ALLUXIO

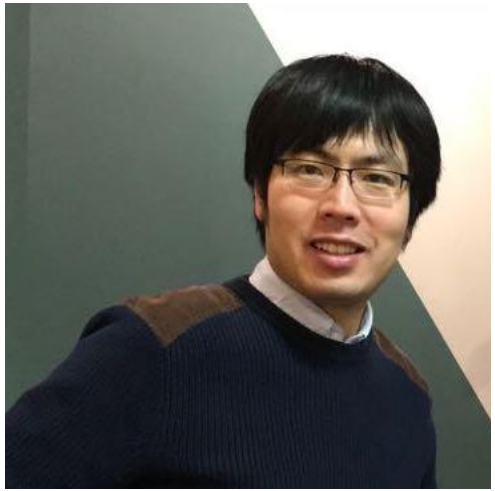
Building a Distributed File System for the Cloud-Native Era

Bin Fan, 05-30-2022 @ Peking University

Agenda

- Evolution From Hadoop Era to Cloud-native Era
- Design & Implement a Distributed File System

About Me



Bin Fan (https://www.linkedin.com/in/bin-fan/)

- Founding Engineer, VP Open Source @ Alluxio
- Alluxio PMC Co-Chair, Presto TSC/committer
- Email: binfan@alluxio.com
- PhD in CS @ Carnegie Mellon University

知 **apc999**

My Research Interests

- Memory-efficient Algorithms for Systems
  - Cuckoo Filter CoNext14
  - Setsep HotOS13
  - SmallCache-based Load balancing SoCC11

- Building Practical Systems
  - SILT SOSP11 - Extremely Mem-efficient KV store on SSD
  - MemC3 NSDI13 - Mem-efficient KV store on DRAM
  - Blizzard NSDI14
  - ScaleBricks SIGCOMM15

- Full Publication List (https://scholar.google.com/citations?user=FzoDCpoAAAAJ)

Joined Google After CMU

Joined Alluxio as Founding Engineer in 2015

The Startup life I was expecting

The Startup life I am experiencing

Alluxio Overview

- Originally a research project (Tachyon) in UC Berkeley AMPLab led by by-then PHD student Haoyuan Li (Alluxio founder CEO)
- Backed by top VCs (e.g., Andreessen Horowitz) with $70M raised in total, Series C ($50M) announced in 2021
- Deployed in production at large scale in Facebook, Uber, Microsoft, Tencent, Tiktok and etc
- More than 1100 Contributors on Github. In 2021, more than 40% commits in Github were contributed by the community users
- The 9th most critical Java-based Open-Source projects on Github by Google/OpenSSF[1]

[1] Google Comes Up With A Metric For Gauging Critical Open-Source Projects

Companies Using Alluxio

TECHNOLOGY

INTERNET

facebook, Baidu 百度, Alibaba.com, Tencent 腾讯, amazon, JD.COM 京东, Rakuten, waze

PUBLIC CLOUD PROVIDERS

aws, Google Cloud, Alibaba Cloud, Microsoft Azure, Tencent Cloud, Baidu Cloud

GENERAL

SAMSUNG, Aunalytics (rethink data), Microsoft, Adobe

E-COMMERCE

唯品会 VIPSHOP.COM 一家专门做特卖的网站, 苏宁易购 suning.com, Shopee, Walmart, Myntra

OTHERS

SAMSUNG SDS, RYTE, TalkingData, CAESARS ENTERTAINMENT, ROBLOX, 链家, yahoo!, NetEase Games, bazaarvoice, nielsen, GUARDANT HEALTH, Lenovo, ESENTIRE, 网易严选, PERCEPTIN, Ctrip, @WalmartLabs, DiDi, Sogou 搜狗, esri, Lucidworks, MOMO, CUELOGIC, QINIU 七牛云存储, 去哪儿 Qunar.Com, ORACLE, walkme Digital Adoption Platform

