系统设计
Distributed System Design
（九章网站下载最新课件）

课程版本 v5.0 本节主讲人：北丐老师

扫描二维码关注微信/微博
获取最新面试题及权威解答

微信: ninechapter
微博: http://www.weibo.com/ninechapter
知乎: http://zhuanlan.zhihu.com/jiuzhang
官网: http://www.jiuzhang.com

什么是分布式系统？

一言以概之：用多台机器去解决一台机器上不能够解决的问题。

比如：存储不够？QPS太大？

Overview 谷歌三剑客

- Distributed File System (Google File System)
  - 怎么有效存储数据？
  - No SQL 底层需要一个文件系统
- Map Reduce
  - 怎么快速处理数据？
- Bigtable = No-SQL DataBase
  - 怎么连接底层存储和上层数据

Overview of today

- Design Distributed File System (Google File System)

你会掌握

- Master Slave Pattern.
- How to check and handle system failure and error.
- How to design Distributed File System.

Design Distributed File System
了解分布式文件系统后可以做什么？

1. Google, Microsoft面试可能会考到.

2. 学习经典系统, 对其他系统设计也有帮助.

比如如何处理failure和recovery.

Distributed File System

Hadoop Distributed File System(HDFS 叫做“很多分手”)
VS
Google File System(GFS)
俗称“高富帅”又叫“刚分手”

Overview

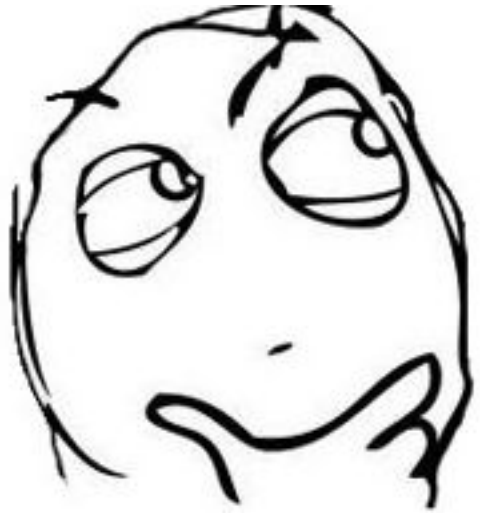
1. 按照4S分析
   - **S**cenario 场景分析
   - **S**ervice 服务
   - **S**torage 存储

2. 理清楚work solution

3. Scale 升级优化

Scenario 场景分析

需要设计哪些功能



Scenario 场景分析

- 需求1
  - 用户写入一个文件，用户读取一个文件.
  - 支持多大的文件？
    - 越大越好？ 比如 >1000T
- 需求2
  - 多台机器存储这些文件
  - 支持多少台机器？
    - 越多越好？10万台, Google 2007 year

