EC/CS 528: Cloud Computing

Distributed Systems for the Cloud

Instructor: Alan Liu

BU Department of Electrical & Computer Engineering

BOSTON UNIVERSITY

Announcements

- The first sprint demo video will be due on Monday.
  - Questions?

- Project check-ins: Each team sends me a date of next meeting with the mentor(s). I will try to coordinate.

- Lecture (Wed 10/5): Ata Turk (State Street Financial)

- Talk (Wed 10/12): Wei Bai (Microsoft Research)

BOSTON UNIVERSITY

Announcements

- Quick team update: project description, sprint plan, etc.

- Cloud resource allocations.

- OpenStack quick demo (later).

GFS and MapReduce

Motivation

- Huge amounts of data to store and process
- Example @2004:

  20+ billion web pages x 20KB/page = 400+ TB

  Reading from one disc 30-35 MB/s

  Four months just to read the web

  1000 hard drives just to store the web

  Even more complicated if we want to process data
- Exp. growth. The solution should be scalable

BOSTON UNIVERSITY

Motivation

- Buy super fast, ultra reliable hardware?
  - Ultra expensive
  - Controlled by third party
  - Internals can be hidden and proprietary
  - Hard to predict scalability
  - Fails less often, but still fails!
  - No suitable solution on the market

BOSTON UNIVERSITY

Motivation

- Use commodity hardware? Benefits:
  Commodity machines offer much better perf/$
  Full control and understanding of internals
  Can be highly optimized for their workloads

- Not that easy:
  Fault tolerance: something breaks all the time
  Applications development
  Debugging, Optimization, Locality
  Communication and coordination
  Status reporting, monitoring

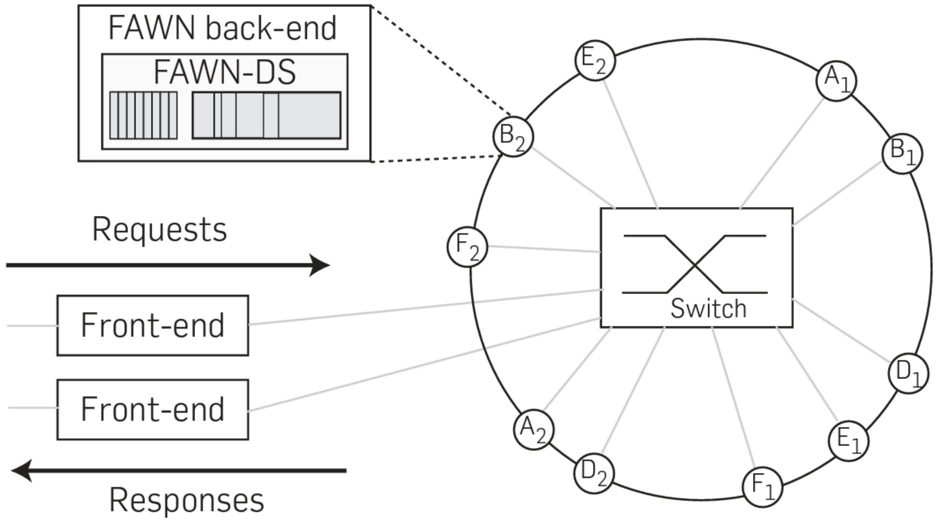
- Handle all these issues for every problem you want to solve

BOSTON UNIVERSITY

How to structure large distributed storage systems?

- An always-interesting research and engineering question. Why?
  - Other requirements: latency, energy consumption, cost.
  - Emerging applications

Figure 1. FAWN-KV architecture.



BOSTON UNIVERSITY

Typical first year for a new Google cluster (circa 2006)

~1 **network rewiring** (rolling ~5% of machines down over 2-day span)

~20 **rack failures** (40-80 machines instantly disappear, 1-6 hours to get back)

~5 **racks go wonky** (40-80 machines see 50% packetloss)

~8 **network maintenances** (4 might cause ~30-min random connectivity losses)

~12 **router reloads** (takes out DNS and external vips for a couple minutes)

~3 **router failures** (have to immediately pull traffic for an hour)

~**dozens of minor 30-second blips** for DNS

~1000 **individual machine failures**

~**thousands of hard drive failures**

slow disks, bad memory, misconfigured machines, flaky machines, etc.