FINANCIAL SERVICES

DBS, ING, BARCLAYS, HUATAI SECURITIES, PayPal, WELLS FARGO

TELCO & MEDIA

COMCAST, 虎牙直播 HUYA.com, HUAWEI, swisscom, 中国移动 China Mobile, China unicom 中国联通

LEARN MORE

Ecosystem Evolution

From 2015 to 2022

7 Years Ago

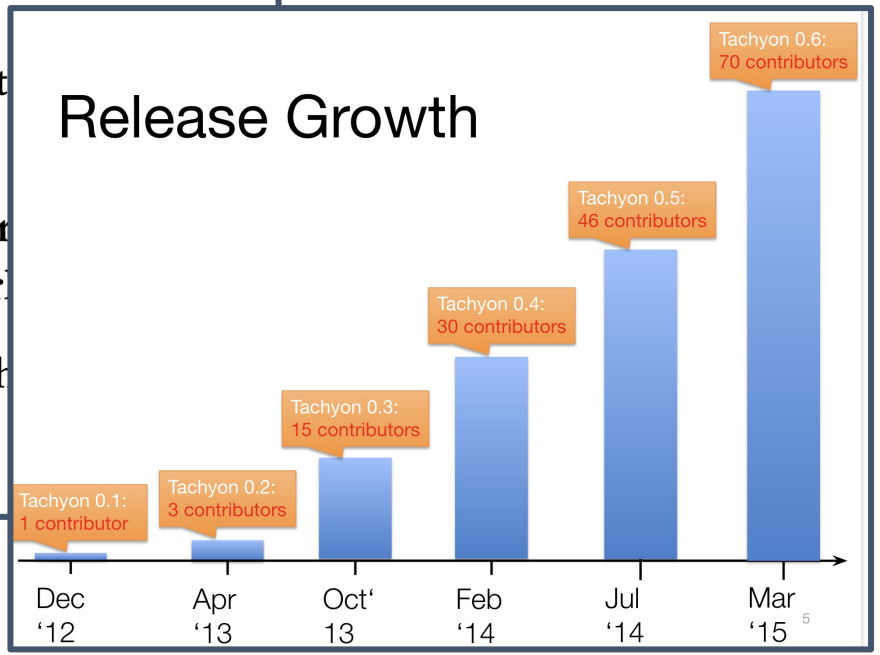
ALLUXIO

Big Data Ecosystem in 2015

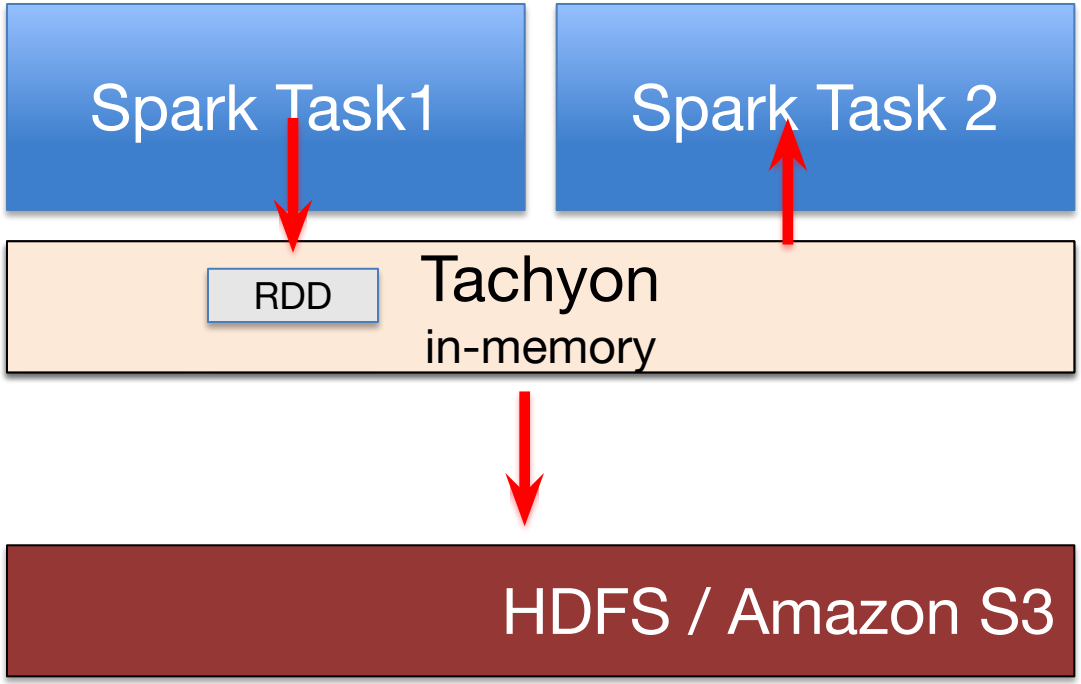
Alluxio (Tachyon) in 2015

What is Tachyon

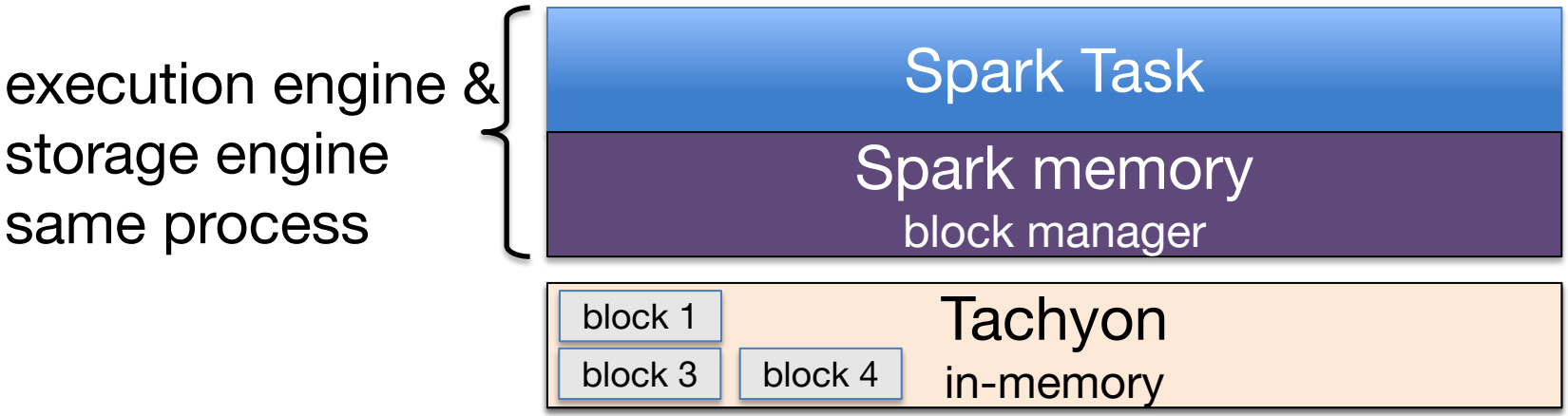
- A Reliable Memory Centric Dist
  Storage System
- Enable memory-speed data shar
  different computation framewor
- Started at AMPLab as a researc
  from the summer of 2012



Alluxio(Tachyon) in 2015: Enable Data Sharing Among Spark Jobs



Alluxio(Tachyon) in 2015: Fast Checkpoint for job reliability



Today

What's Different

Topology

- On-prem Hadoop → Cloud-native, Multi- or Hybrid-cloud, Multi-datacenter

Computation

- MR/Spark → Spark, Presto, Hive, Tensorflow, Pytorch ….
- More mature frameworks (less frequent OOM etc)

Data access pattern

- Sequential-read (e.g., scanning) on unstructured files → Ad-hoc read into structured/columnar data
- Hundred to thousand of big files → millions of small files

