系统设计
Distributed System Design

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

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，No-SQL 第二节课我们已经讲过

Overview of today

- Distributed File System (Google File System)

- Design a Lookup Service

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的解决的问题。

比如：存储不够？QPS太大?

了解分布式文件系统后可以做什么？

1. Google，Yahoo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。

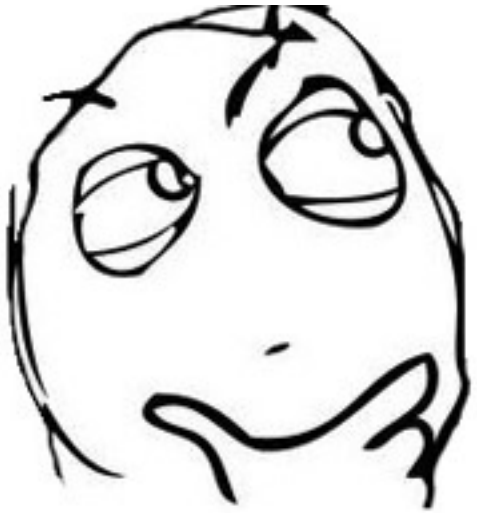
Hadoop's Distributed File System

Amazon's Simple Storage Service(s3)

Google File System

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.
- 需求2
  - 多台机器存储这些文件

Needs 需求

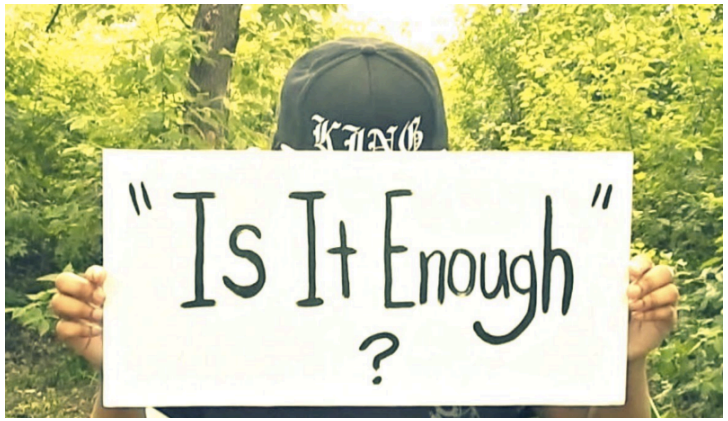
需要设计多牛的系统

Needs 需求

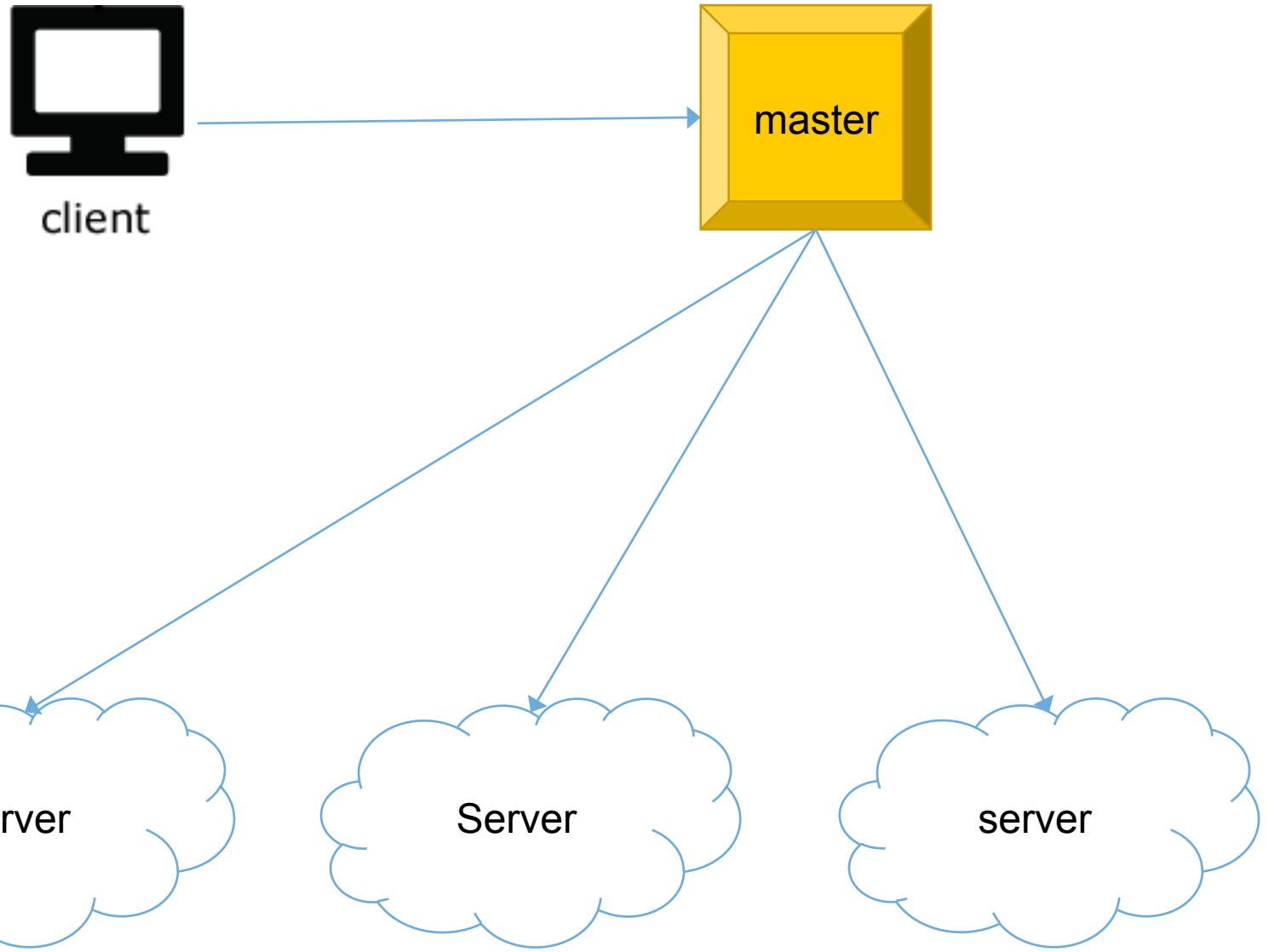
- 支持多大的文件?
  - 越大越好? 比如 >1000T
- 支持多少台机器?
  - 越多越好? 10万台, Google 2007 year

Application 应用

系统由哪些部分构成



Application 应用



Kilobyte 数据

数据如何存储

Kilobyte 数据

- 大文件存在哪?
  - 内存? 数据库? 文件系统?

Kilobyte 数据

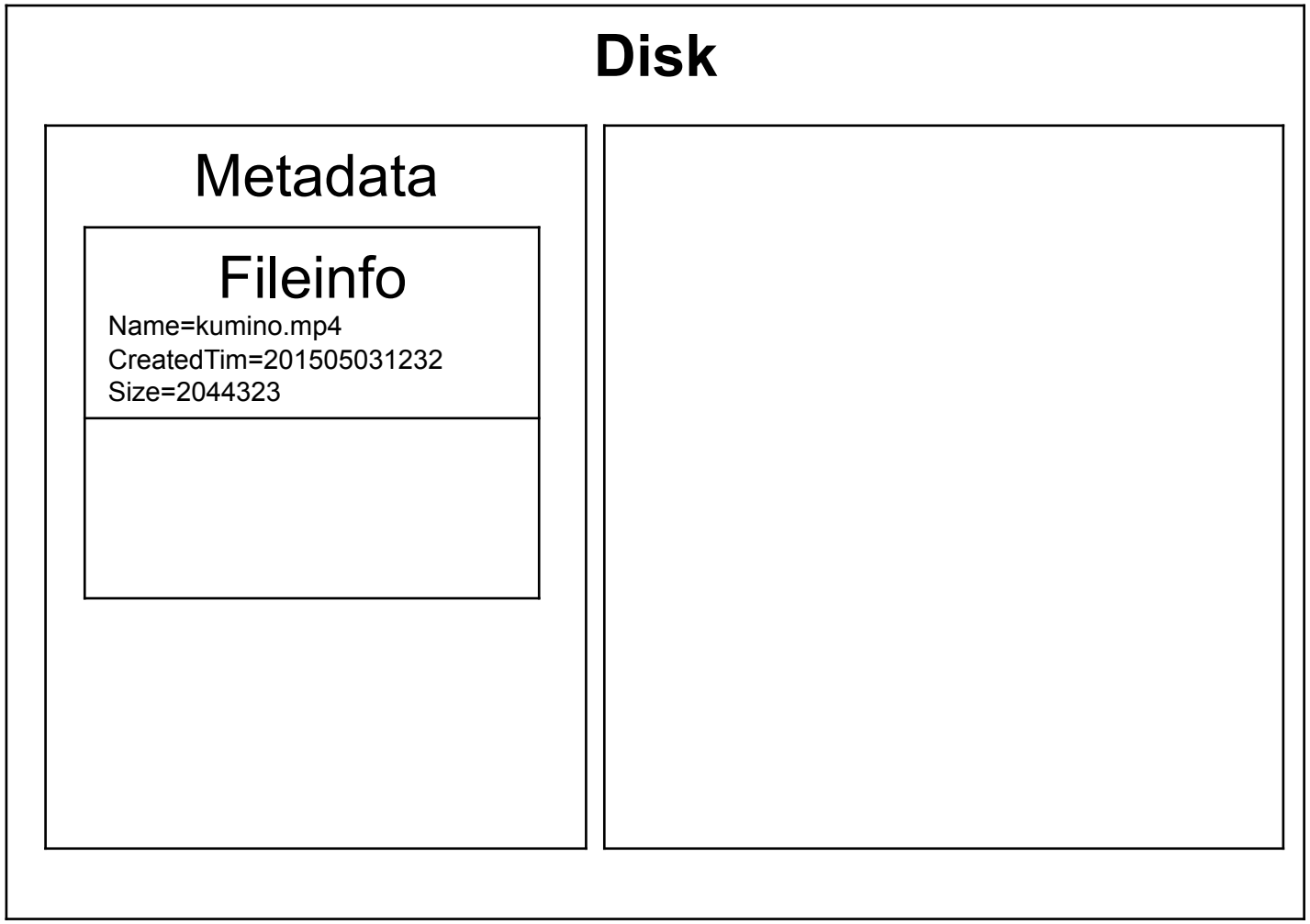
- 大文件存在哪?
  - 内存? 数据库? 文件系统?

- 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?

