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

课程版本 v6.0　　本节主讲人：北丐老师

版权声明：九章课程不允许录像，否则将追究法律责任，赔偿损失

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？

Vs

Overview 谷歌三剑客

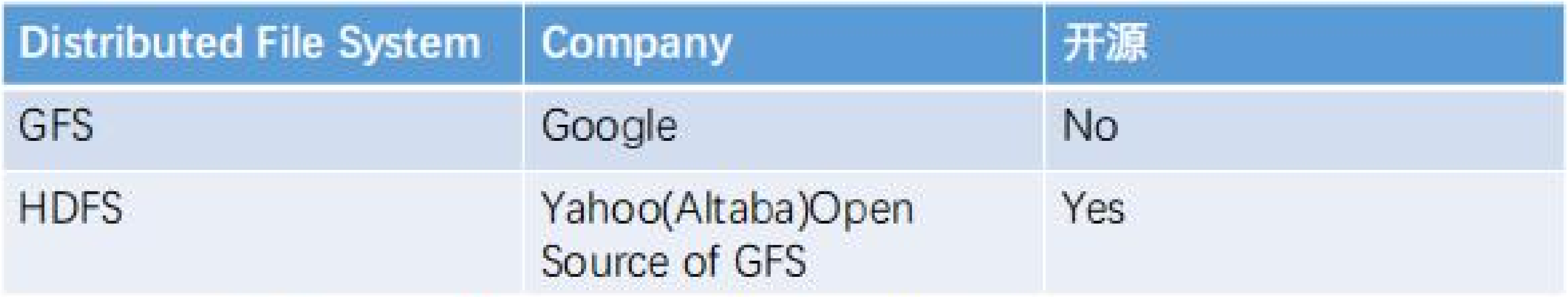
- Distributed File System (Google File System)
  - 怎么有效存储数据？
  - No SQL 底层需要一个文件系统
- Map Reduce
  - 怎么快速处理数据？
- Bigtable = No-SQL DataBase
  - 怎么连接底层存储和上层数据

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

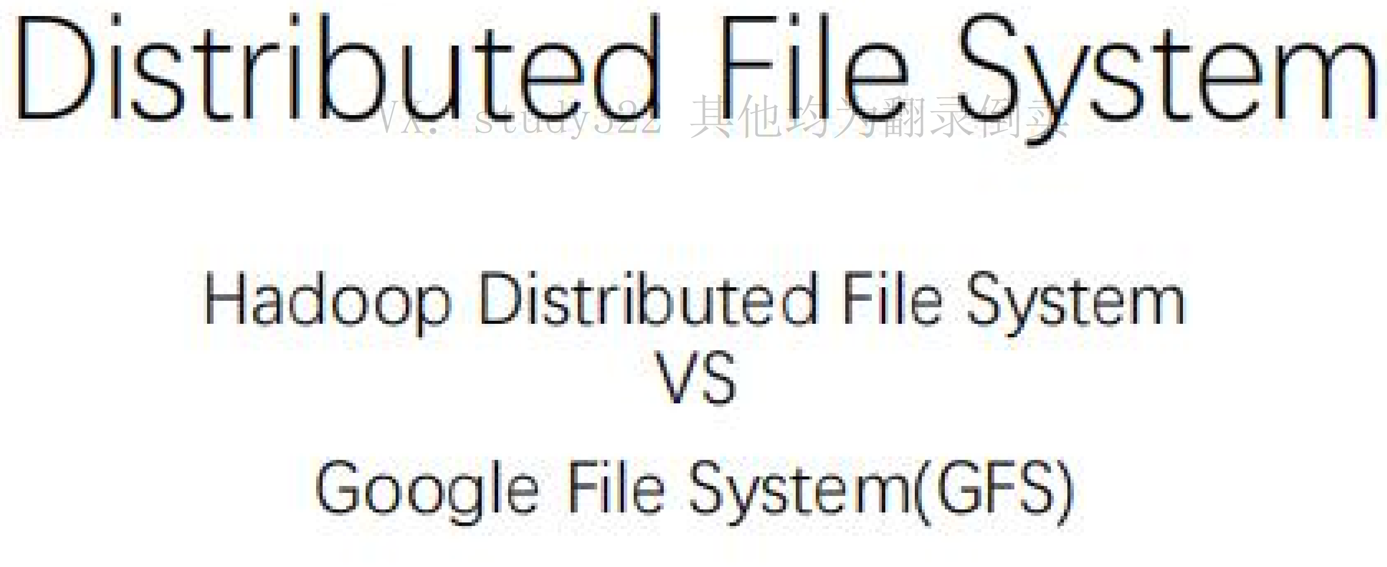
1. Google，Microsoft面试可能会考到.

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	Company	开源
GFS	Google	No
HDFS	Yahoo(Altaba)Open Source of GFS	Yes



Distributed File System

Hadoop Distributed File System
VS
Google File System(GFS)

Overview

1. 按照4S分析
   - **S**cenario 场景分析
   - **S**ervice 服务
   - **S**torage 存储
   - **S**cale 升级优化