The Evolution from Hadoop to Cloud-native Era

Data Storage

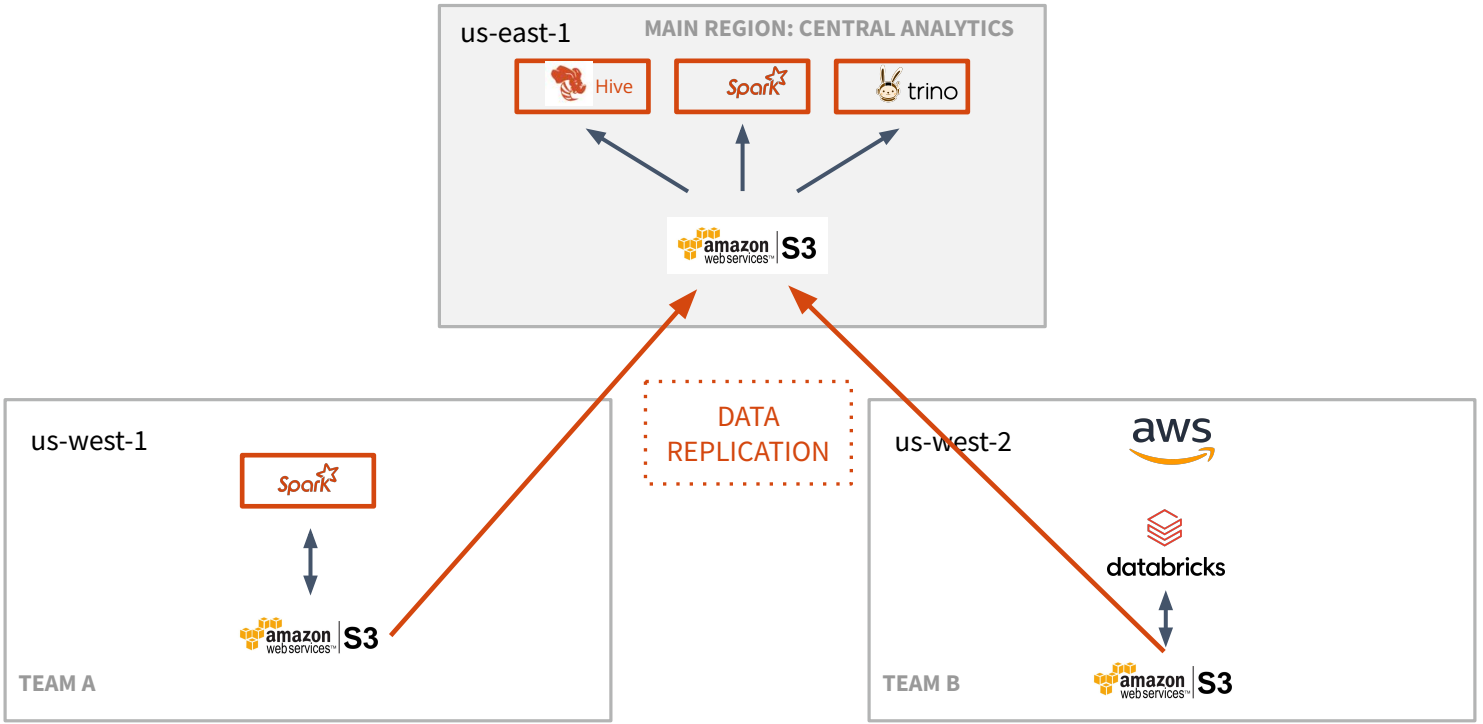
- On-prem & colocated HDFS → S3 !!! and other object stores (possibly across regions like us-east & us-west), and legacy on-prem HDFS in service

Resource/Job Orchestration

- YARN → K8s
  - Lost focus on data locality

ALLUXIO

A Real-world Example

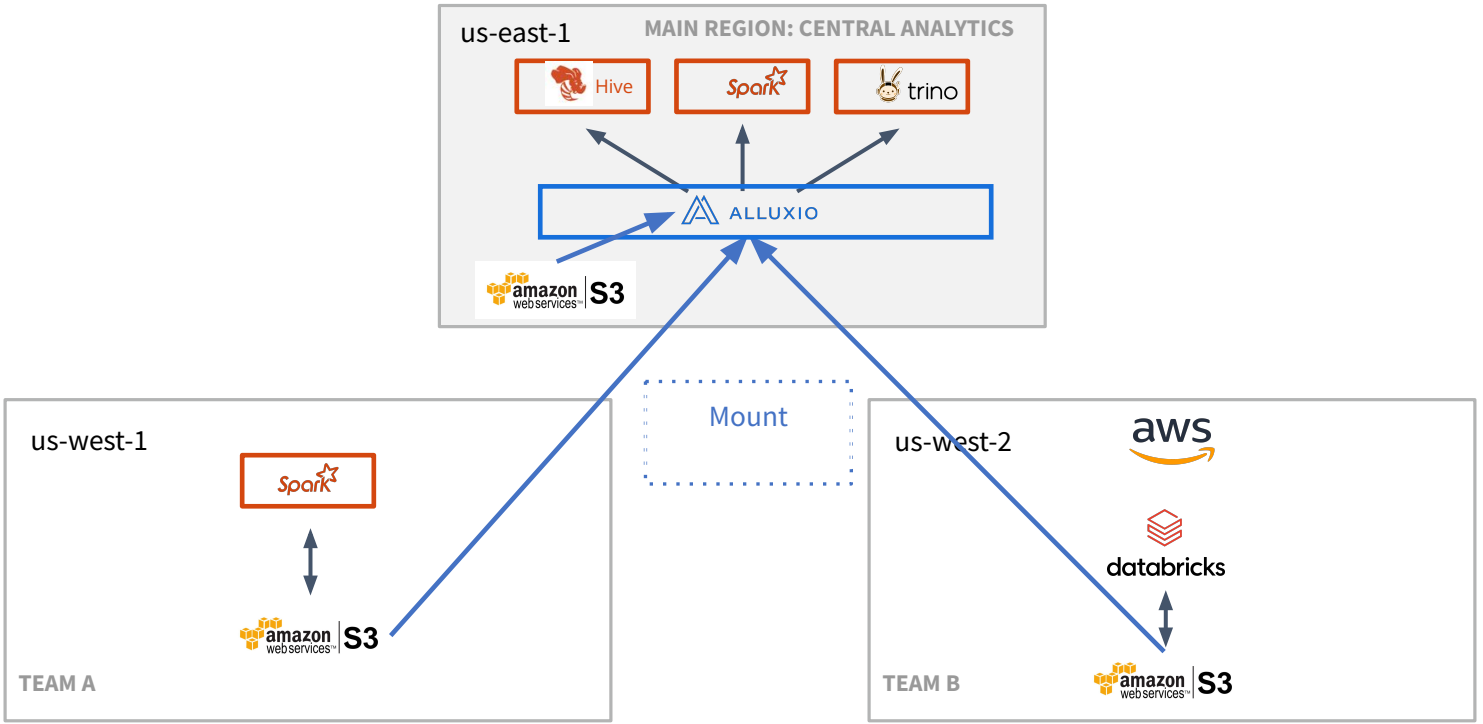


Analytics & AI in the Hybrid and Multi-Cloud Era


Available:

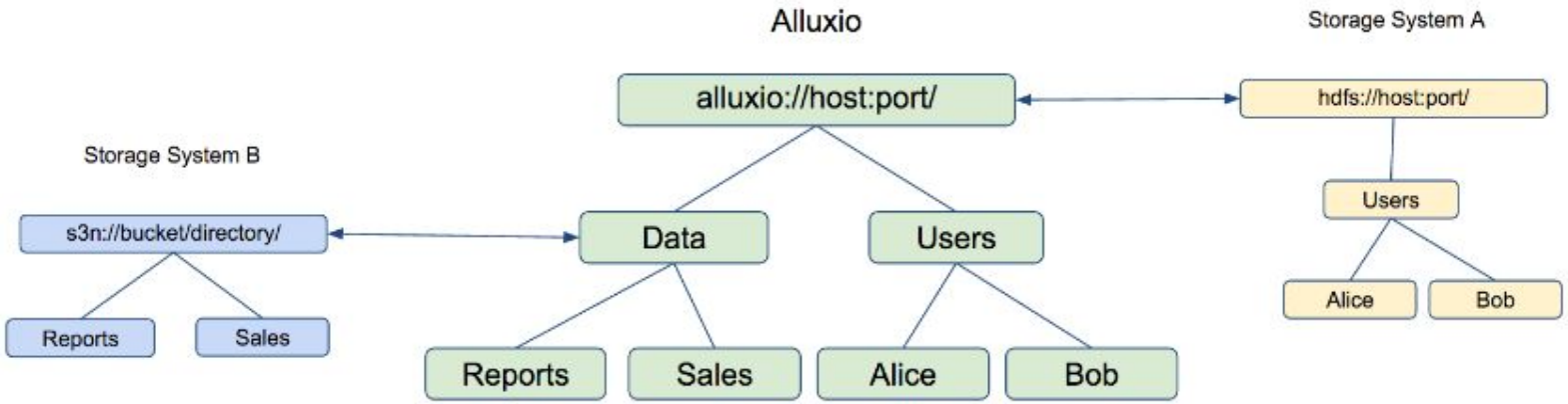
ALLUXIO

A Real-world Example



A Strongly Consistent Logical File System

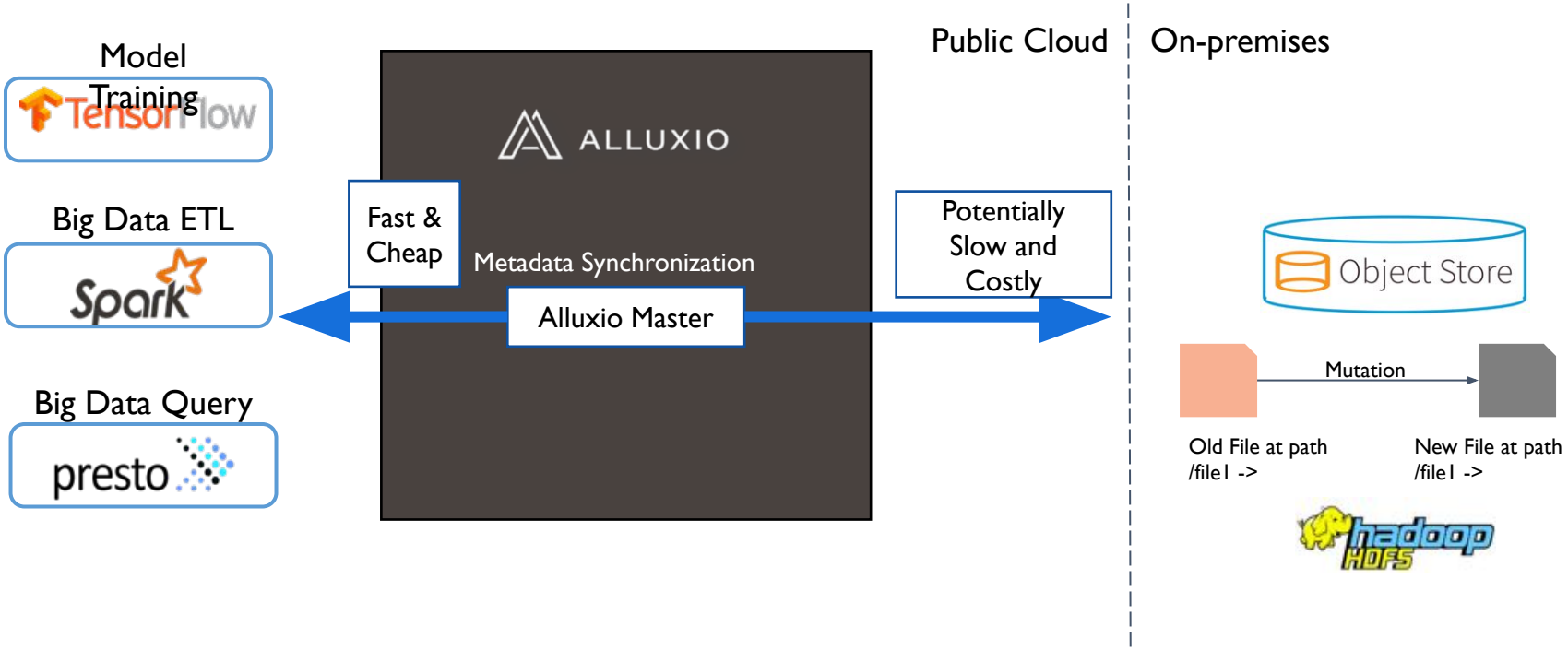
Mount individual storage systems to providing users a **Unified Namespace**



- Extension: Single logical Alluxio path backed by multiple storage systems
  - Example customized data policy: Migrate data older than 7 days from HDFS to S3

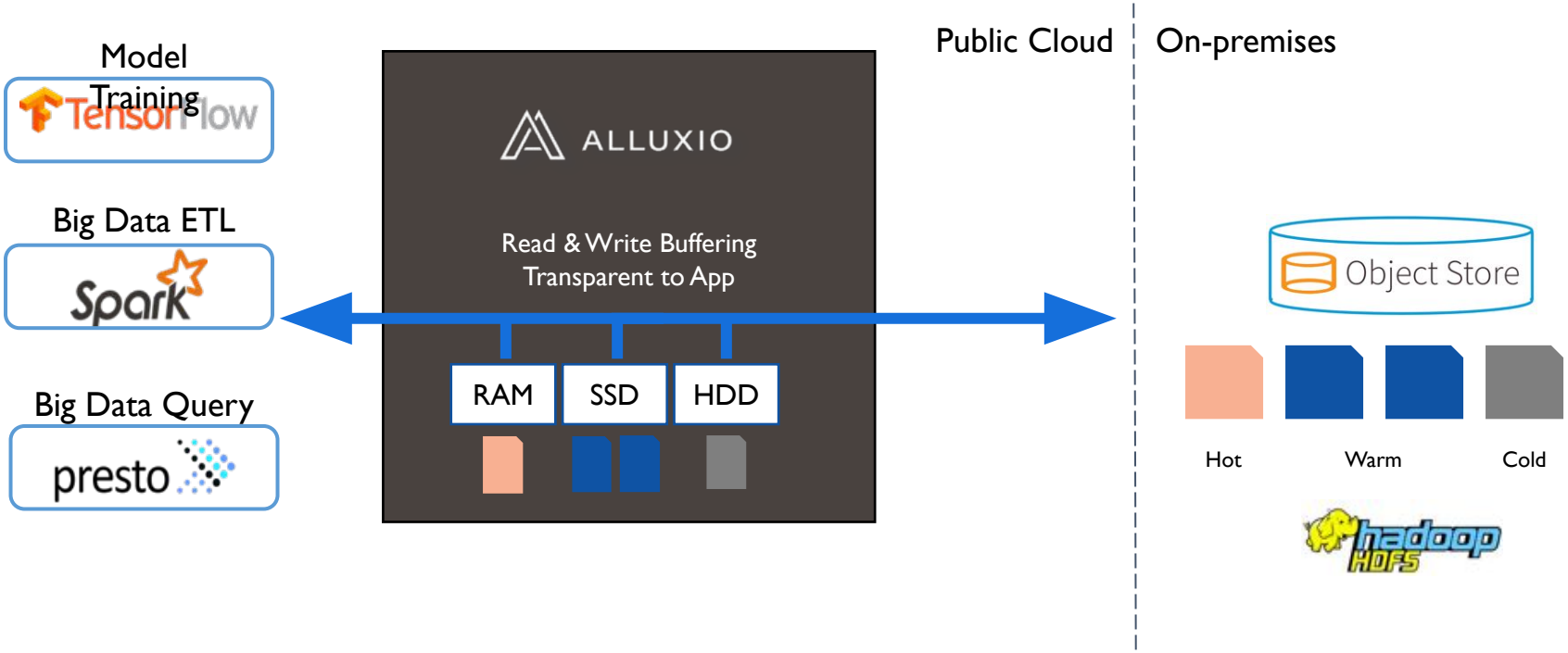
Scalable and Consistent Metadata Locality

