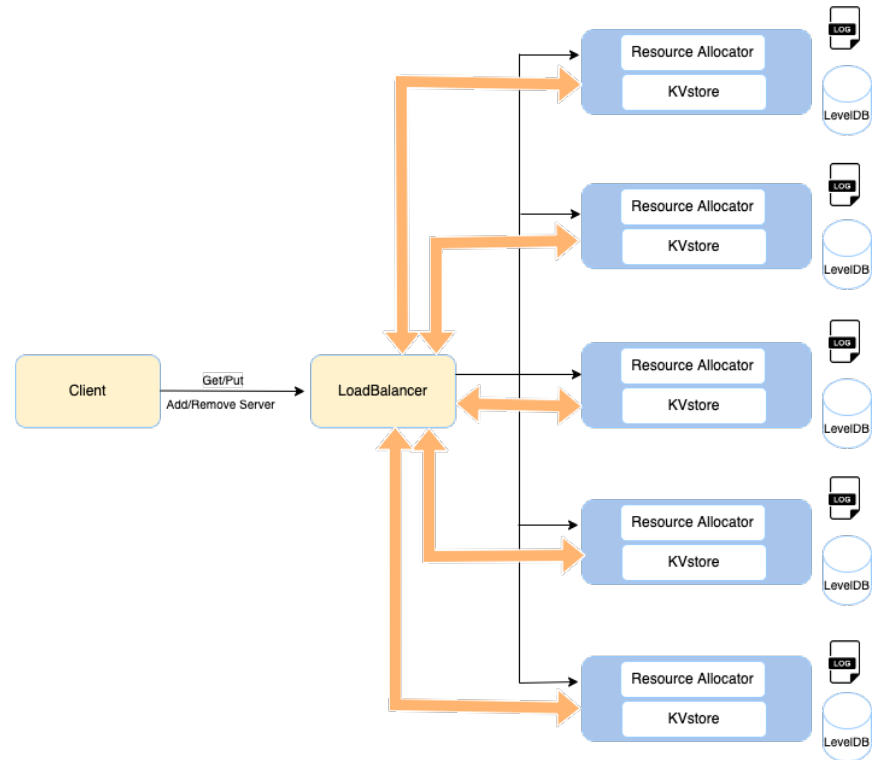


AddServer()

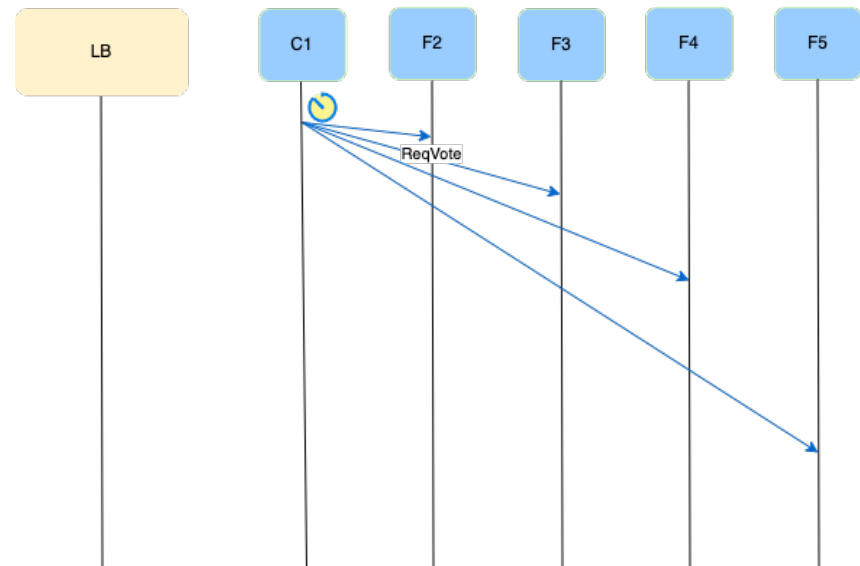


DeleteServer()

System Overview

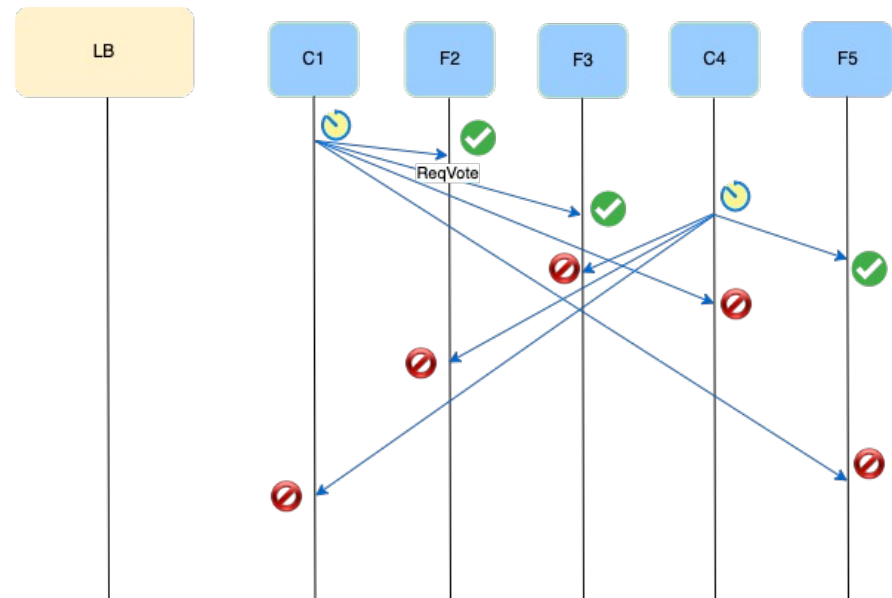


Leader Election



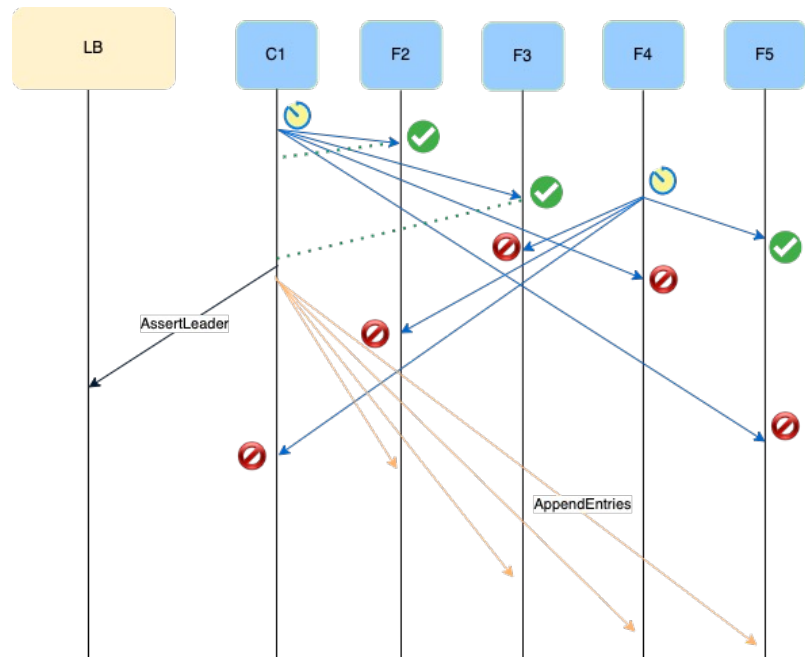
LB: Load Balancer
F: Follower
C: Candidate

