# Replicated Block Store

CS 739, Project - 3, Group 14

Aakarsh Agarwal
Muhammad Asad Khan
Bijan Tabatabai
Salman Munaf

(Dept. of Computer Sciences/CDIS)

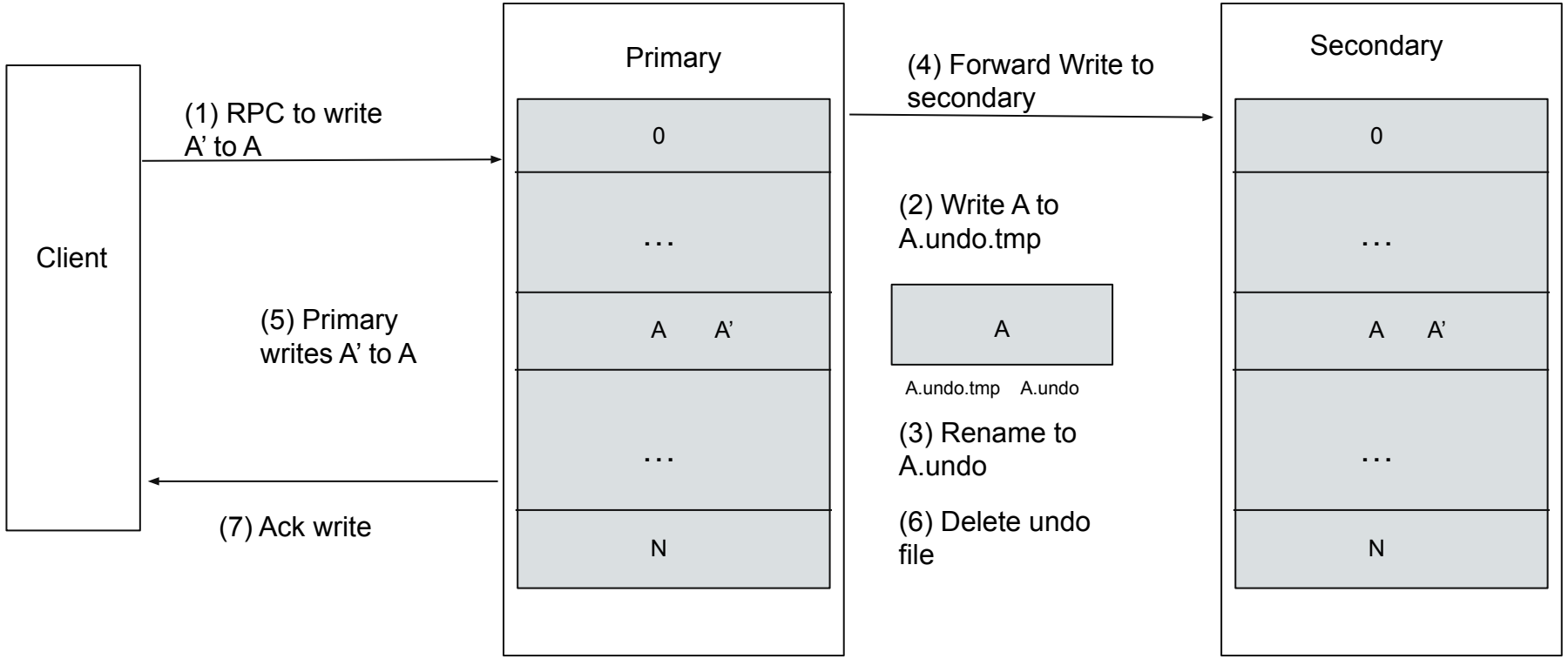
# Design decisions - Replication Strategy

- Primary/Backup Replication
  - 2 servers - one is started as Primary, other one as Backup
  - Client sends read and write requests only to Primary
  - Reads involve Primary reading from its disk and sending data back to the client
  - Writes involve both Primary and Backup writing the data on their respective disks (replication of data)
  - Communication between Primary and Backup
    - Heartbeats: To identify server crash
    - Copy/Replication of data: Write data is copied to the other server and other server acknowledges it after copy has been made
  - Communication between Client and Server
    - Read request
    - Write request