Long distance links: **wild dogs, sharks, dead horses, drunken hunters, etc.**

Reliability Must Come From Software

BOSTON UNIVERSITY

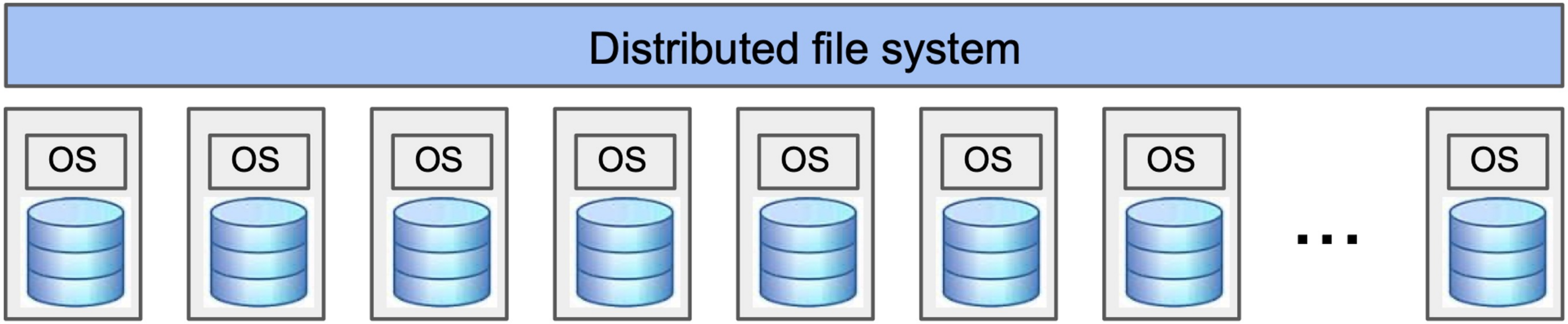
A Series of Steps,
All With Common Theme:

Provide Higher-Level View Than
"Large Collection of Individual Machines"

Self-manage and self-repair as much as possible

BOSTON UNIVERSITY

First Step:
Abstract Away Individual Disks



The Google File System

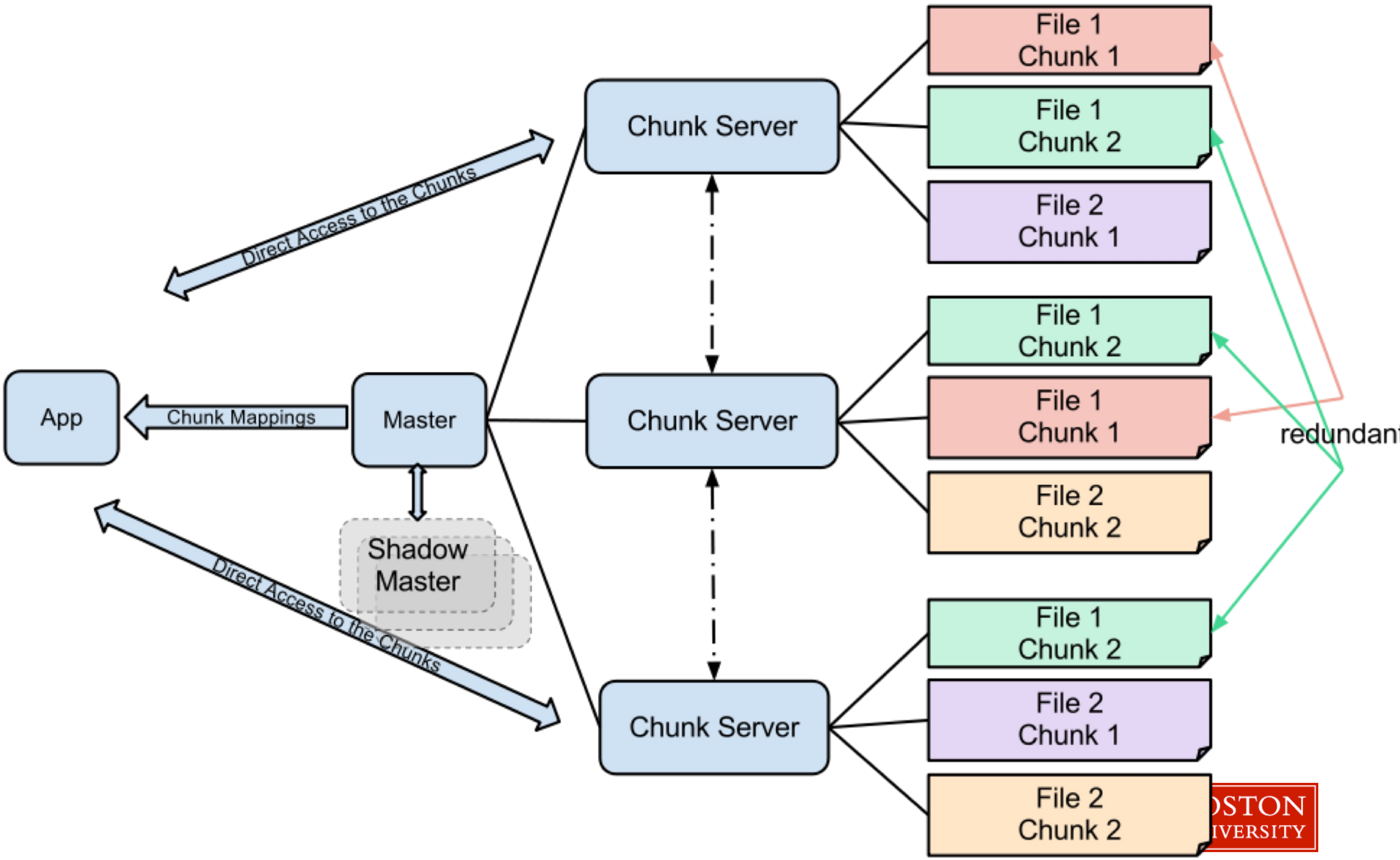
S. Ghemawat, H. Gobioff, S. Leung. SOSP, 2003