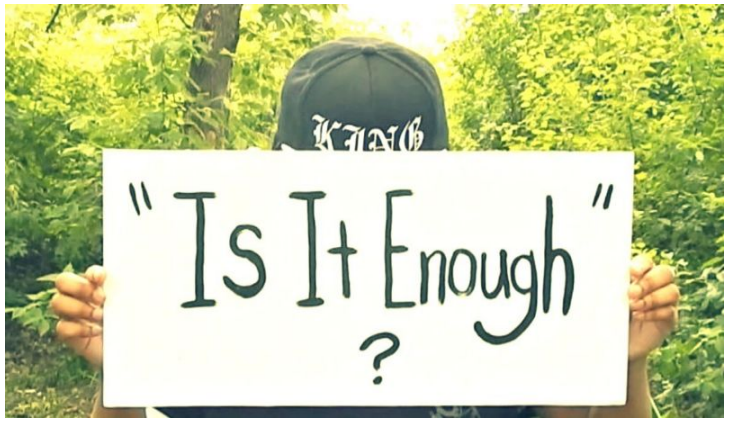
Service 服务

Service 服务

Client

+

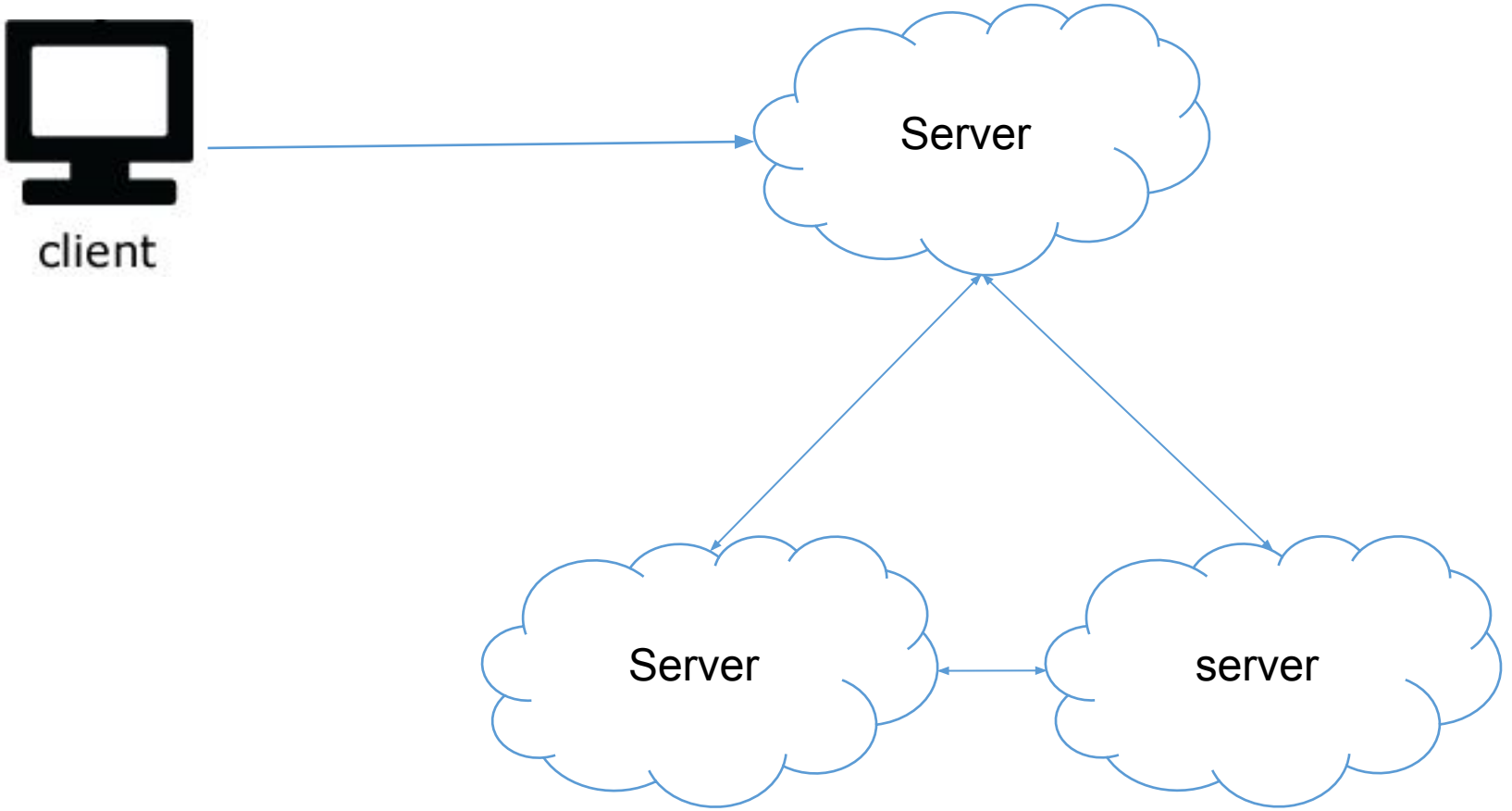
Server



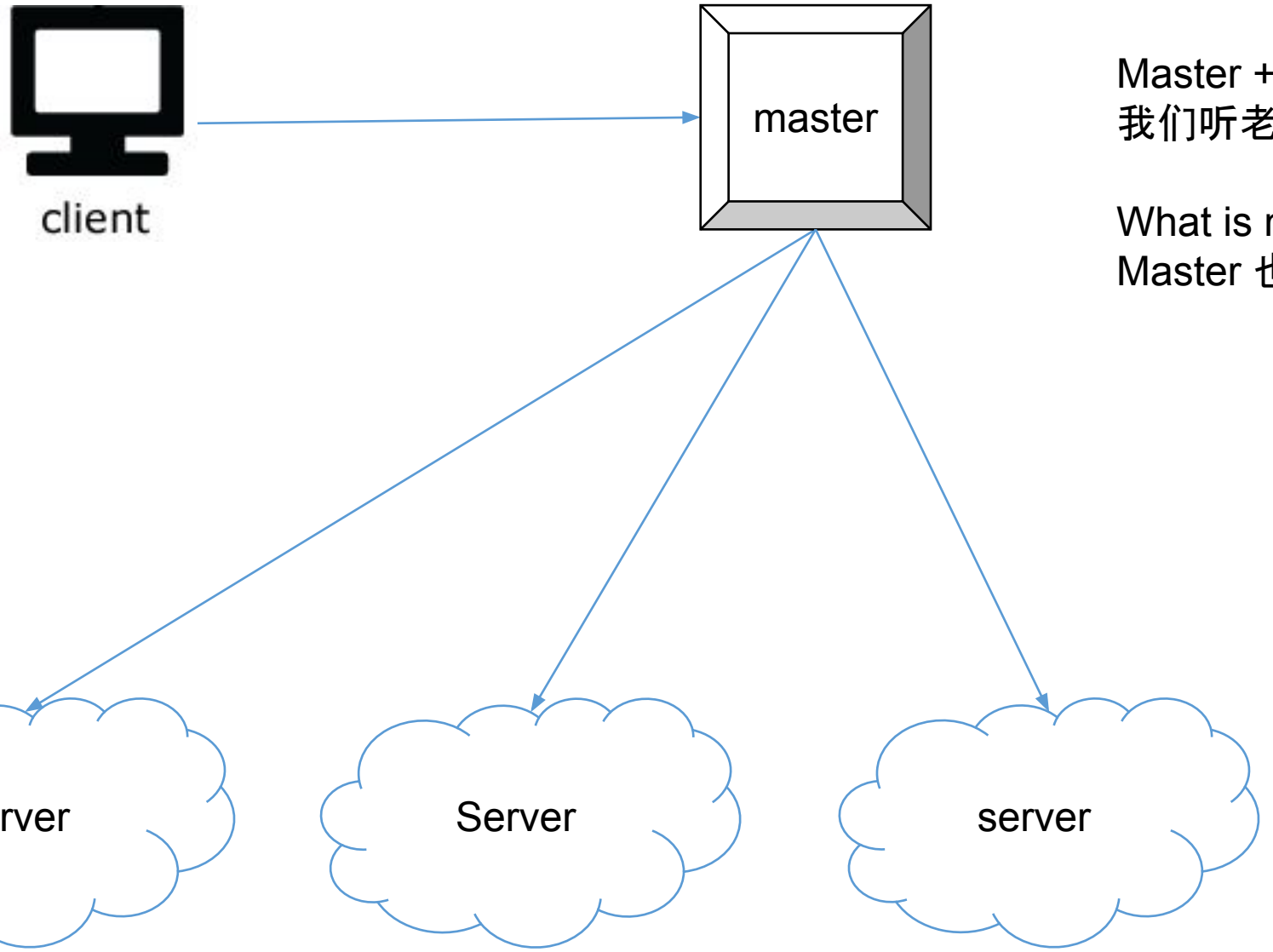
多台机器怎么沟通？

Service 服务

Peer to peer
谁也看不惯谁



Service 服务



Master + Slave
我们听老大的

What is master?
Master 也是一台机器

Storage 存储

- Peer 2 Peer (BitComet, Cassandra)
  - Advantage
    - 一台机器挂了还可以工作
  - Disadvantage
    - 多台机器需要经常通信保持他们数据一致

- Master Slave
  - Advantage
    - Simple Design
    - 数据很容易保持一致
  - Disadvantage
    - 单master要挂

- Final Decision
  - Master + Slave
  - 单master挂了重启就是。挂的概率在 0.1%

Storage 存储

Storage 存储

- 大文件存在哪？
  - 内存？数据库？文件系统？

Storage 存储

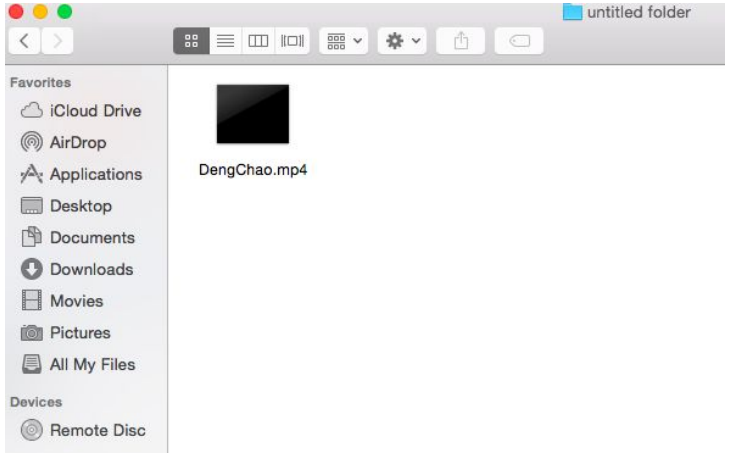
- 大文件存在哪？
  - 内存？数据库？文件系统？

- 怎么存在文件系统里面呢？
  - 操作系统基础知识怎么存文件的？

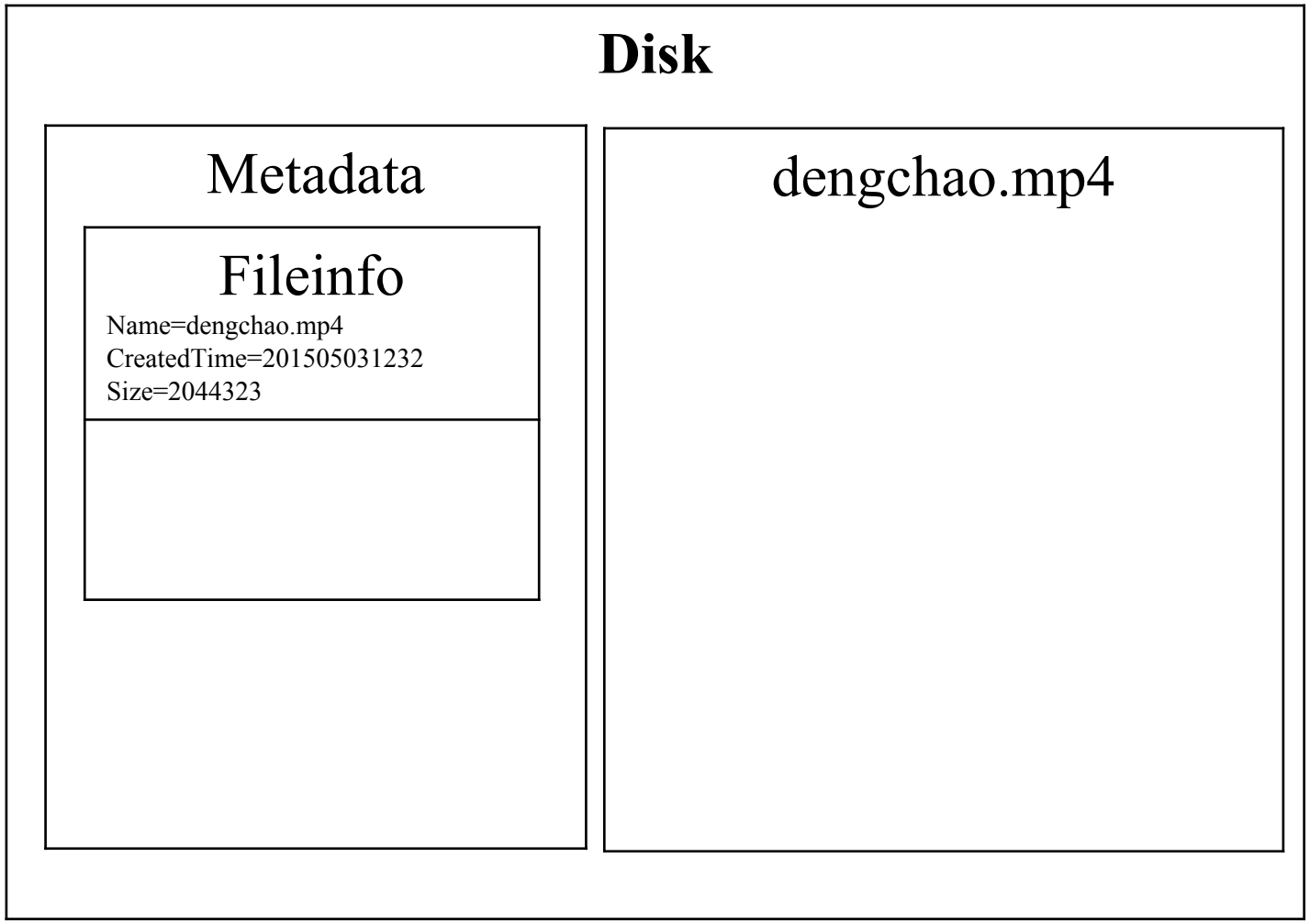
Interviewer: How to save a file in one machine？