# Design decisions - Durability

- Data stored in single file
- Undo logging
  - Before overwriting data, save old data into undo file
  - When write completes, delete undo file
  - If undo file exists on server startup, apply undo to data
- Reader/Writer locks serialize writes to 4K aligned blocks of data
  - If write is unaligned, must grab two locks

# Design decisions - Walkthrough of normal write



# Design decisions - Crash Recovery Protocol - Crash identification

- Primary sends heartbeat to Backup every second
- If Primary does not get back ack from Backup, Backup crash is identified
- If Backup does not receive heartbeat from Primary for 5 seconds, Primary crash is identified

# Design decisions - Crash Recovery Protocol (Primary crashes)

- Backup identifies itself as primary and starts serving clients
- New Primary keeps on sending heartbeats to identify old primary (now backup) reboot
- New Primary keeps logs of all writes till backup is rebooted
- Once backup is live, new Primary plays the log and sends all writes one by one

# Design decisions - Crash Recovery Protocol (Backup crashes)

- No change in reads handled by primary
- Primary keeps on sending heartbeats to identify backup reboot
- Primary keeps logs of all writes till backup is rebooted
- Once backup is live, Primary plays the log and sends all writes one by one

# Design decisions - Crash Recovery Protocol (Client library)

- Client needs to hide crash from user
- Client makes the initial call to any one of the two nodes
- In case of crash, client will alternate between two nodes till Primary is found and request is processed successfully
- Possible responses
  - Blockstore_Success - Client called Primary and request was executed successfully
  - Blockstore_Not_Primary - Client called Backup. Client will now call the other node
  - Failure/Crash Case
    - Client call to Primary (in crashed state) fails
    - Client makes call to other node
      - Blockstore_Success - If Backup has identified itself as Primary
      - Blockstore_Not_Primary - Backup yet to identify itself as Primary, calls 1st node again

# Experimental Setup

- For latency measurements, 3 Cloudlab C220g1 nodes connected on a 1Gbps network link
  - Primary, Backup, and client ran on separate nodes

- For crash demonstrations, servers and client were run on one machine
  - Not ideal, but we had a limited number of machines and did not want to interfere with performance results

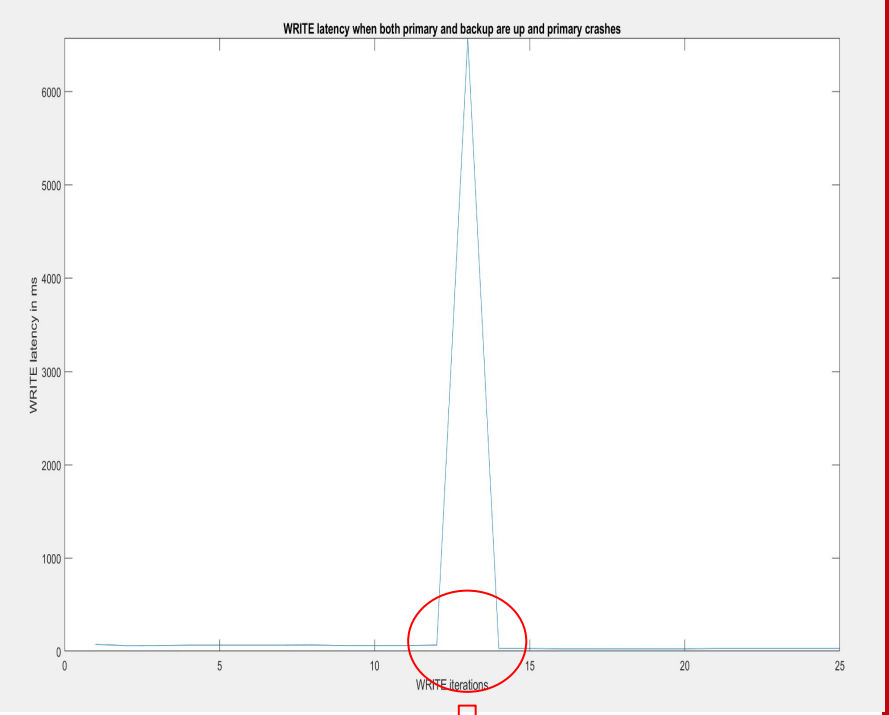
# Read latency (replicated vs single server)