Leader Election



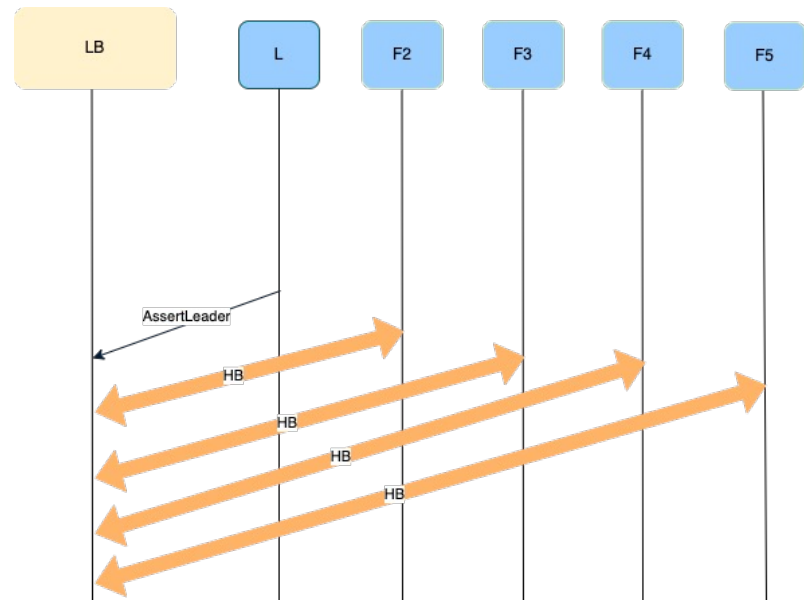
LB: Load Balancer
F: Follower
C: Candidate