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？

Service 服务

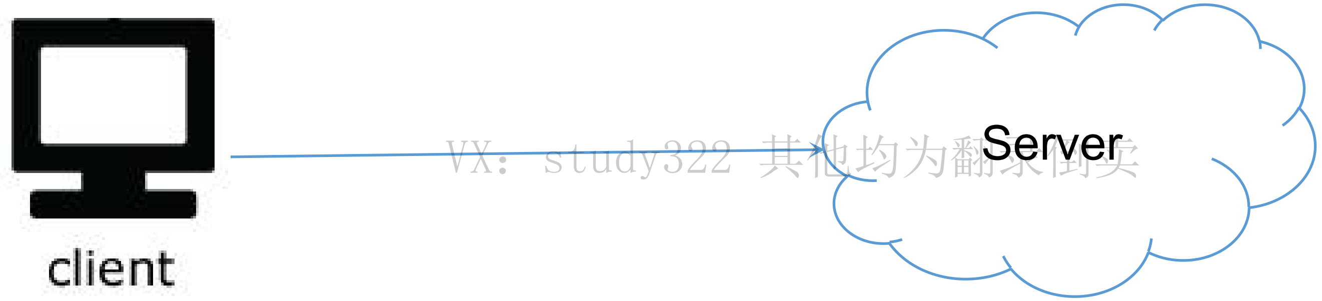
Service 服务

Client

+

Server

Service 服务



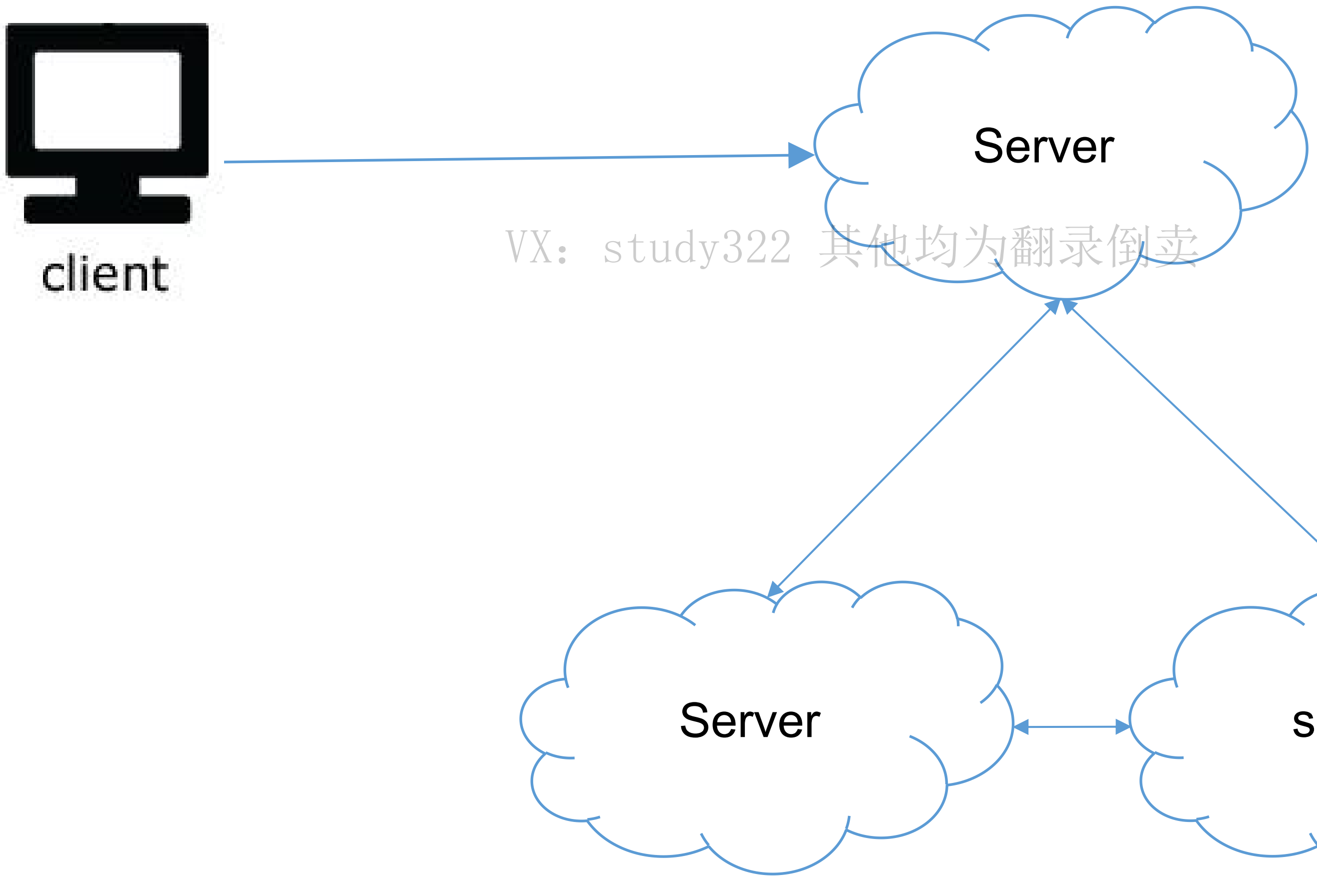
多台机器怎么沟通？

社会主义 or 资本主义

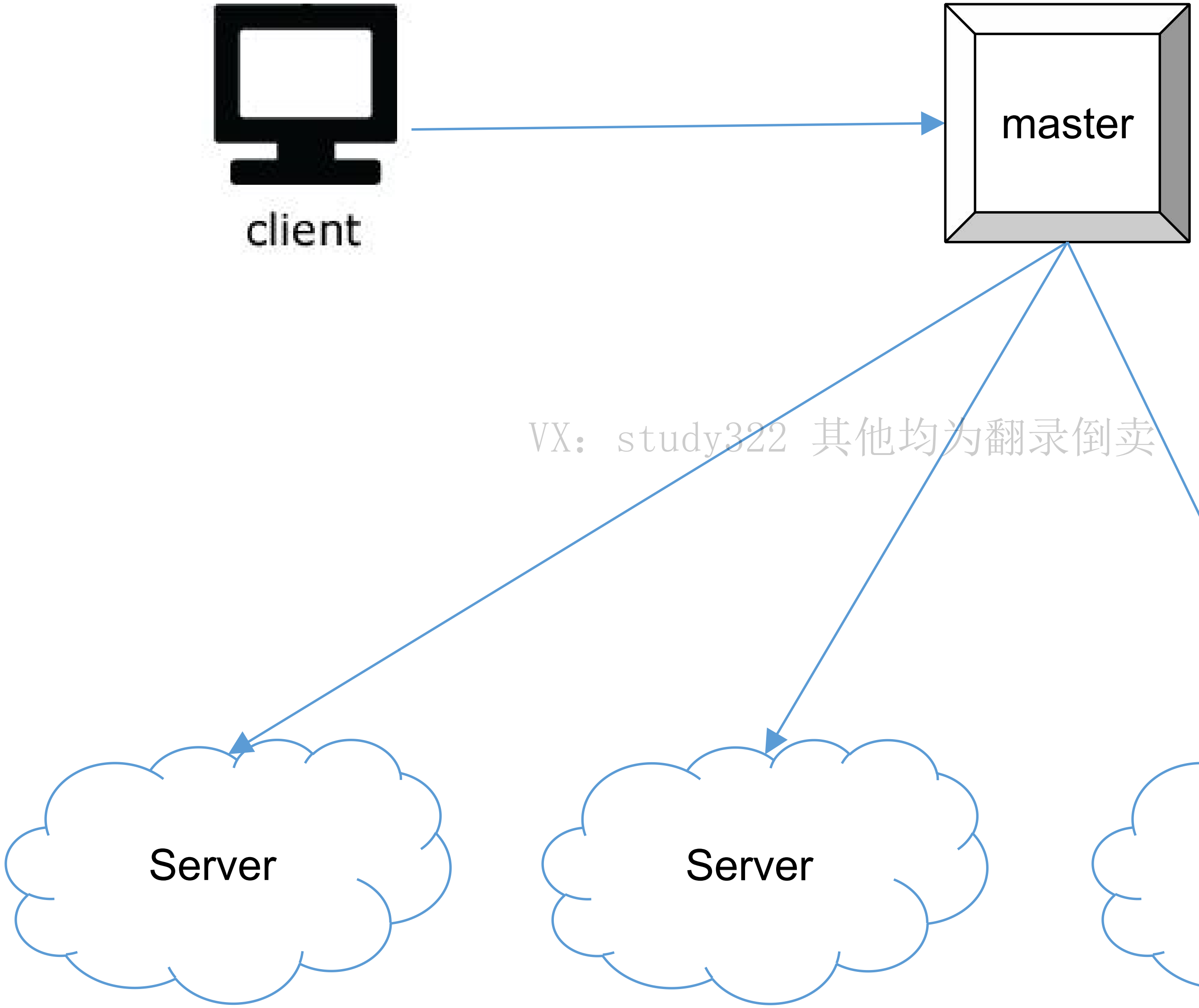
Service 服务

Peer to peer
谁也看不惯谁

社会主义



Service 服务



Master + Slave
我们听老大的

资本主义
好比"大哥和他的小弟"

what is master?
Master 也是一台机器

Service 服务

- Peer 2 Peer
  - 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

数据如何存储

Storage 存储

- 大文件存在哪？
  - 内存？硬盘？

Storage 存储

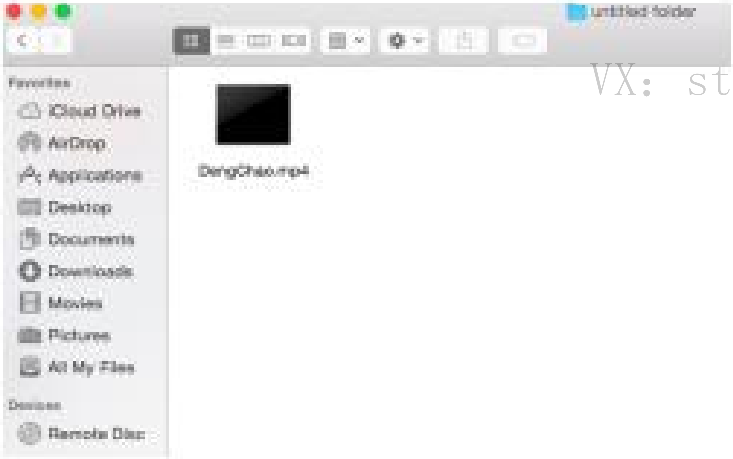
- 大文件存在哪？
  - 内存？硬盘？

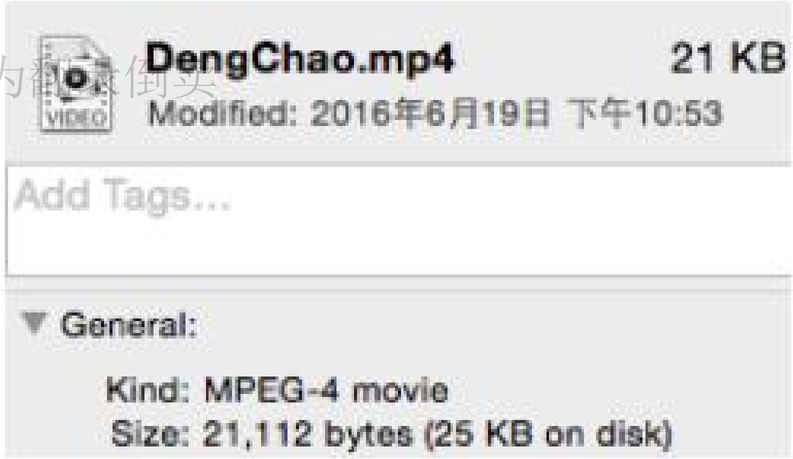
- 怎么存在文件系统里面呢？
  - 怎么设计GFS？

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？





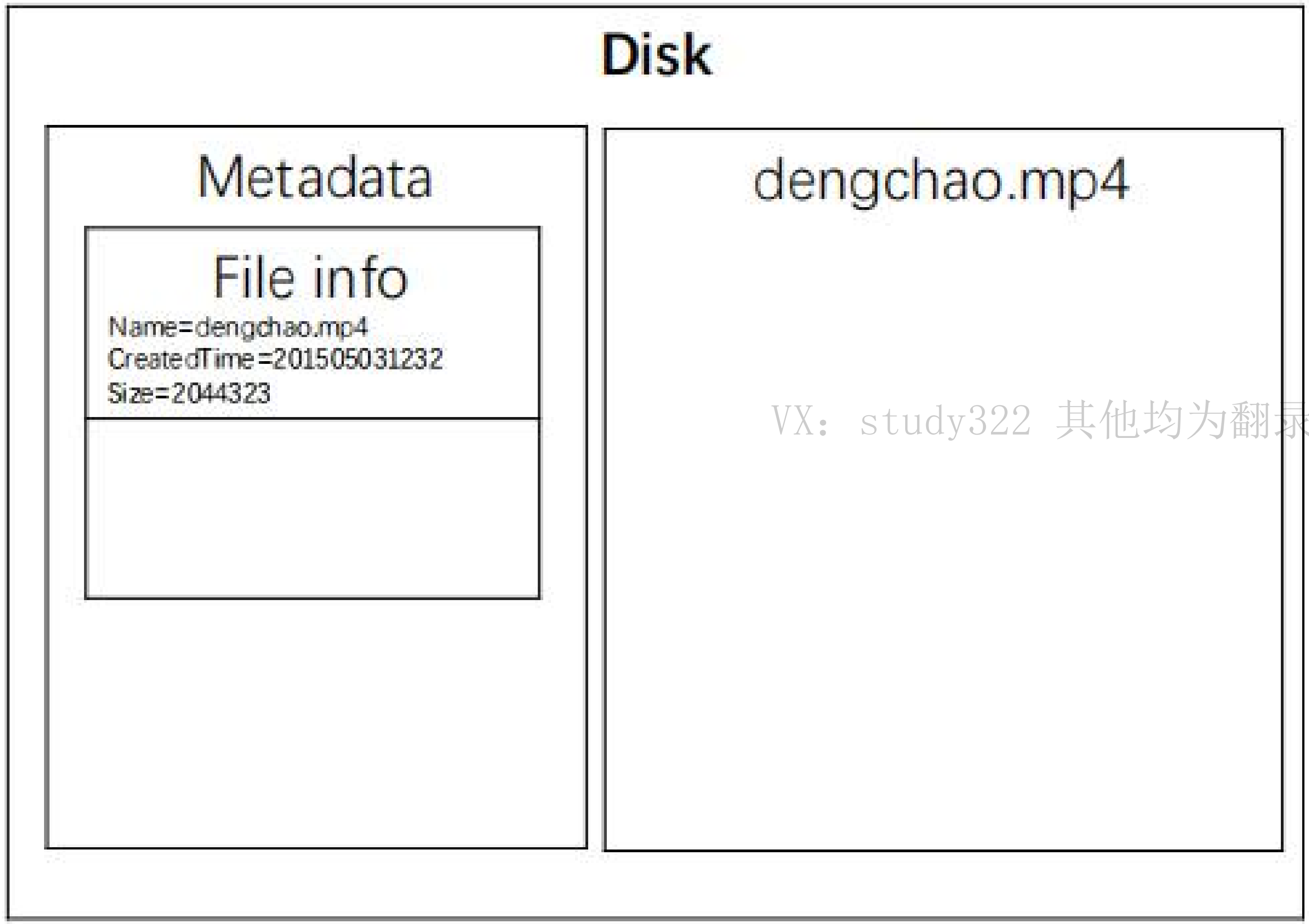
DengChao.mp4 21 KB
Modified: 2016年6月19日 下午10:53

Add Tags...

▼ General:

Kind: MPEG-4 movie
Size: 21,112 bytes (25 KB on disk)

How to save a file in one machine



Disk

Metadata

File info

Name=dengchao.mp4
CreatedTime=201505031232
Size=2044323

dengchao.mp4

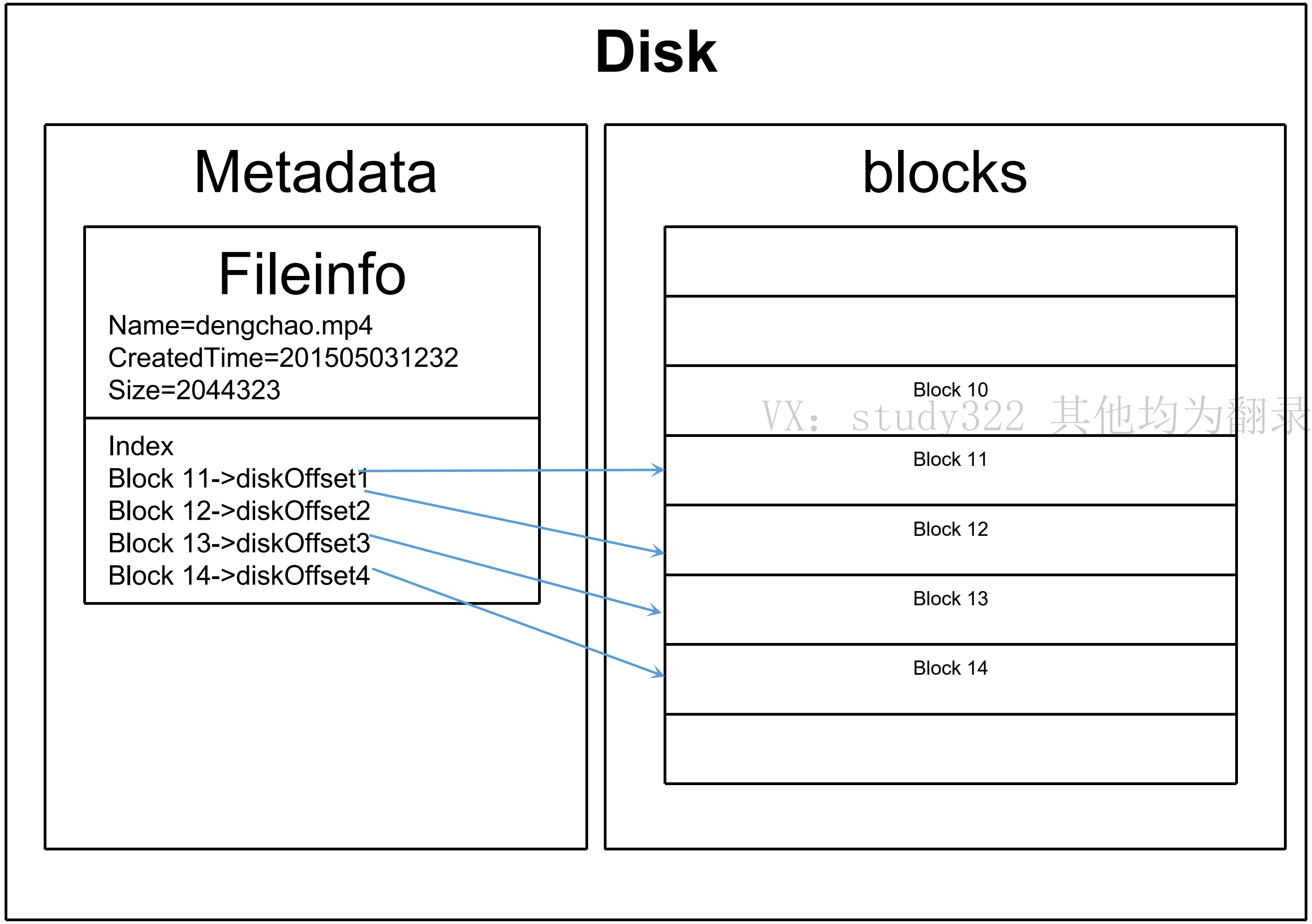
Metadata: 描述"其他数据"而存储的信息

Metadata 访问 常常多于 内容的访问

Metadata 和文件内容是存在一起还是分开?