Assumptions:

- Inexpensive commodity hardware -> need for fault tolerance / recovery
- Workload (e.g., crawling -> indexing -> PR -> ...)
  - Multiple clients
  - Large streaming reads, Small random writes
  - Concurrent appends to the same file
- High Throughput > Low Latency

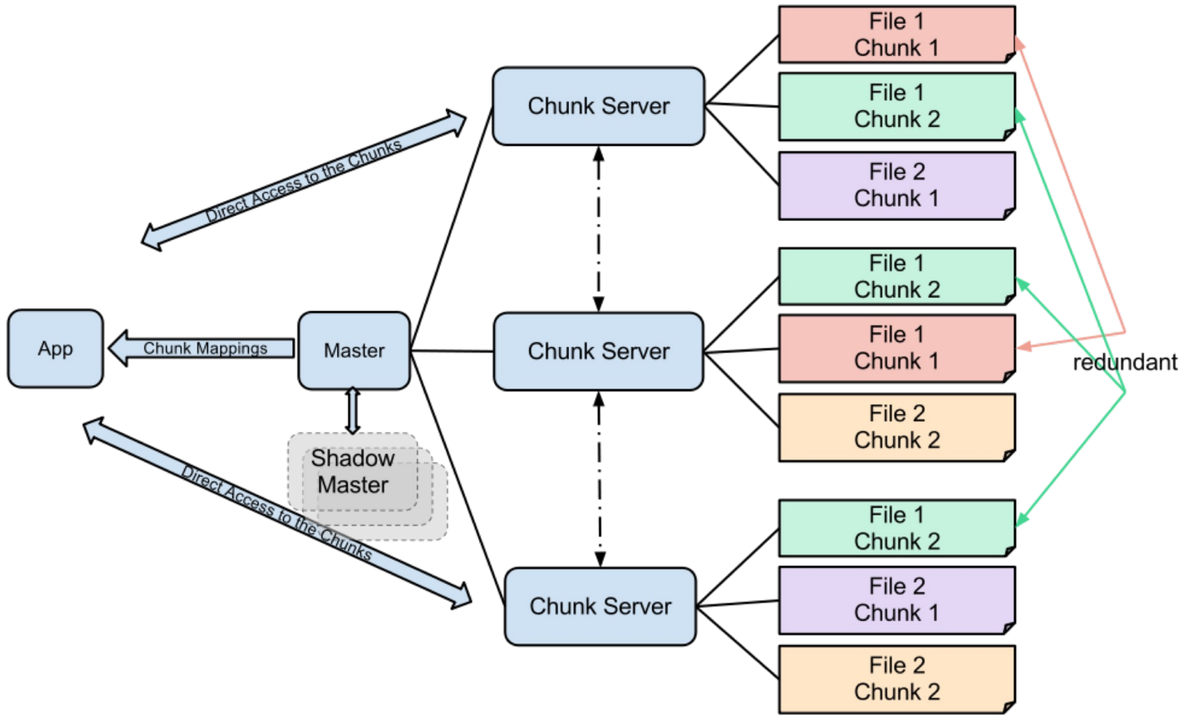
BOSTON UNIVERSITY

Architecture



Architecture

- User-level process running on commodity Linux machines
- Consists of Master Server and Chunk Servers
- Files broken into chunks (typically 64 MB),
- 3x redundancy
- Data transfers happen directly between clients and Chunk Servers



BOSTON UNIVERSITY

Master Node

- Centralization for simplicity & global knowledge for chunk placement/replication
- Namespace and metadata management
- Managing chunks
  - Where they are (file<-chunks, replicas)
  - Where to put new
  - When to re-replicate (failure, load-balancing)
  - When and what to delete (garbage collection)
- Fault tolerance
  - Shadow masters
  - Monitoring infrastructure outside of GFS
  - Periodic snapshots
  - Mirrored operations log

Master Node

- All Metadata is kept in Master's memory – it's fast!
  - A 64 MB chunk needs less than 64B metadata => for 640 TB less than 640MB
- Master learns ChunkServer-to-chunk mapping from Chunk Servers when
  - Master starts
  - A Chunk Server joins the cluster
- Master exchanges periodic heartbeat with Chunk Servers
  - state monitoring & instructions
- Operation log to keep file-to-chunk mapping persistent
  - Is used for serialization of concurrent operations
  - Replicated in master's disk and on remote machines
  - Respond to client only when log is flushed locally and remotely