普通的操作系统怎么做的呢？100G

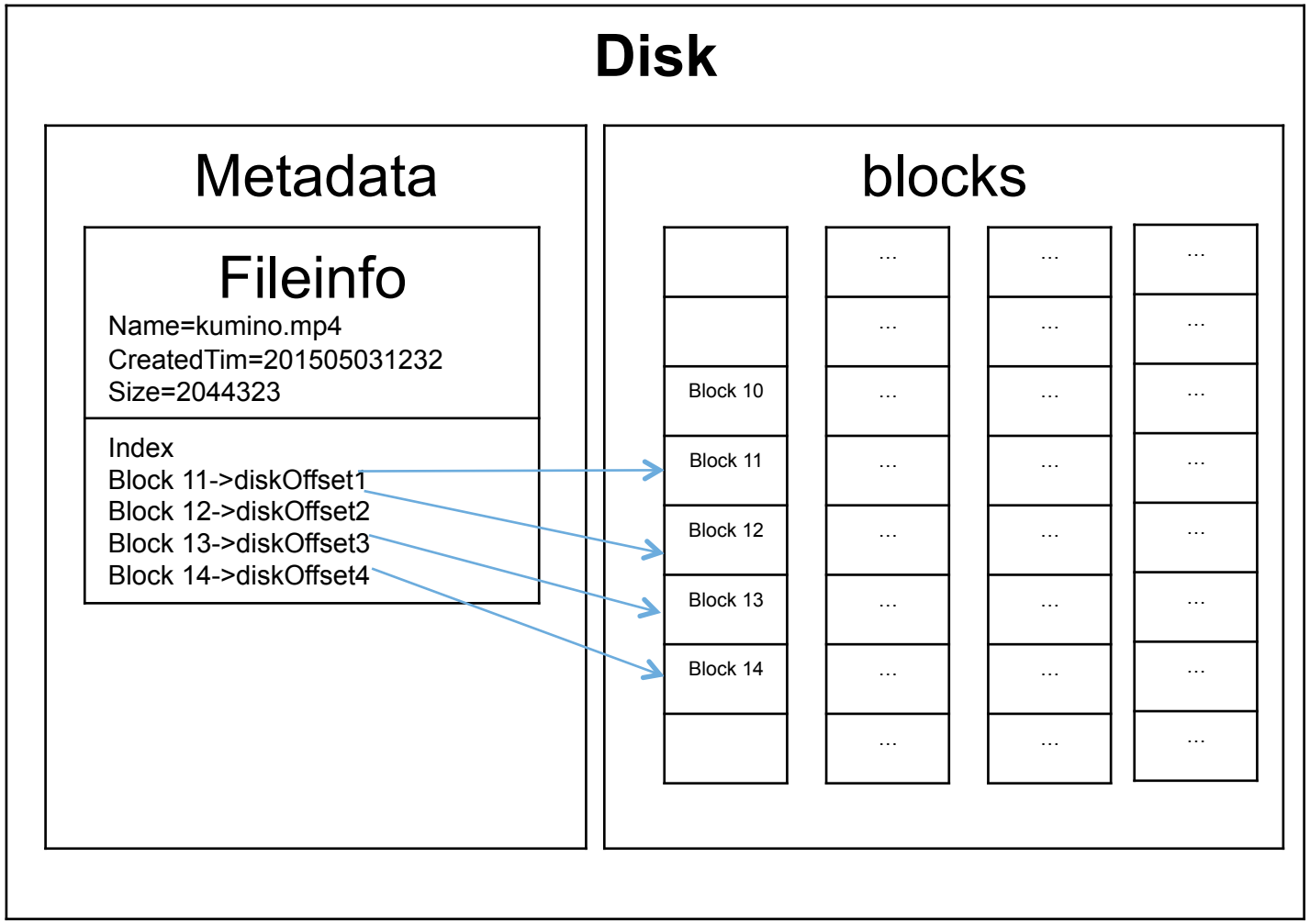
How to save a file in one machine



Metadata 访问 尝尝多于 内容的访问

文件是分开存储的呢？还是连续存储的呢？

How to save a file in one machine



Key point

- 1 block = 1024Byte
- Block Advantage
  - check sum
  - 碎片化

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

Is block size big enough?

100T

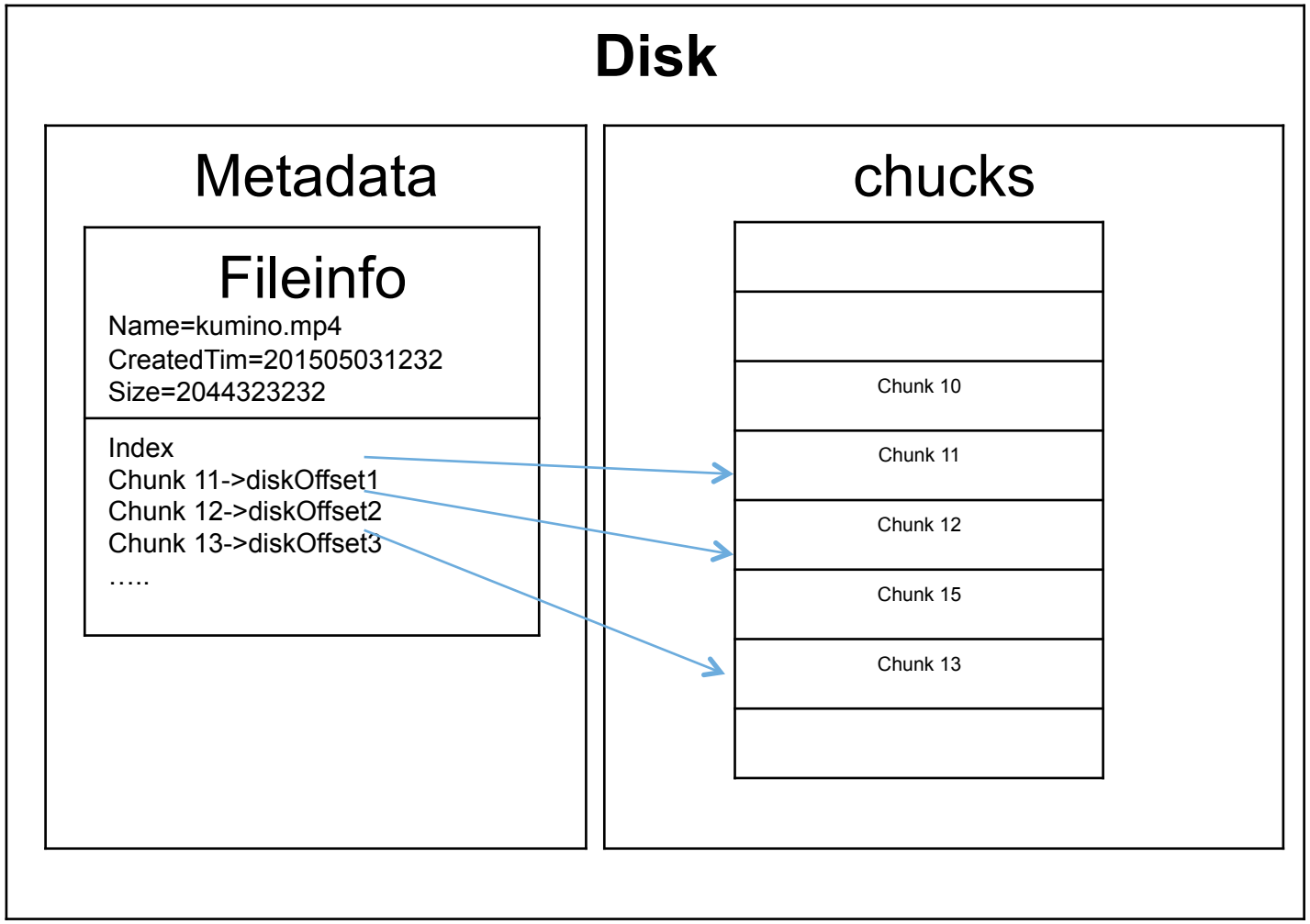
=100*1000G

=100*1000*1000M

=100*1000*1000*1000K

=100*1000*1000*1000block

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



Key point

- 1 chunk= 64M
  = 64*1024K
- 1 new block = 64K
- 1 chunk = 1024 new blocks

Advantage

- Reduce size of metadata
- Reduce traffic

Disadvantage

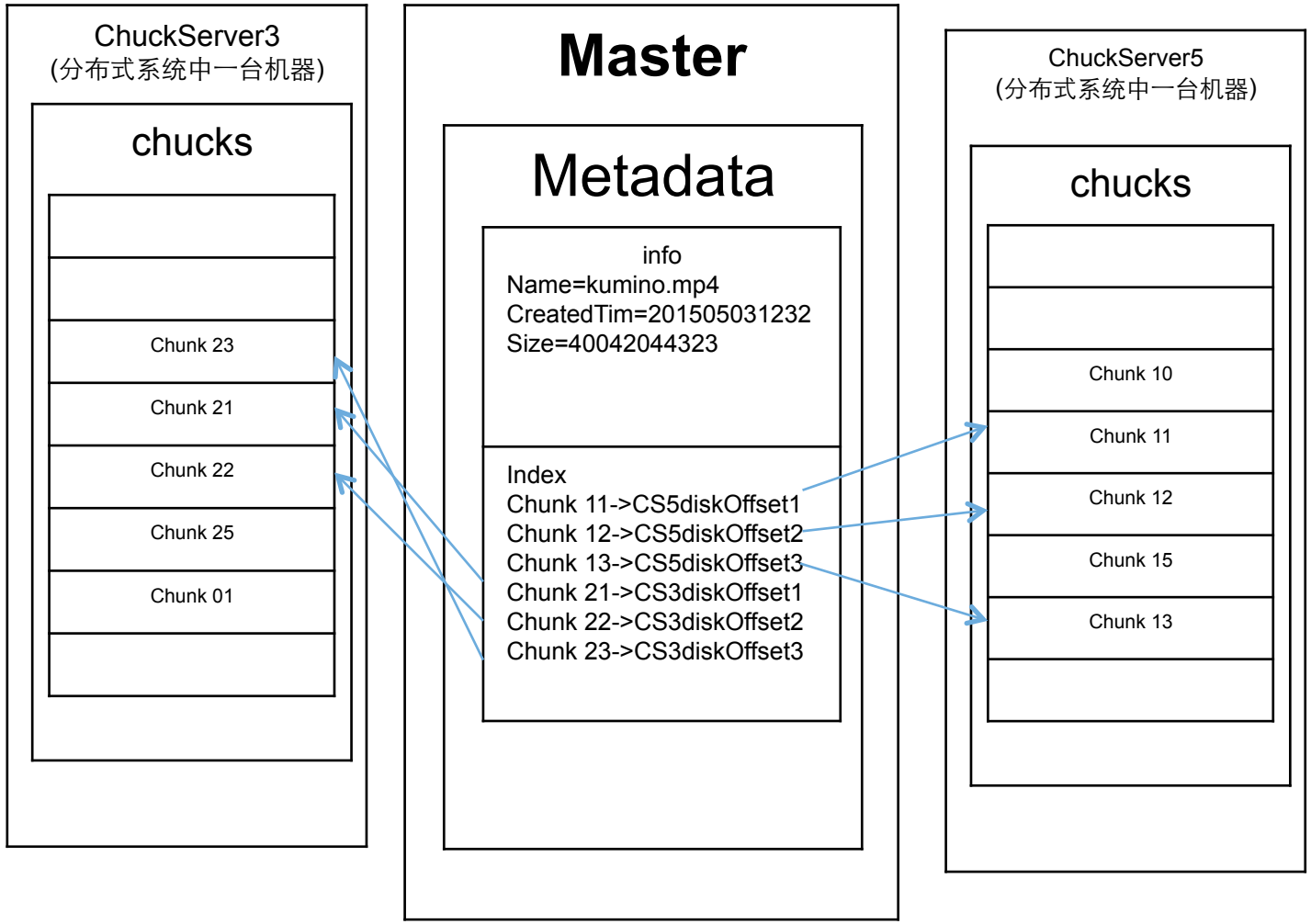
- Waste space for small files

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

100P

Is one machine big enough?

How to save extra-large file in several machine?



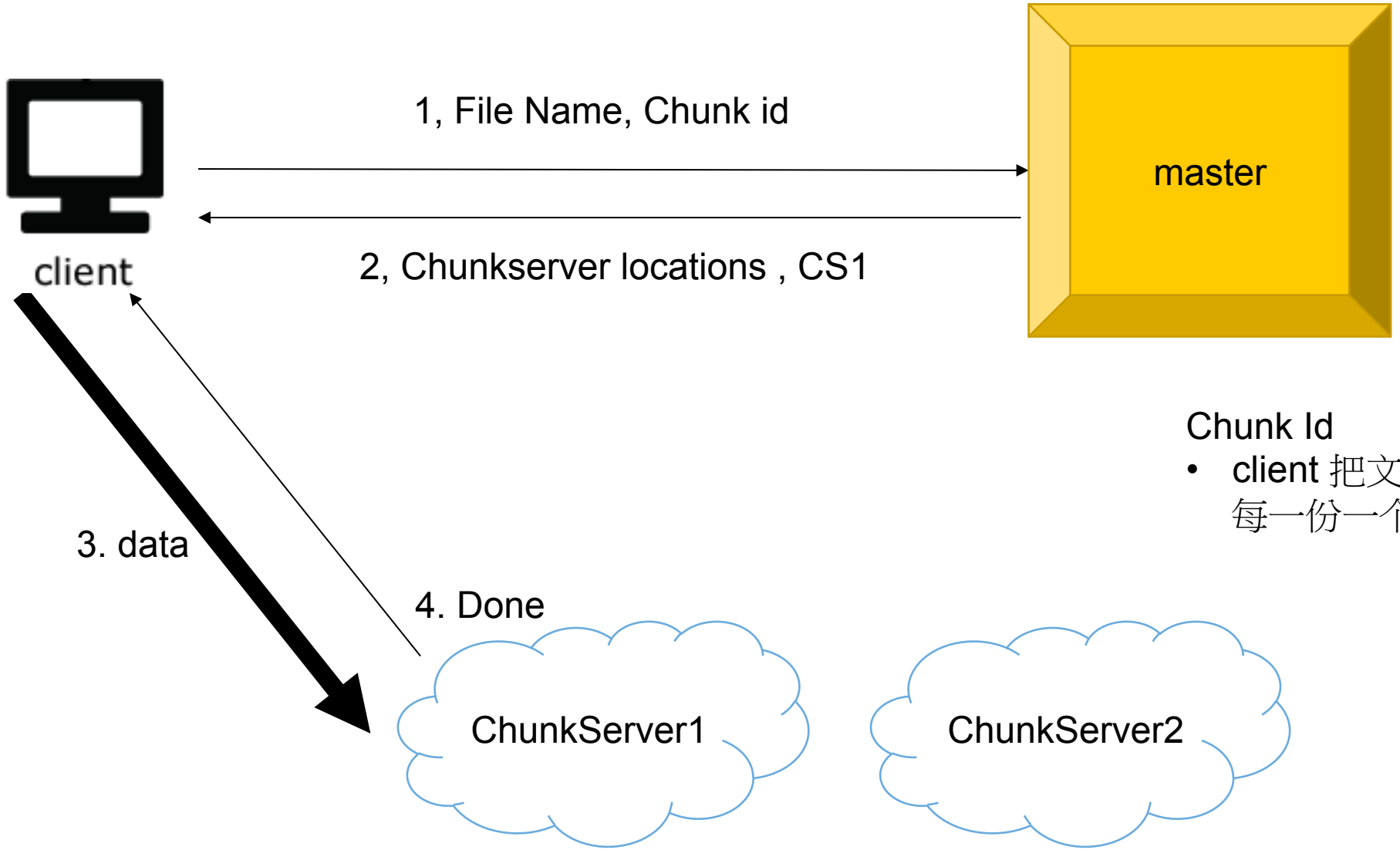
Key point

- One master + many ChunkServers

One Work Solution for Read / Write

Interviewer: How to write a file?