Leader Election



LB: Load Balancer
C: Candidate
F: Follower

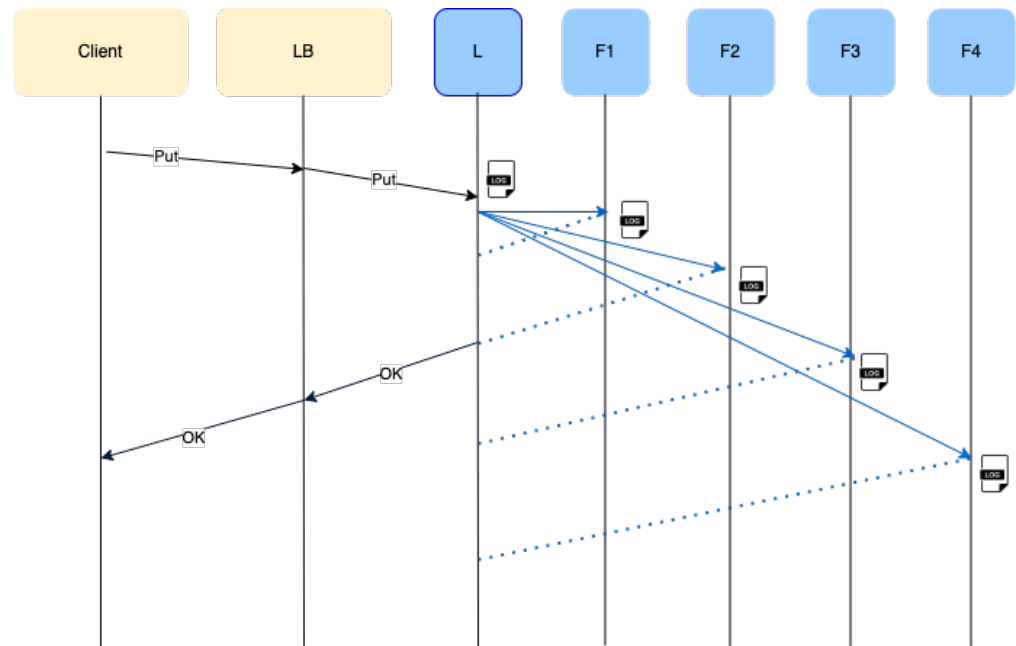
Leader Election



LB: Load Balancer
F: Follower
L: Leader

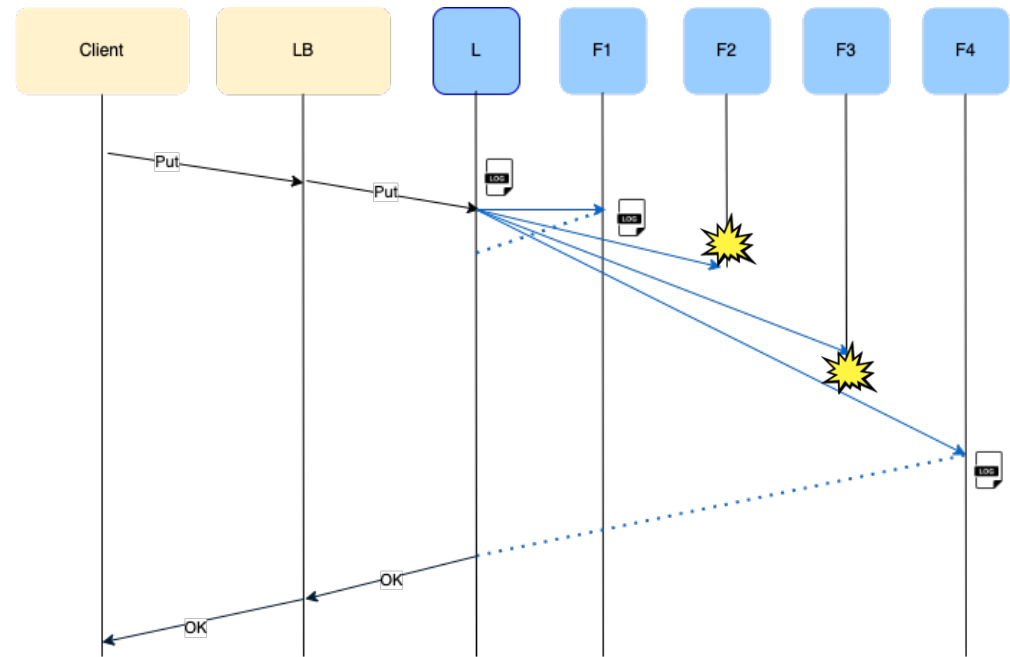


Put without Crash

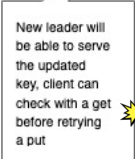


LB: Load Balancer
L: Leader
F: Follower

Put with Crash on Follower

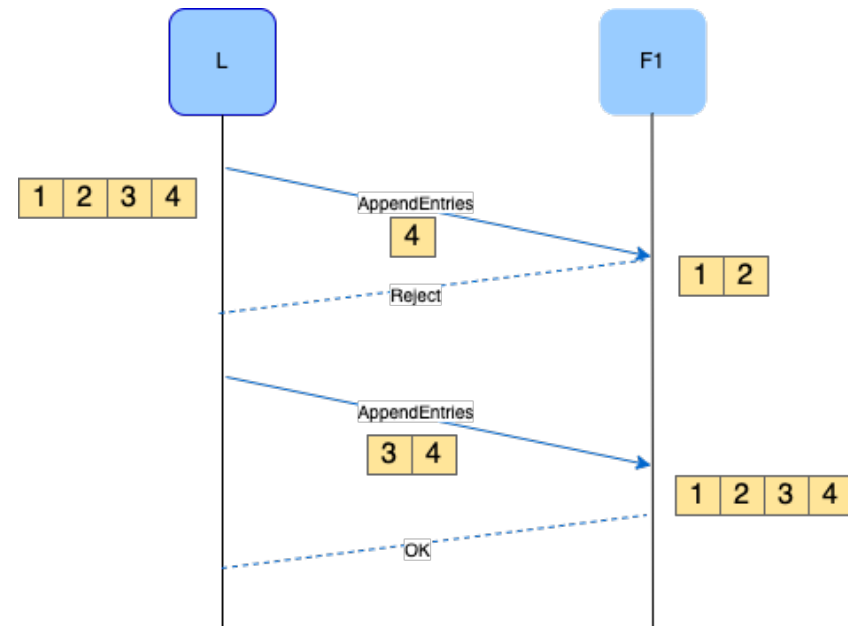