普通的操作系统是怎么做的呢？100G

DengChao.mp4
一个文件有什么东西？



How to save a file in one machine



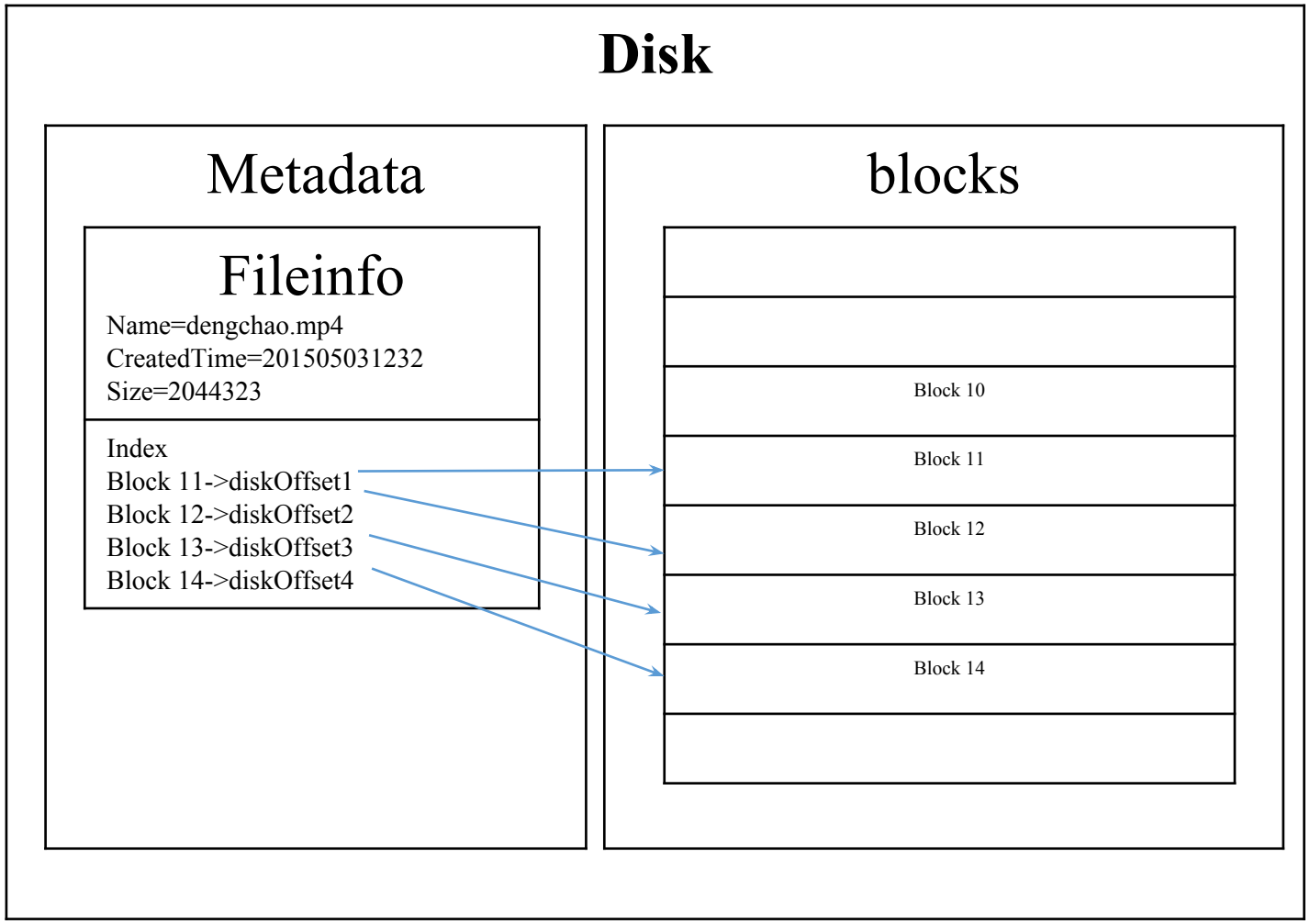
Metadata: 描述“其他数据”而存储的信息

Metadata 访问 常常多于 内容的访问

文件是分开存储的呢？还是连续存储的呢？

Windows就是连续存储，
Linux就是分开存储
（画个图）

How to save a file in one machine



Key point

- 1 block = 1024Byte

- Block Advantage?
  - 方便检查错误
  - 碎片化 (fragmentation)

Interviewer: How to save a large file in one machine？

Is block size big enough?

100T(多文件)

=100*1000G

=100*1000*1000M

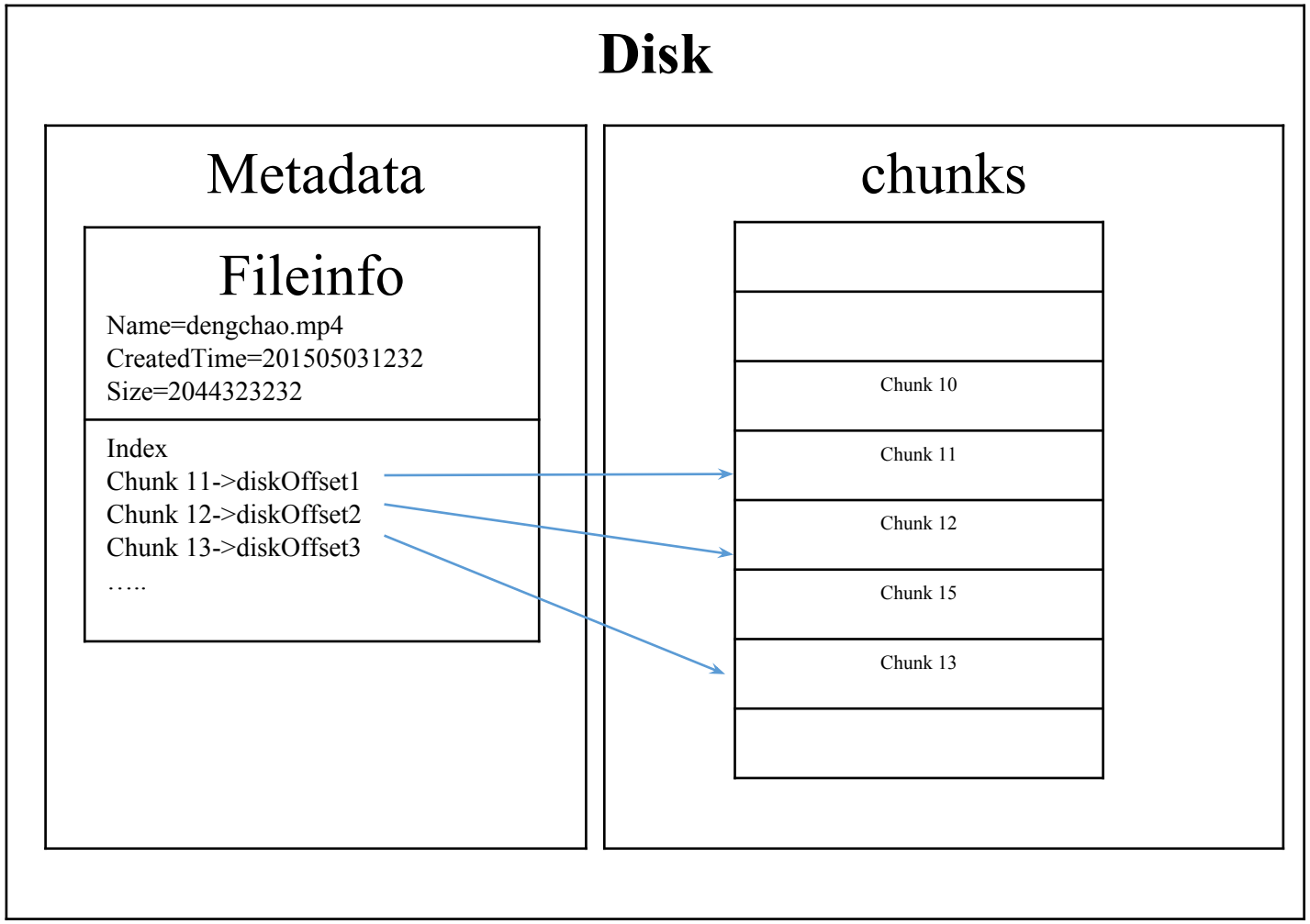
=100*1000*1000*1000K

=100*1000*1000*1000block

通常大文件是一些数据文件合成的
file-merged.txt = 0.3G

```
[pradeep@node35 data]$ hdfs dfs -ls /user/pradeep/demo/merge-example
Found 4 items
-rw-r--r--   3 pradeep supergroup   74363750 2016-02-09 22:16 /user/pradeep/demo/merge-example/1.txt
-rw-r--r--   3 pradeep supergroup   74363750 2016-02-09 22:16 /user/pradeep/demo/merge-example/2.txt
-rw-r--r--   3 pradeep supergroup   74363750 2016-02-09 22:16 /user/pradeep/demo/merge-example/3.txt
-rw-r--r--   3 pradeep supergroup   74363750 2016-02-09 22:16 /user/pradeep/demo/merge-example/4.txt
[pradeep@node35 data]$
[pradeep@node35 data]$ hdfs dfs -getmerge /user/pradeep/demo/merge-example file-merged.txt
[pradeep@node35 data]$
[pradeep@node35 data]$ ls -lart file-merged.txt
-rw-r--r--. 1 pradeep ldapusers 297455000 Feb  9 22:19 file-merged.txt
```