文件内容是分开存储的呢？还是连续存储的呢?

How to save a file in one machine



Key point

- 1 block = 4096Byte

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

Is block size big enough?

100T(多文件)

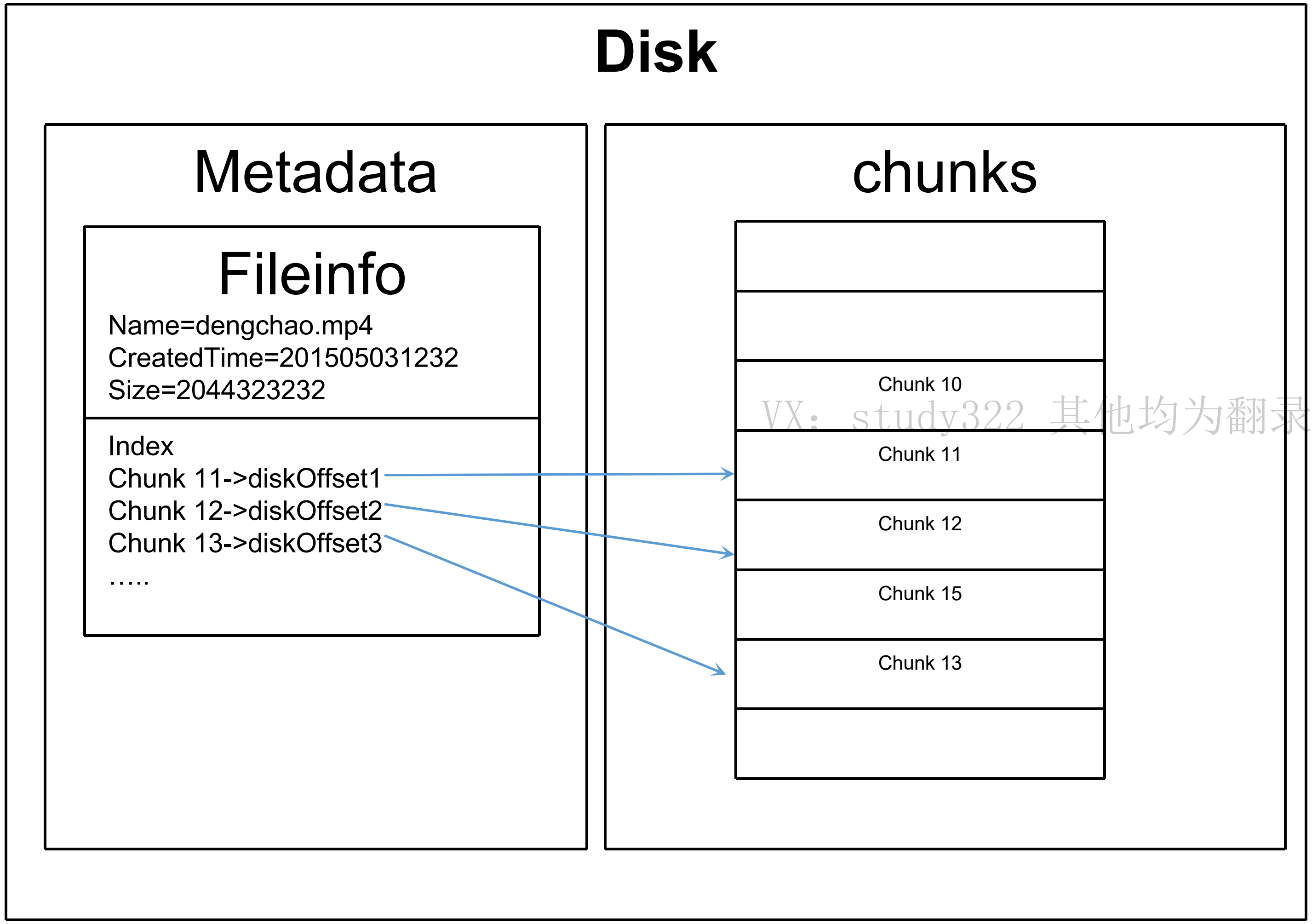
=100*1000G

=100*1000*1000M

=100*1000*1000*1000K

=100*1000*1000*1000block

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



Key point

- 1 chunk= 64M
  = 64*1024K

Advantage

- Reduce size of metadata
- Reduce traffic

Disadvantage

- Waste space for small files

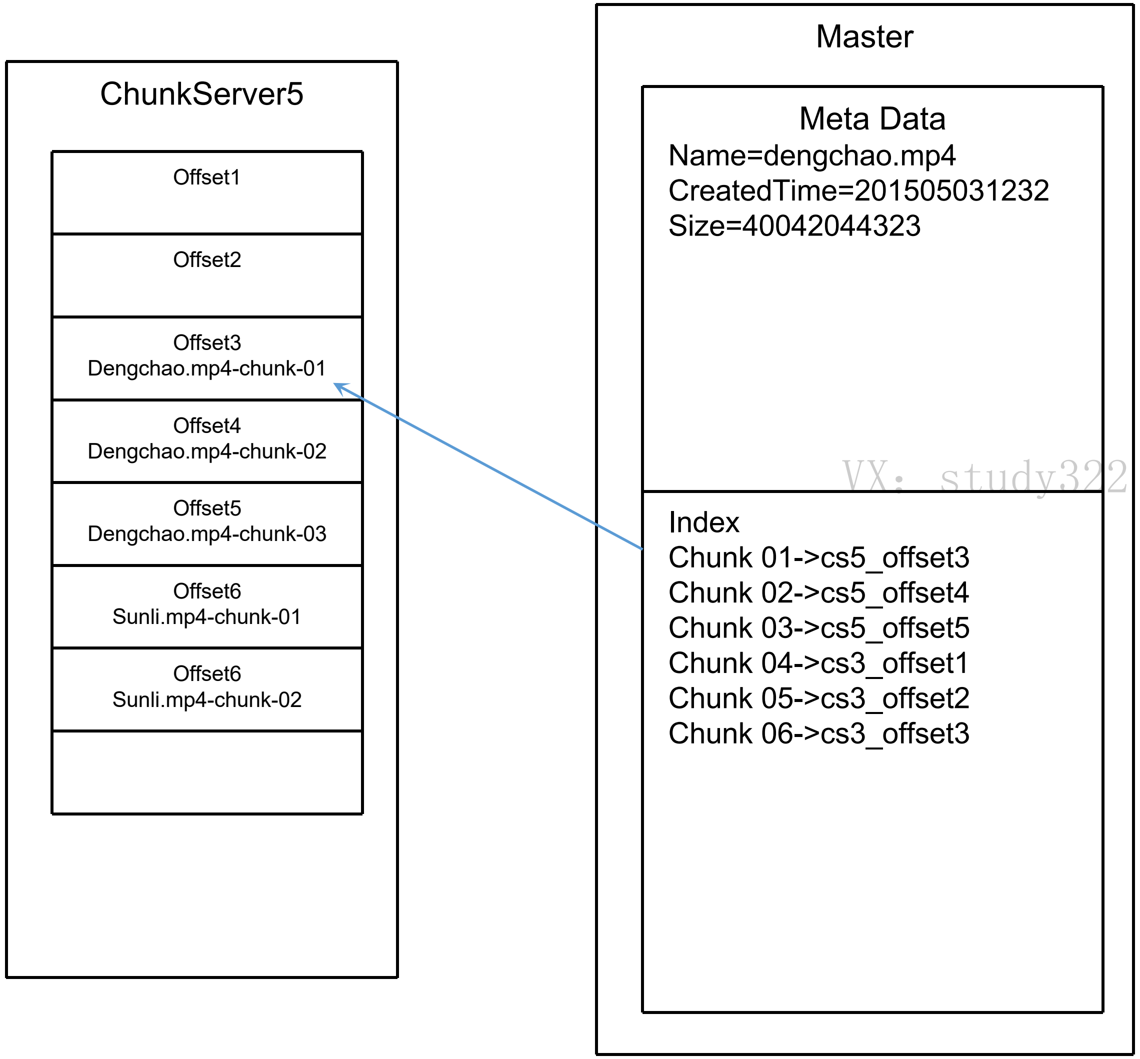
Interviewer: How to save extra-large file in several machine？

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



ChunkServer5

Offset1

Offset2

Offset3
Dengchao.mp4-chunk-01

Offset4
Dengchao.mp4-chunk-02

Offset5
Dengchao.mp4-chunk-03

Offset6
Sunli.mp4-chunk-01

Offset6
Sunli.mp4-chunk-02

Master

Meta Data
Name=dengchao.mp4
CreatedTime=201505031232
Size=40042044323

Index
Chunk 01->cs5_offset3
Chunk 02->cs5_offset4
Chunk 03->cs5_offset5
Chunk 04->cs3_offset1
Chunk 05->cs3_offset2
Chunk 06->cs3_offset3

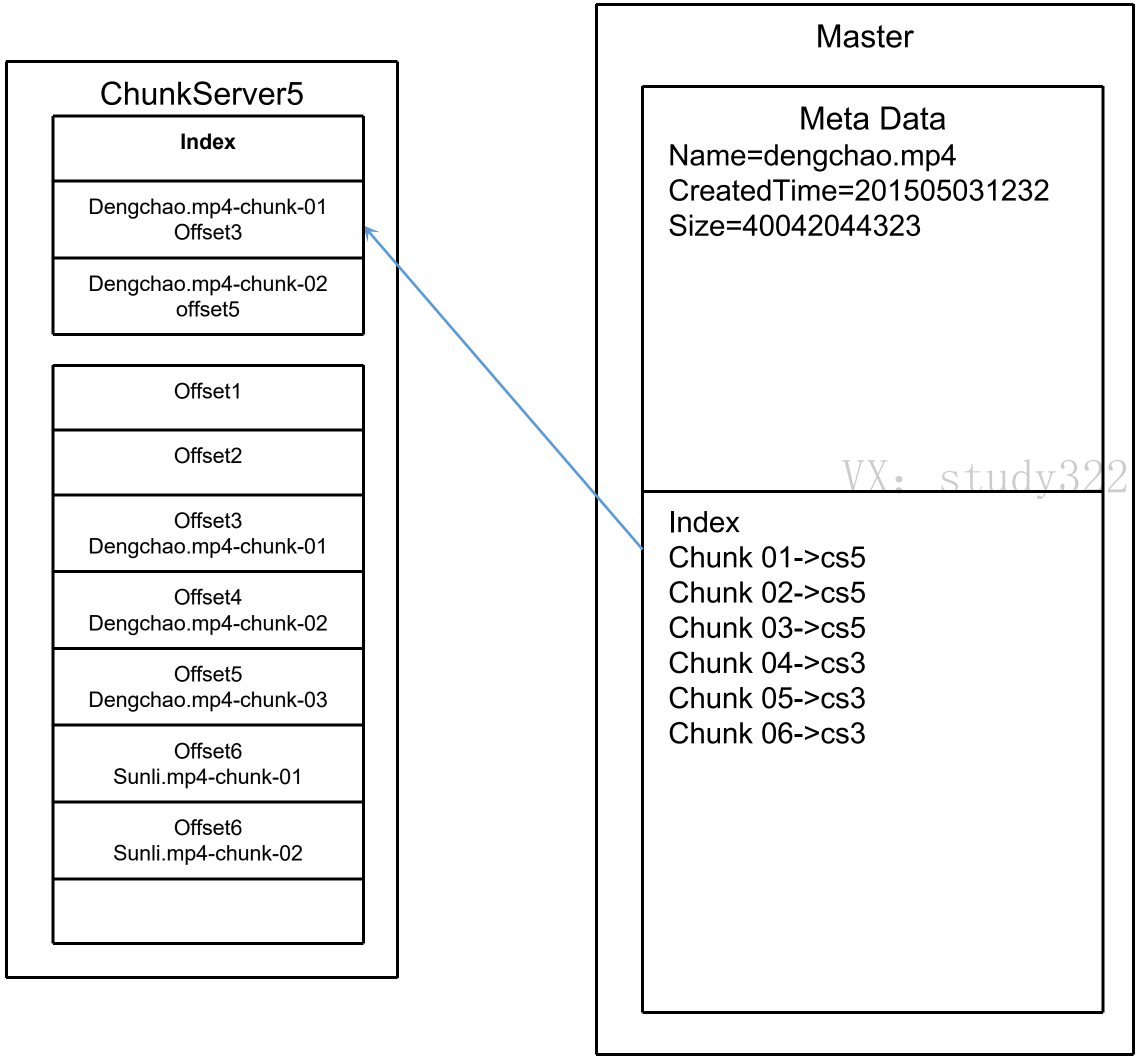
Key point

- One master + many ChunkServers

Slave Servers = Chunk Servers

每个chunk的Offset偏移量可不可以不存在master上面？

Scale about the Storage



ChunkServer5

Index

Dengchao.mp4-chunk-01
Offset3

Dengchao.mp4-chunk-02
offset5

Offset1

Offset2

Offset3
Dengchao.mp4-chunk-01

Offset4
Dengchao.mp4-chunk-02

Offset5
Dengchao.mp4-chunk-03

Offset6
Sunli.mp4-chunk-01

Offset6
Sunli.mp4-chunk-02

Master

Meta Data
Name=dengchao.mp4
CreatedTime=201505031232
Size=40042044323

Index
Chunk 01->cs5
Chunk 02->cs5
Chunk 03->cs5
Chunk 04->cs3
Chunk 05->cs3
Chunk 06->cs3