Synchronization of changes across clusters



Scale-out Data Caching for Higher Data Locality

Local I/O performance for remote data with intelligent multi-tiering

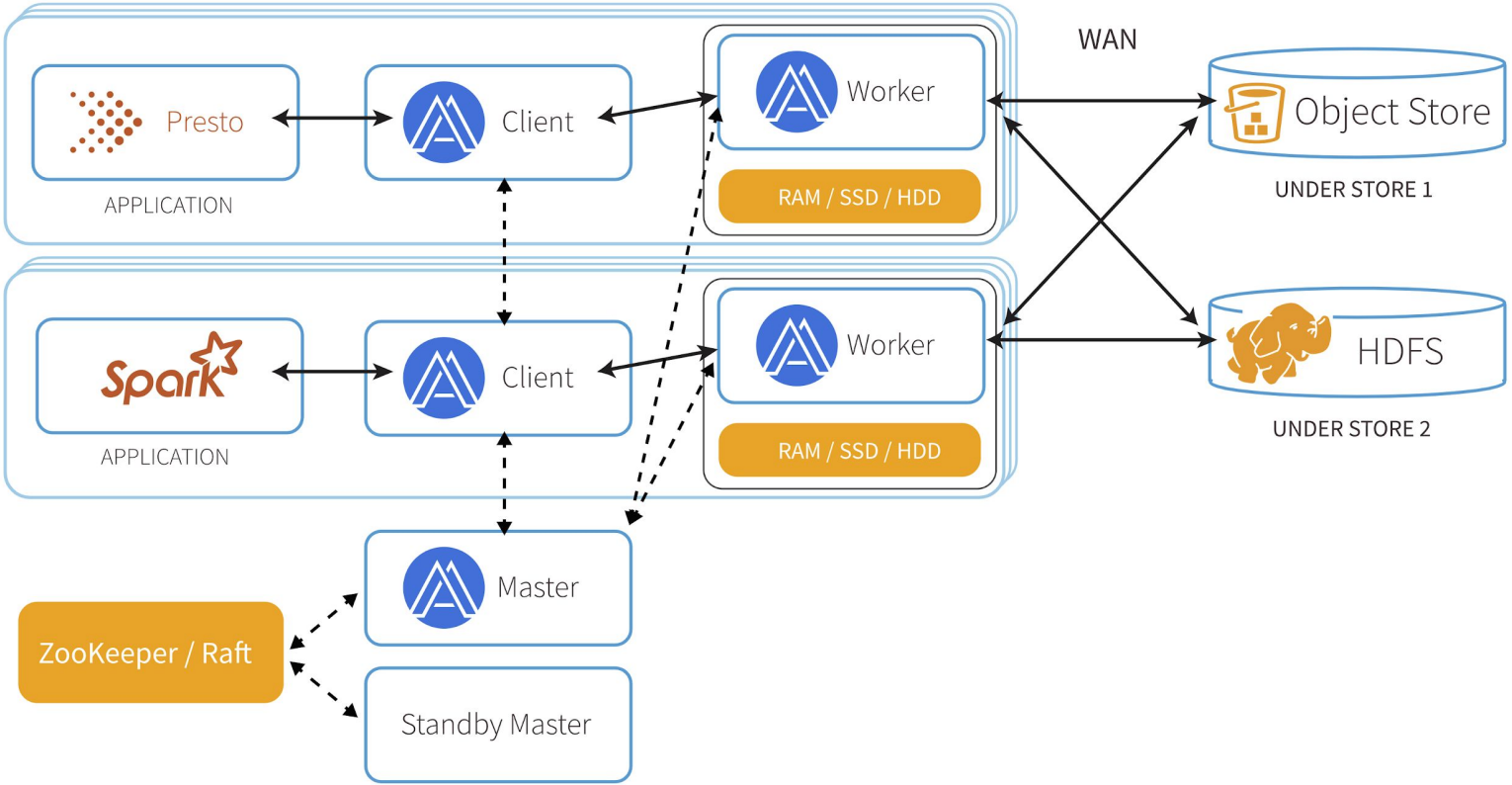


Design & Implement a Distributed File System

Challenges & Solutions

知乎:设计开源分布式文件系统Alluxio用到了哪些知识?

Alluxio Architecture



Metadata

What is File System Metadata

- Data structure of the Filesystem Hierarchy: Often an Inode tree to represent parent dir, children, permission bits, ower/group, modification time
  - Each node on this inode tree corresponding to one file or directory
  - Commonly seen in all file systems
  - Can include mounts of other file systems in Alluxio and the size of the tree can be very large!
- Sub-file blocks information (block ID -> workers)
  - Index for a distributed system to point to the data server

Factors w.r.t. Design a Scale Metadata Service

- # of Alluxio Servers in a cluster
  - Heartbeat:
    - node -> master
  - Load balancing
    - Workload skew
- # of concurrent clients
- # of files/dirs in this logical file system
- Throughput of metadata RPCs
  - Read ops
  - Write ops
- Speed to fail over to other stand-by masters (avoid Single node of failure)