Interviewer: How to save a large file in one machine？



Key point

- 1 chunk= 64M
  = 64*1024K

Advantage

- Reduce size of metadata

Disadvantage

- Waste space for small files

系统设计没有标准答案
就是TradeOff的过程

Interviewer: How to save extra-large file in several machine？

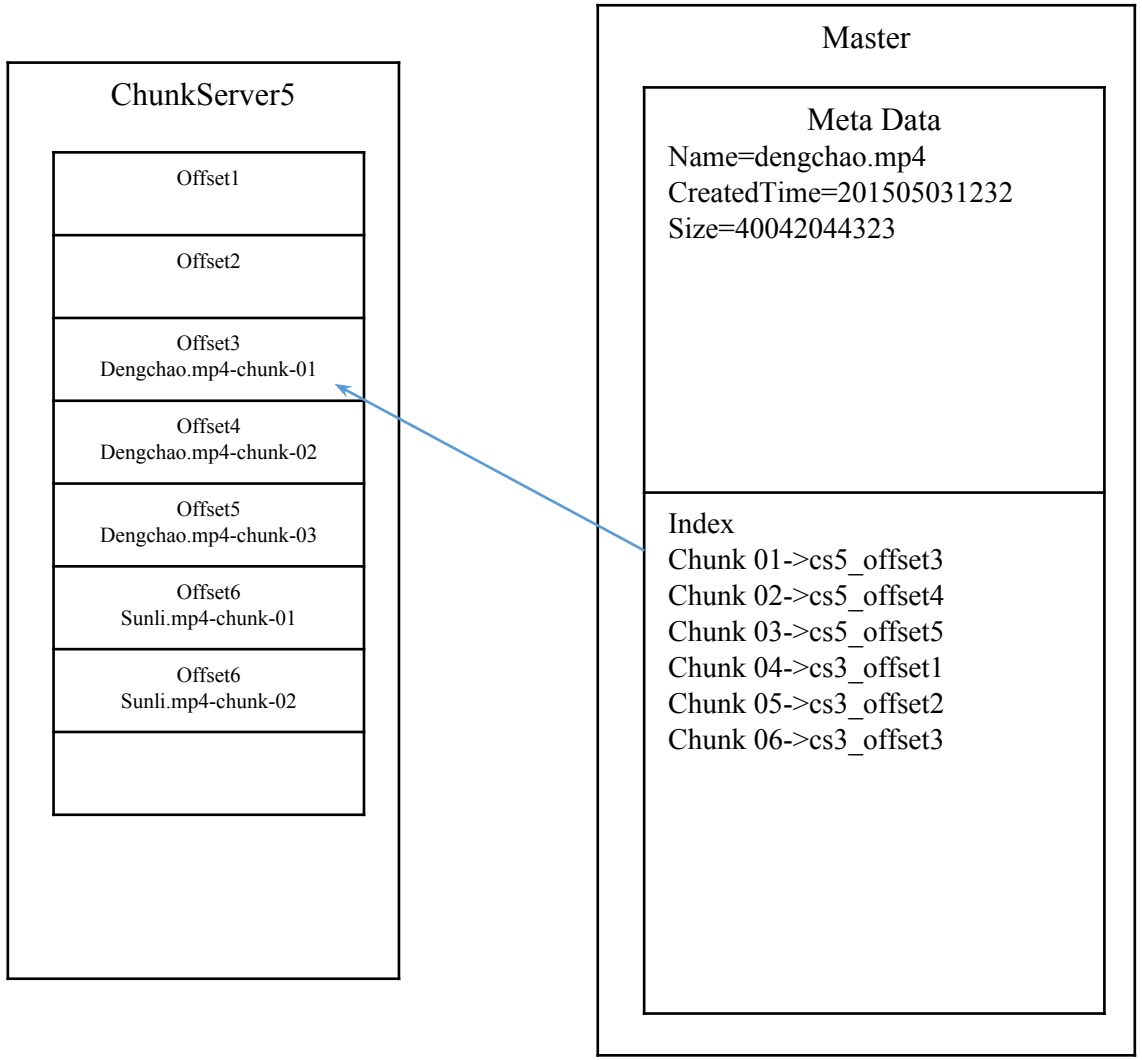
10P

Is one machine big enough?

这里的文件并不是指一个dengchao.mp4就那么大

而是很多个文件

Scale about the Storage

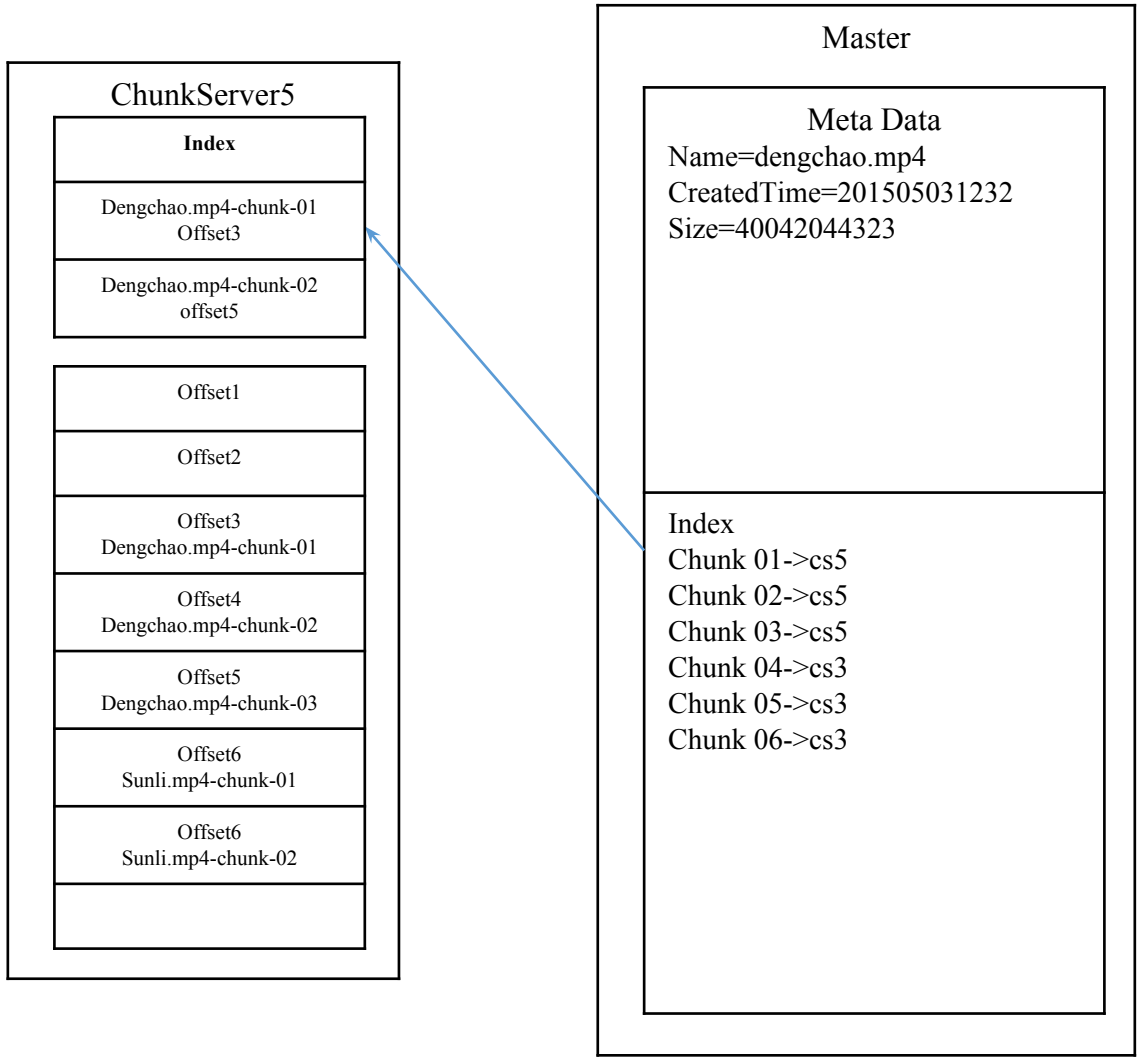


Key point

- One master + many ChunkServers

每个chunk的Offset偏移量可不可以不存在master上面？

Scale about the Storage



Key point

• The master don't record the diskOffset of a chunk

Advantage

• Reduce the size of metadata in master

• Reduce the traffic between master and ChunkServer（chunk offset改变不需要通知master）

Master 存储10P 文件的metadata 需要多少容量?

1 chunk = 64MB needs 64B. (经验值)

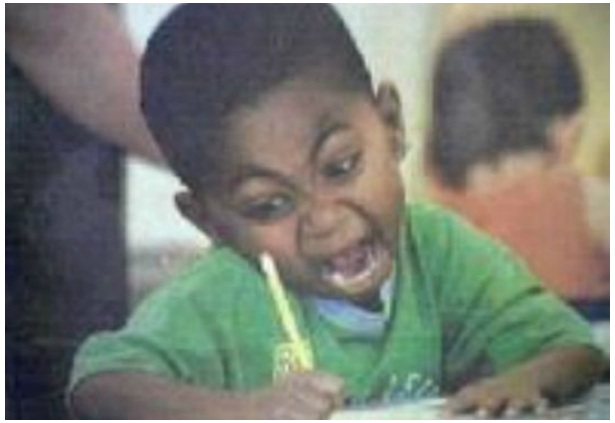
10P=16*10^6 chunk needs 10 G

Overview

- 按照4S分析
  - **S**cenario 场景分析
  - **S**ervice 服务
  - **S**torage 存储
- 理清楚work solution
- Scale 升级优化

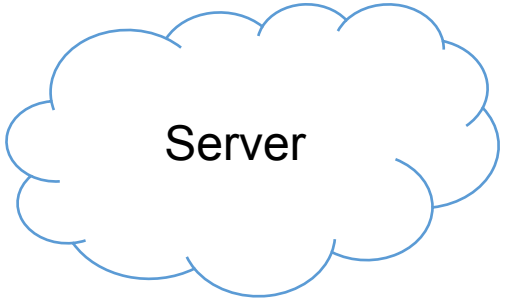
One Work Solution for Read / Write

Interviewer: How to write a file?