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
  - **S**cale 升级优化

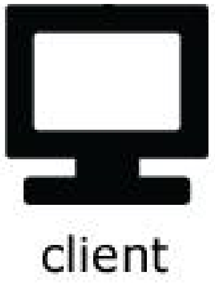
- 理清楚work solution

- 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？



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。

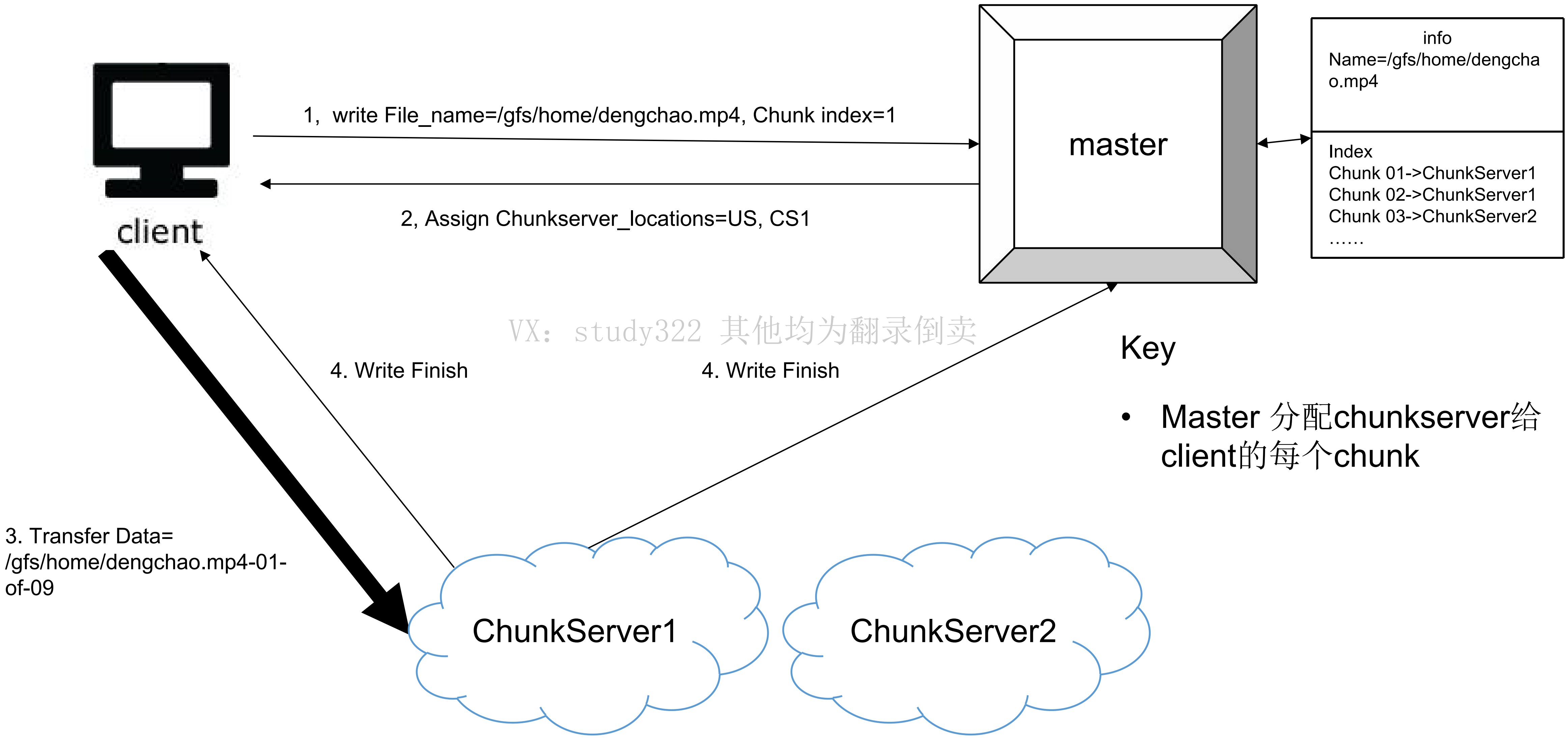
- 如果是分成多份多次写入，那么每一份的大小?
  - 文件本来是按照Chunk来存储的，所以传输单位也是Chunk

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?


client
1, write File_name=/gfs/home/dengchao.mp4, Chunk index=1
2, Assign Chunkserver_locations=US, CS1
master
4. Write Finish
4. Write Finish
3. Transfer Data=
/gfs/home/dengchao.mp4-01-of-09
ChunkServer1
ChunkServer2

info
Name=/gfs/home/dengchao.mp4

Index
Chunk 01->ChunkServer1
Chunk 02->ChunkServer1
Chunk 03->ChunkServer2
……

Key

- Master 分配chunkserver给client的每个chunk

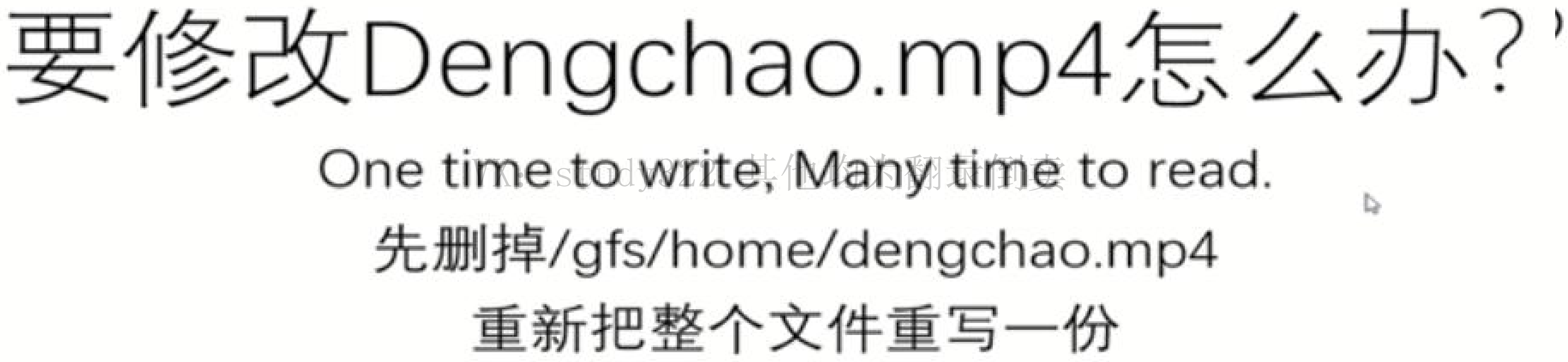
要修改Dengchao.mp4怎么办?

/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?

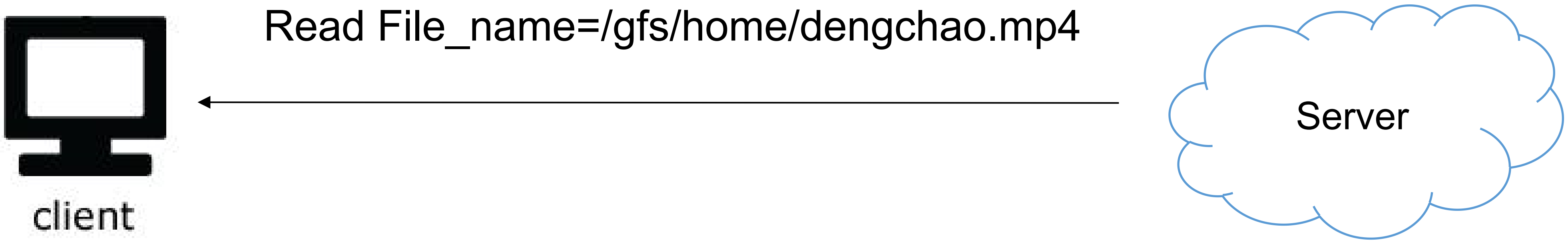
One time to write, Many time to read.

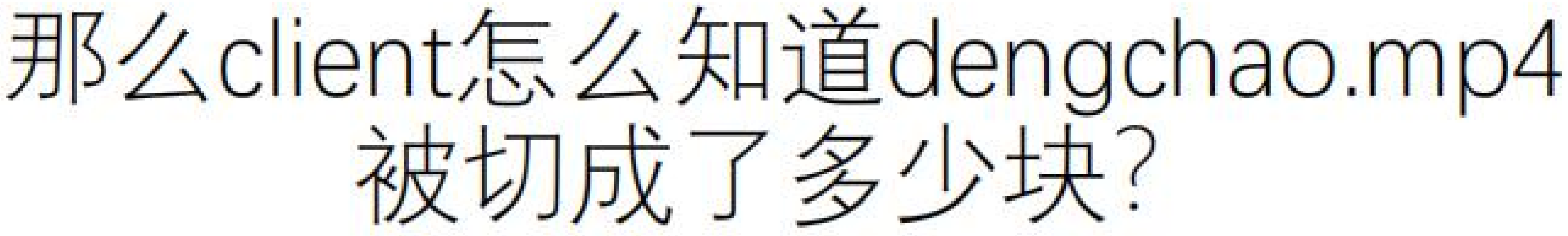
先删掉/gfs/home/dengchao.mp4

重新把整个文件重写一份

Interviewer: How to read a file?