How to write a file?

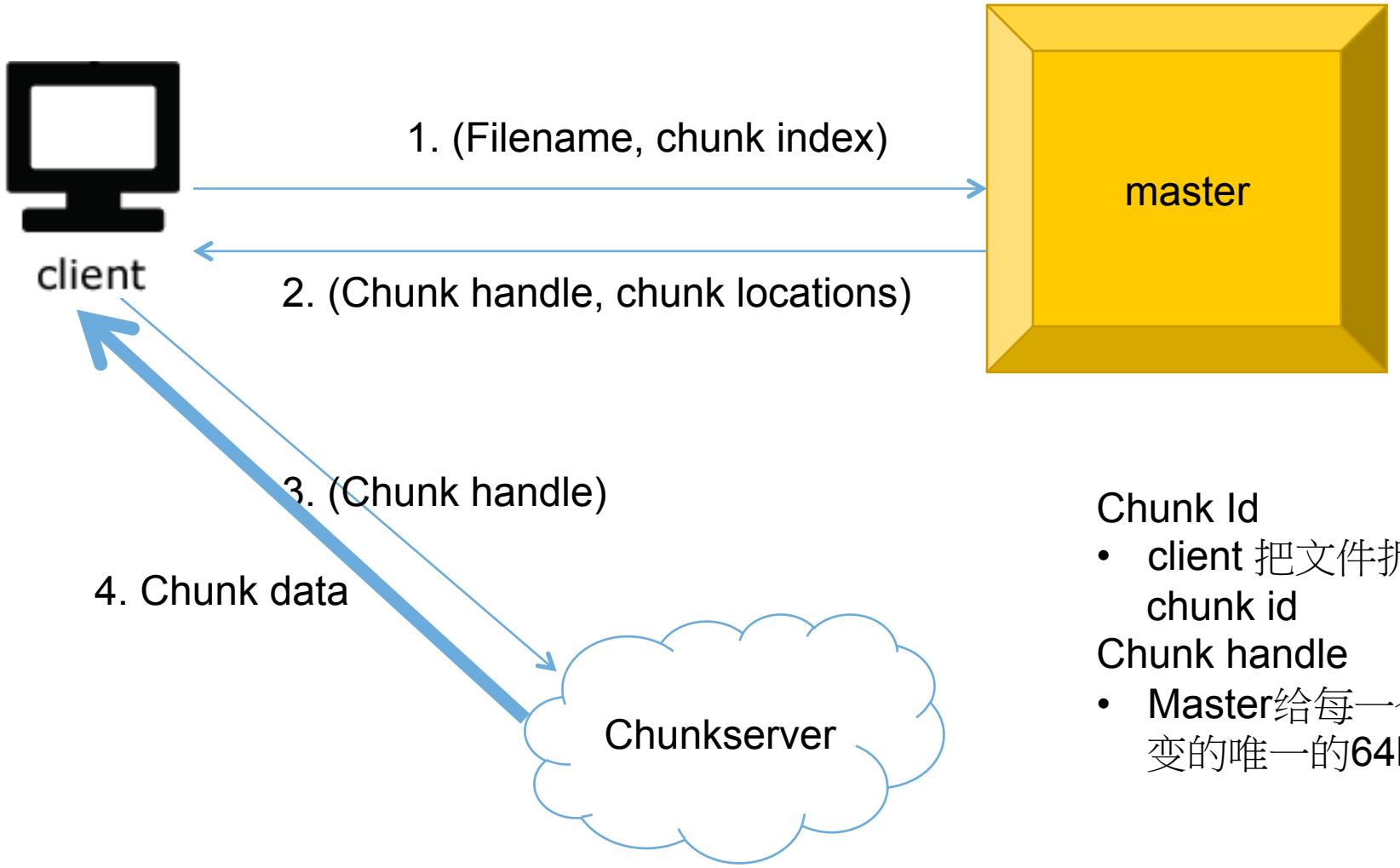


Chunk Id

- client 把文件拆分为n份，每一份一个chunk id

Interviewer: How to read a file?

How to read from a file?



Chunk Id

- client 把文件拆分为n份，每一份一个chunk id

Chunk handle

- Master给每一个ChunkServer分配的不变的唯一的64bit的编号

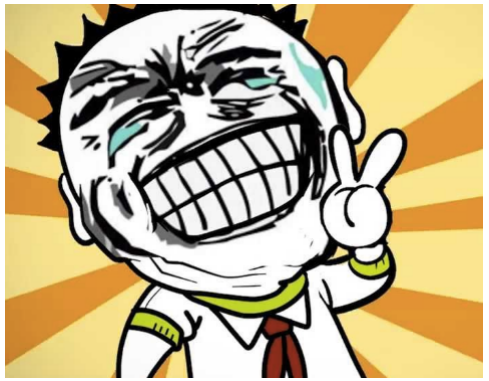
Master Task

- 存储各个文件数据的metadata
- 存储Map(file name + chunk id -> 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

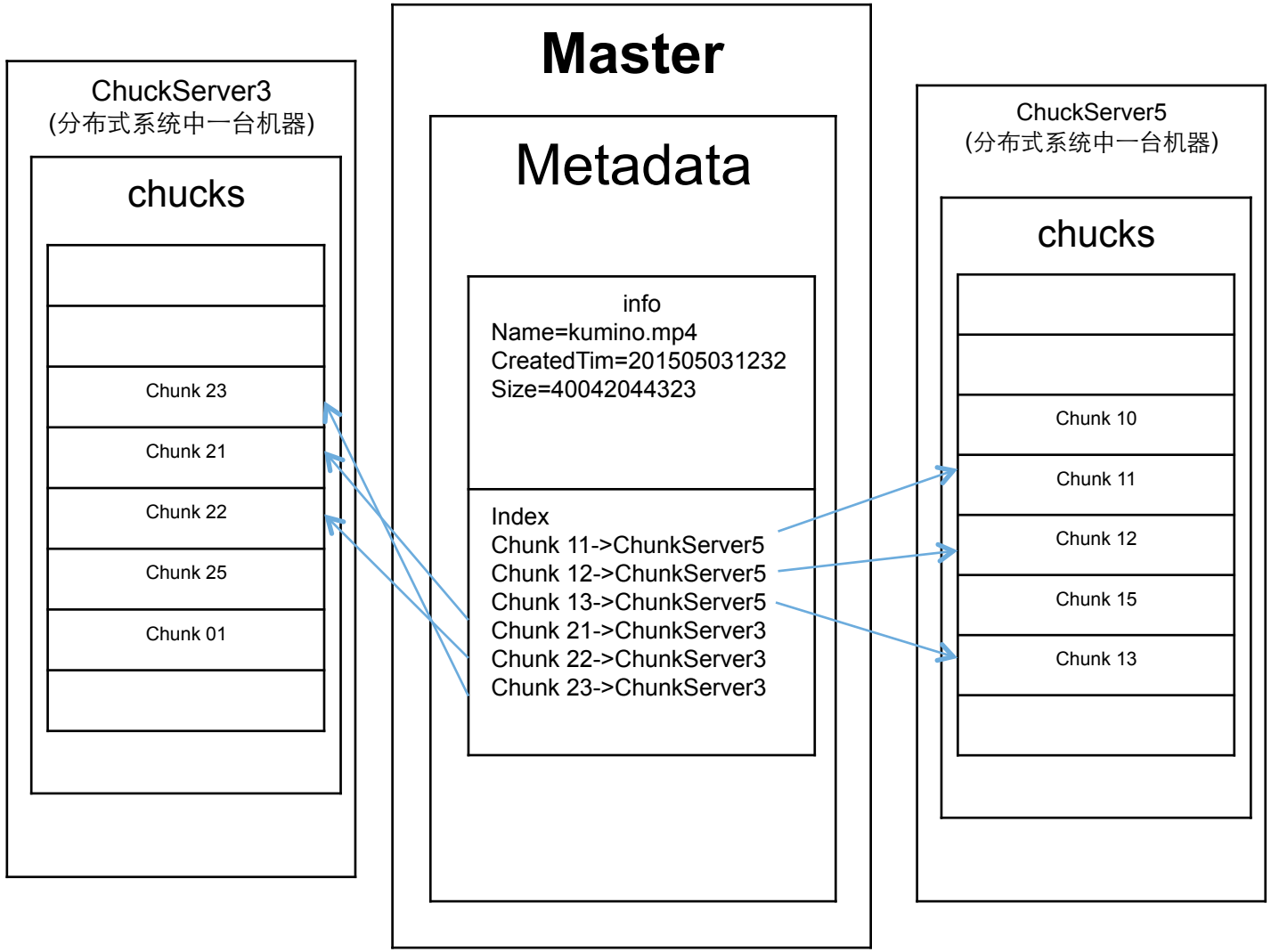
- 存储
  - 普通文件系统 Meta Data，Block
  - 大文件存储： Block-> Chunk
  - 多台机器超大文件: Chunk Server + Master
- 写入
  - Master+Client+ChunkServer 沟通流程
  - Master 维护metadata 和 chunkserver 表
- 读出
  - Master+Client+ChunkServer 沟通流程



Evolve 进化

系统如何优化与维护

How to save extra-large file in several machine?