LB: Load Balancer
L: Leader
F: Follower



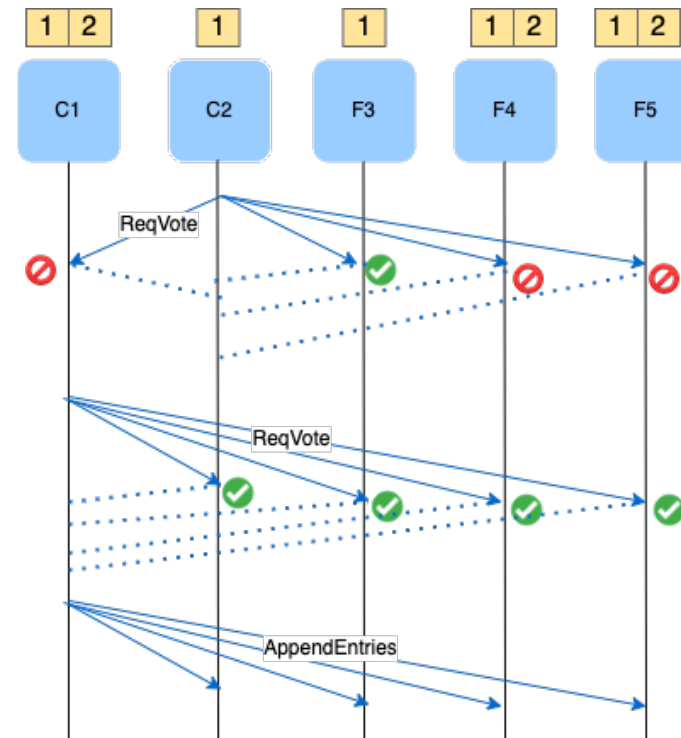
F: Follower

Correctness: Append Entries



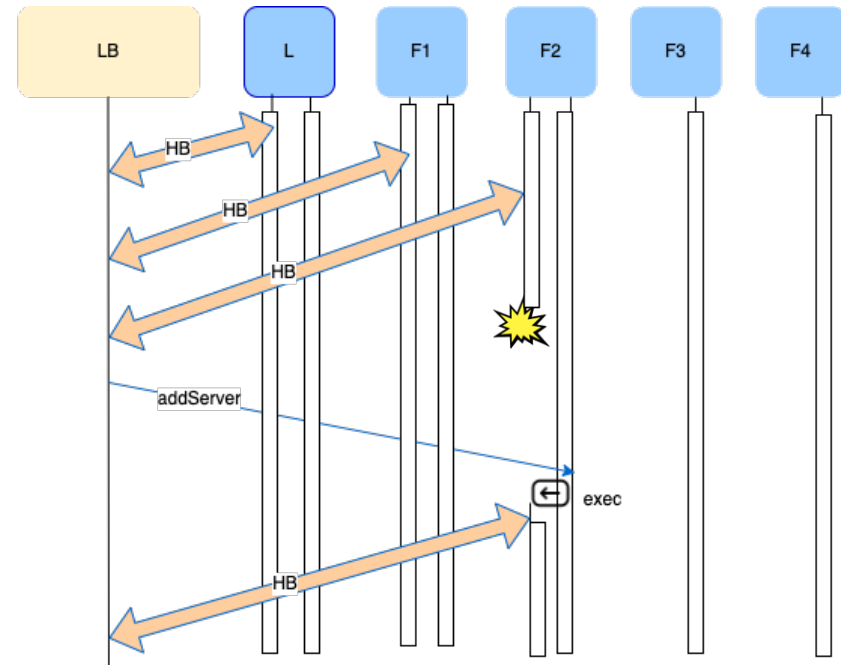
L: Leader
F: Follower

Correctness: Leader Election



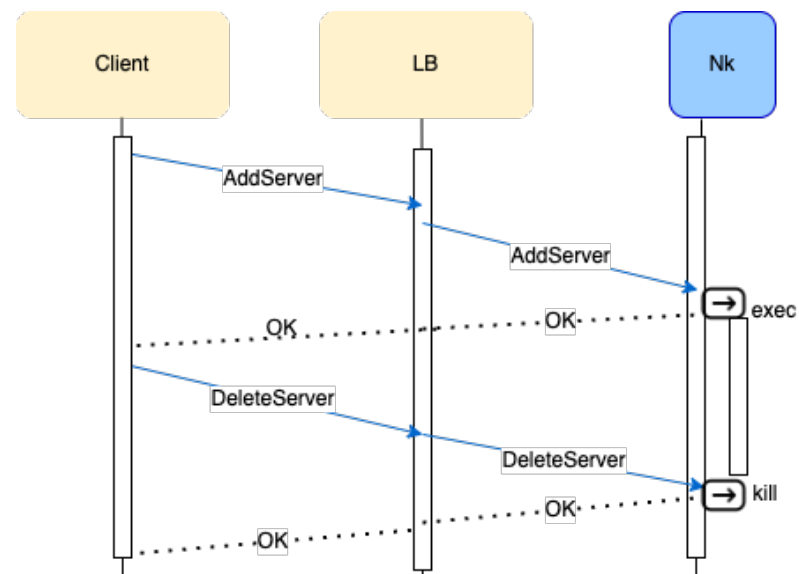


Resource Allocator: Autoscaling



LB: Load Balancer
L: Leader
F: Follower

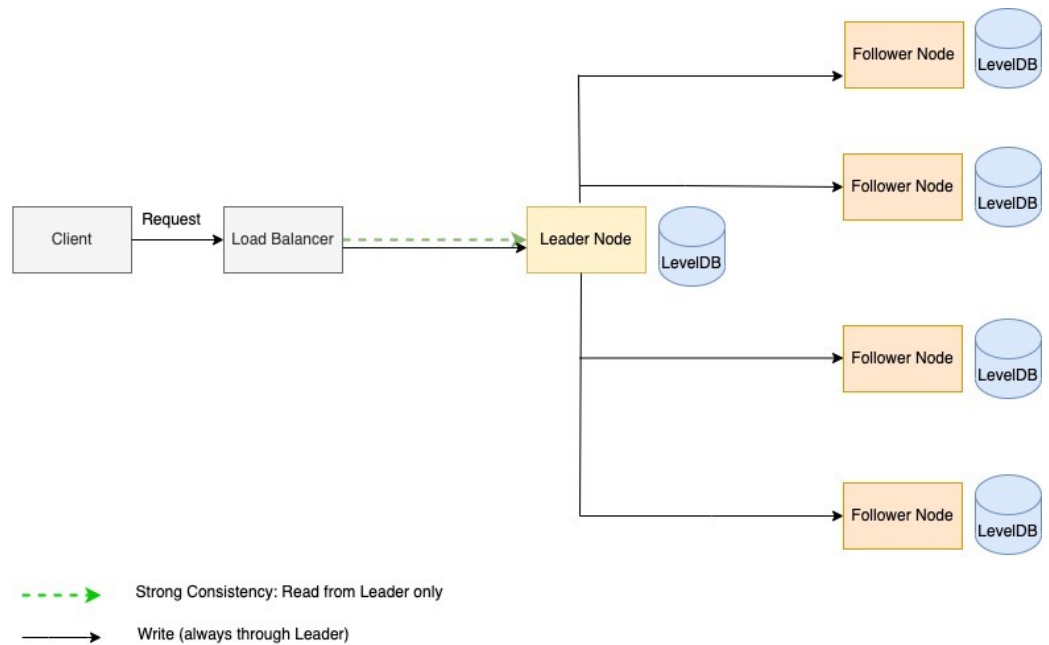