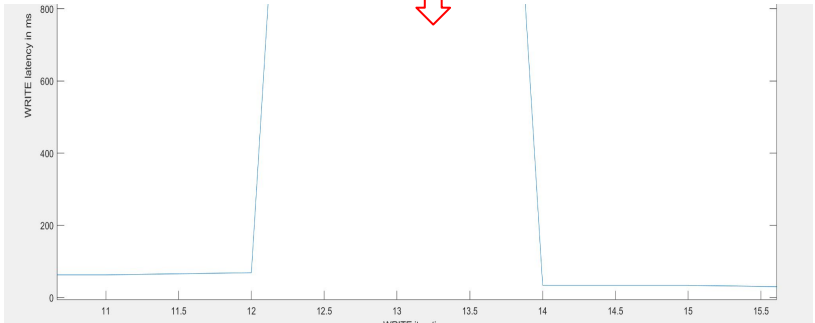
- The READ latency for replication and single server is same as we service READ requests only from the primary

- The average READ latency was 910 microseconds

# Write latency

- Write latency before crash: 64 ms
- Write latency after backup becomes new primary: 31ms
- Write on single-server system (without replication): 31ms



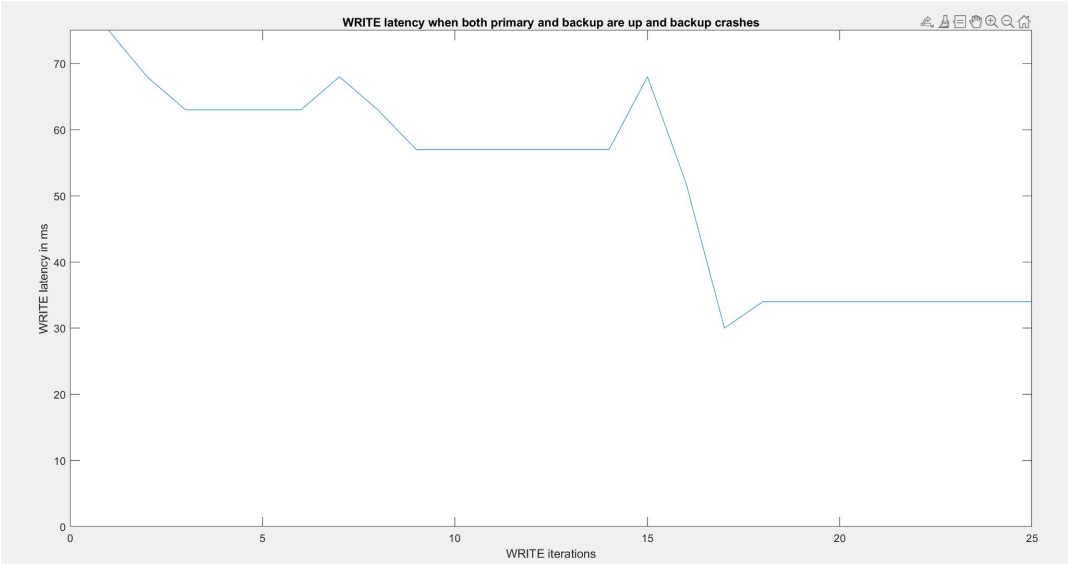
# Read latency with crashes (mean)

- After Backup is promoted to primary, READ latency is the same as before

- Backup crashes don’t affect read latency

# Write latency (Backup crash) (mean, std deviation)

- The average WRITE latency in case of backup crash is 51 ms and the standard deviation is 14.32 ms