一次写入
还是拆分成多份多次写入？

write File_name=/gfs/home/dengchao.mp4



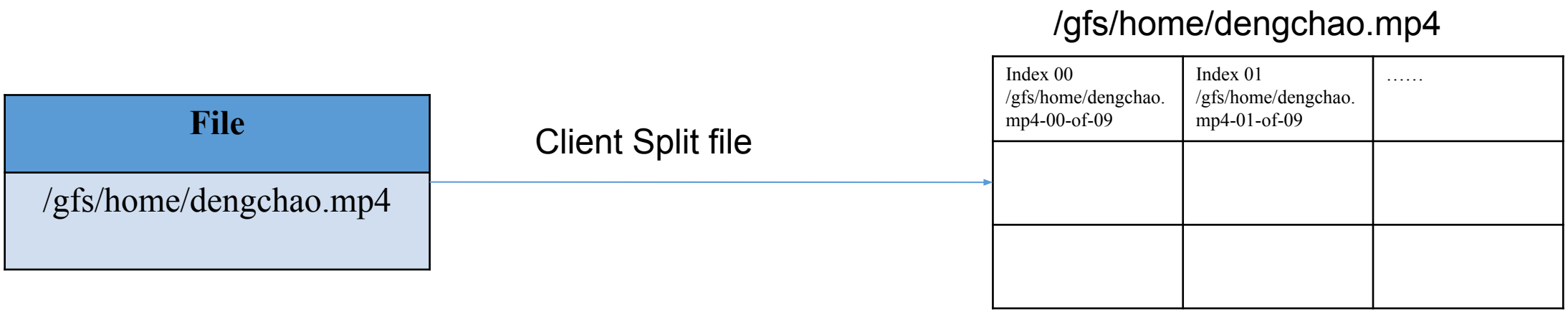
把大胖子直接写入呢？

还是把大胖子碎尸万段了后写入呢？

一次 VS 多次

- 写入过程中出错了，那么需要重新写入，哪一种方法更好？
  - 一次传输得重新传输整个文件，多次只用重新传一小份。
- 如果是分成多份多次写入，那么每一份的大小？
  - 文件本来是按照Chunk来存储的，所以传输单位也是Chunk
- 如果是分成多份多次写入，那么是告诉master让master来切分？还是client自己切分？
  - client自己按照文件大小切分。
  - 比如 /gfs/home/dengchao.mp4 size = 576M. 那么可以切分问 576M/64M = 9个chunk。

How to write a file?



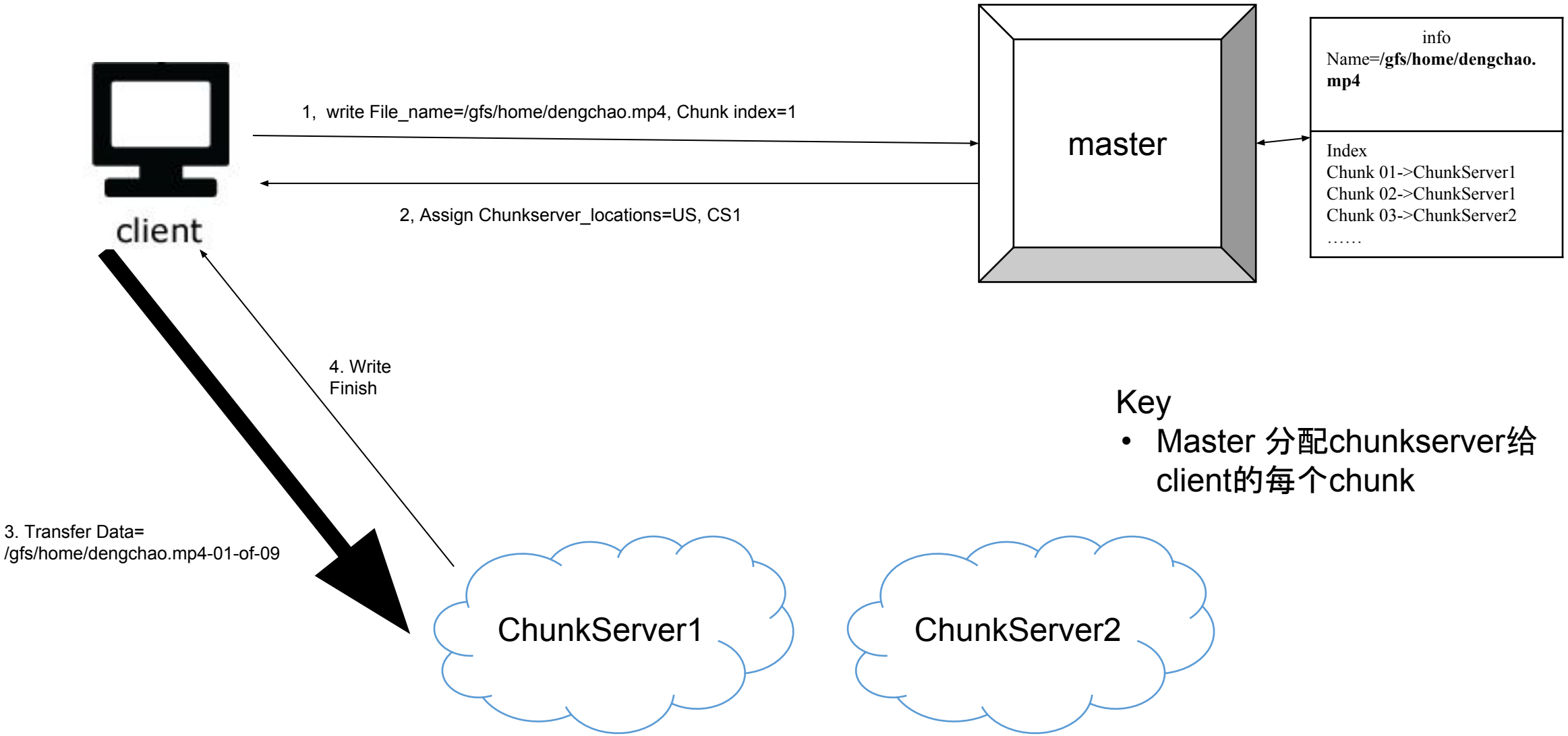
- client 把文件拆分为n份，每一份一个chunk index
- 所以并不是一下就把胖子写进去

那每一个chunk是怎么写入server的呢？

直接写到chunk server？

需要先个master沟通，再写入chunk server？

How to write a file?



Key

- Master 分配chunkserver给client的每个chunk

这个地方的client是User么？

How to write a file?

Client	Server
User	Browser
Broswer	Webserver
Webserver	Database
Database	GFS
Webserver	GFS(Google File System)

要修改Dengchao.mp4怎么办？

/gfs/home/dengchao.mp4

要修改的部分在哪个chunk？

修改了过后chunk变大了要怎么处理？

修改了过后chunk变小了要怎么处理？

要修改Dengchao.mp4怎么办？

One time to write, Many time to read.