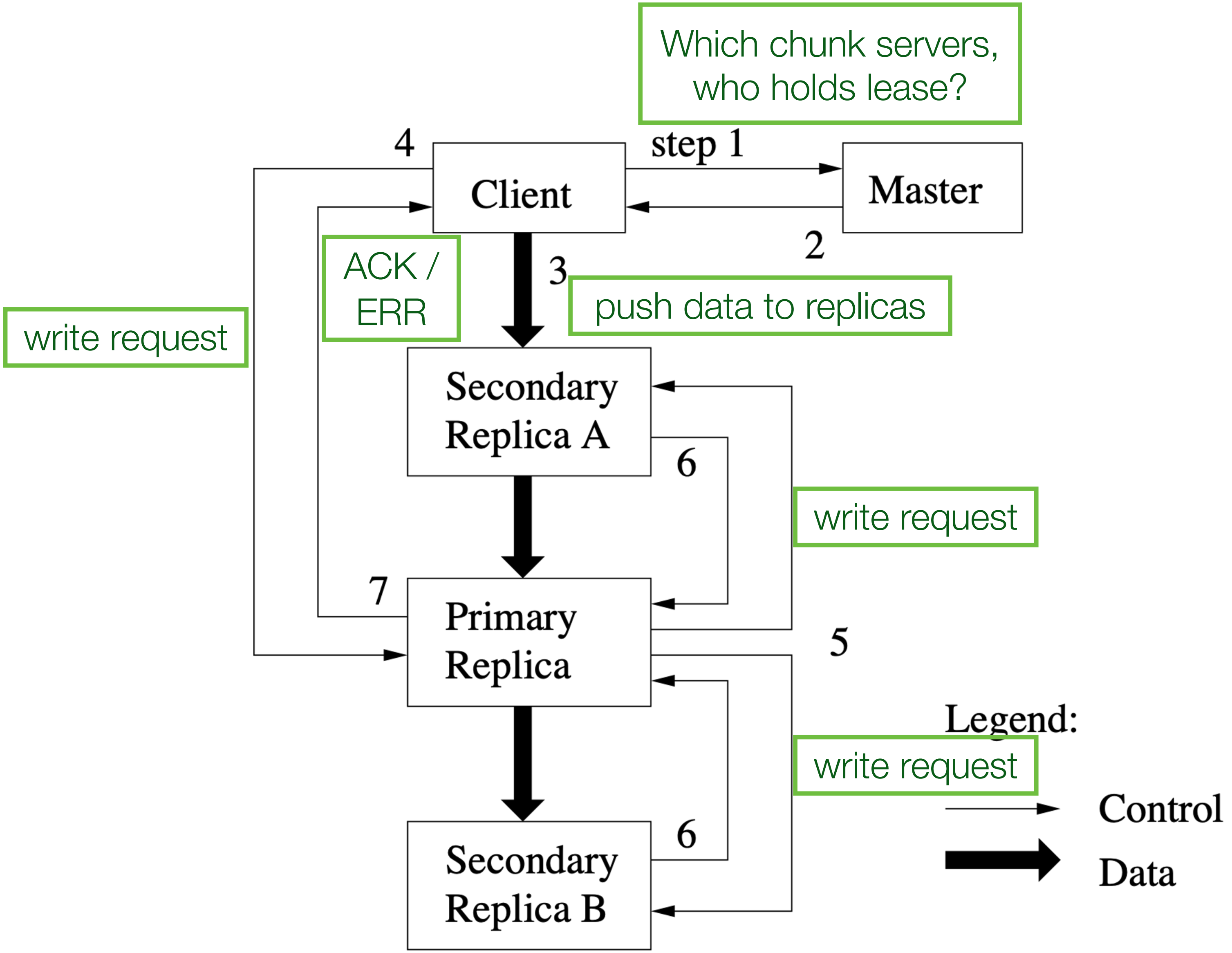
BOSTON UNIVERSITY

Chunk Servers

- 64MB chunks as Linux files
  - Reduce size of the master data structures
  - Reduce client-master interaction
  - Internal fragmentation => allocate space lazily

- Fault tolerance
  - Heart-beat to the master
  - Something wrong => master inits replication

Append Control and Data Flow



Control & Dataflow

- Decouple control flow and data flow
- Control flow
  - Master -> Primary -> Secondaries
- Data flow
  - Carefully picked chain of Chunk Servers
    - Forward to the closest first
    - Distance estimated based on IP

BOSTON UNIVERSITY

Some other important notes

- Smart chunk creation policy