# Recovery timeline

- The average failover time is 5548 ms

- The average time to reintegrate 20 WRITE requests is 679 ms

# Read and Write (aligned vs unaligned address)

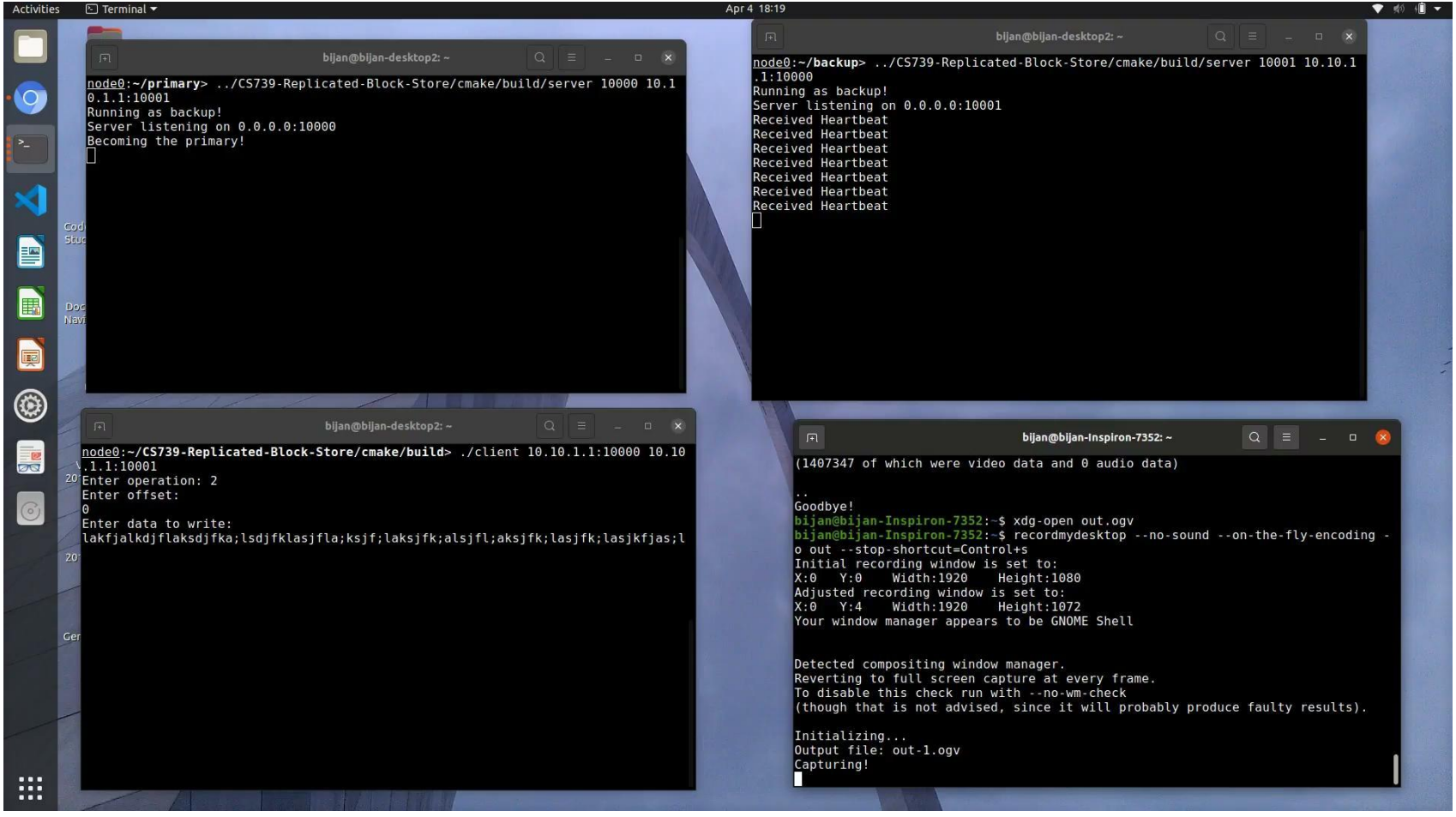
- The read and write latency for aligned and unaligned address is similar.
- There is not much difference because for unaligned reads or writes we are acquiring two locks instead of one.