Key point

- One master + many ChunkServers

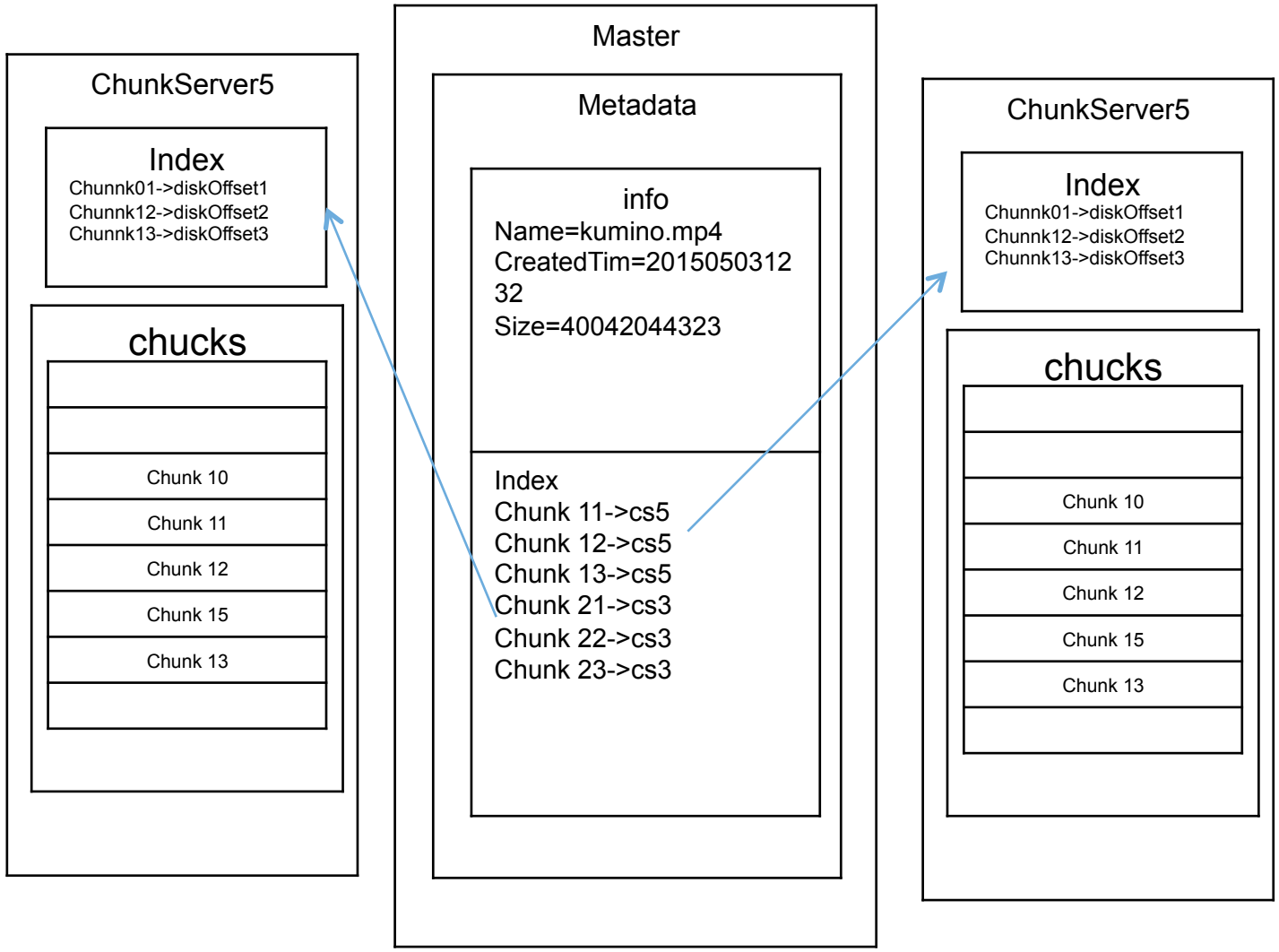
Disadvantage

- Any change of a chunk in a ChunkServer need to notify the Master

Evolve about the Storge

How to reduce traffic and storage of master?

Evolve about the Storage



Key point

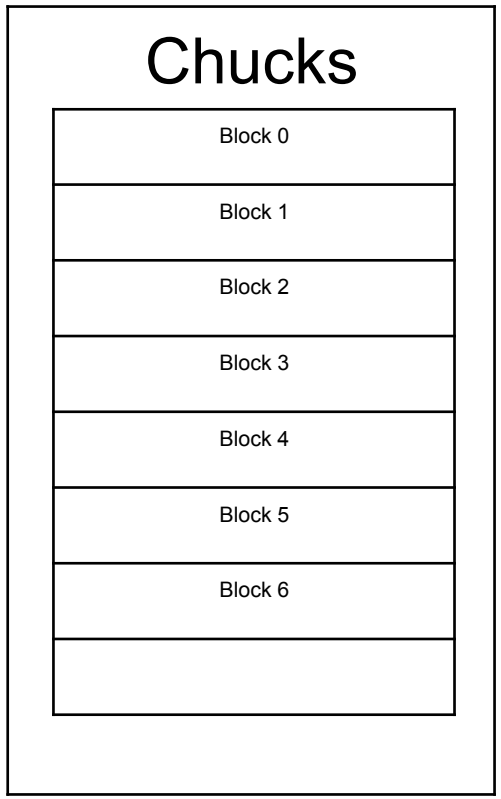
- The master don't record the diskOffset of a chunk

- Advantage
- Reduce the size of metadata in master
- Reduce the traffic between master and ChunkServer

Evolve about the Failure and Recover

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

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



Metadata

Block0-> checksum
Block1-> checksum
Block2-> checksum

Key point

- 1 chunk = a list of blocks
- 1 block = 64KB
- Each block has a checksum
- 1 checksum = 32bit
- The size of checksum of 1T file
- 1T/1KB*32bit = 64MB
- Load the chucksum in the memory

When to check checksum?

- It will compare its checksum when it reads

Check Sum

原来

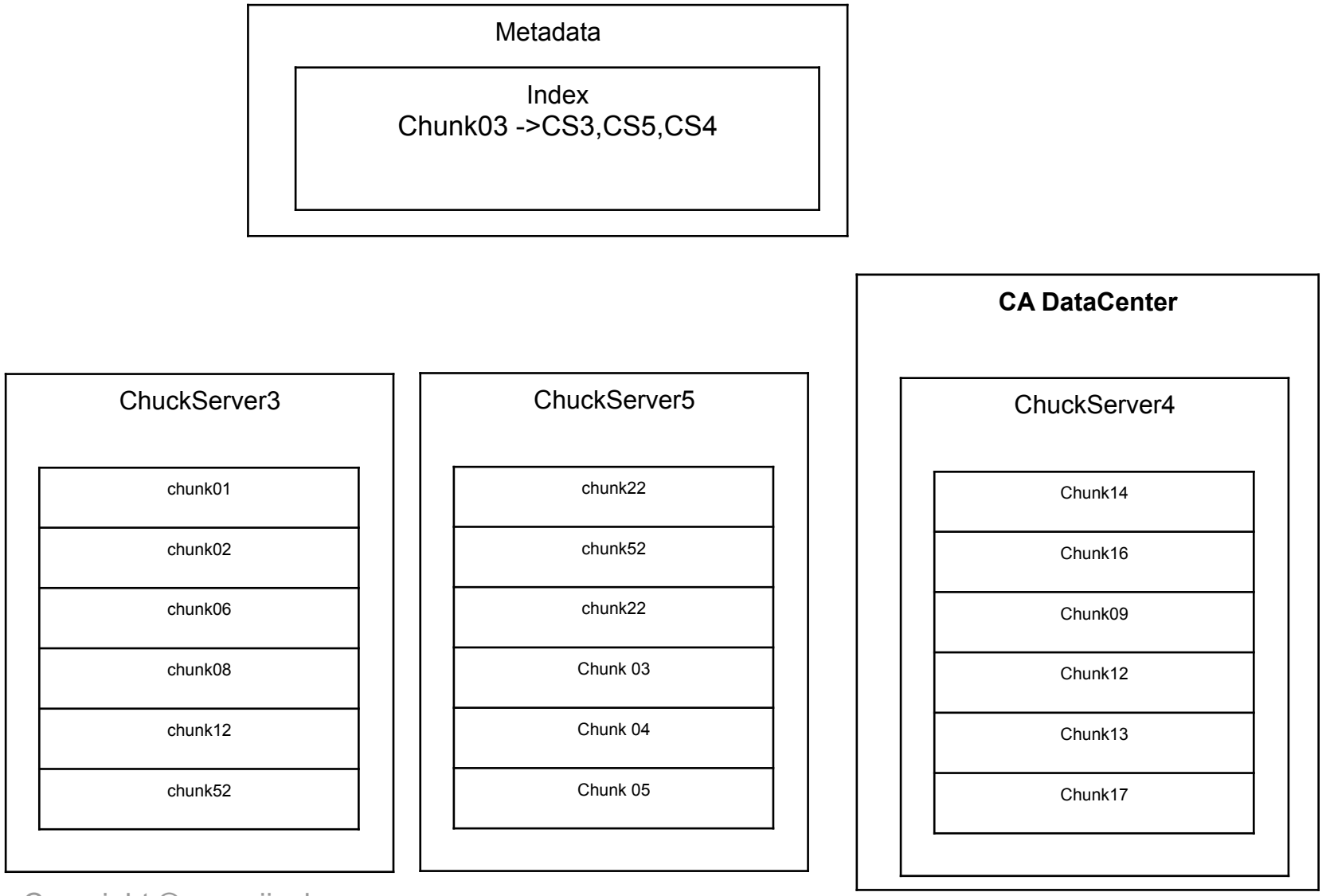
数据	1	2	3	Checksum(xor)
二进制表示	1	10	11	11

错误后

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

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

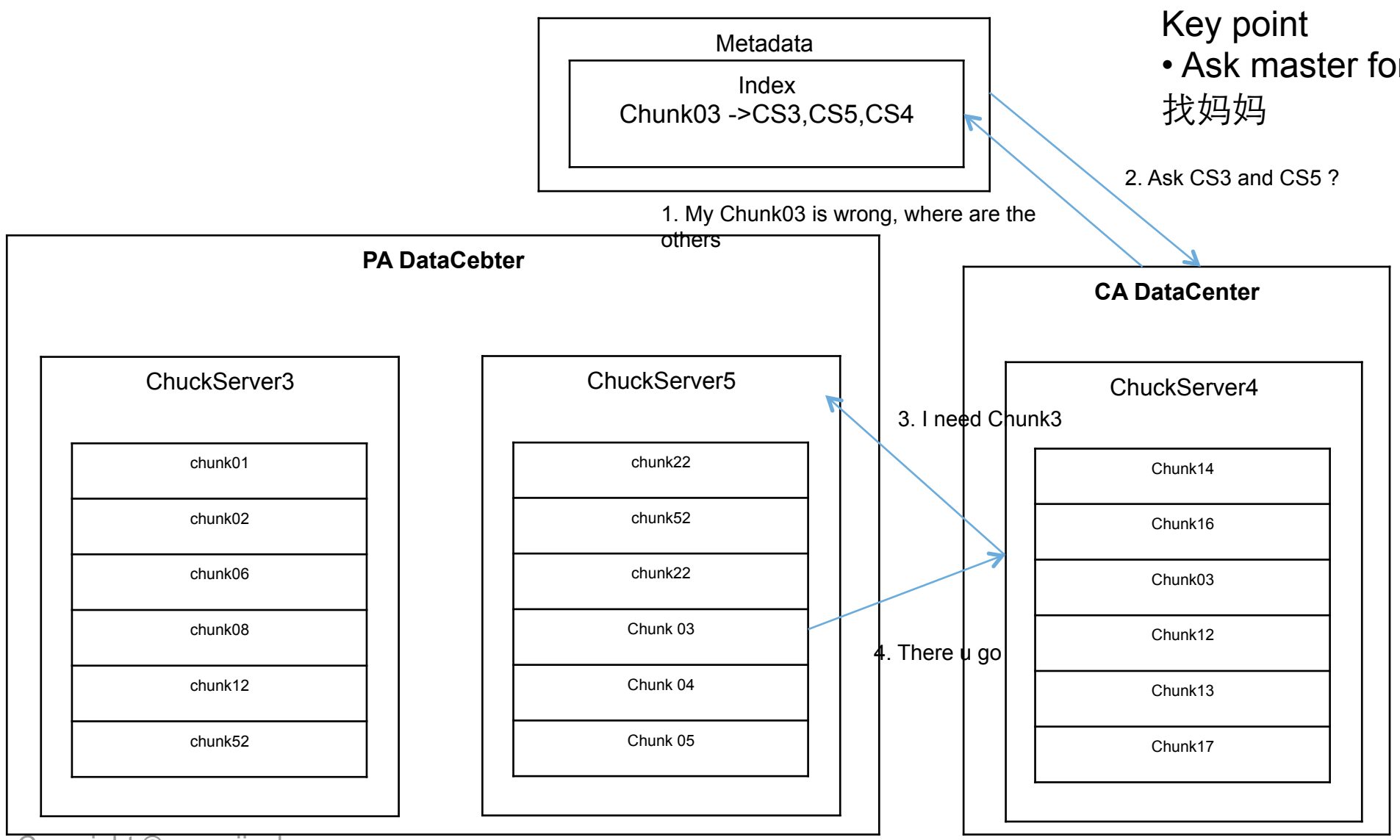
How to avoid data loss when a ChunkServer is down/fail?



Key point

- Replicate the Chunks
- How many replications?
  - 3
  - Across racks: 2+1
- How to select ChunkServers of a chunk
  - Server with below-average disk space utilization
  - Limited number of “recent” creation， prevent new disk write

Interviewer: How to recover when a chunk is broken?



Key point
• Ask master for help Question?
找妈妈

How to find whether a ChunkServer is down?

How to find whether a ChunkServer is down?

Interviewer: HeartBeat.