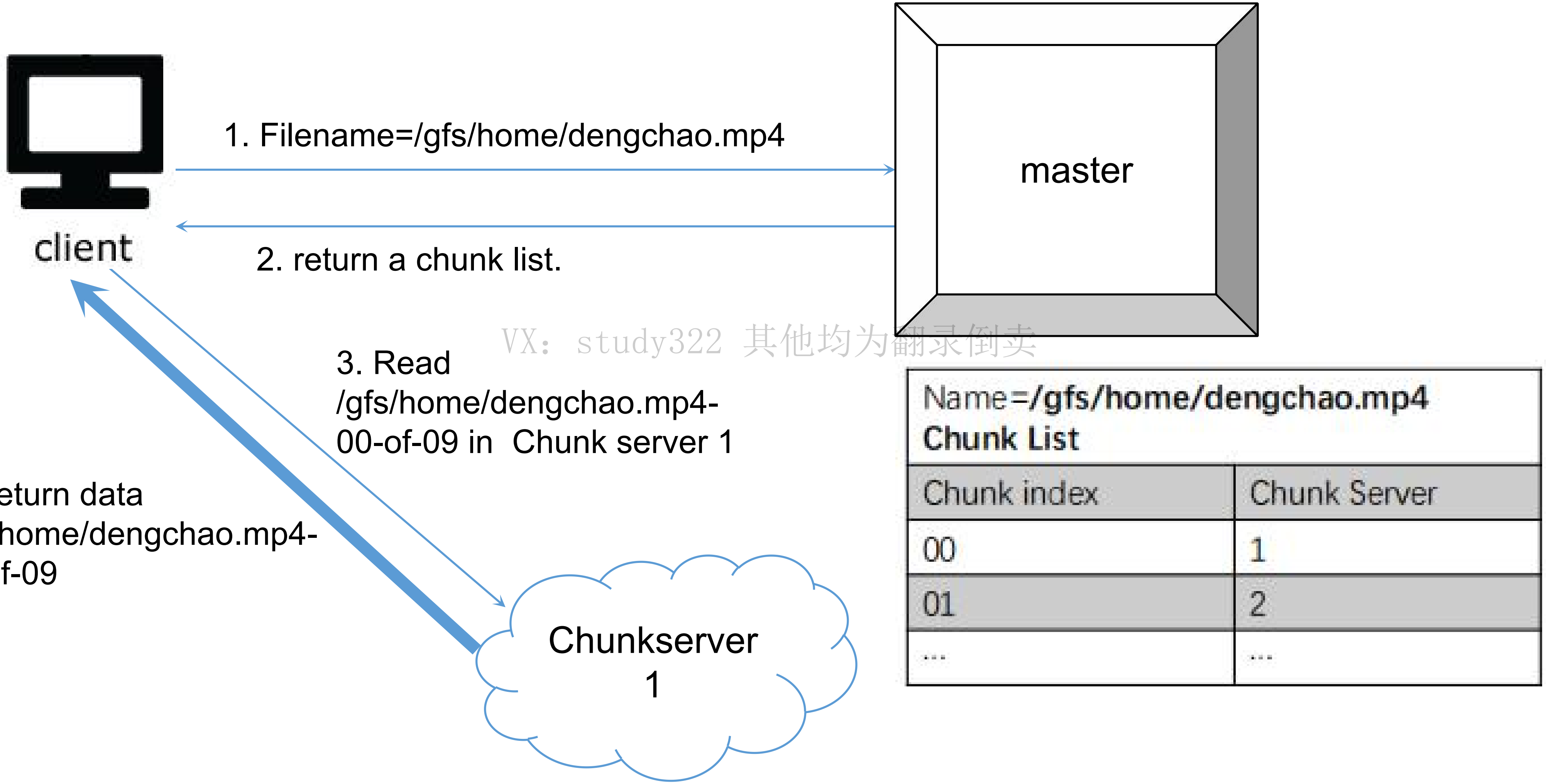
一次读整个文件？
还是拆分成多份多次读入？





那么client怎么知道dengchao.mp4
被切成了多少块?

How to read from a file?



client
1. Filename=/gfs/home/dengchao.mp4
master
2. return a chunk list.
3. Read /gfs/home/dengchao.mp4-00-of-09 in Chunk server 1
4. Return data /gfs/home/dengchao.mp4-00-of-09
Chunkserver 1

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

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

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

Paper: Apache Hadoop Goes Realtime at Facebook

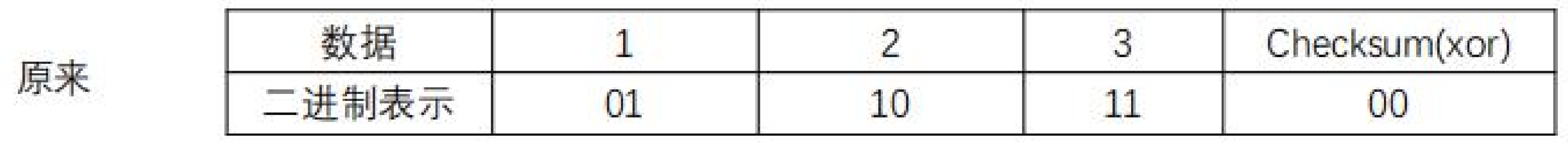
Multi Master

Paper: Paxos Algorithm

Scale about the Failure and Recover

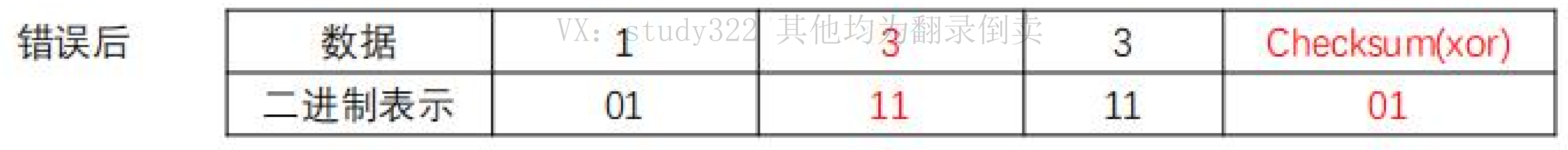
Interviewer: How to identify whether a chunk on the disk is broken?

CheckSum



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

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

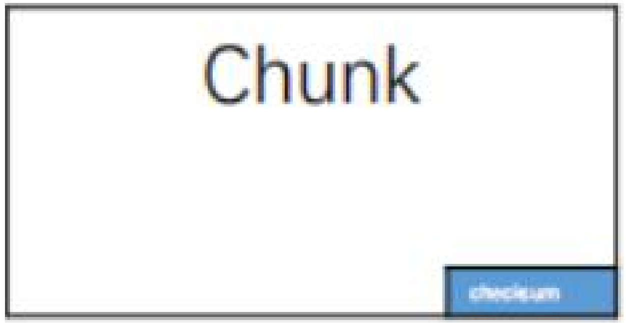
- 1 checksum size?
- 4bytes = 32bit
- 1 chunk = 64MB
- Each block has a checksum
- The size of checksum of 1T file
- 1P/64MB*32bit = 62.5 MB

- Add check sum for blocks is acceptable.

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）

做备份

需要多少个备份？
每个备份放在哪？

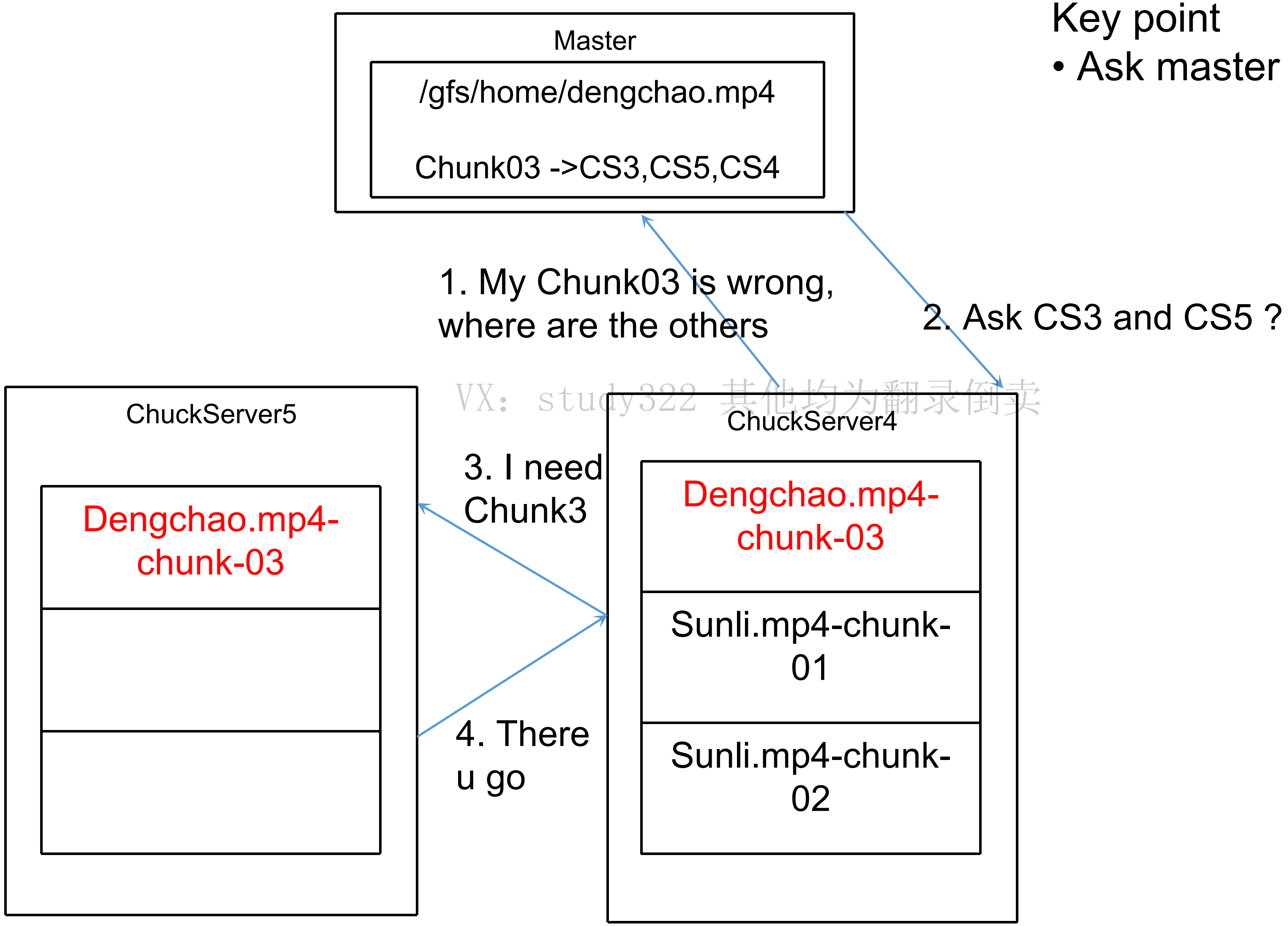
需要多少个备份？
每个备份放在哪？

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）

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 Chunk Server is down?

How to find whether a ChunkServer is down?

Interviewer: HeartBeat.

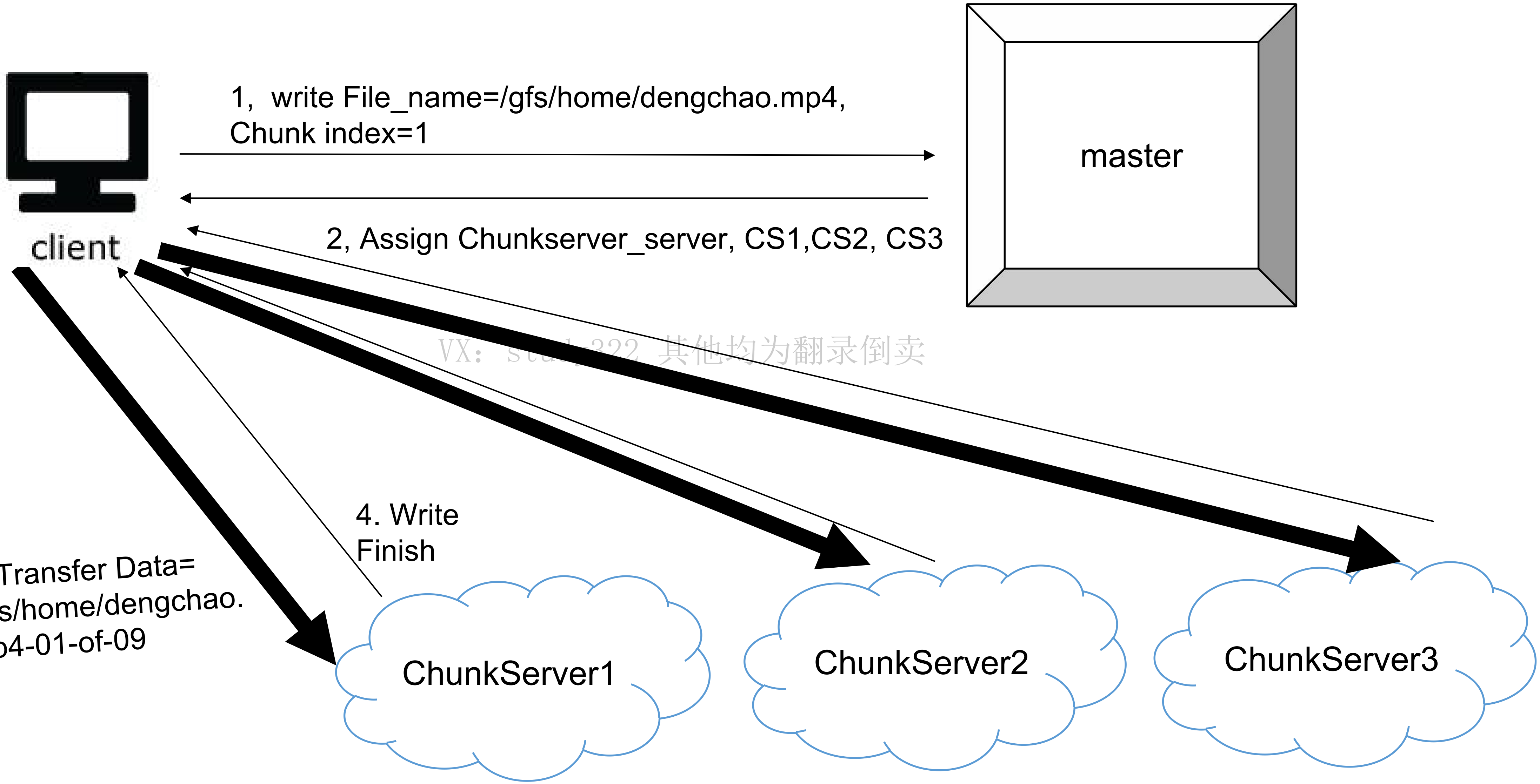
A: master -> chunkservers?