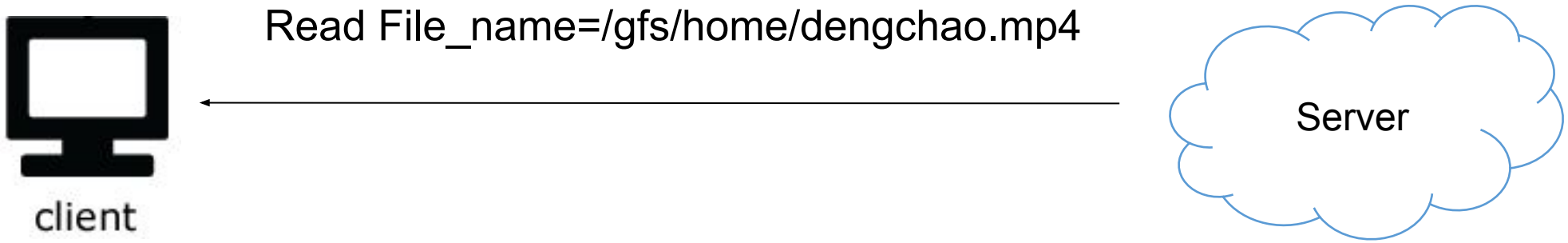
先删掉/gfs/home/dengchao.mp4

重新把整个文件重写一份

跟化妆一样，画丑了，洗掉重画

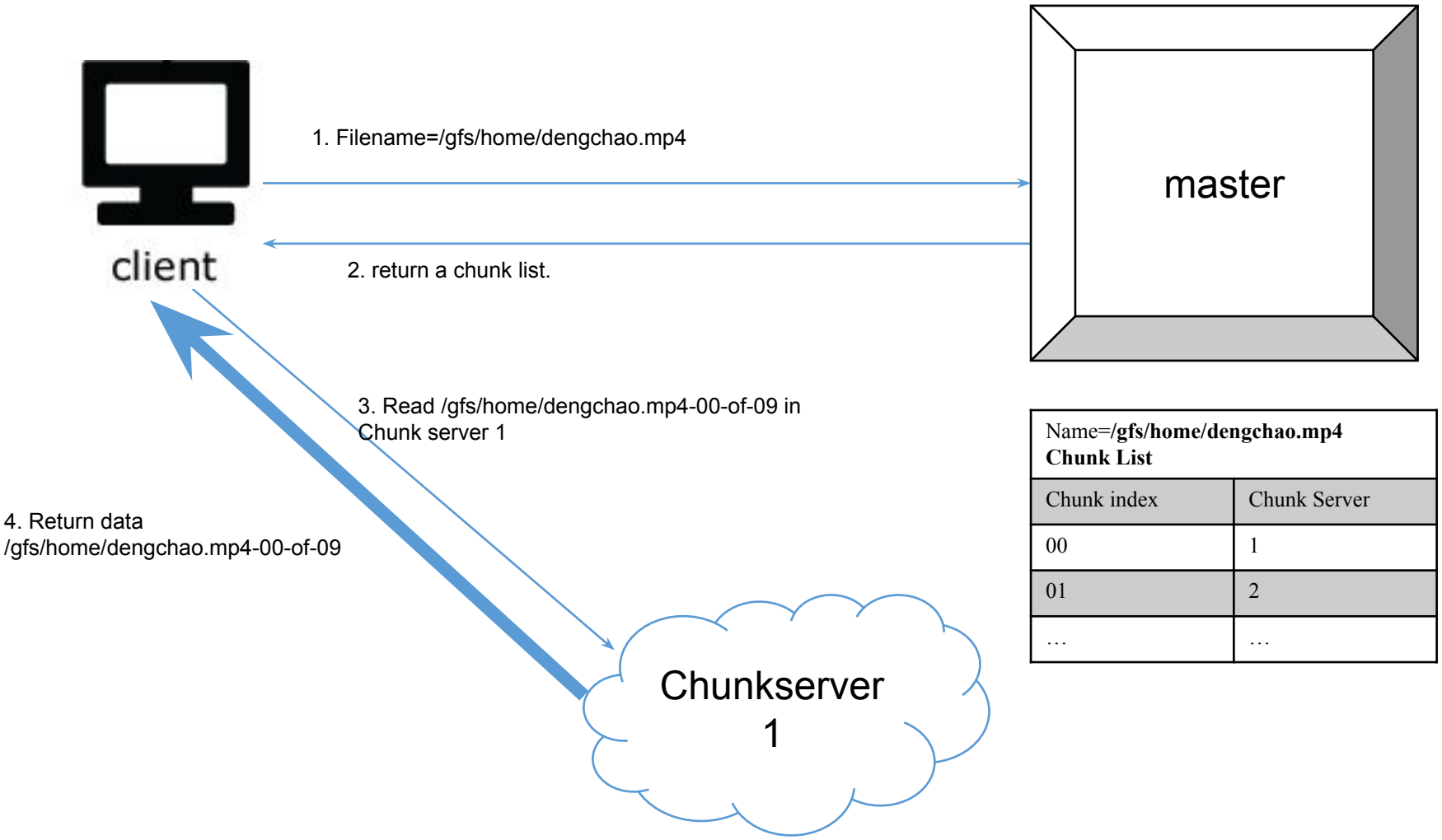
Interviewer: How to read a file?

一次读整个文件？
还是拆分成多份多次读入？



那么client怎么知道dengchao.mp4被切成了多少块？

How to read from a file?



Name=**/gfs/home/dengchao.mp4**
Chunk List

Chunk index	Chunk Server
00	1
01	2
…	…

Master Task

- 存储各个文件数据的metadata
- 存储Map(file name + chunk index -> chunk server)
  - 读取时找到对应的chunkserver
  - 写入时分配空闲的chunkserver

- Question?
  - 为什么不把数据直接给master 让master 去写？
    - Master bottleneck

One Work Solution

- 存储
  - 普通文件系统 Meta Data，Block
  - 大文件存储：Block-> Chunk
  - 多台机器超大文件: Chunk Server + Master
- 写入
  - Master+Client+ChunkServer 沟通流程
  - Master 维护metadata 和 chunkserver 表
- 读出
  - Master+Client+ChunkServer 沟通流程

Overview

- 按照4S分析
  - **S**cenario 场景分析
  - **S**ervice 服务
  - **S**torage 存储
- 理清楚work solution
- Scale 升级优化

休息5分钟

Scale 升级

系统如何优化与维护

GFS的精髓

单Master 够不够？

单Master 够不够？

工业界90%的系统都采用单master

Simple is perfect

Single Master Failure

Double Master 双龙戏珠

Paper: Apache Hadoop Goes Realtime at Facebook

Multi Master

Paper: Paxos Algorithm

Scale about the Failure and Recover

Interviewer: How to identify whether a chunk on the disk is broken?

CheckSum

Check Sum 检查一位错误

原来

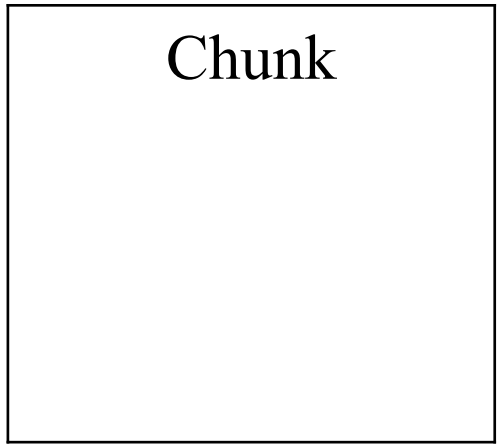
数据	1	2	3	Checksum(xor)
二进制表示	01	10	11	00

错误后

数据	1	3	3	Checksum(xor)
二进制表示	01	11	11	01

- CheckSum Method (MD5, SHA1, SHA256 and SHA512)
- Read More: https://en.wikipedia.org/wiki/Checksum

Add checksum for chunk?



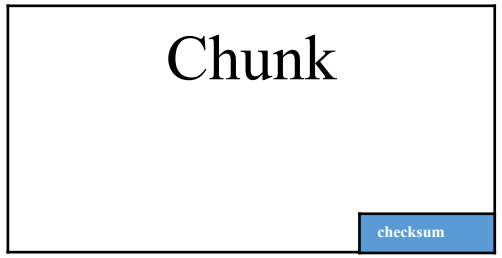
How to identify whether a chunk on the disk is broken?

- 1 checksum size?
- 4bytes = 32bit
- 1 chunk = 64MB
- Each chunk has a checksum
- The size of checksum of 1T file
- 1P/64MB*32bit = 62.5 MB

什么时候写入checksum?

什么时候写入checksum?

Answer: 写入一块chunk的时候顺便写入



什么时候检查checksum?

什么时候检查checksum?

Answer: 读入这一块数据的时候检查

1. 重新读数据并且计算现在的checksum
2. 比较现在的checksum和之前存的checksum是否一样

Interviewer: How to avoid chunk data loss when a ChunkServer is down/fail?

Interviewer: How to avoid data loss when a ChunkServer is down/fail?

Answer: Replica（专业词汇）

说得好听叫做双保险

需要多少个备份？
每个备份放在哪？

需要多少个备份？
每个备份放在哪？

1. 三个备份都放在一个地方(加州)。
2. 三个备份放在三个相隔较远的地方(加州，滨州，纽约州)
3. 两个备份相对比较近，另一个放在较远的地方(2个加州，1个滨州)