Resource Allocator: Scaling out and in



LB: Load Balancer
L: Leader
N: Node

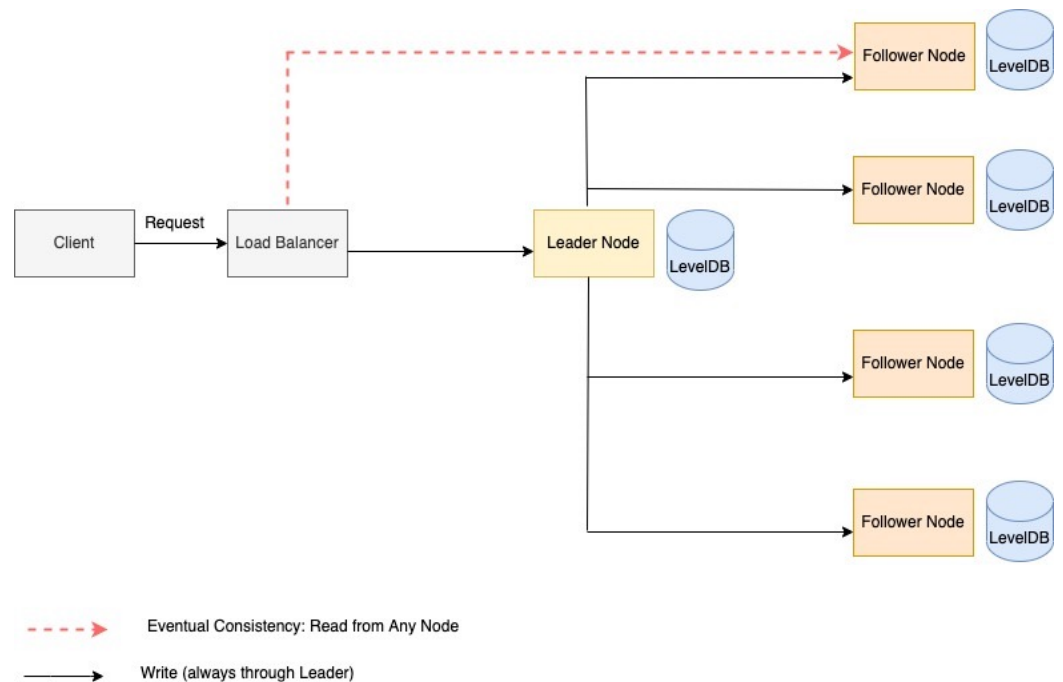


Get: Strong Consistency






Get: Eventual Consistency (Quorum of 1)



Measurements





Testing Strategy and System Specs



Crash points: Macros that cause a seg fault



Automated scripts for Get() and Put()



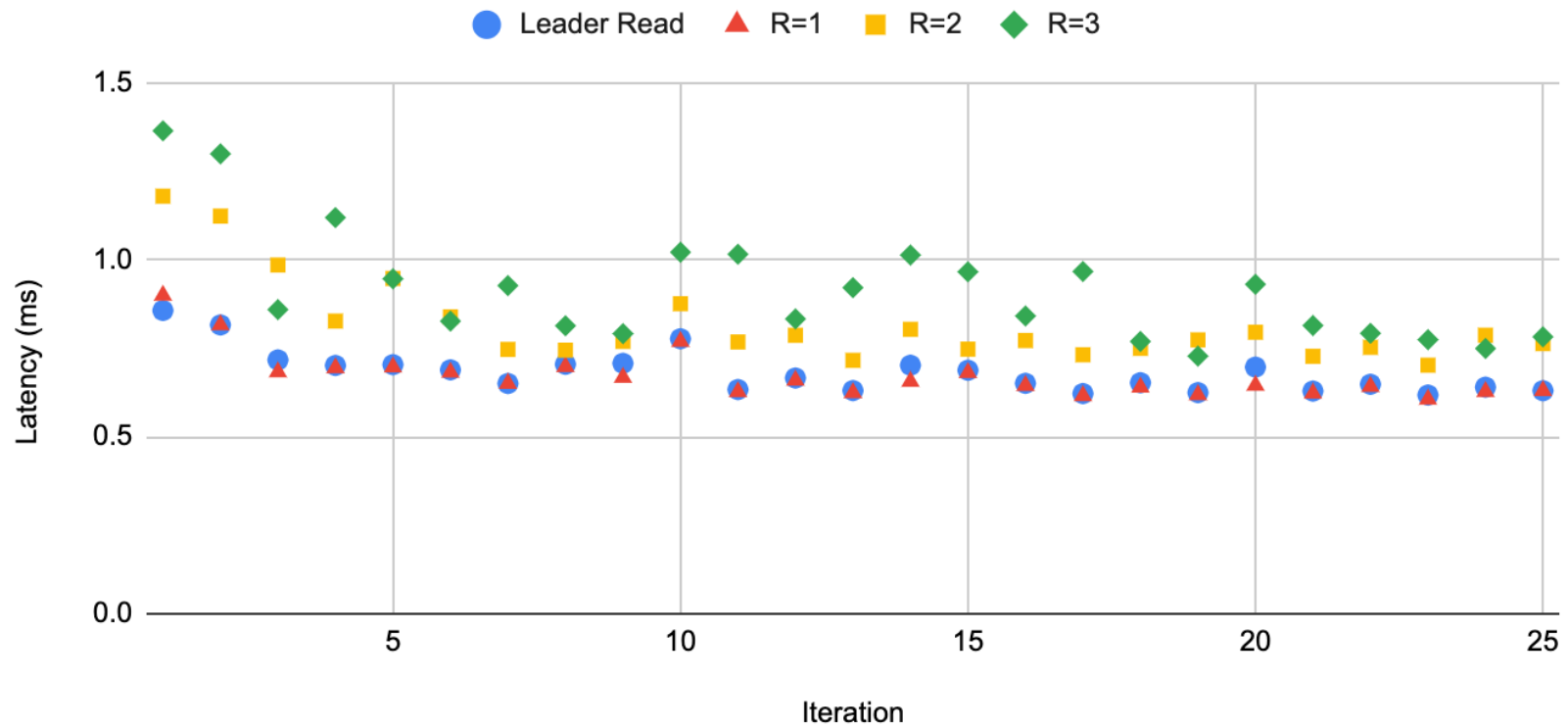
System Details :