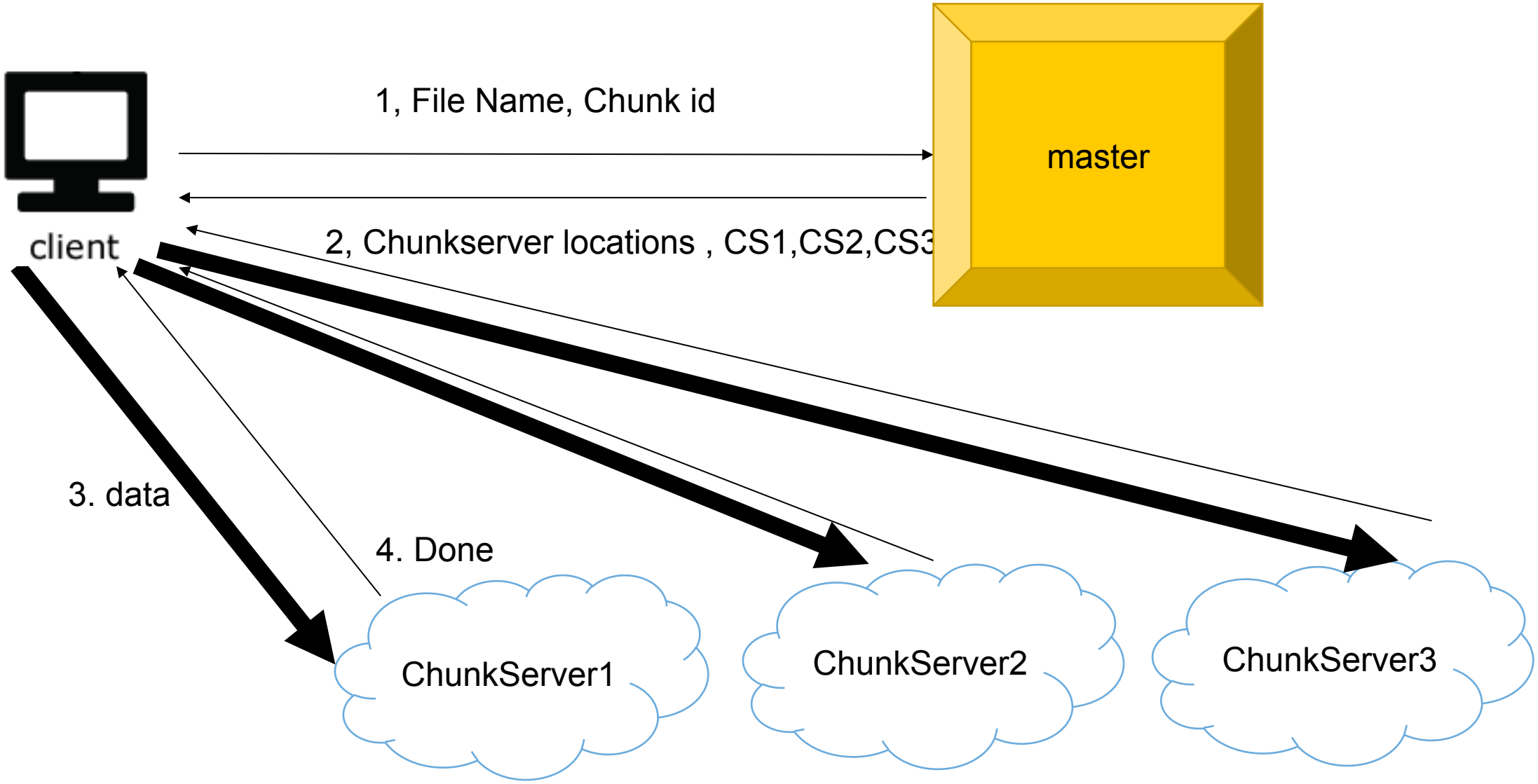
A: master -> chunkservers?

B: chunkservers->master?

Evolve about the Write

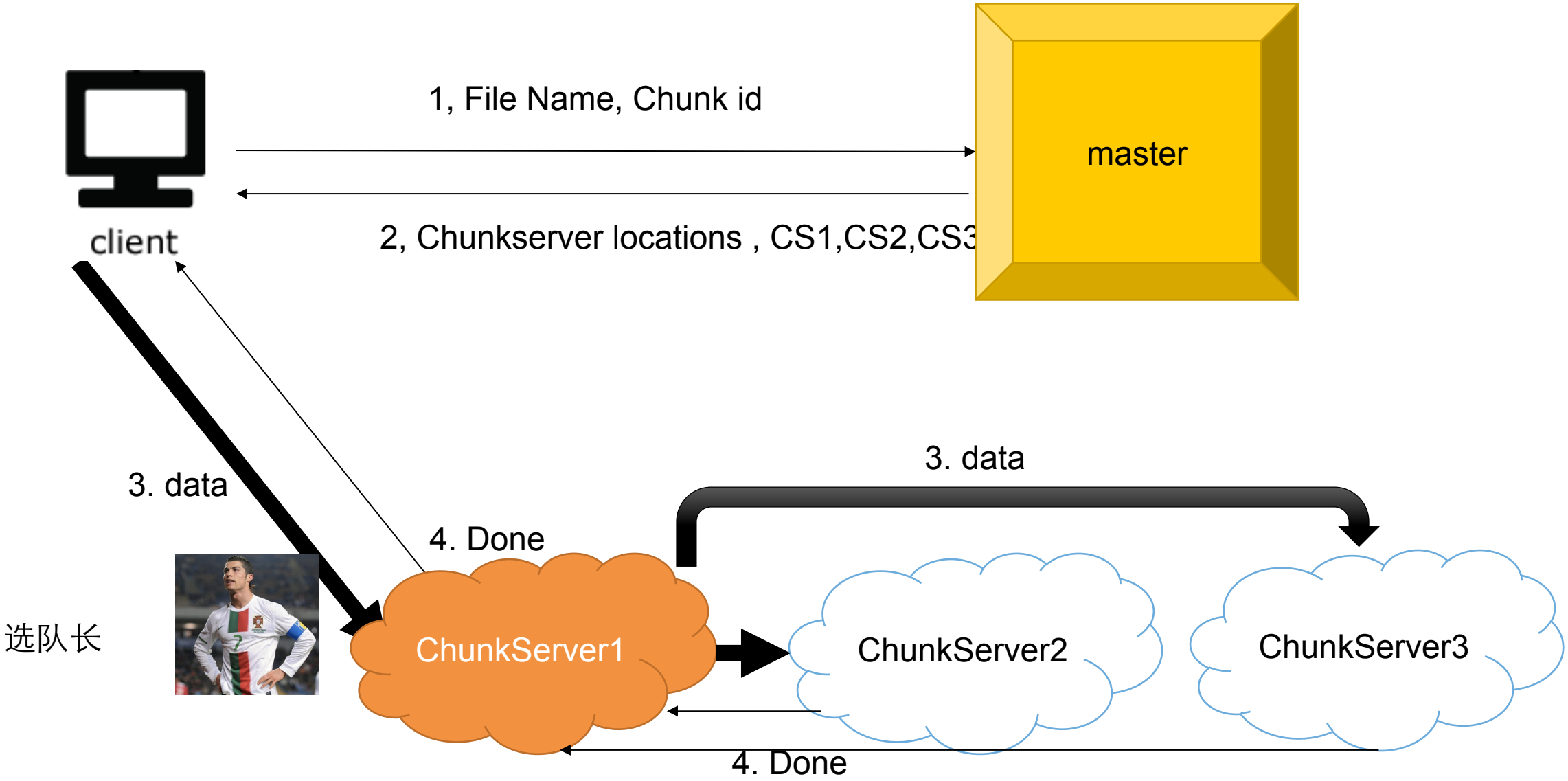
Interviewer: Whether write to only one server is safe?

How to write a file?



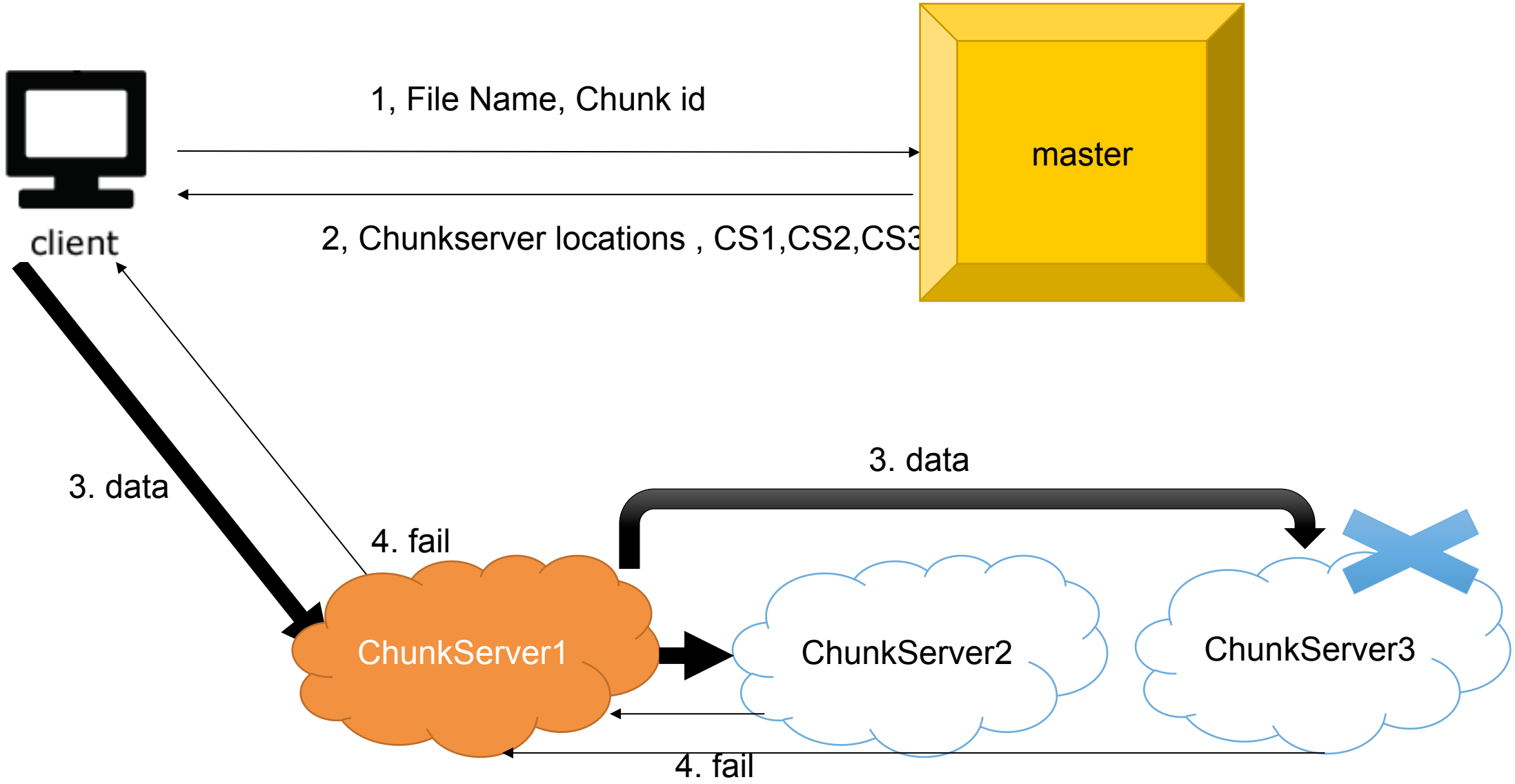
Interviewer: How to solve Client bottleneck?

How to solve Client bottleneck?



Interviewer: How to solve ChunkServer failure?

How to solve ChunkServer failure?



How to solve ChunkServer failure?