Single Master Scalability

How to Store File System Metadata

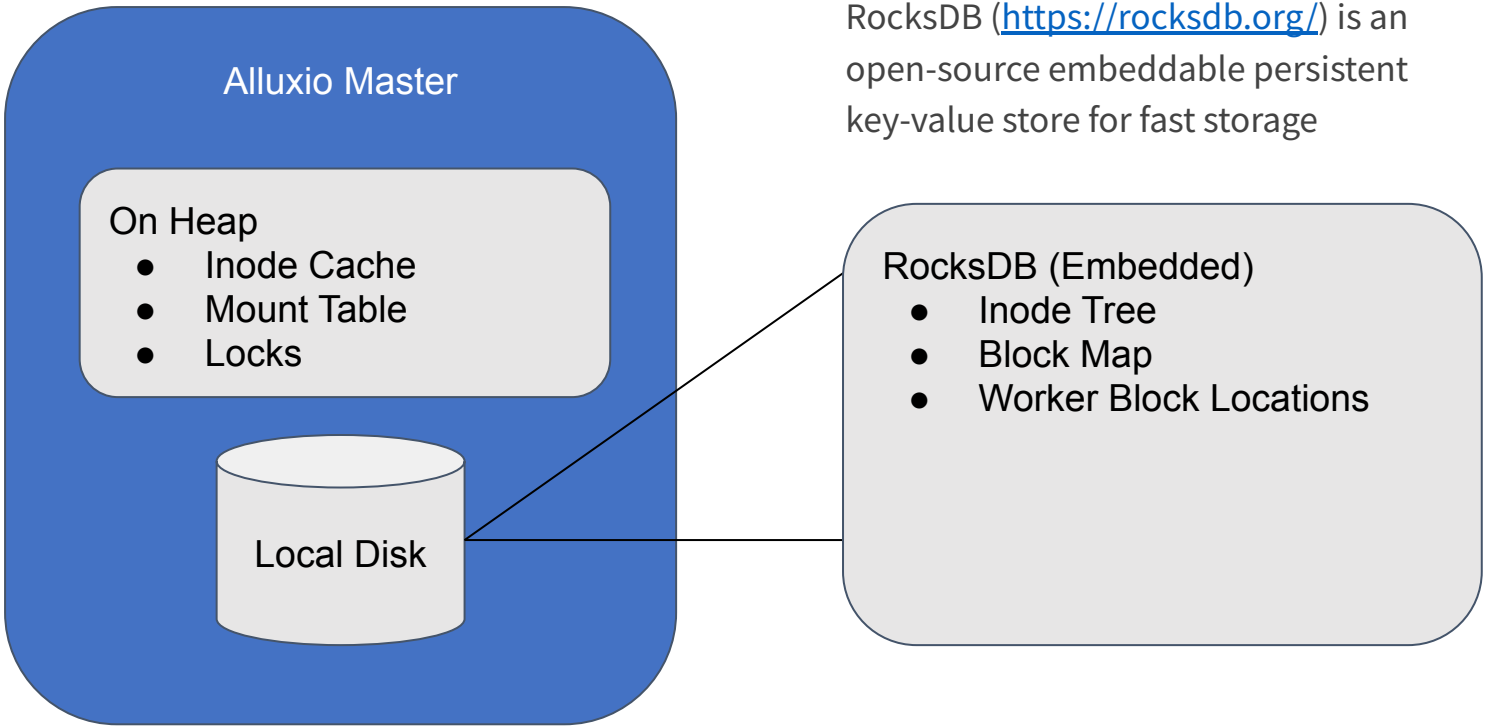
Federating Multiple Storage

=> We need to handle a “logical file system” multiple times bigger

Storing the raw metadata becomes a problem with a large number of files

- On average, each file takes 1KB of on-heap storage
- 1 billion files would take 1 TB of heap space!
- A typical JVM runs with < 64GB of heap space
- GC becomes a big problem when using larger heaps

Off-Heap Metadata Storage => 1 Billion Files



RocksDB (https://rocksdb.org/) is an open-source embeddable persistent key-value store for fast storage

Other Metadata Serving Challenges

- Common file operations (ie. getStatus, create) need to be fast
  - On heap data structures excel in this case
- Operations need to be optimized for high concurrency
  - Generally many readers and few writers for large-scale analytics
- The metadata service also needs to sustain high load
  - A cluster of 100 machines can easily house over 5k concurrent clients!
- Connection life cycles need to be managed well
  - Connection handshake is expensive
  - Holding an idle connection is also detrimental

High Availability

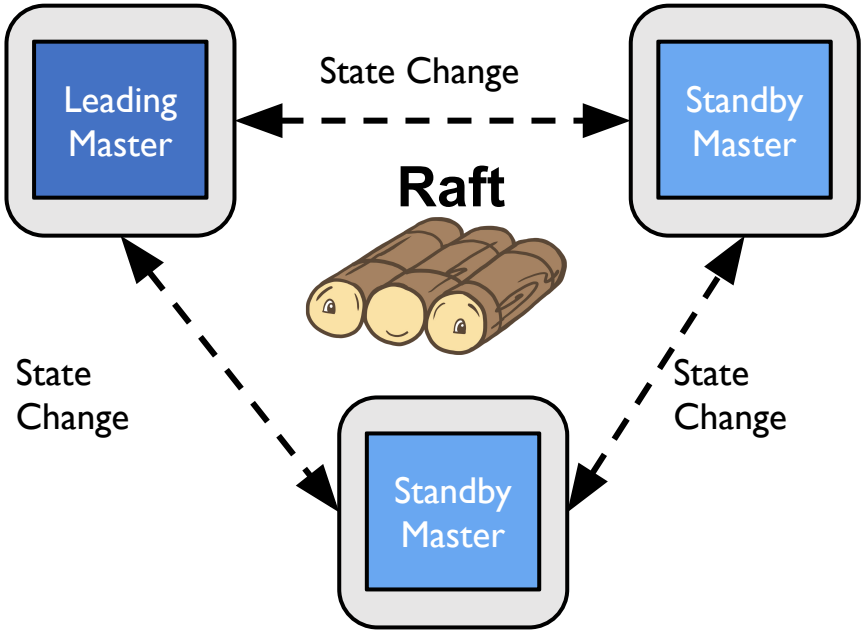
Built-in Fault Tolerance

- Alluxio cluster can recover from restarts, and avoid single-point of failure
  - File system status must be able to be recovered
  - This was previously done utilizing an external fault tolerance storage
- Our approach: Self-Managed Quorum for Leader Election and Journal Fault Tolerance Using Raft
  - Raft is a consensus algorithm that is designed to be easy to understand. It's equivalent to Paxos in fault-tolerance and performance
  - Enables hot standbys for rapid recovery in case of single node failure