CloudLab

Ubuntu
20.04 LTS

32 cores

Read Latency

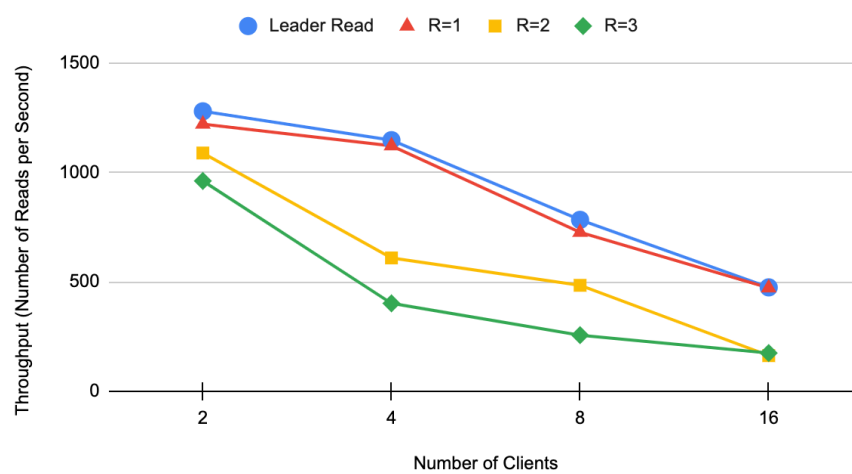


Total Servers = 3
sizeof(key) = 16 B
sizeof(value) = 100 B

Throughput of Concurrent Reads on Same Key

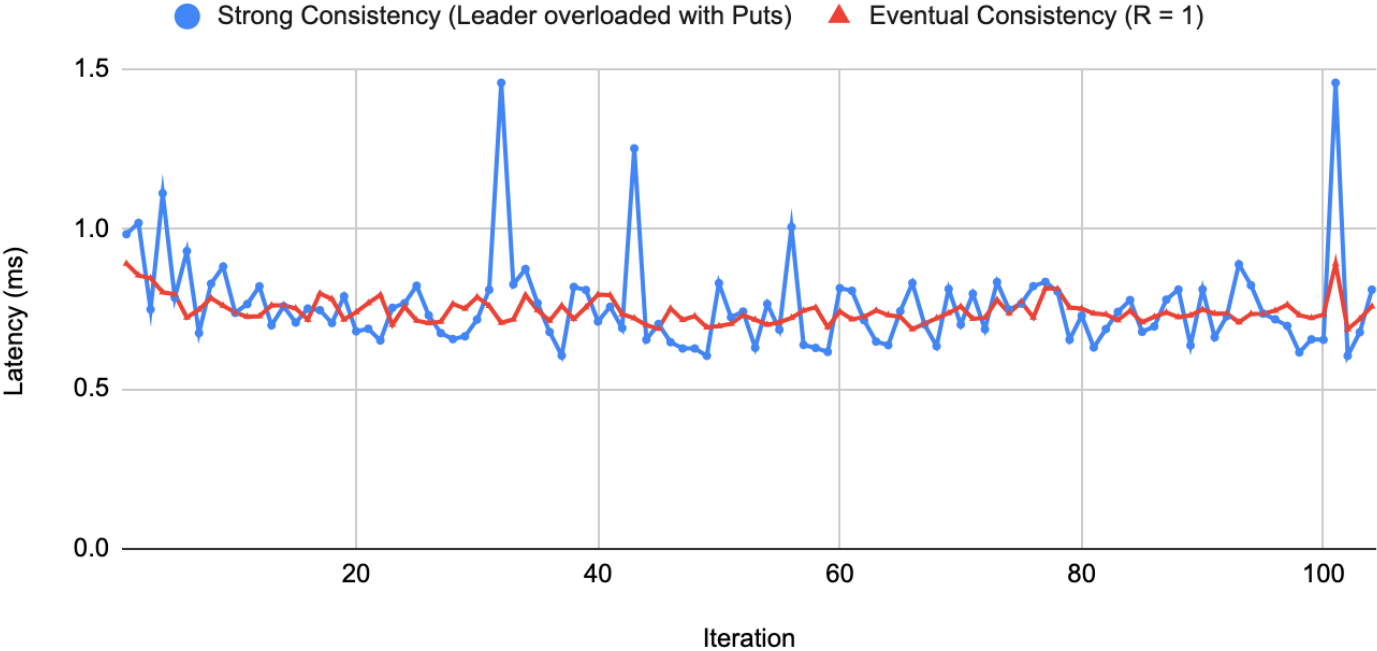


Throughput of Concurrent Reads on Different Keys



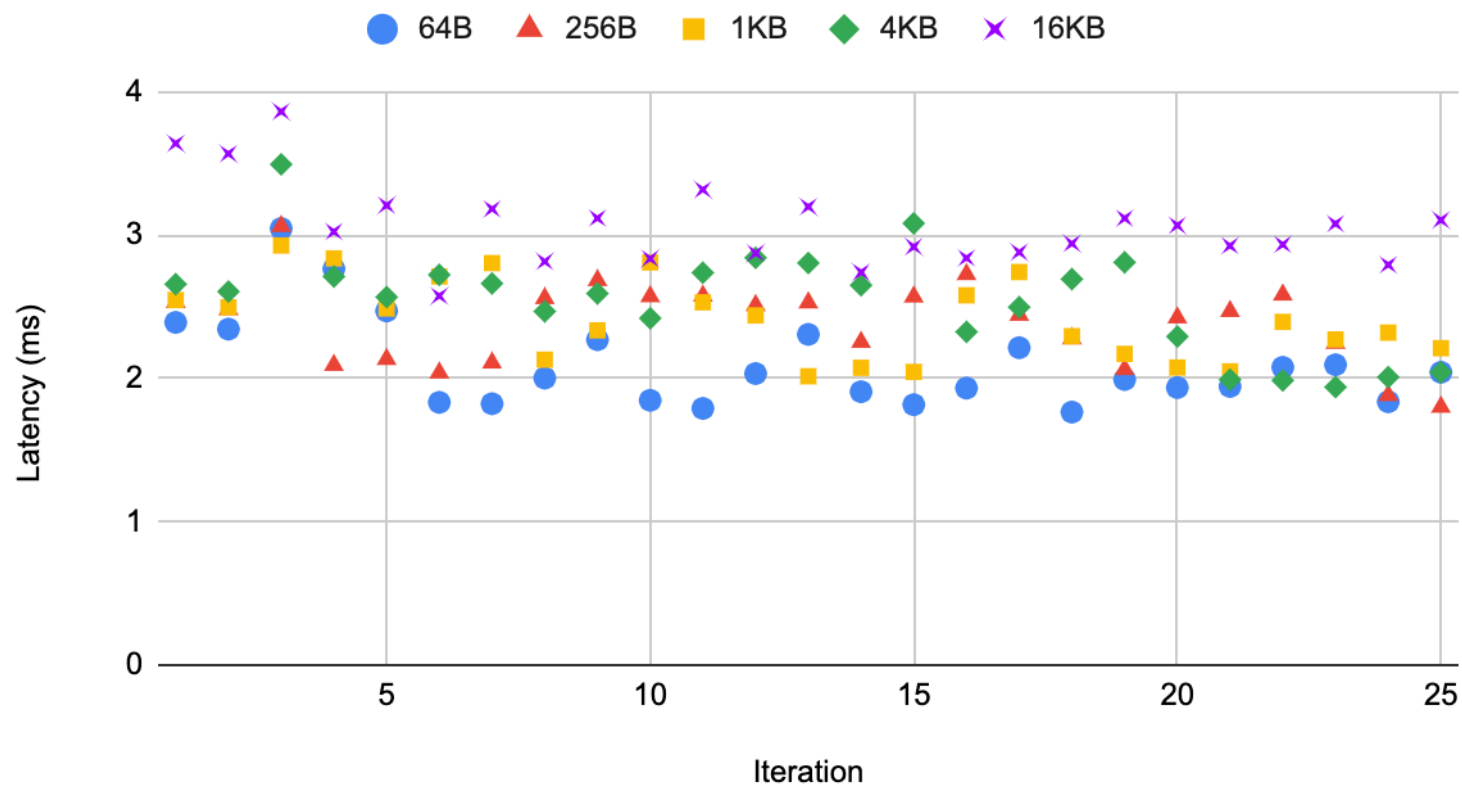
Total Servers = 3
sizeof(key) = 16 B
sizeof(value) = 100 B

Strong Consistency vs Eventual Consistency



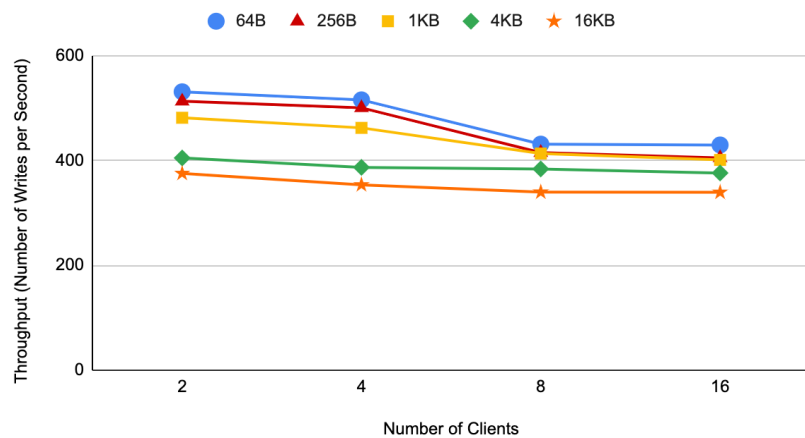
Total Servers = 3
sizeof(key) = 16 B
sizeof(value) = 100 B
Same Key