B: chunkservers->master?

Scale about the Write

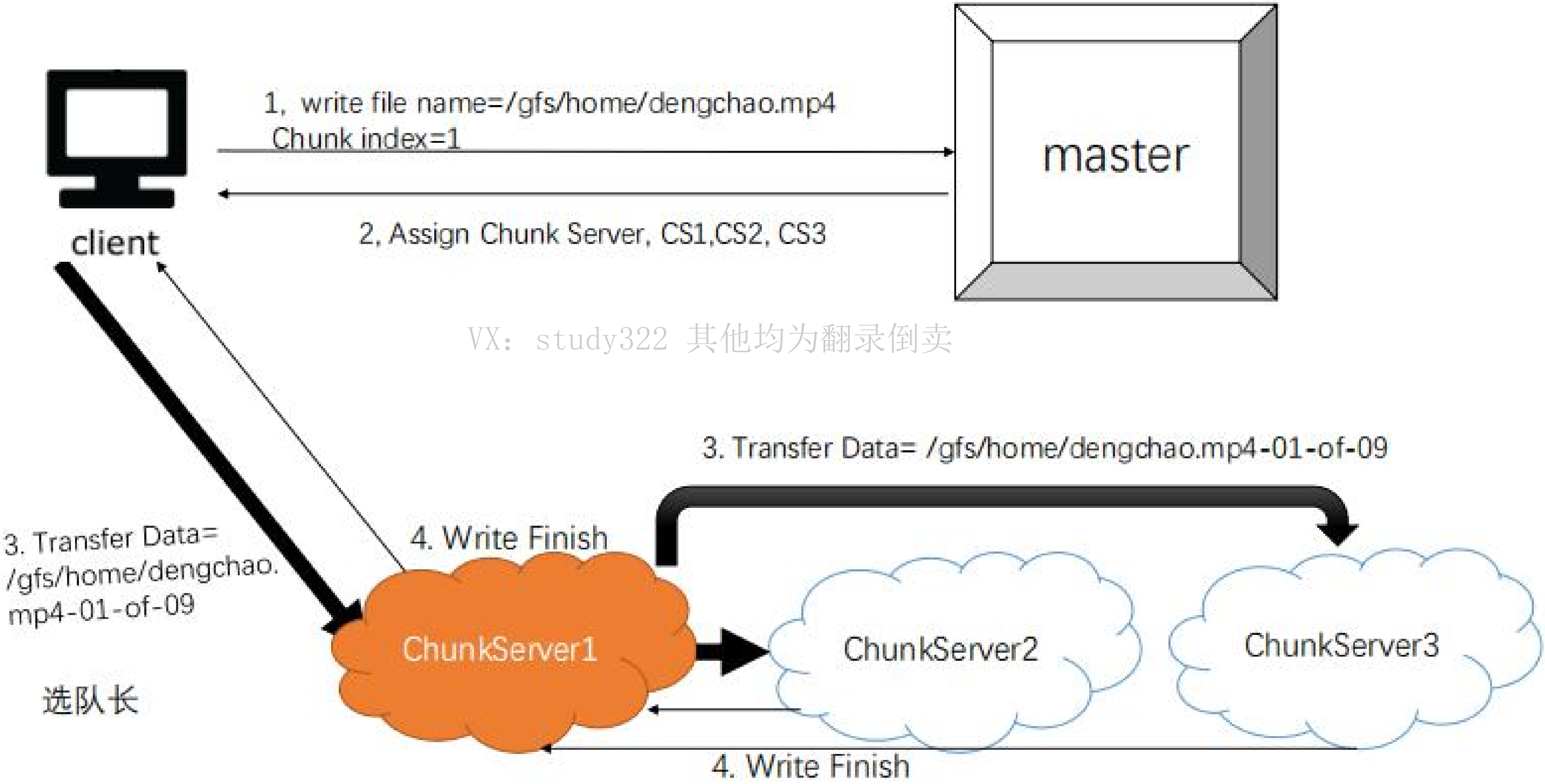
Interviewer: Whether write to only one server is safe?

How to write a file?



Interviewer: How to solve Client bottleneck?

How to solve Client bottleneck?



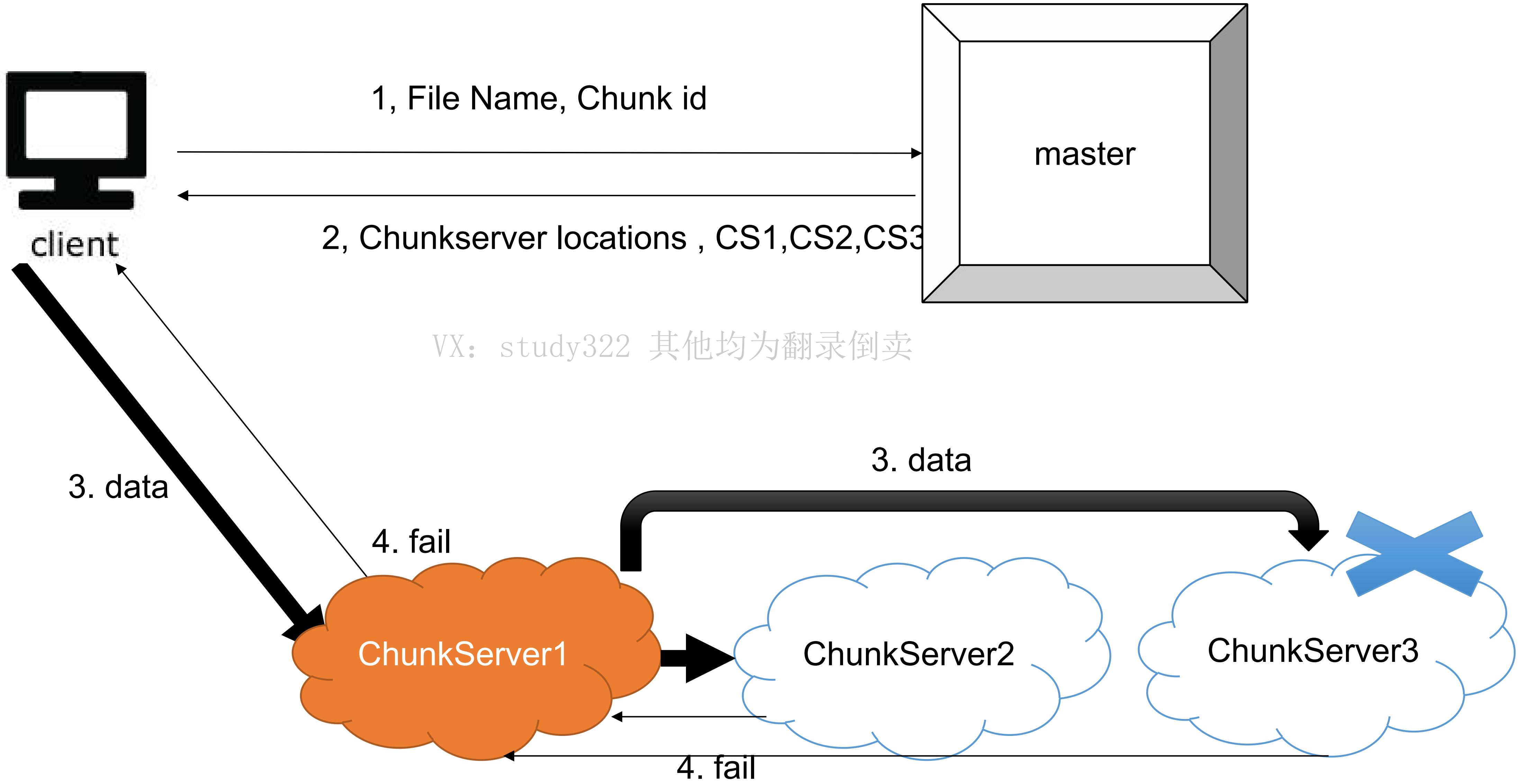
选队长

Interviewer: 怎么样选队长?

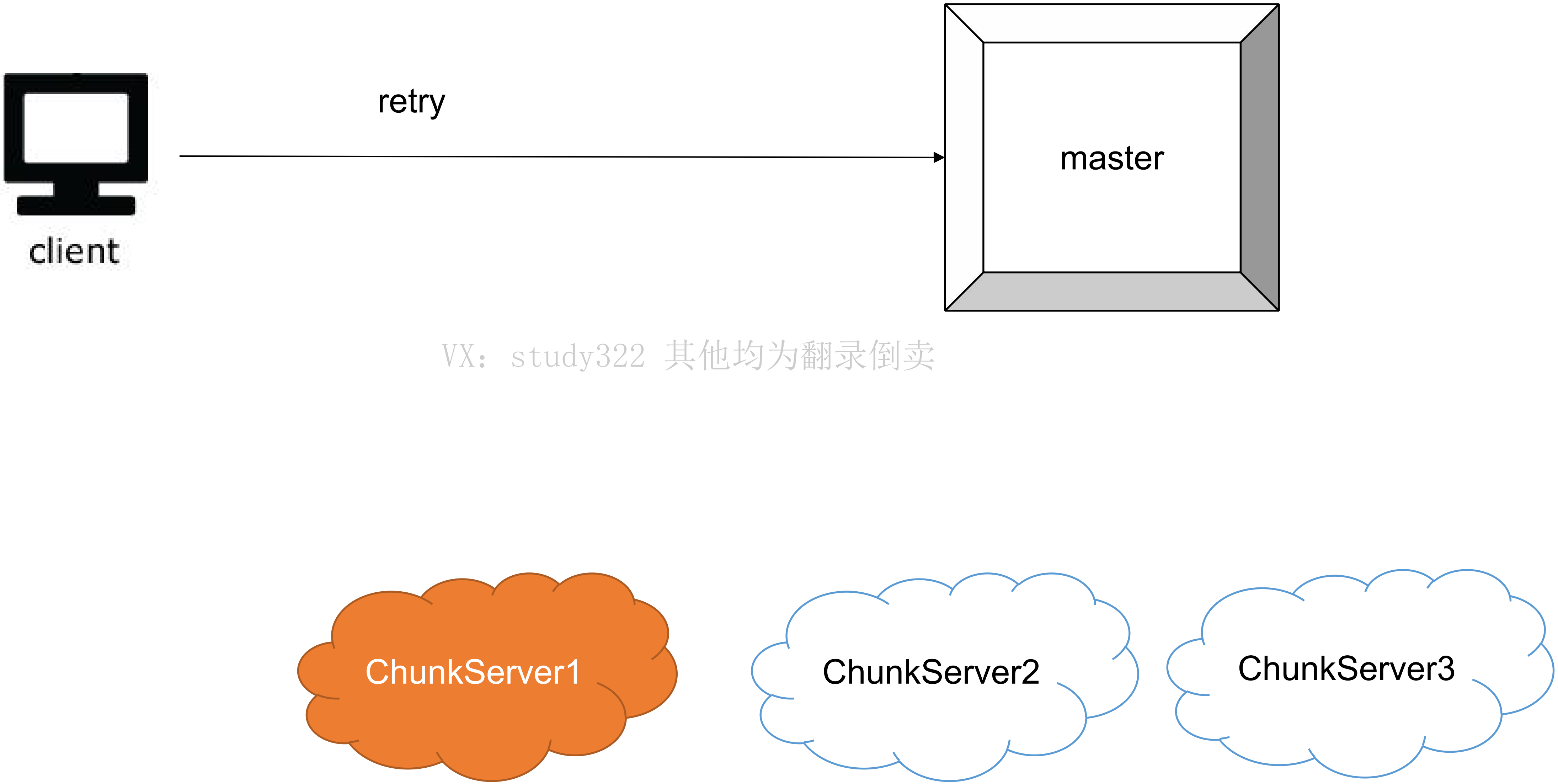
1. 找距离最近的（快）
2. 找现在不干活的（平衡traffic）

Interviewer: How to solve Chunk Server 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