  - Chunk Servers with below-average disk utilization, limited # of recent, distribute chunks across racks
- Smart re-replication policy
  - Under replicated first
  - Chunks that are blocking client
  - Live files first (rather than deleted)
- Rebalance and Garbage Collect periodically

Some other important notes

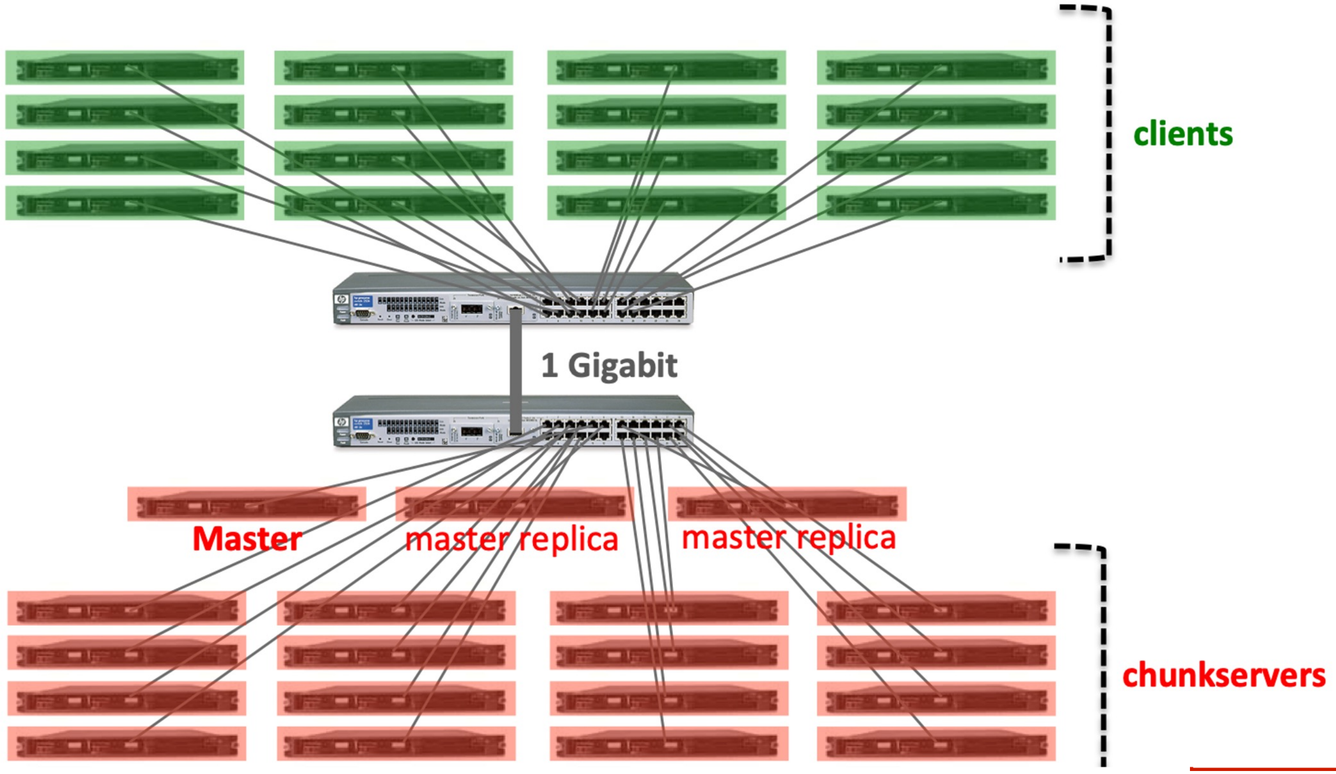
- Fault tolerance at master
  - master state is replicated
    - operation logs and checkpoints are replicated
    - a mutation is considered committed after it is written to log and log replicas
    - shadow-masters for read availability
- Data integrity handled at chunk servers
  - 32 bit checksums in memory for each 64K block
  - realize corruption during read
  - restore from other chunk servers