Write Latency

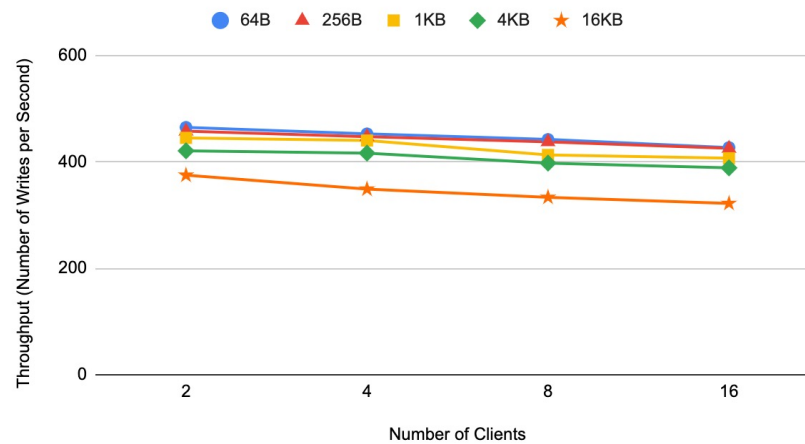


Total Servers = 3
sizeof(key) = 16 B

Throughput of Concurrent Writes on Same Keys

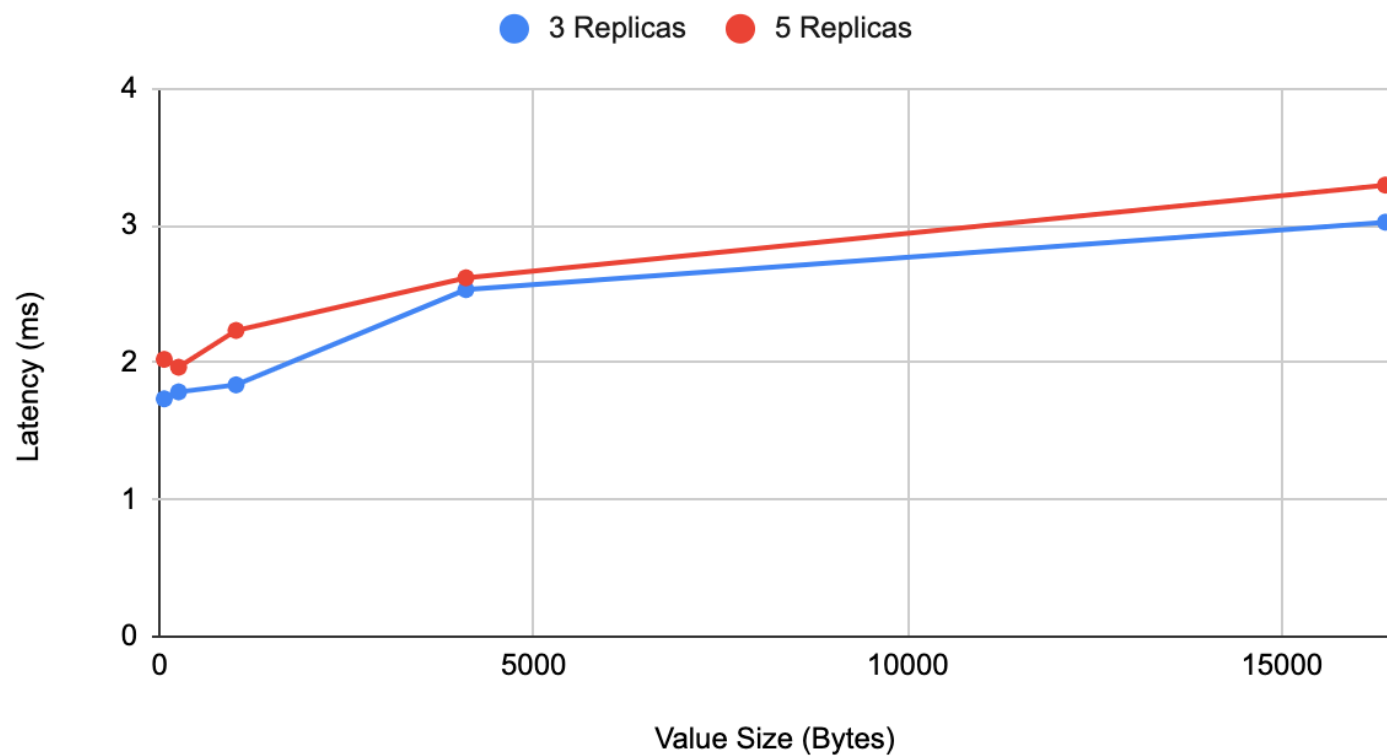


Throughput of Concurrent Writes on Different Keys

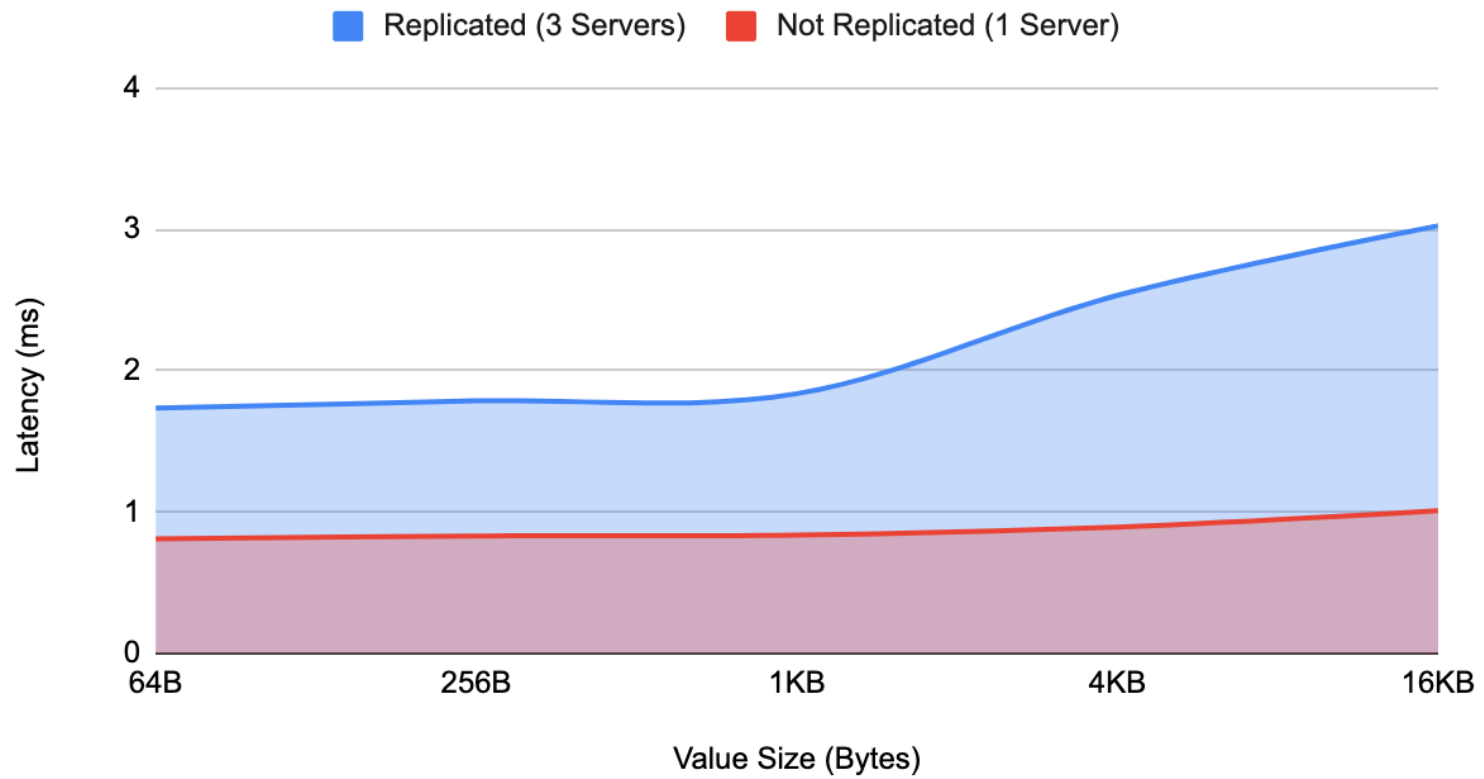


Total Servers = 3
sizeof(key) = 16 B

Write Latency: 3 Replicas vs 5 Replicas



Replication Overhead



sizeof(key) = 16 B

Demos





Demo List

Raft In Action

Autoscaling

Correctness: Leader Election

Leader Crash

Tunable Consistency

Raft in Action

The screenshot displays a terminal window with four tabs, each showing the output of a Raft-related command. The tabs are labeled 'ssh' and show the following content:

- Top-left tab:** Shows the command `node-1:~/CS739-P4/src/cmake/build> ./keyvalue_client 0.0.0.0:50051 -p key value`. The output is empty.
- Top-right tab:** Shows the command `node-1:~/CS739-P4/src/cmake/build> ./loadbalancer 0.0.0.0:50051 0.0.0.0:50052`. The output is `LB Server for Client listening on 0.0.0.0:50051 LB Server for Nodes listening on 0.0.0.0:50052`.
- Bottom-left tab:** Shows the command `node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60001 0.0.0.0:60002 0.0.0.0:50052`. The output includes:

```
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/term_vote0.0.0.0:60001
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/replicated_log0.0.0.0:60001
[INFO] KeyValue Server listening on 0.0.0.0:60001
```
- Bottom-middle tab:** Shows the command `node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60011 0.0.0.0:60012 0.0.0.0:50052`. The output includes:

```
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/term_vote0.0.0.0:60011
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/replicated_log0.0.0.0:60011
[INFO] KeyValue Server listening on 0.0.0.0:60011
```
- Bottom-right tab:** Shows the command `node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052`. The output includes:

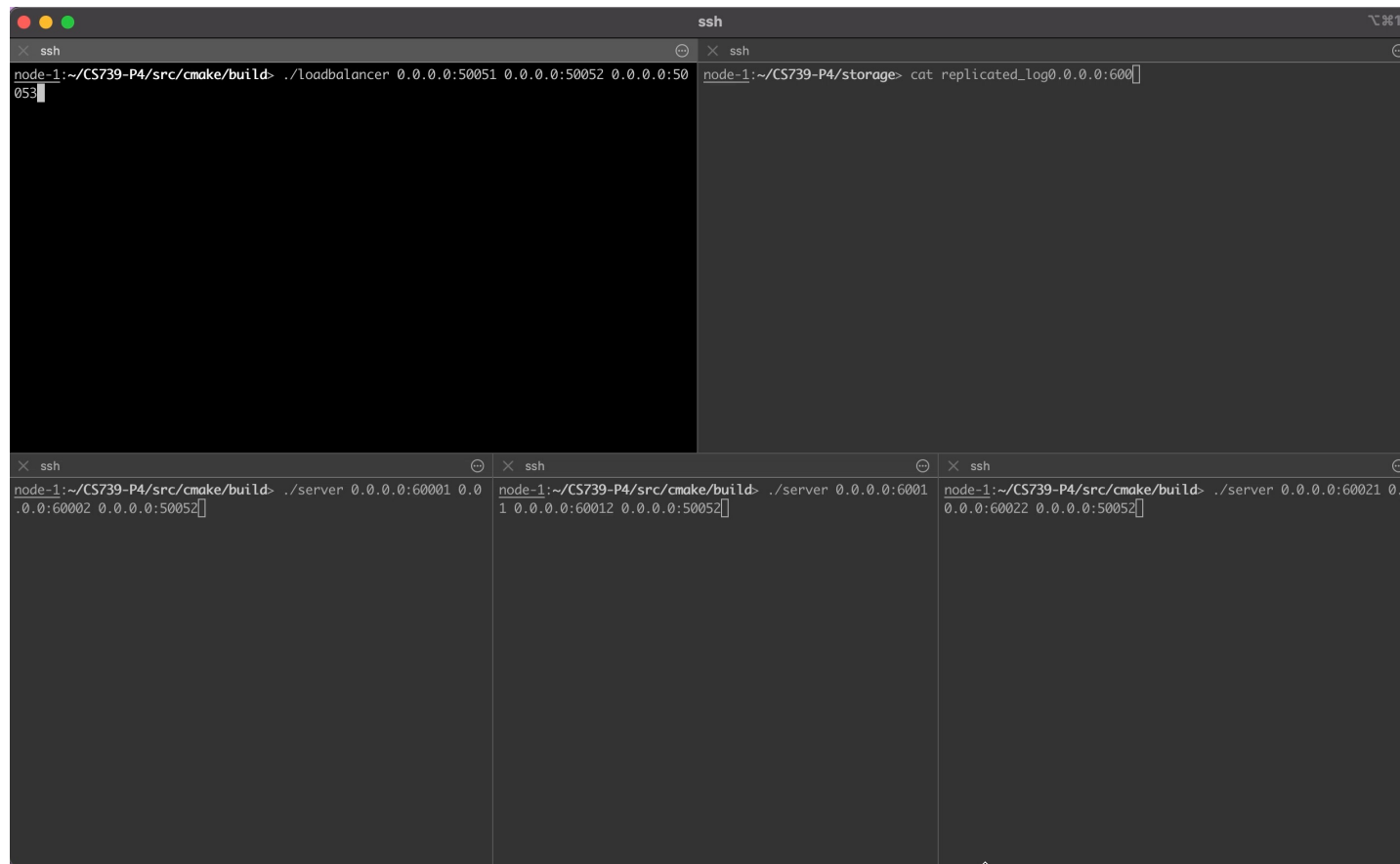
```
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/term_vote0.0.0.0:60021
[INFO]: Logging at: /users/ssharma/CS739-P4/storage/replicated_log0.0.0.0:60021
[INFO] KeyValue Server listening on 0.0.0.0:60021
```

Autoscaling

The screenshot shows a terminal window with four tabs, all titled 'ssh'. The first tab shows the command `./loadbalancer 0.0.0.0:50051 0.0.0.0:50052 0.0.0.0:50053` being executed. The second tab shows the command `./server 0.0.0.0:60001 0.0.0.0:60002 0.0.0.0:50052`. The third and fourth tabs show the command `./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052`. The terminal output is mostly black, indicating that the processes are running successfully.

```
node-1:~/CS739-P4/src/cmake/build> ./loadbalancer 0.0.0.0:50051 0.0.0.0:50052 0.0.0.0:50053  
  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60001 0.0.0.0:60002 0.0.0.0:50052  
  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052  
  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052
```


Correctness: Leader Election



The screenshot displays a terminal window with four panes, each representing a different node in a distributed system. The panes are arranged in a 2x2 grid. The top-left pane shows a node-1 prompt with a command to run a loadbalancer. The top-right pane shows a node-1 prompt with a command to cat a replicated_log file. The bottom-left pane shows a node-1 prompt with a command to run a server. The bottom-right pane shows a node-1 prompt with a command to run a server. The terminal output shows the results of these commands, including IP addresses and port numbers.

```
node-1:~/CS739-P4/src/cmake/build> ./loadbalancer 0.0.0.0:50051 0.0.0.0:50052 0.0.0.0:50053  
node-1:~/CS739-P4/storage> cat replicated_log0.0.0.0:6000  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60001 0.0.0.0:60002 0.0.0.0:50052  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60011 0.0.0.0:60012 0.0.0.0:50052  
node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052
```

Leader Crash

```
node-1:~/CS739-P4/src/cmake/build> ./keyvalue_client 0.0.0.0:50051 -p
key2 value2

node-1:~/CS739-P4/src/cmake/build> ./loadbalancer 0.0.0.0:50051 0.0.0.0:50052

node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60001 0.0.0.0:60002 0.0.0.0:50052

node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60011 0.0.0.0:60012 0.0.0.0:50052

node-1:~/CS739-P4/src/cmake/build> ./server 0.0.0.0:60021 0.0.0.0:60022 0.0.0.0:50052
```

Tunable Consistency

The screenshot shows a terminal window with four panes, each representing a different node in a distributed system. The top-left pane shows a client command being executed. The top-right pane shows a load balancer command. The bottom-left and bottom-right panes show server commands. The middle-left and middle-right panes show the output of the server commands, displaying a list of IP addresses and ports.

```
node-1:~/CS739-P4/src/cmake/build> ./keyvalue_client 0.0.0.0:50051 -p key6 value61
```

```
node-1:~/CS739-P4/src/cmake/build> ./loadbalancer 0.0.0.0:50051 0.0.0.0:50052
```

```
node-1:~/CS739-P4/src/cmake/build> ./server 128.105.144.139:60001 128.105.144.139:60002 128.105.144.139:50052
```

```
node-1:~/CS739-P4/src/cmake/build> ./server 128.105.144.139:60031 128.105.144.139:60032 128.105.144.139:50052
```

```
node-1:~/CS739-P4/src/cmake/build> ./server 128.105.144.139:60011 128.105.144.139:60012 128.105.144.139:50052
```

```
node-1:~/CS739-P4/src/cmake/build> ./server 128.105.144.139:60021 128.105.144.139:60022 128.105.144.139:50052
```



Work In Progress

SNAPSHOT:

- For Log Compaction, save KV State with the Last Index and Last Term of System when Snapshot is taken.
- Snapshot is taken periodically at every node.
- Leaders shares its updated snapshot with newly joining nodes in cluster for quick updates.



Conclusion

- LevelUpDB, our cloud native version of LevelDB, uses RAFT as the consensus protocol to maintain consistency between replicas.
- Replication comes with a tradeoff on performance
- To provide flexibility for different use-cases, LevelUpDB offers tunable consistency levels, wherein clients can specify how they want their read requests to be processed.
- It offers a Resource Allocator to manage the number of replicas running and offers capability to dynamically add and remove servers while also managing crashes.



Benita Britto
benitakbritto



Reetuparna Mukherjee
reetuparna

Thank you!



Krishna Ganeriwal
ganeriwalk



Shreyansh Sharma
shreyansh21sharma