选chunk server的时候有什么策略？

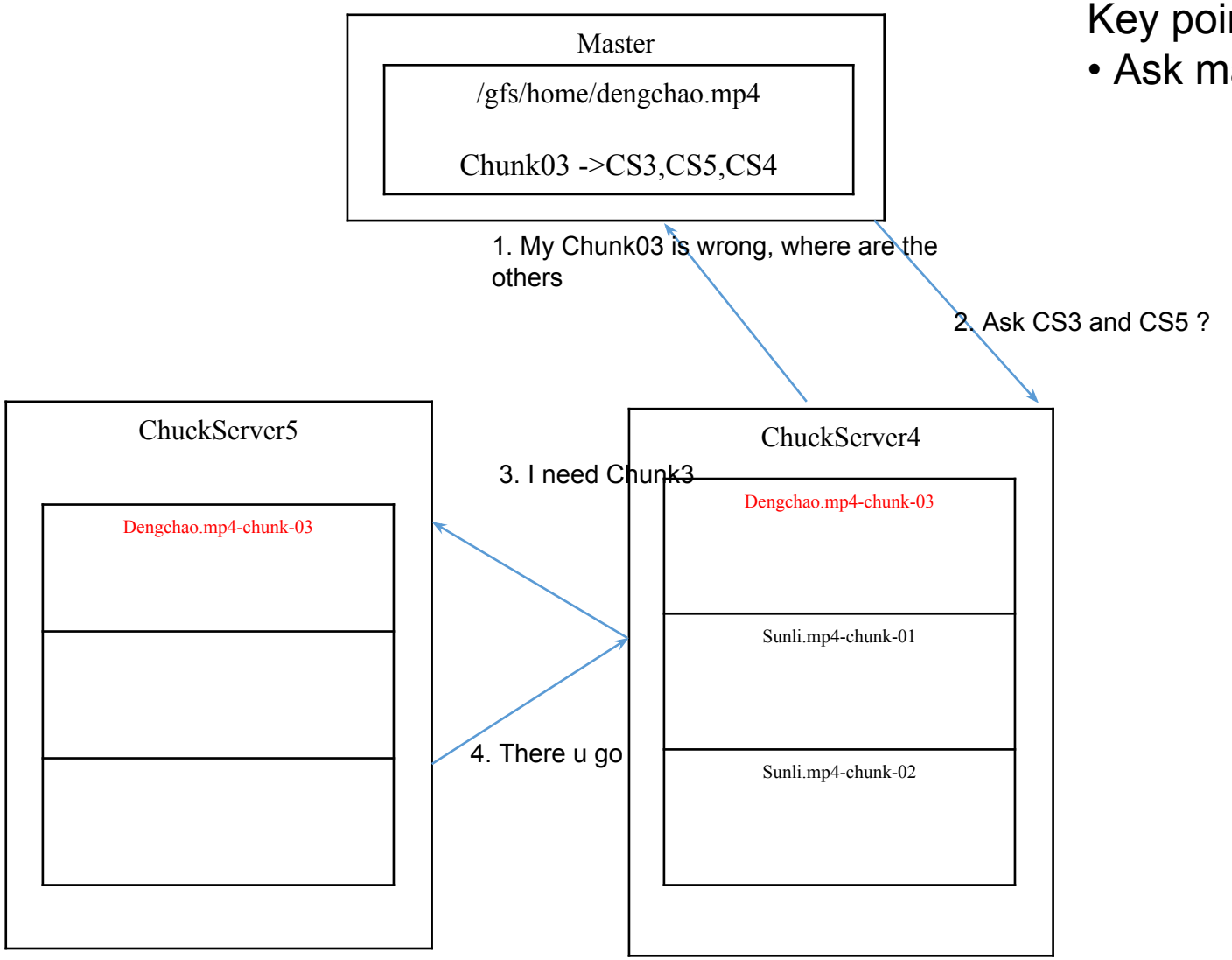
选chunk server的时候有什么策略？

1. 最近写入比较少的。(LRU)
2. 硬盘存储比较低的。

Interviewer: How to recover when a chunk is broken?

Interviewer: How to recover when a chunk is broken?

Answer: Ask master for help



Key point

- Ask master for help

How to find whether a ChunkServer is down?

How to find whether a ChunkServer is down?

Interviewer: HeartBeat.

画一个图阐释

A: master -> chunkservers?

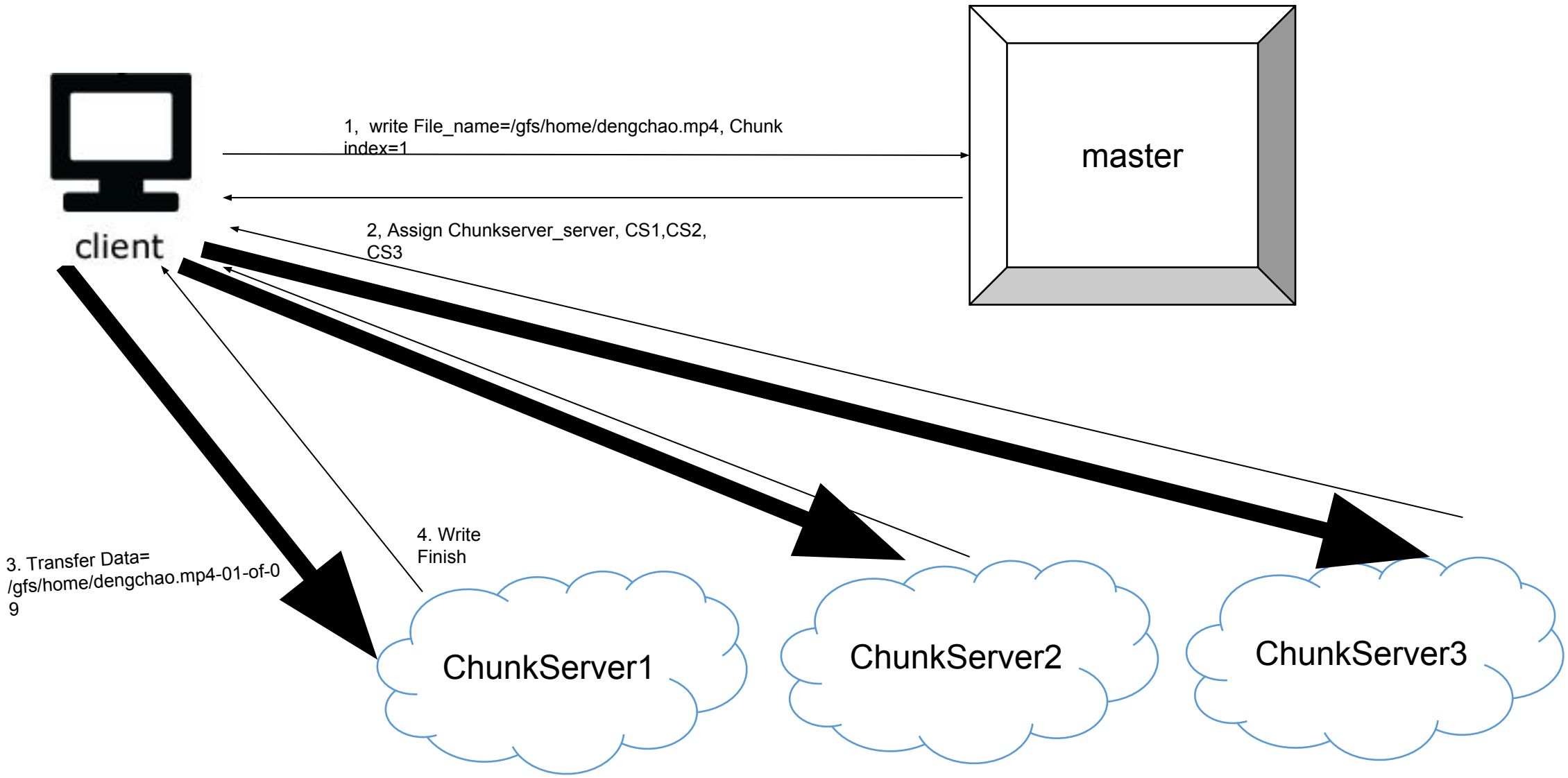
B: chunkservers->master?

备胎?

Scale about the Write

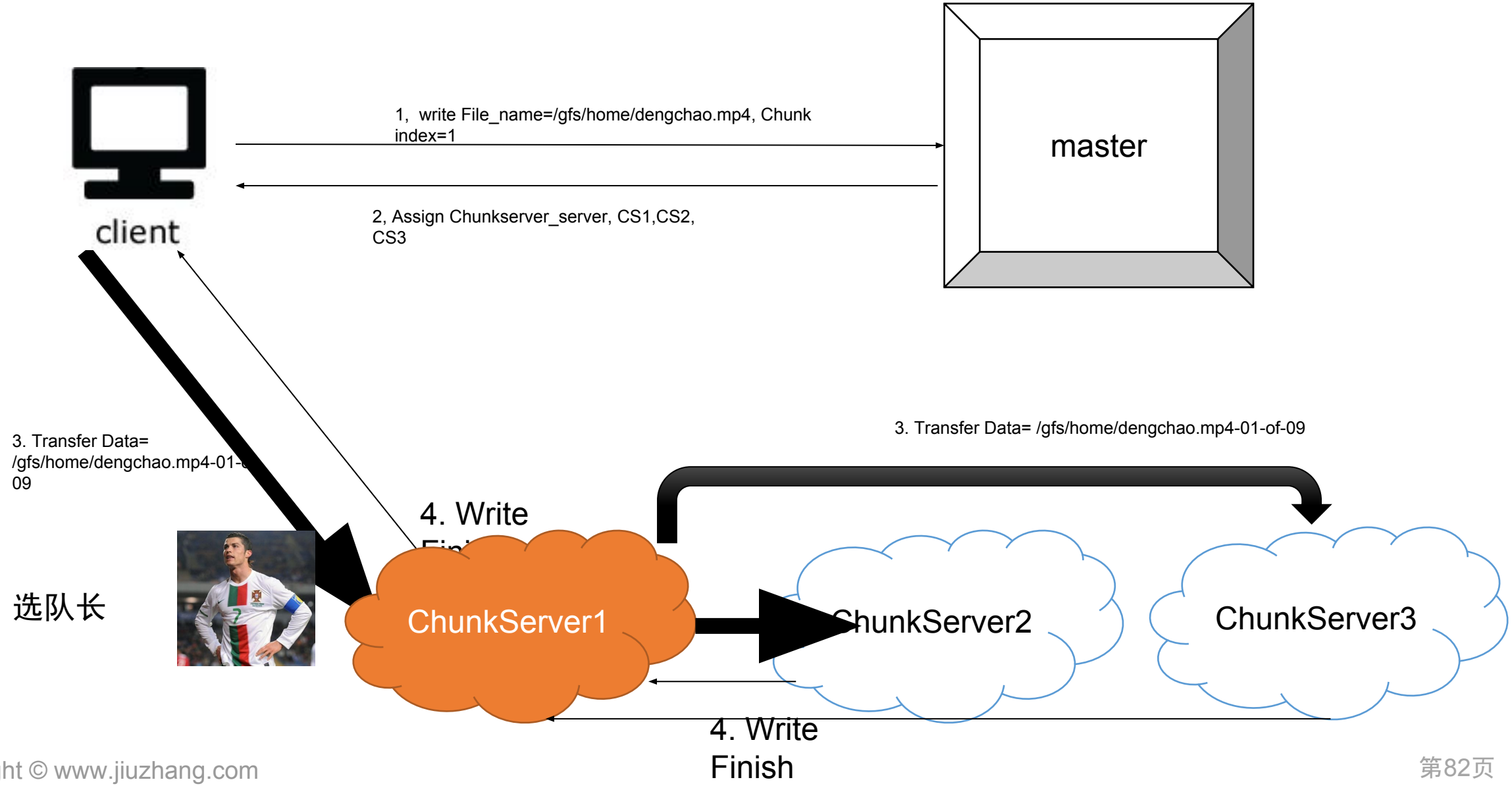
Interviewer: Whether write to only one server is safe?