Google onsite non-abstract large scale system design 真题

https://www.jiuzhang.com/qa/627/

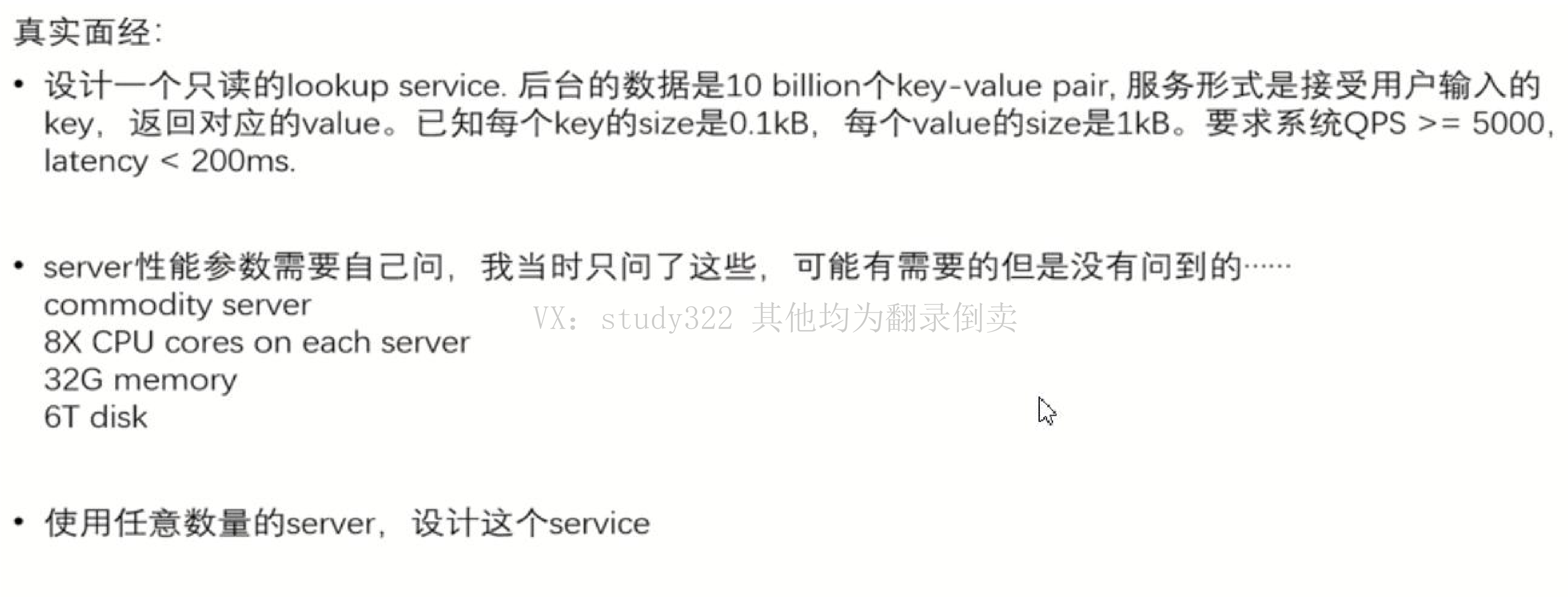
Read More

- 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

GFS实战

设计lookup service

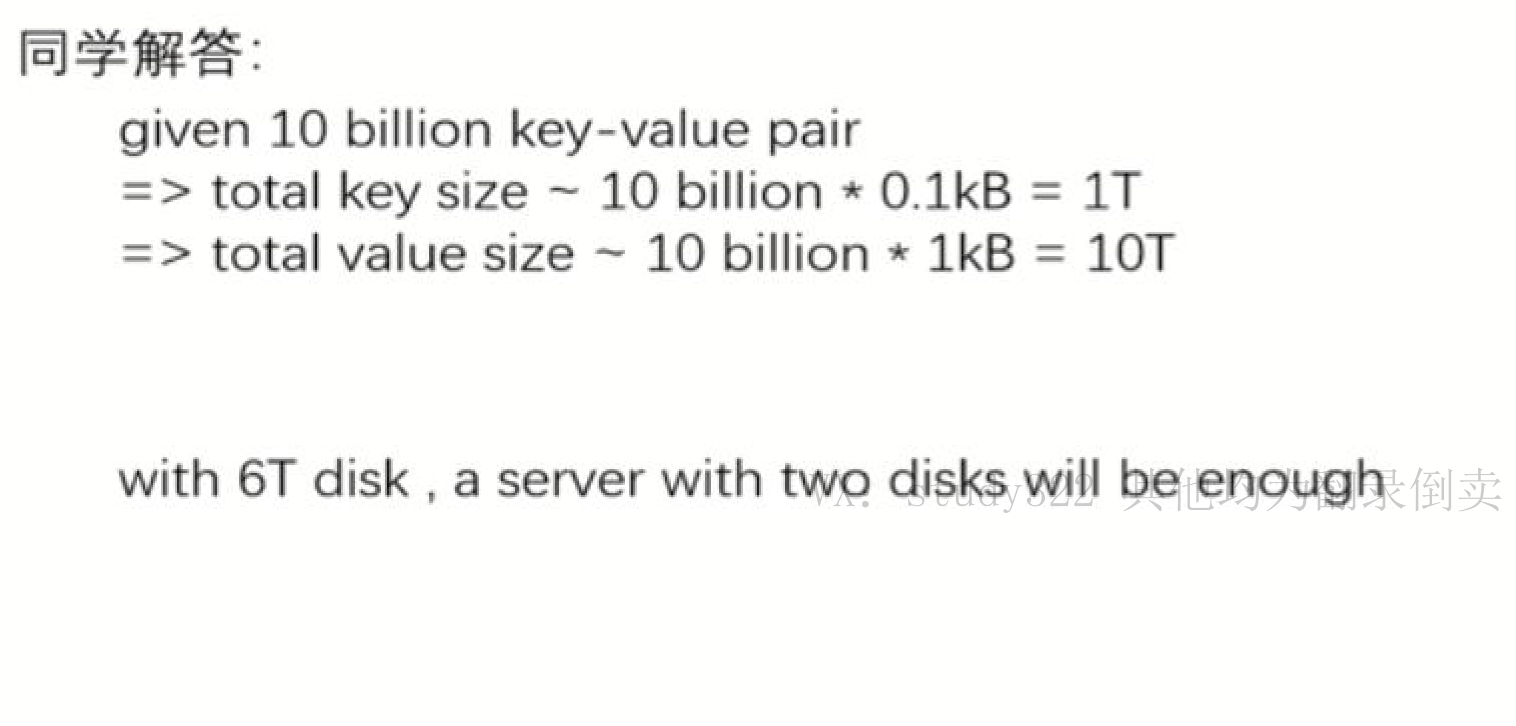


真实面经：

- 设计一个只读的lookup service. 后台的数据是10 billion个key-value pair, 服务形式是接受用户输入的key，返回对应的value。已知每个key的size是0.1kB，每个value的size是1kB。要求系统QPS >= 5000, latency < 200ms.

- server性能参数需要自己问，我当时只问了这些，可能有需要的但是没有问到的……
  commodity server
  8X CPU cores on each server
  32G memory
  6T disk

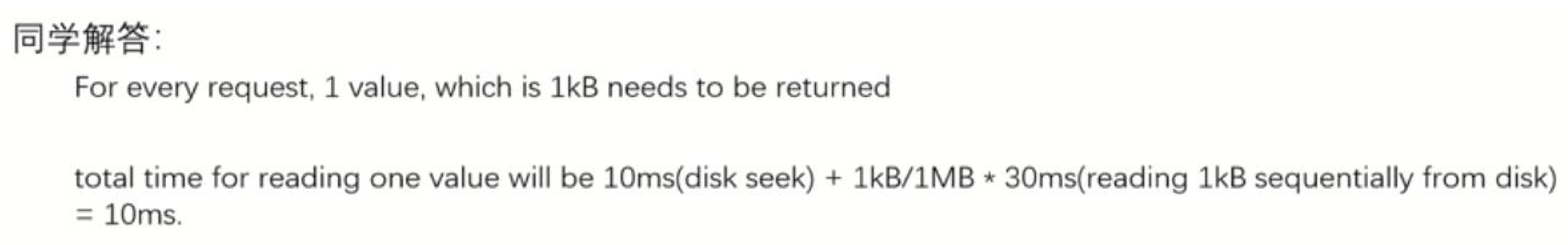
- 使用任意数量的server，设计这个service



同学解答:

given 10 billion key-value pair
=> total key size ~ 10 billion * 0.1kB = 1T
=> total value size ~ 10 billion * 1kB = 10T

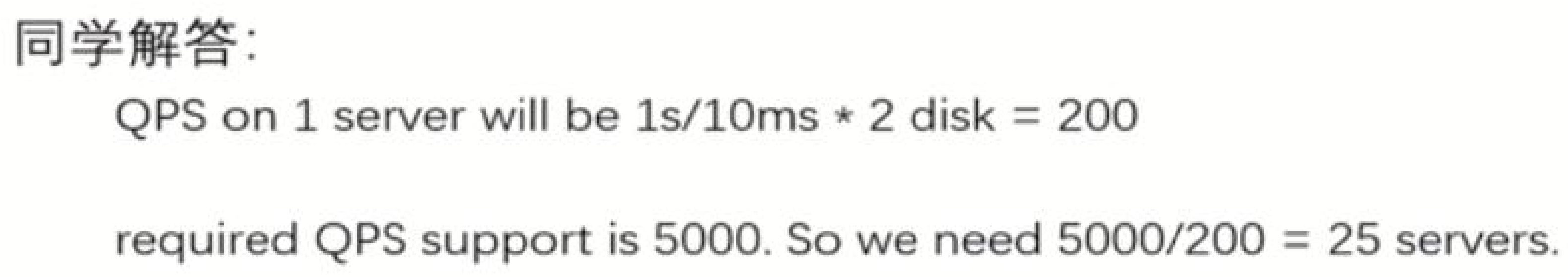
with 6T disk , a server with two disks will be enough



同学解答:

For every request, 1 value, which is 1kB needs to be returned

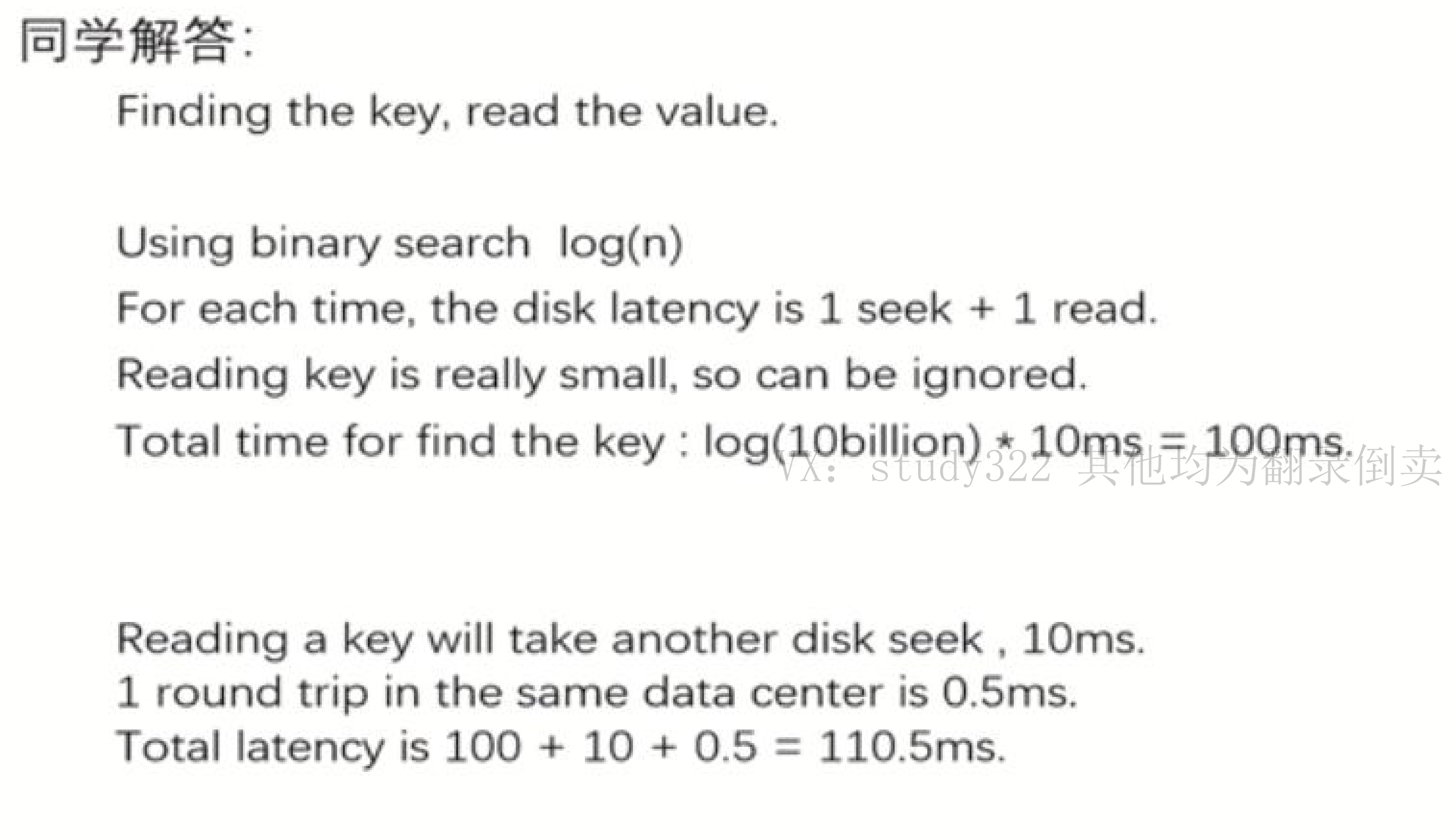
total time for reading one value will be 10ms(disk seek) + 1kB/1MB * 30ms(reading 1kB sequentially from disk) = 10ms.



同学解答:

QPS on 1 server will be 1s/10ms * 2 disk = 200

required QPS support is 5000. So we need 5000/200 = 25 servers.



同学解答:

Finding the key, read the value.

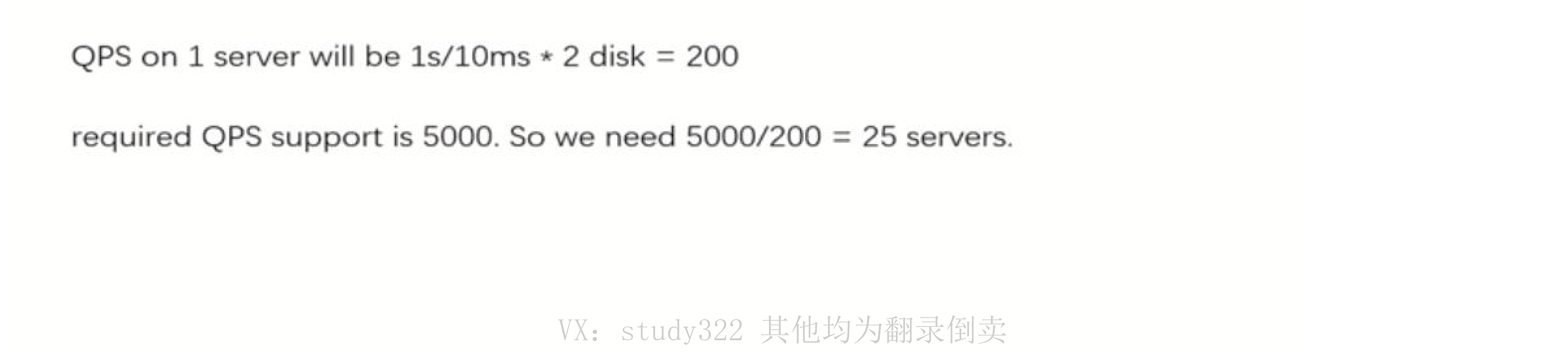
Using binary search log(n)

For each time, the disk latency is 1 seek + 1 read.

Reading key is really small, so can be ignored.

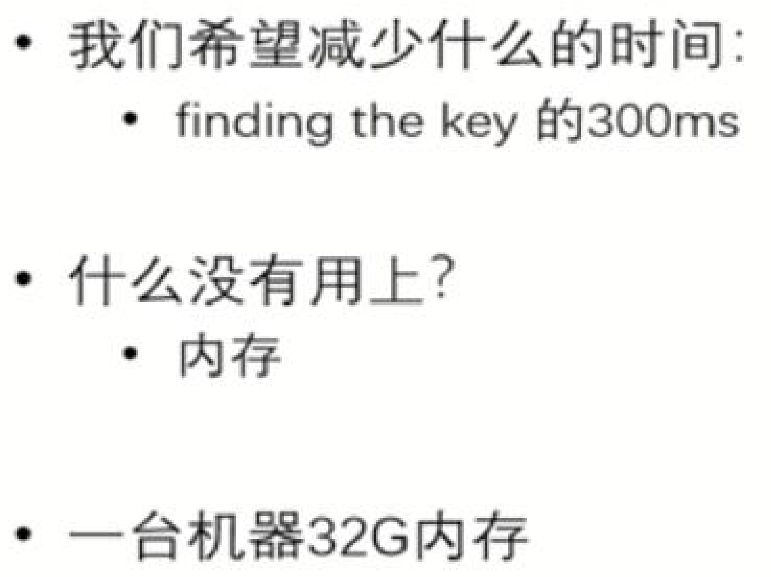
Total time for find the key : log(10billion) * 10ms = 100ms.

Reading a key will take another disk seek , 10ms.
1 round trip in the same data center is 0.5ms.
Total latency is 100 + 10 + 0.5 = 110.5ms.



QPS on 1 server will be 1s/10ms * 2 disk = 200

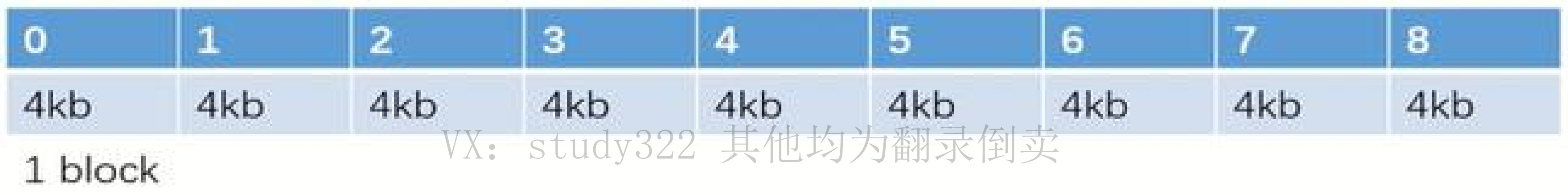
required QPS support is 5000. So we need 5000/200 = 25 servers.



- 我们希望减少什么的时间:
  - finding the key 的300ms
- 什么没有用上?
  - 内存
- 一台机器32G内存
  - 40台机器就可在内存中装下所有的<key, 硬盘地址>这样的键值对
  - 内存中二分查找，30次，时间可以忽略不计
  - so total latency is 10 + 0.5 = 10.5ms

GFS常见问题解答

问：什么是文件系统中的block?



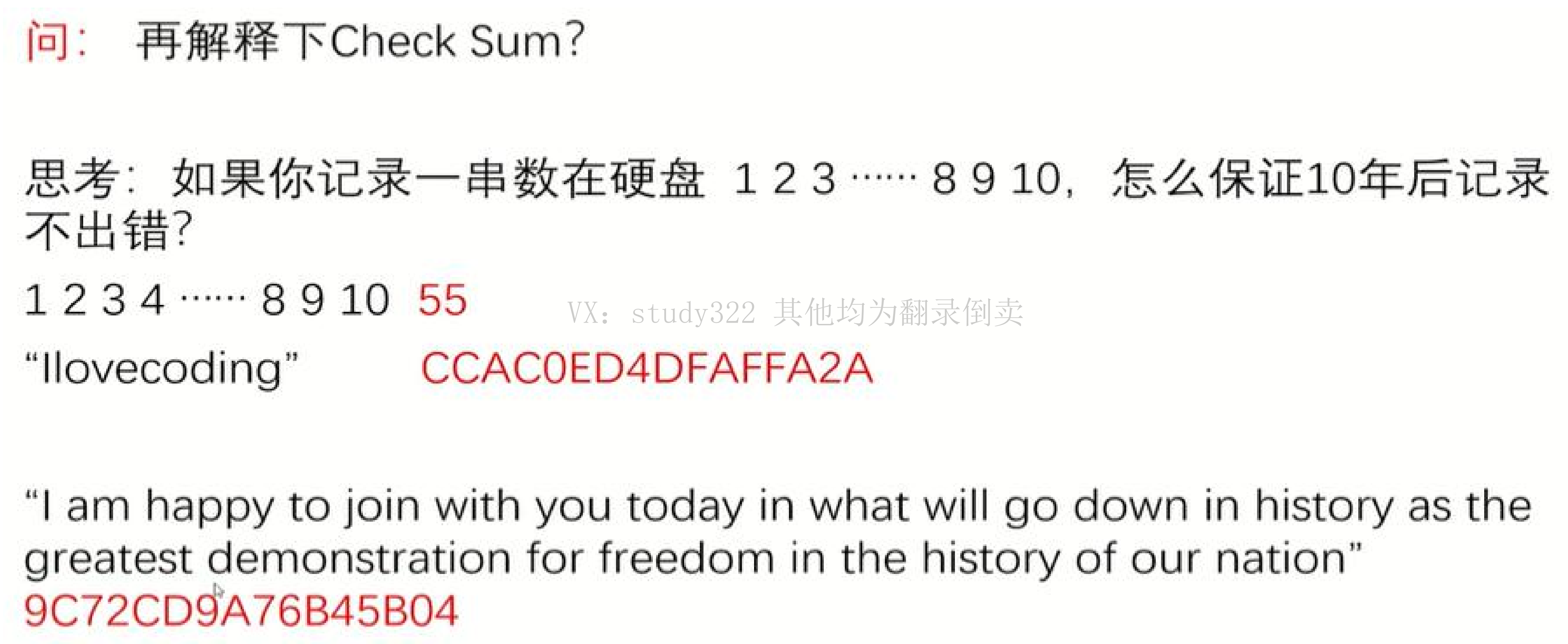
0	1	2	3	4	5	6	7	8
4kb	4kb	4kb	4kb	4kb	4kb	4kb	4kb	4kb

1 block

问：什么是异或（XOR）操作？

XOR	0	1
0	0	1
1	1	0

相同为0，不同为1



问：　再解释下Check Sum？

思考：如果你记录一串数在硬盘 1 2 3 …… 8 9 10，怎么保证10年后记录不出错？

1 2 3 4 …… 8 9 10　55

“Ilovecoding”　　CCAC0ED4DFAFFA2A

“I am happy to join with you today in what will go down in history as the greatest demonstration for freedom in the history of our nation”
9C72CD9A76B45B04

THANK YOU