| **Read** | |
|---|---|
| Aligned (µs) | Unaligned (µs) |
| 917 | 908 |

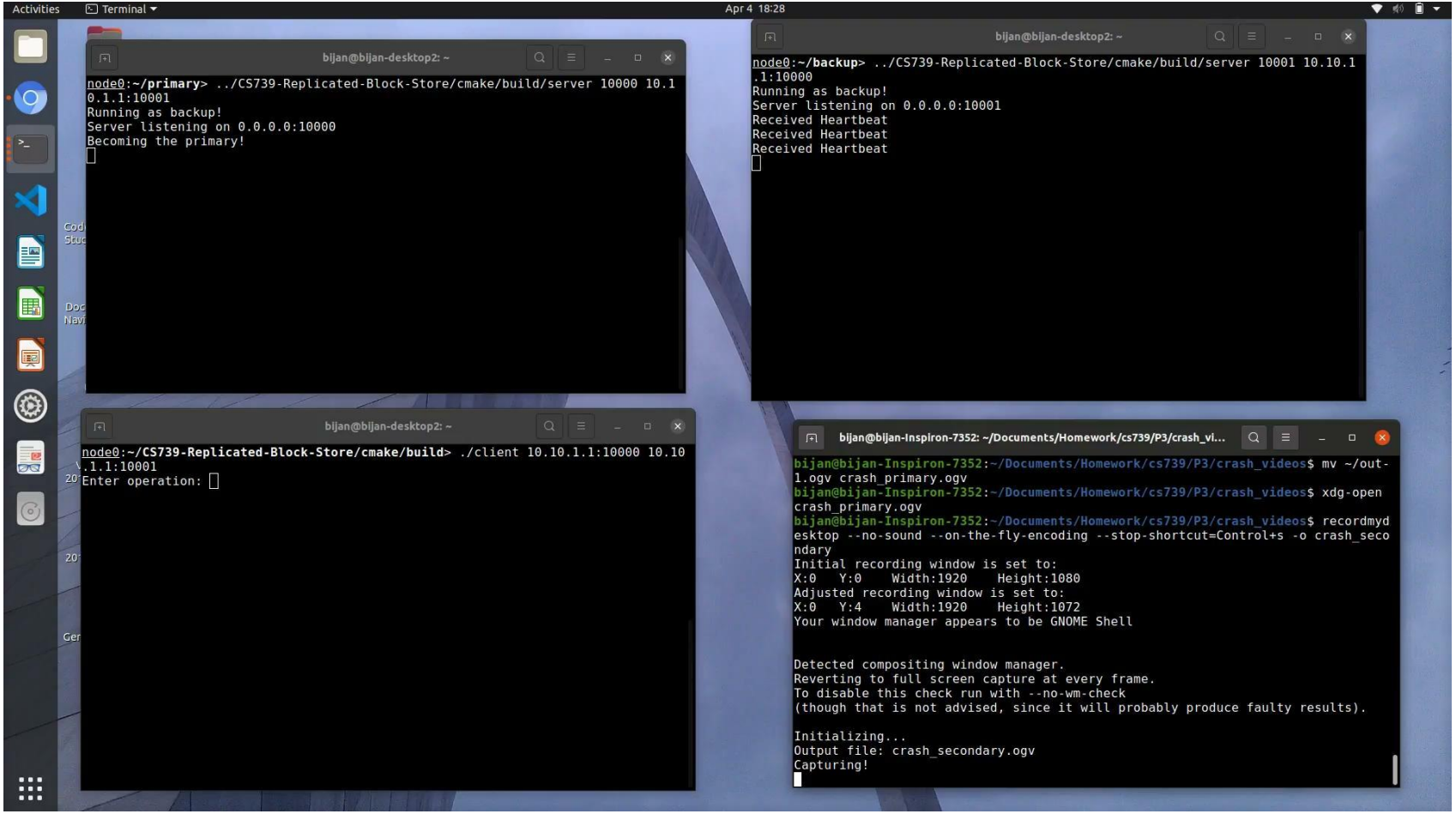
| **Write** | |
|---|---|
| Aligned (ms) | Unaligned (ms) |
| 62 | 71 |