How to write a file?



Interviewer: How to solve Client bottleneck?

How to solve Client bottleneck?

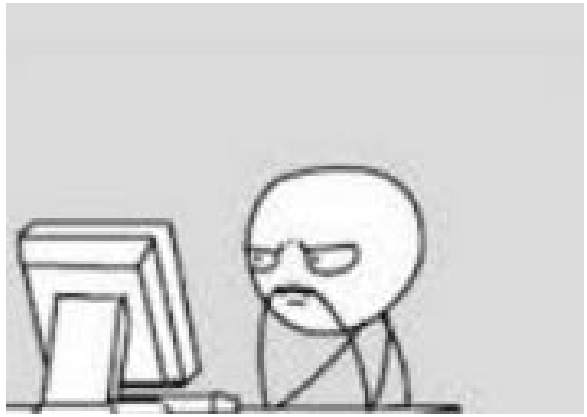


Interviewer: 怎么样选队长?

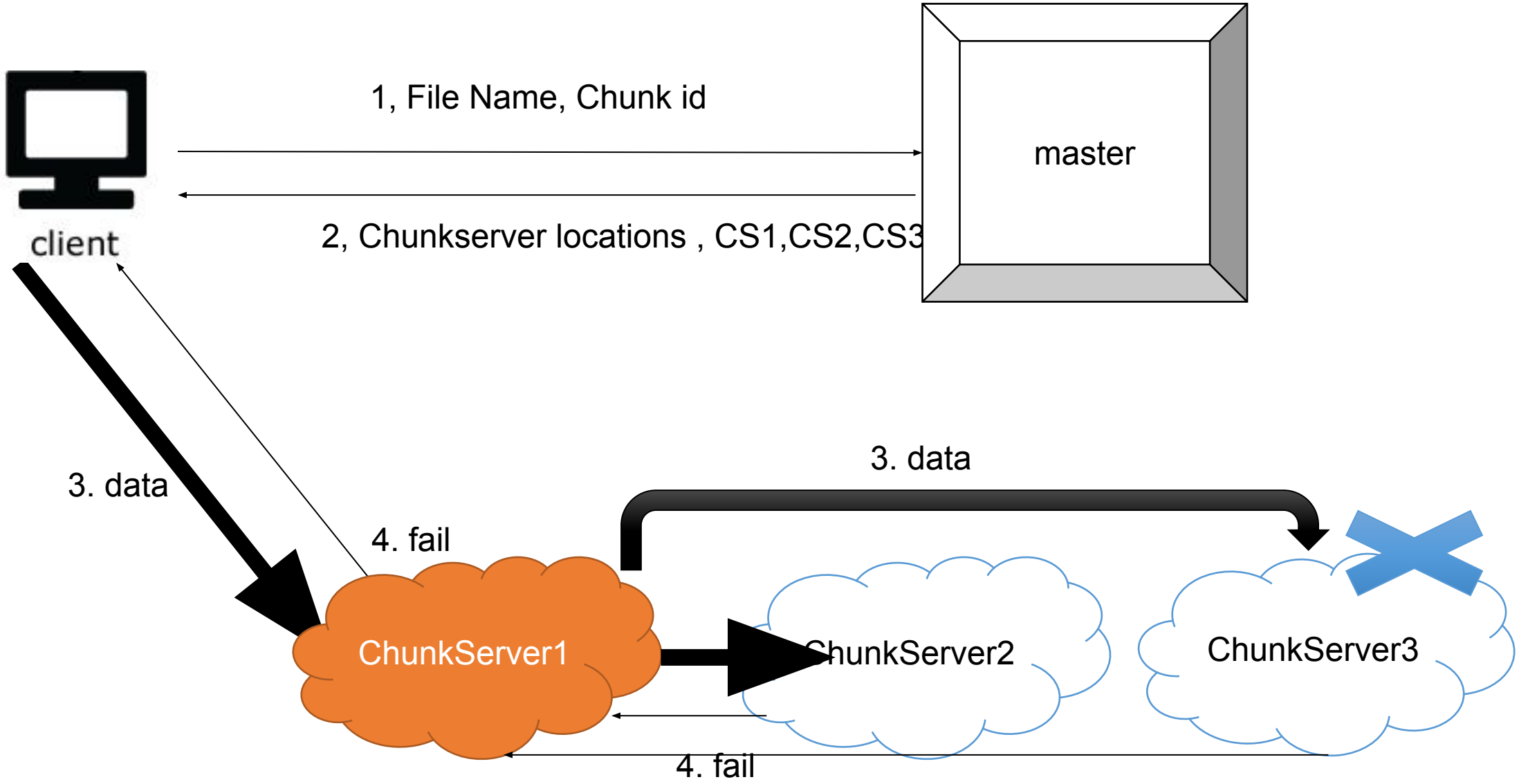
1. 找距离最近的(快)
2. 找现在不干活的(平衡traffic)

Interviewer: 队长一直不变的么？

Interviewer: How to solve ChunkServer failure?



How to solve ChunkServer failure?



How to solve ChunkServer failure?

总结 Summary

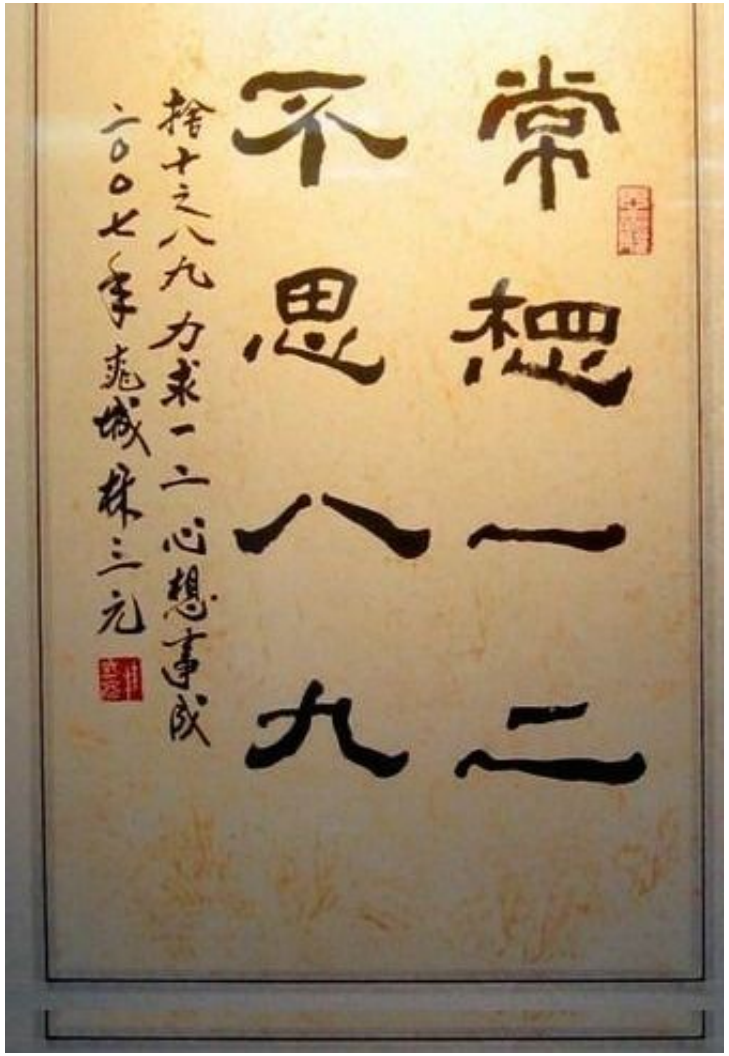
- Key Point: Master-Slave
- Storage:
  - Save a file in one machine -> a big file in one machine -> a extra big file in multi-machine
  - Multi-machine
    - How to use the **master?**
    - How to traffic and storage of master?
- Read:
  - The process of reading a file
- Write:
  - The process of writing a file
  - How to reduce master traffic?
    - Client 和 Chunk Server沟通
  - How to reduce client traffic?
    - Leader Election
- Failure and Recover (key)
  - Discover the failure a chunk?
    - **Check Sum**
  - Avoid the failure a chunk?
    - **Replica**
  - Recover the failure?
    - Ask master
  - Discover the failure of the chunkserver?
    - **Heart Beat**
  - Solve the failure of writing ChunkServer?
    - Retry

Read More

- Expert/Master, http://url.cn/dOLFCs
- Expert/Master, http://url.cn/eErkhm
- Expert/Master, http://url.cn/LqTkoa

- 为什么说学习GFS对我们其他的系统设计也有好处呢？
  - Master Slave Pattern
  - How to handle failure
  - How to use GFS

鸡血一刻



- 人生不如意處十之八九，要多想餘下那一二得意之處。
- 我們對於難得的成功要極度珍惜，保持一顆感恩的心和一個樂觀的態度。