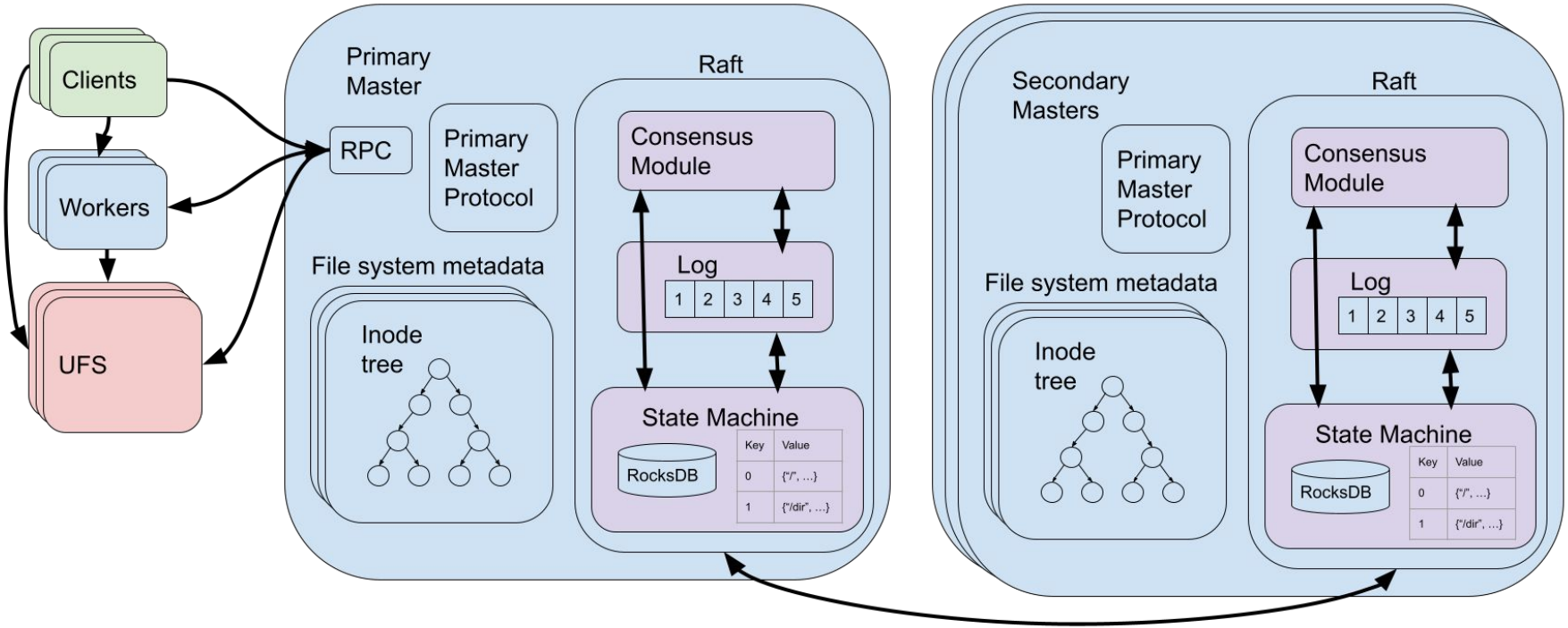
拓展阅读:知乎:漫话分布式系统共识协议: Paxos篇

Built-in Self-Managed Quorum-based Journal

- **Consensus achieved internally**
  - Leading masters commits state change
- **Benefits**
  - Local disk for journal
- **Challenges**
  - Performance tuning