# On to the fun part..
# Demo!

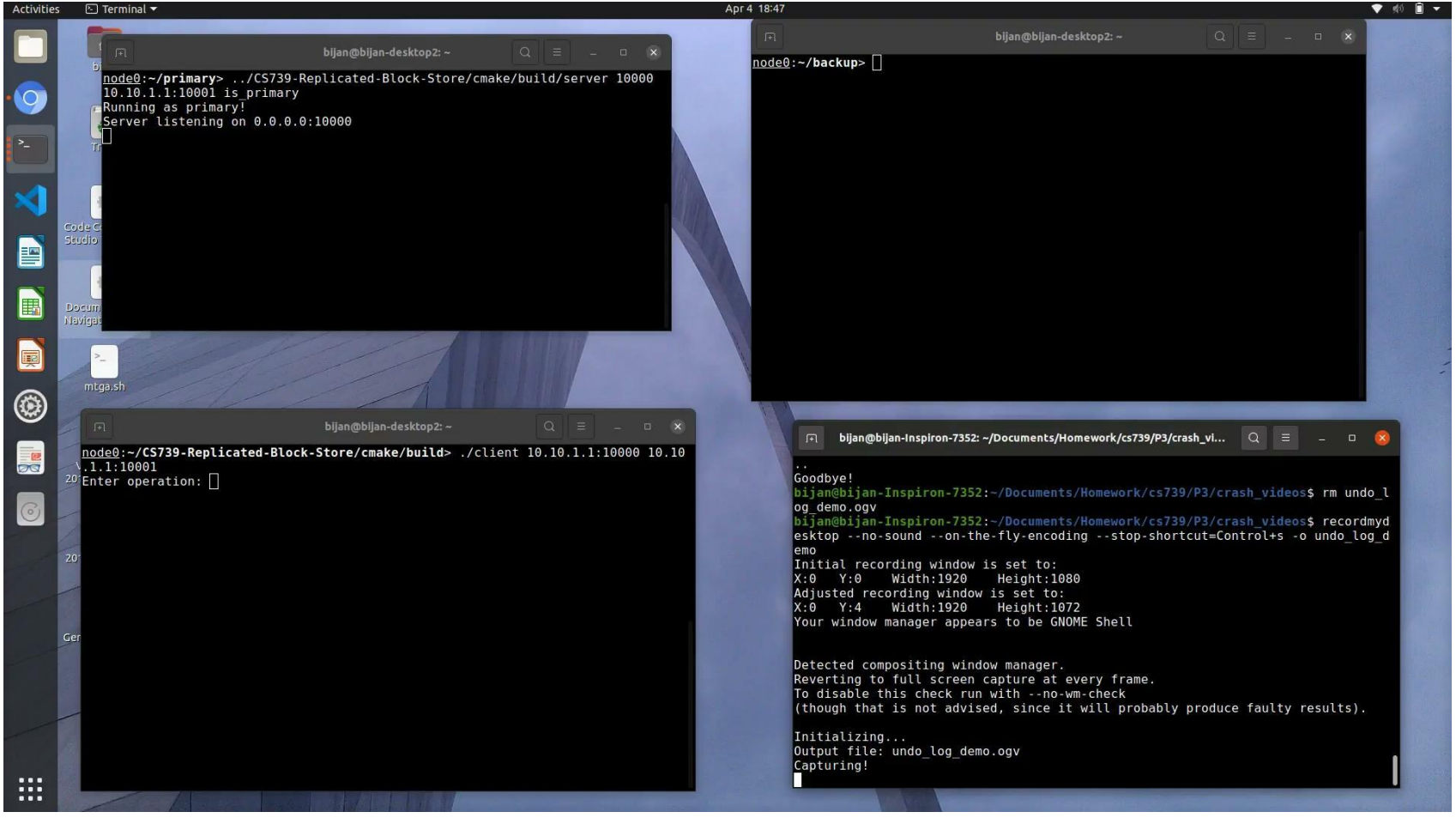
# Demonstration - Availability (Primary Crash)