BOSTON UNIVERSITY

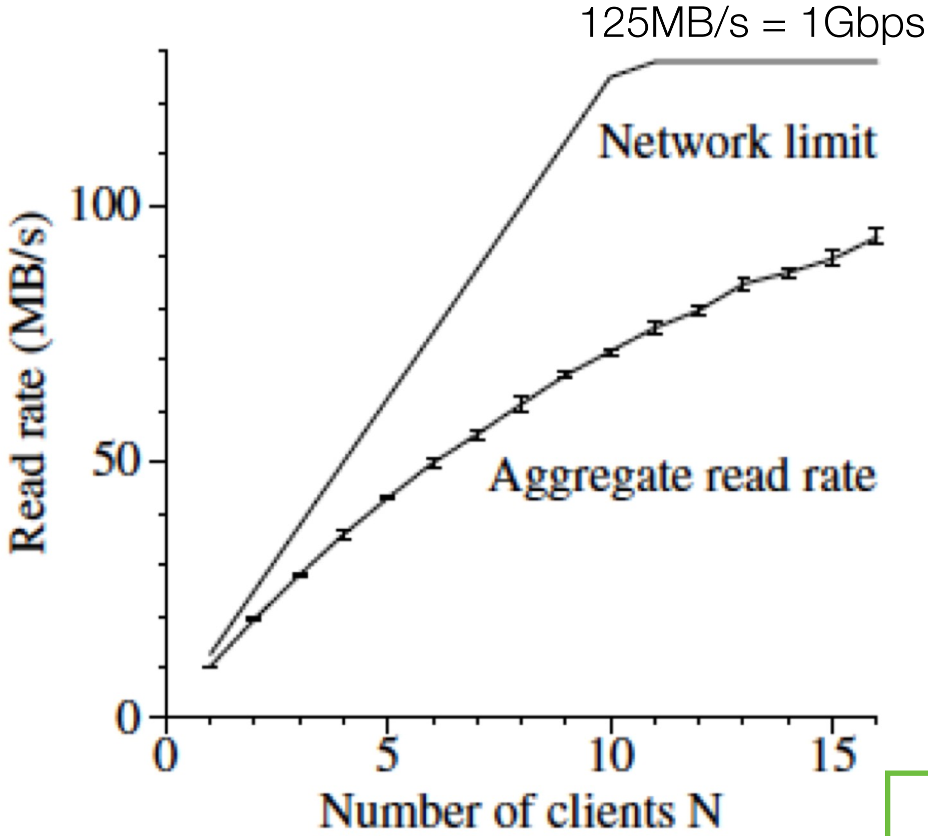
Measurements (2003)