总结 Summary

- 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?
    - 选队长
- Failure and Recover
  - 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

- 下课后我会总结一个文字版比较全的读取和写入流程放在ppt里面一起发给大家

Microsoft Interviewer: Design a Lookup Service

Key: Url; Value: Page content

Key: UserId; Value: User profile

Design a Lookup Service

- 问题：设计一个只读的lookup service
  - 要求
    - 10 billion key-value pair
    - Key Size < 0.1kB, 总共1000G
    - Value Size = 100KB, 总共 1P
  - 解题思路SNAKE：
    - Scenario 比较明确 Lookup
      - 怎么样实现Look Up?
    - Needs 题目要求
    - Application client + server*
    - **K**ilobyte
      - 存哪些数据
        - Key Value
      - 怎么存？存储架构

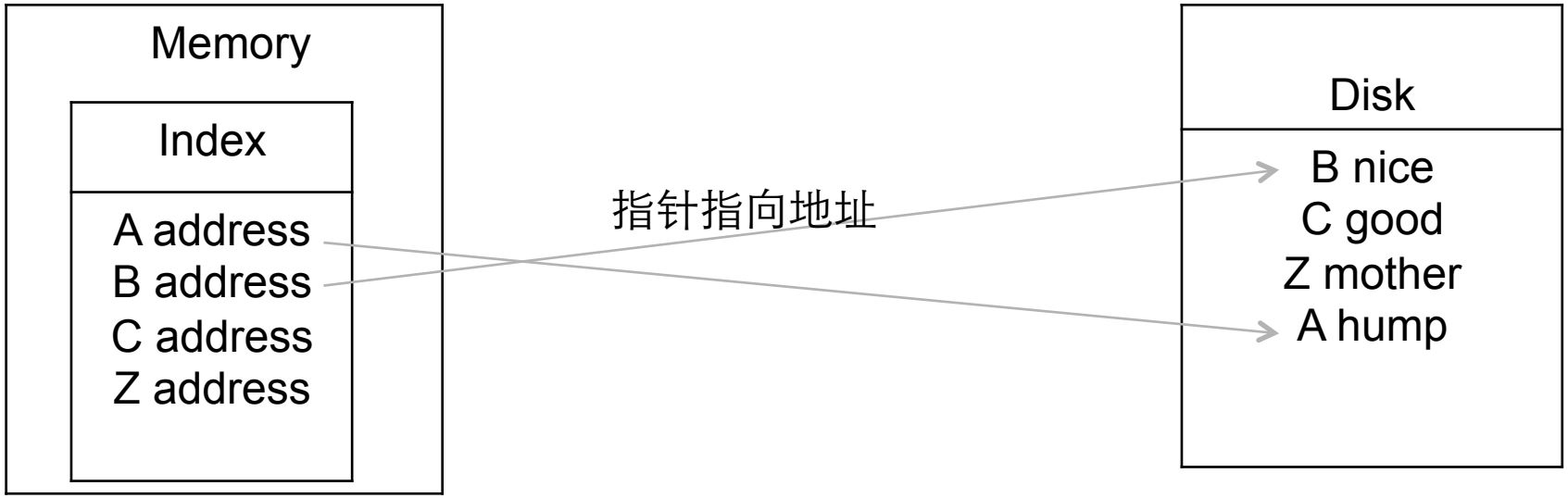
如果选择存储架构

- Hash Map
  - Value = 1P
  - 内存不够怎么办?
- NoSql 数据库
  - 今天我们就是要讲Bigtable, 不要用概念直接回答。我们要深入。
  - Bigtable 带写操作，这道题不带写操作其实不需要那么复杂。
  - 我们需要设计一个**只读**的lookup service **更高效**的系统
- GFS
  - **放**到磁盘上面，那要具体怎么设计呢?

Interview: How to support lookup for one file on disk

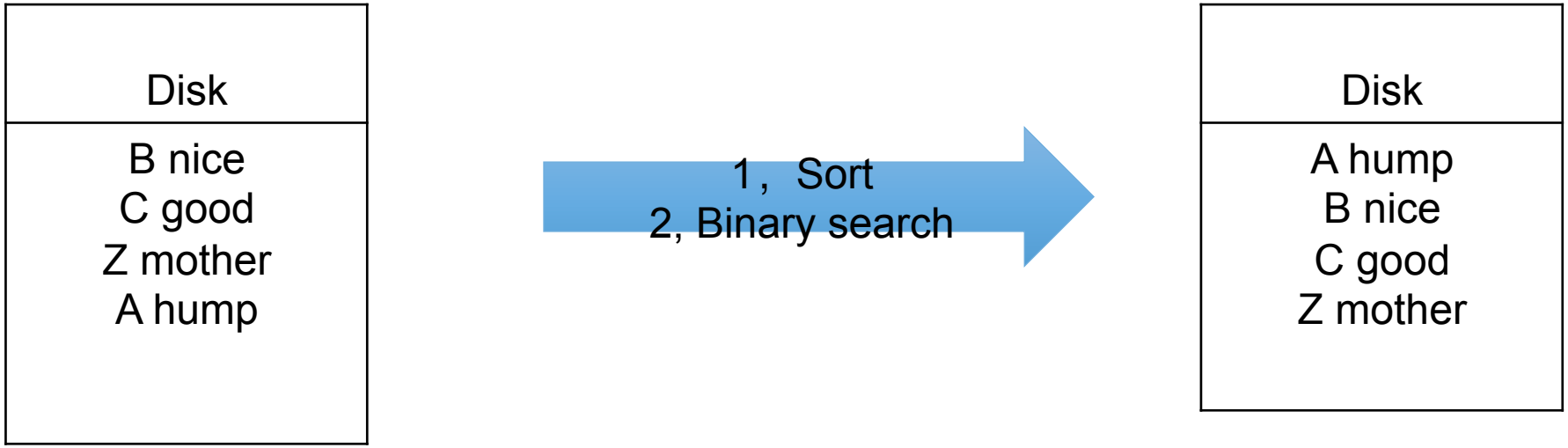
How to support lookup for a file on disk

方法一



How to support lookup for a file on disk

方法二



你觉得上面两个方法哪个好?
Index VS Sort?