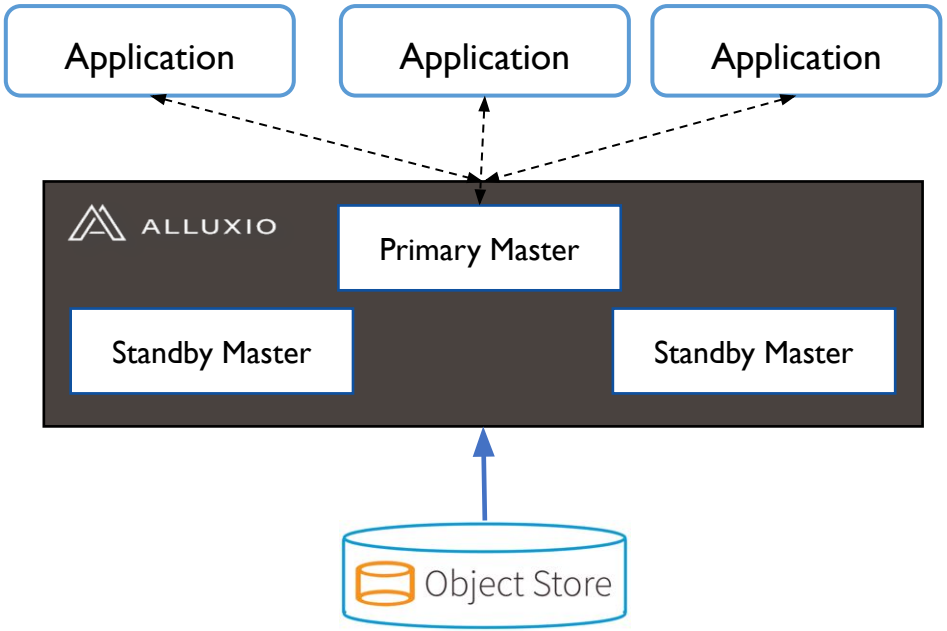
Alluxio + Raft architecture



Consistency

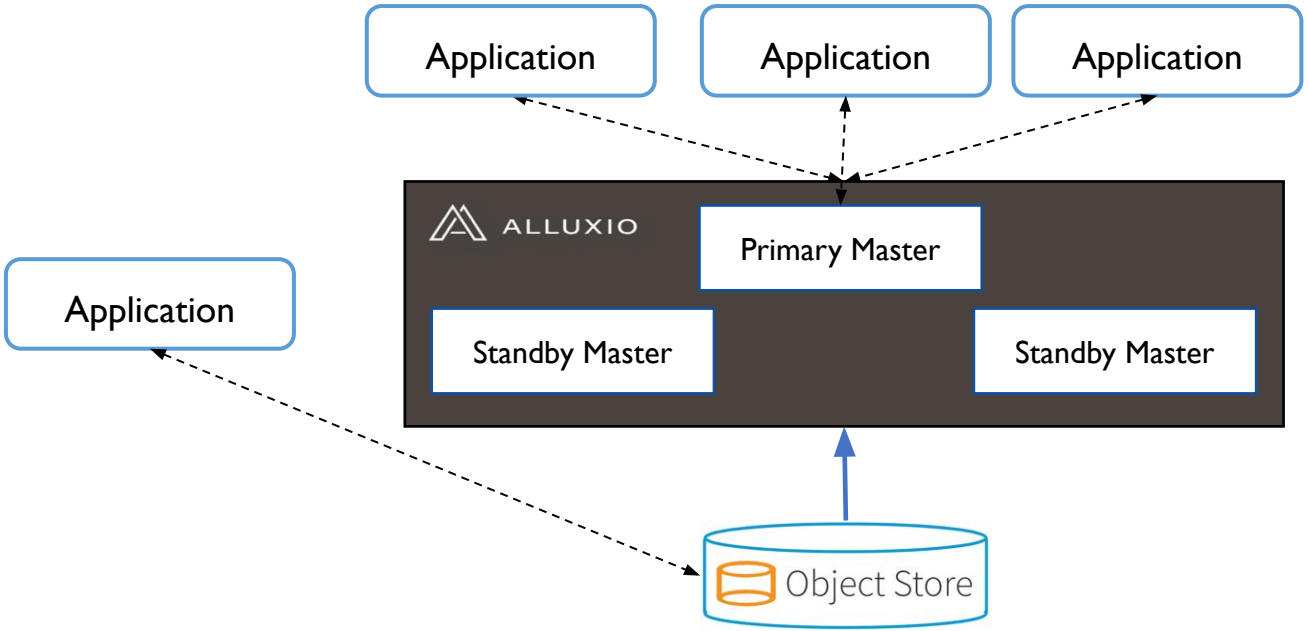
Consider Alluxio File System Alone

- If clients only query and modify Alluxio File System through Alluxio masters, the semantics is strongly consistent



Consider Alluxio File System + UFS

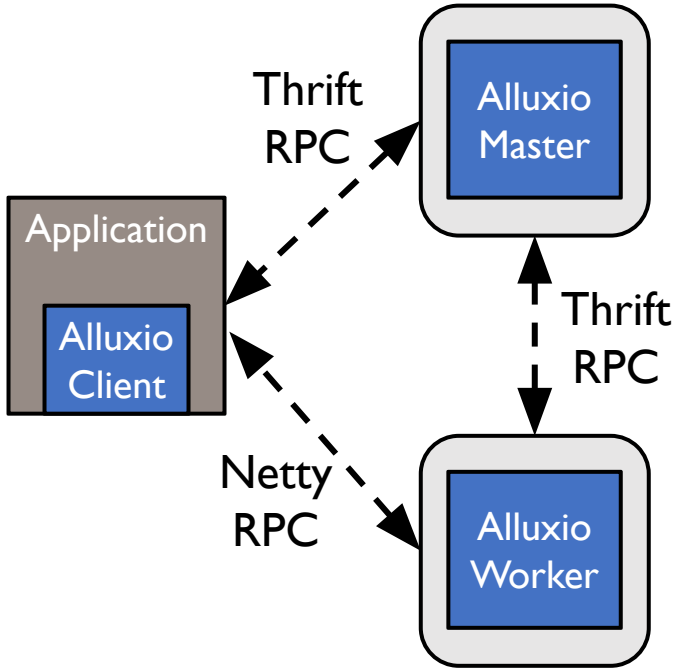
- When clients can modify UFS, Alluxio masters provide synchronization between Alluxio namespace and UFS



Serving Data

RPC System in Alluxio 1.x

- **Master RPC using Thrift**
  - Filesystem metadata operations
- **Worker RPC using Netty**
  - Data operations
- **Problems**
  - Hard to maintain and extend two systems
  - Thrift is not maintained, no streaming RPC support

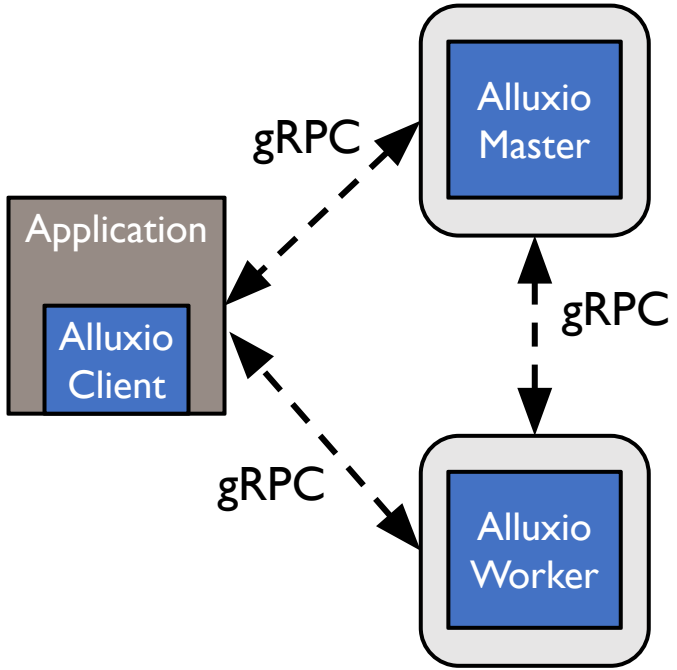


gRPC

- https://grpc.io/
- gRPC is a modern open source high performance RPC framework that can run in any environment
- Works well with Protobuf for serialization

Unified RPC Framework in Alluxio 2.0

- **Unify all RPC interfaces using gRPC**
- **Benefits**
  - Streaming I/O
  - Protobuf everywhere
  - Well maintained & documented
- **Challenges**
  - Performance tuning



gRPC Transport Layer

- Connection multiplexing to reduce the number of connections from # of application threads to # of applications
  - Solves the connection life cycle management problem
- Threading model enables the master to serve concurrent requests at scale
  - Solves the high load problem
- High metadata throughput needs to be matched with efficient IO
  - Consolidated Thrift (Metadata) and Netty (IO)

Check out this blog for more details:
https://www.alluxio.com/blog/moving-from-apache-thrift-to-grpc-a-perspective-from-alluxio

Corner Cases

Implement a Prototype is Easy

- Make it production ready is HARD
- All sorts of corner cases are the enemy
  - AWS S3 outage can happen every year
  - Race conditions: Concurrent reader/write, write/write
  - Resource-leaking can be unintentional
  - HDFS is considered reliable; when writing critical information (e.g., journals) be careful (and good luck)
  - Disk can fail without warning
  - External service can behave really weird
  - Human errors (mis-configuration)

Summary

Summary

- Designing & Implementing a distributed system is hard but also fun
- First you need to well understand the design requirements
- Consistency, Scalability, Reliability – We spent most of our time to fight for
- Do not reinvent the wheel, but also be cautious when introducing new building blocks
- Building a good open-source system is hard, building a thriving open-source community is hard^2

拓展阅读: *知乎:为什么在中国搞不出 Spark 和 Hadoop 这种东西？*

- Interested in 👆? Contact me and let's work on an intern project

Questions?

Website

www.alluxio.io

Slack

https://alluxio.io/slack

Social Media

Twitter.com/alluxio

Linkedin.com/alluxio