Micro-benchmarks: GFS CLUSTER

- 1 **master**, 2 **master replicas**, 16 **chunkservers** with 16 **clients**
- **Dual 1.4 GHz PIII** processors, **2GB RAM**, **2x80GB** 5400 rpm disks, **FastEthernet NIC** connected to one HP 2524 Ethernet switch 24 ports 10/100 + **Gigabit uplink**



Read measurements



- each client read random 4MB region 256 times (1GB) from 320GB file

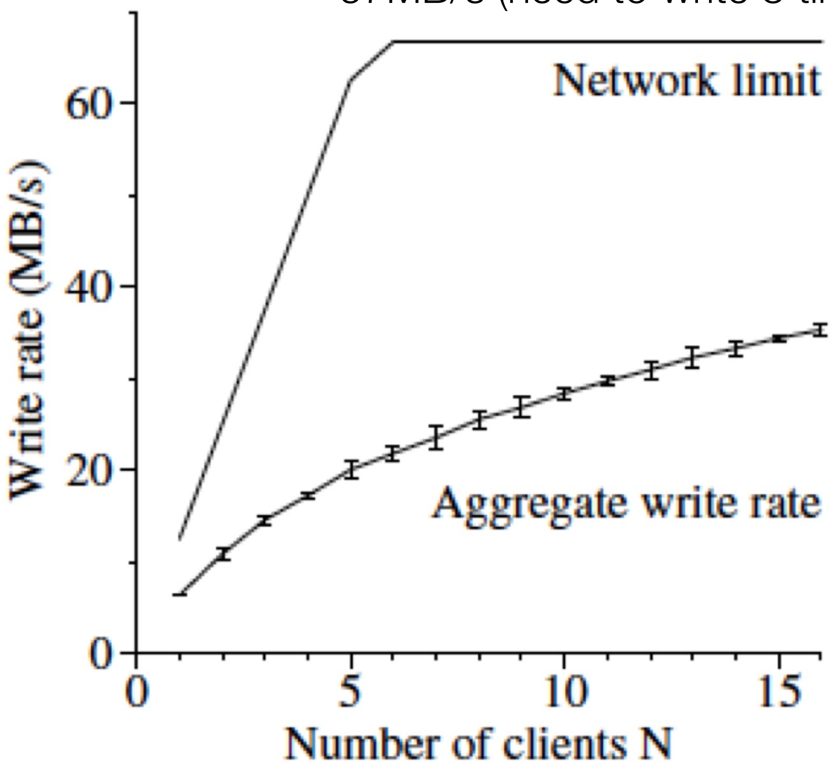
75-80% efficiency

94MB/s =>75% for 16 clients

BOSTON UNIVERSITY

Write measurements

67MB/s (need to write 3 times)



- each client writes 1GB data to a new file in a series of 1MB writes

50% efficiency

~35MB/s => 50% for 16 clients
- collision is more likely (3 replicas)

BOSTON UNIVERSITY

Measurements (2003)

- **Real world clusters: STORAGE & METADATA**
  - Cluster A used regularly for R&D by +100 engineers
  - Cluster B used for production data processing

	R&D	Production
Cluster	A	B
Chunkservers	342	227
Available disk space	72 TB	180 TB
Used disk space	55 TB	155 TB
Number of Files	735 k	737 k
Number of Dead files	22 k	232 k
Number of Chunks	992 k	1550 k
Metadata at chunkservers	13 GB	21 GB
Metadata at master	48 MB	60 MB

Cluster A store 55/3= **18 TB** of data
Cluster B store 155/3= **52 TB** of data

Chunkservers metadata = checksums for each 64 KB data block + chunk version number

BOSTON UNIVERSITY

Measurements (2003)

- **Real world clusters: READ/WRITE RATES & MASTER LOAD**
  - Cluster A used regularly for R&D by +100 engineers
  - Cluster B used for production data processing

	R&D	Production
Cluster	A	B
Read rate (last minute)	583 MB/s	380 MB/s
Read rate (last hour)	562 MB/s	384 MB/s
Read rate (since restart)	589 MB/s	49 MB/s
Write rate (last minute)	1 MB/s	101 MB/s
Write rate (last hour)	2 MB/s	117 MB/s
Write rate (since restart)	25 MB/s	13 MB/s
Master ops (last minute)	325 Ops/s	533 Ops/s
Master ops (last hour)	381 Ops/s	518 Ops/s
Master ops (since restart)	202 Ops/s	347 Ops/s