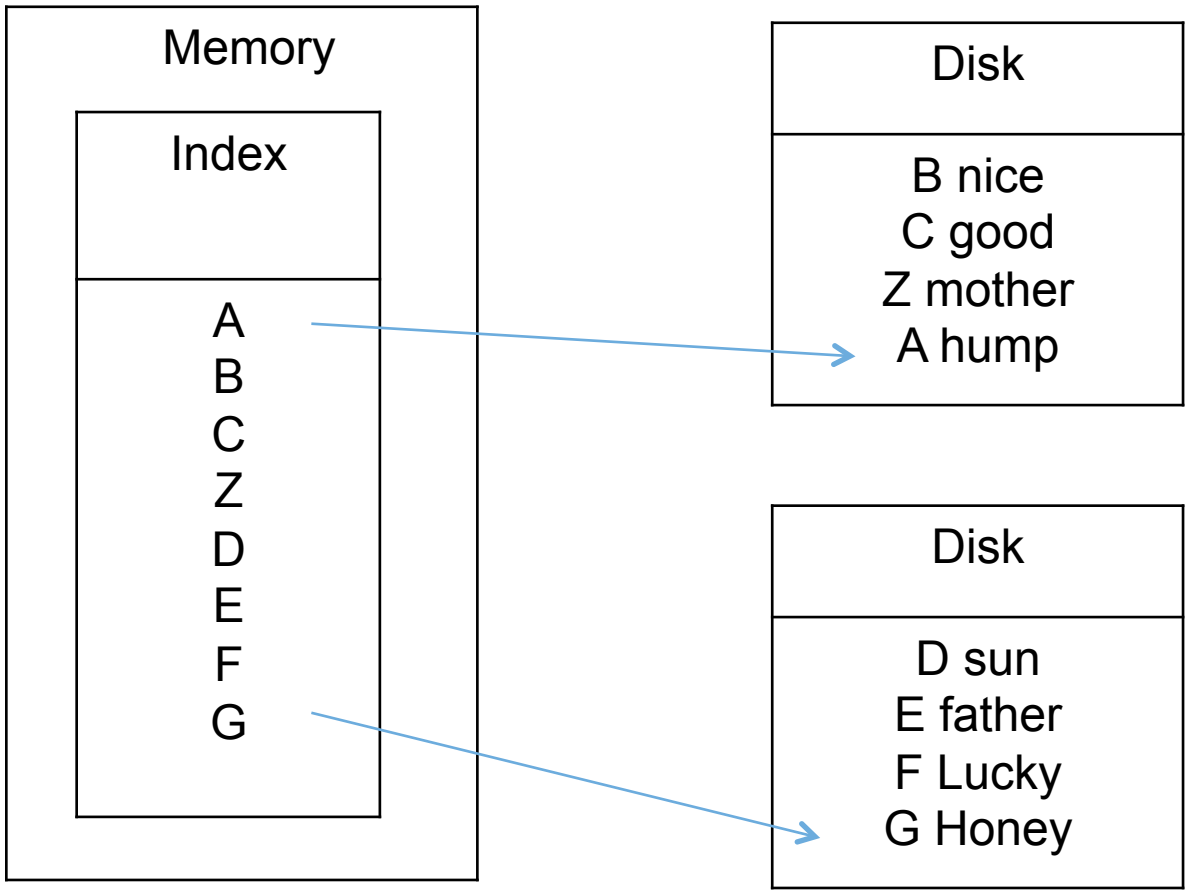
随着文件增大

Index < Sort

Interview: How to support lookup for multi-files on disk

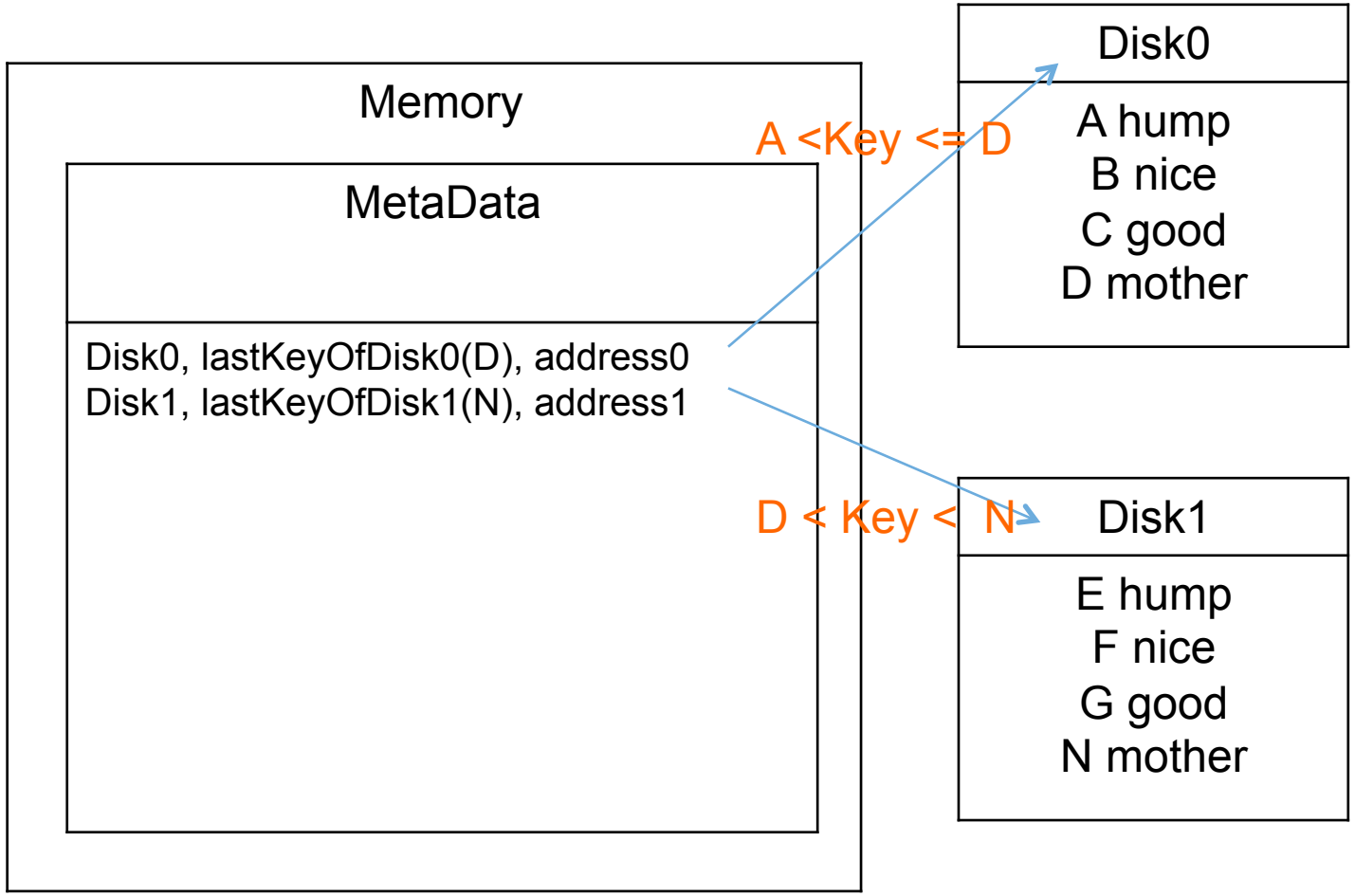
Lookup File Size increase

How to support lookup for multi files on disk



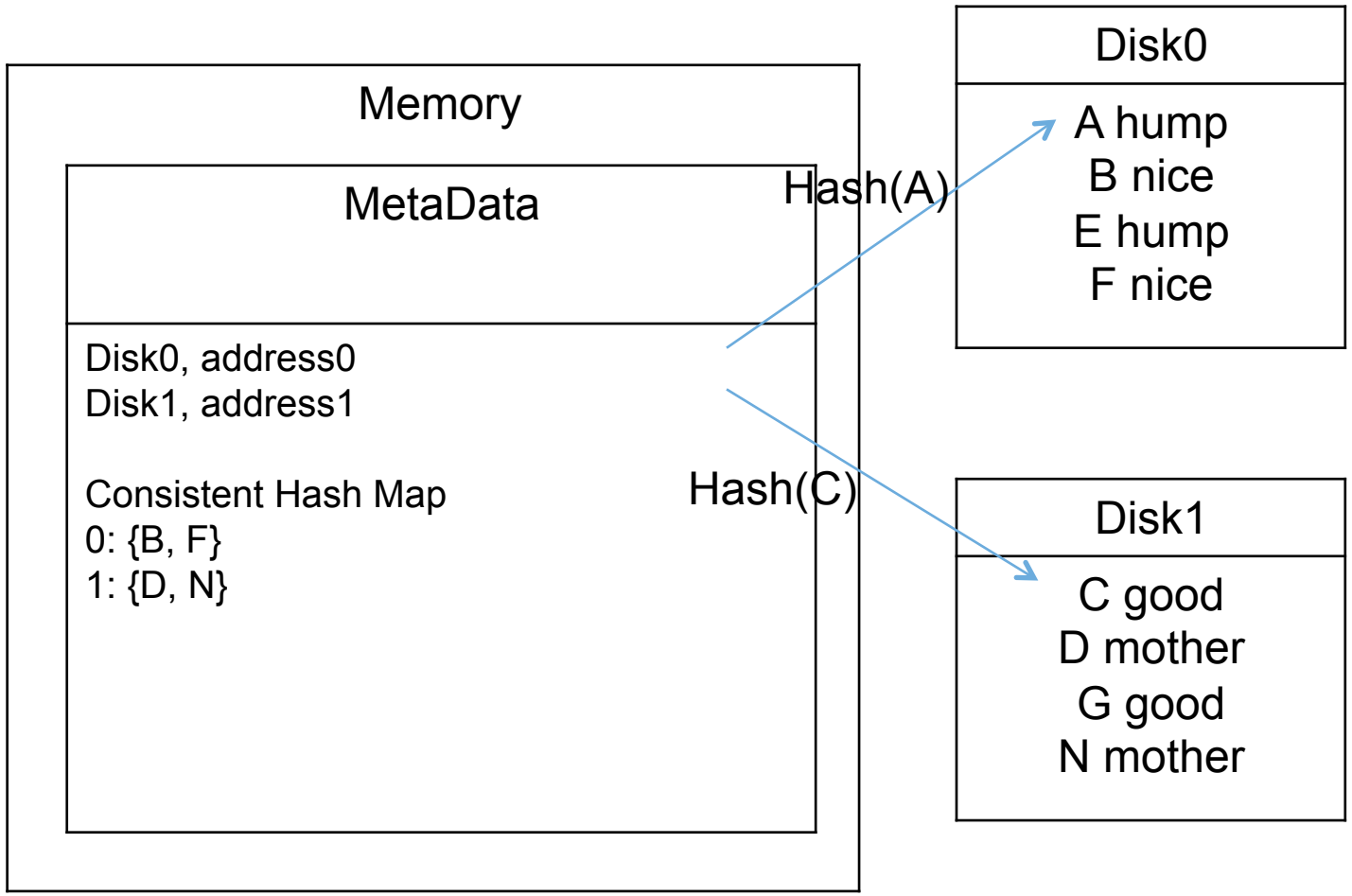
- Problem
  - Load too many key in the memory
  - 1TB= 10 billion key * 0.1 KB

How to support lookup for a file on disk



- Key
  - Split key
  - Don't store every key in the memory

How to support lookup for a file on disk



- Key
  - Consistent Hash Map
  - Don't store every key in the memory
- http://www.lintcode.com/zh-cn/problem/consistent-hashing-ii/

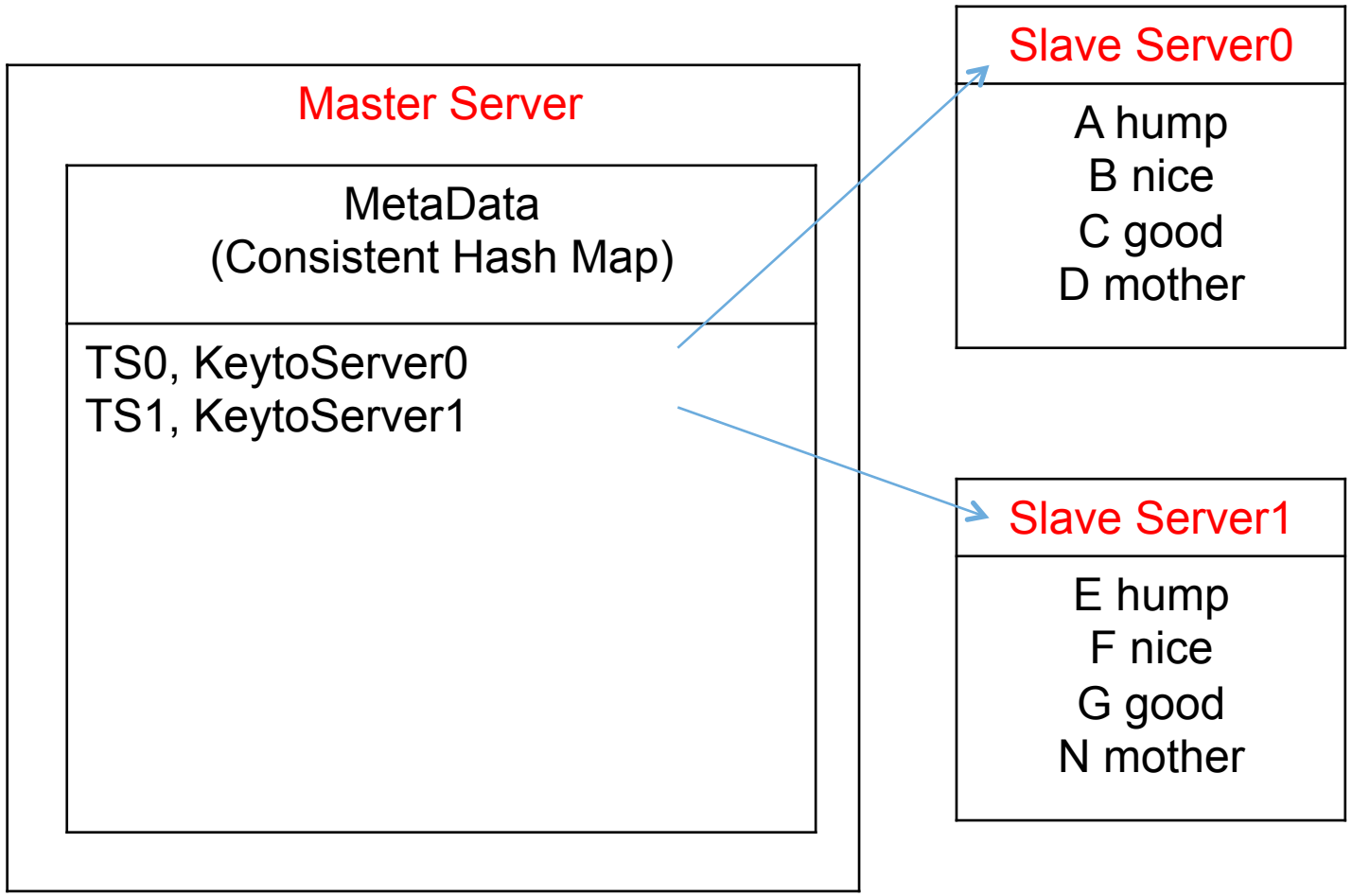
Interviewer: Disks increases as key value increases，What should we do?

Data(Key Value pair) Size is 1P.