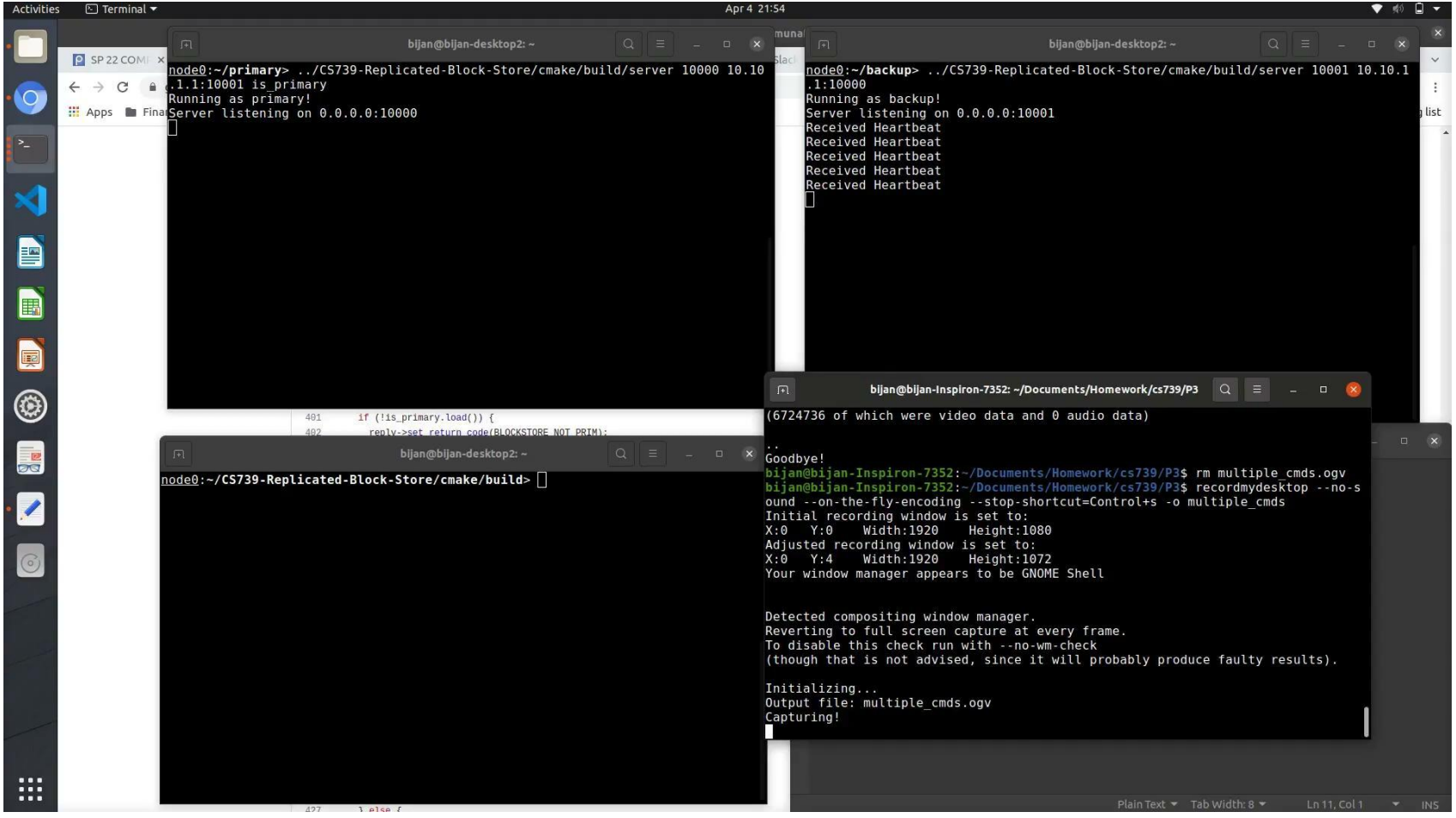
# Demonstration - Availability (Secondary Crash)



# Demonstration - Consistency (Crash mid write)



# Demonstration - Consistency (Concurrent writes)



# Thank You!

# Q/A?


WISCONSIN
UNIVERSITY OF WISCONSIN–MADISON