Cluster A network configuration can support read rate of 750 MB/s
Cluster B network configuration can support read rate of 1300 MB/s

BOSTON UNIVERSITY

Measurements (2003)

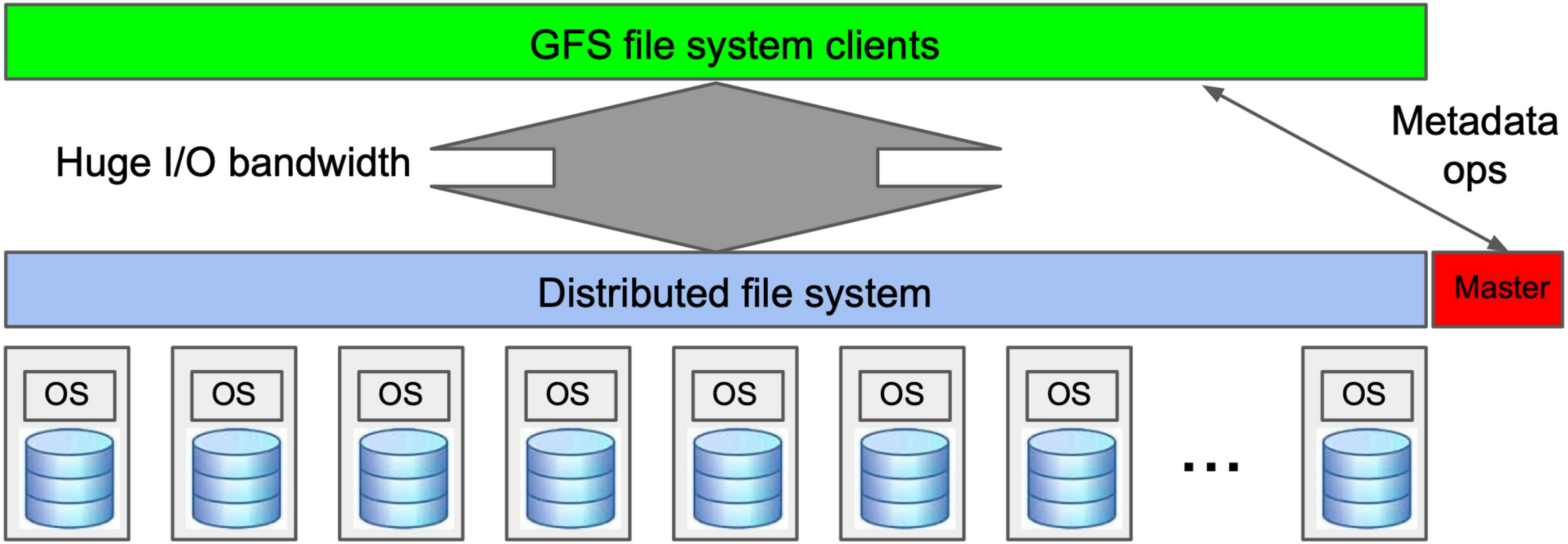
- **Real world clusters: RECOVERY TIME**
  - **Kill 1 chunkserver in cluster B** (Production)
    - 15.000 chunks on it (= 600 GB of data)
    - Cluster limited to 91 concurrent chunk cloning (= 40% of 227 chunkservers)
    - Each clone operation consume at most 50 Mbps
    - **15.000 chunks restored in 23.2 minutes effective replication rate of 440 MB/s**
  - **Kill 2 chunkservers in cluster B** (Production)
    - 16.000 chunks on each (= 660 GB of data)
    - This double failure reduced 266 chunks to having a single replica… ☹
    - **These 266 chunks cloned at higher priority and were all restored to a least 2xreplication within 2 minutes**

BOSTON UNIVERSITY

The Google File System
in one slide

Google File System (Ghemawat, Gobioff, & Leung, SOSP‘03)

- Centralized master manages metadata
- 1000s of clients read/write directly to/from 1000s of disk serving processes
- Files chunks of 64 MB, each replicated on 3 different servers
- High fault tolerance + automatic recovery, high availability



Q&A