Answer: Need a lot of server to maintain a lot of disks

Master + Slave

How to support lookup for a file on disk



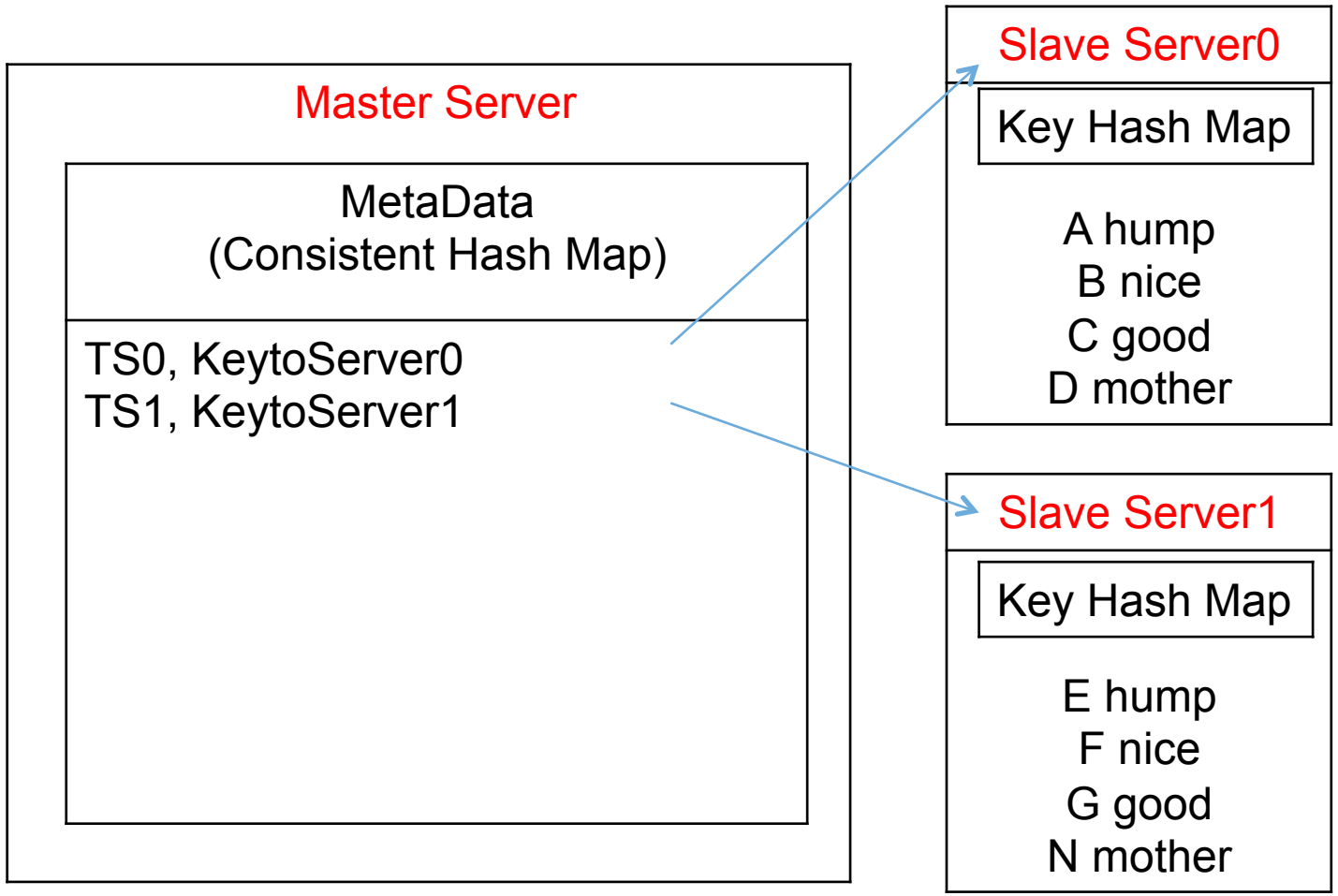
Disk -> Server

- Disk这么多，需要多台机器
- Single Point failure
- 降低每台机器的QPS
- 这么多台机器需要一个管理

- Key
  - Consistent Hash
  - Master Slave 模型

- 有点类似web server的模型.
- Master = web server
- Slave Server = db

How to support lookup for a file on disk



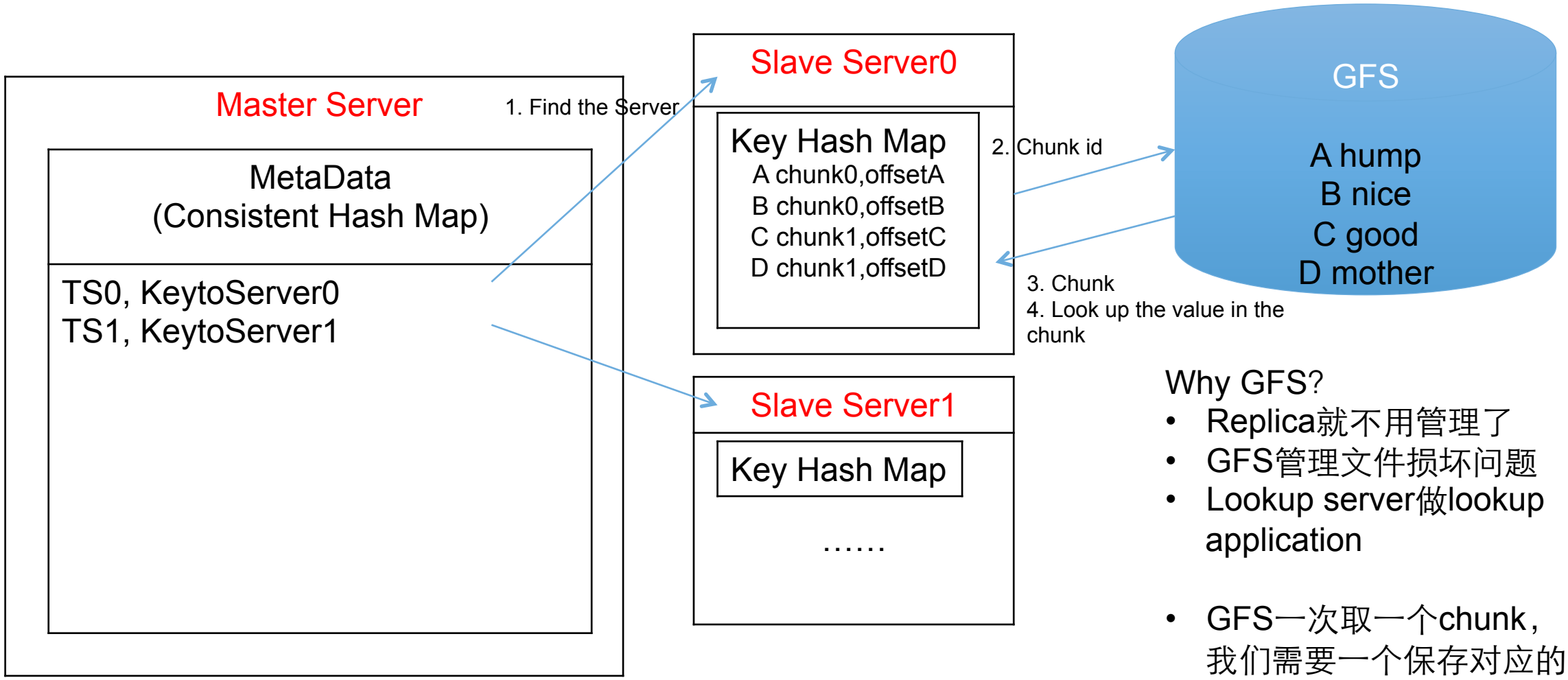
Key
- Master Slave 模型
- Consistent Hash

Disk
- 没有内存
- Binary search查询key value

Server
- 有内存
- 一个server 上面可以load key 到内存一个hash map， value 是对应的key value的地址
- Key 在内存里面
- Key Value 存在disk里面

- Question?
  - Why key value 都存在disk里面？不是只存value?

How to support lookup for a file on disk



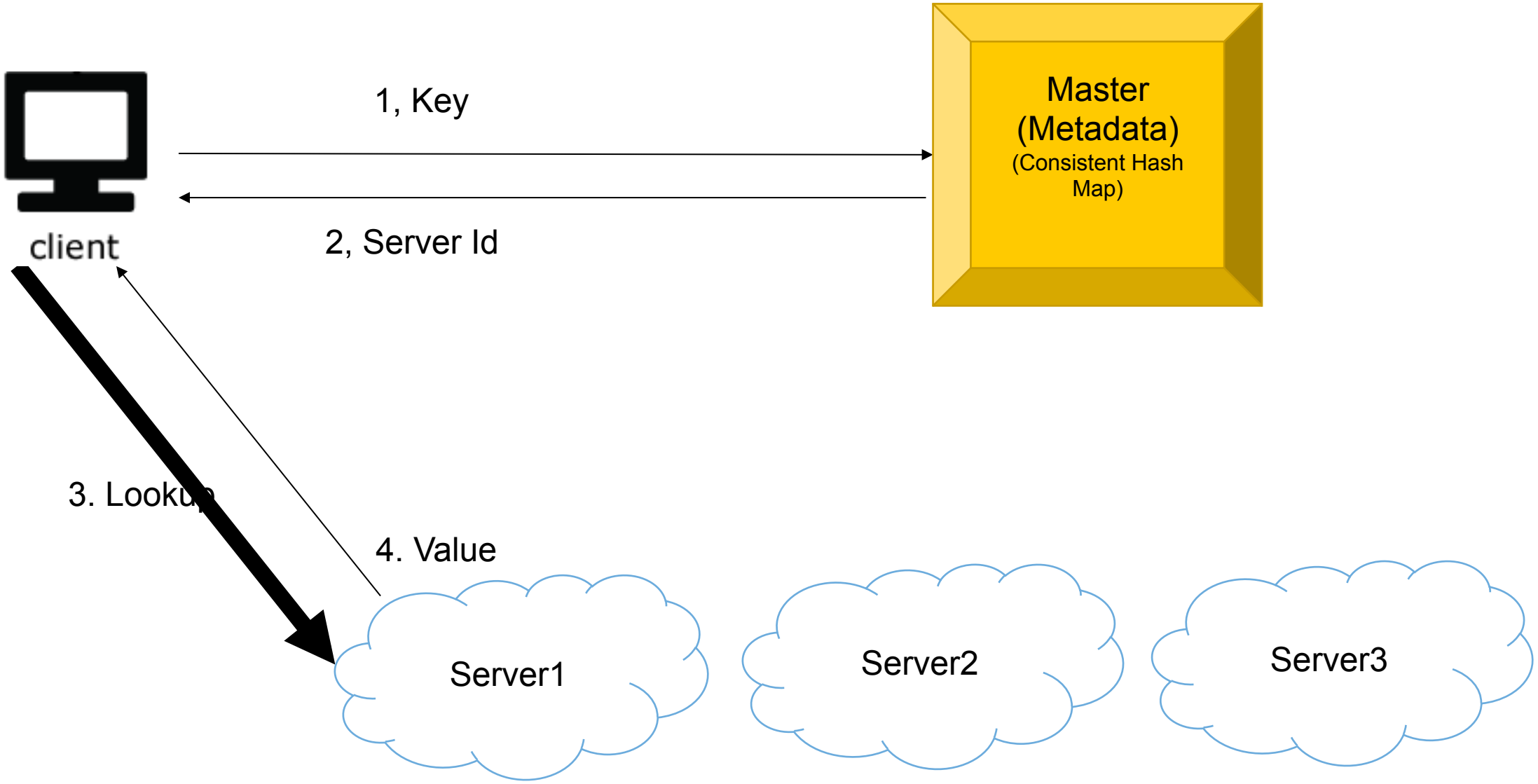
Why GFS?

- Replica就不用管理了
- GFS管理文件损坏问题
- Lookup server做lookup application

- GFS一次取一个chunk，我们需要一个保存对应的值在chunk上面的位置

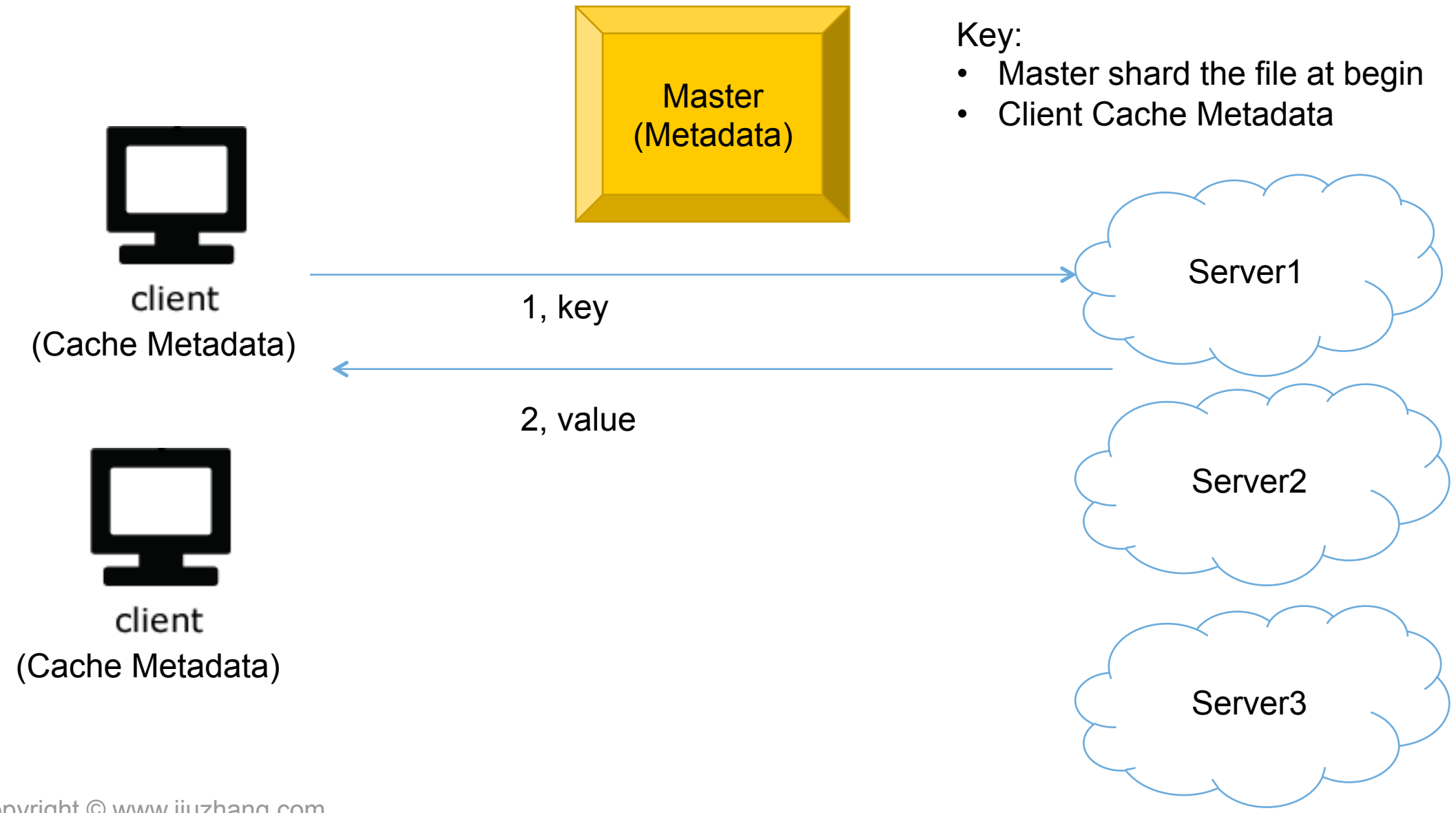
Interviewer: How to look up a key?

Interview: How to look up a key?



Interviewer: Master is bottle neck?

Interview: How to look up a key?



Summary of Lookup Service

- Design
  - Client + Master + Server
- Client
  - Look up
  - MetaData
- Server
  - Maintain the Data (Key value pairs)
- Master
  - Shard the file
  - Maintain the MetaData
  - Manage the servers health
- Key
  - Shard + Consistent Hash
  - Master + Slave

Summary of Today

- Distributed File System
  - 怎么有效存储数据?
    - Store
    - Read
    - Write
    - Failure and Recovery（加分）
- Lookup Service
  - 怎么把已学的东西运用
    - Master
    - Client
    - Server
    - How to connect to GFS
- Key Point
  - Master Slave (Master bottle neck)
  - Failure Detect: checksum, heart beat
  - Recovery: Ask master for help
  - Consistent Hash

参考阅读

- One Slave Server has 1000 G data.
- LookUp Service Application
  - Web Page
    - 100,000,000,000 pages * 10 versions per page * 20KB / Version
    - 20 PB data
  - Google Maps
    - 100 TB satellite image data
- 参考文献
- https://www.cs.rutgers.edu/~pxk/417/notes/23-lookup.html
- http://michaelnielsen.org/blog/consistent-hashing/
- http://www.paperplanes.de/2011/12/9/the-magic-of-consistent-hashing.html

、