About Me

- Independent software engineer
- Today’s real example is from ClickFunnels
- > 70,000 customers, > 1.8B of payments processed
- Billions of rows of OLTP data (Amazon Aurora MySQL)

- Ryan Worl
- @ryanworl on Twitter
- ryantworl@gmail.com
Agenda

- How FoundationDB Works
- “Everyday” data problems
- Why FoundationDB can be the solution
- ClickFunnels’ recent data problem
- FoundationDB for YOUR data problems
Coordinators elect & heartbeat Cluster Controller (Paxos)

Coordinators store core cluster state, used like ZooKeeper

All processes register themselves with the Cluster Controller
Cluster Controller (CC) assigns Master Role
CC assigns TLog, Proxy, Resolver, and Storage Roles
CC assigns TLog, Proxy, Resolver, and Storage Roles
CC assigns TLog, Proxy, Resolver, and Storage Roles
CC assigns TLog, Proxy, Resolver, and Storage Roles
On Start: Your App Connects and Asks CC For Topology

YOUR APP
Client Library Asks a Proxy for Key Range to Storage Mapping
Data Distribution Runs On Master, Key Map Stored in Database
Start a Transaction: Ask Master for Latest Version
Start a Transaction: Ask Master for Latest Version (Batched)
Perform Reads at Read Version Directly to Storage
Consequences

- All replicas participate in reads
- Client load balances among different replicas
- Failures of all but one replica for each range keep the system alive
Buffer Writes Locally Until Commit

YOUR APP

00
55
AF
FF
Commit Part 1: Send R/W Conflict Ranges + Mutations to Proxy
Part 2: Proxy Batches Txns to Master To Get Commit Version
Consequences

- Master is not a throughput bottleneck
- Intelligent batching makes Master workload small
- Conflict ranges and mutations are not sent to Master at all

@ryanworl
Part 3: Send Conflict Ranges to Resolvers for Conflict Detection
Part 4: If Isolation Passes, Send Mutations to Relevant TLogs
Part 5: (Async) Storages Pull Mutations from Their Buddy TLogs
Failure Detection: Cluster Controller Heartbeats
Initiate Recovery on Any Transaction Role Failure

YOUR APP

00 55 AF FF
Cluster Controller Failure: Coordinators Elect New One

YOUR APP
Storage Server Failure: No Recovery, Repair in Background
Status Quo

- Most apps start uncomplicated
- One database, one queue
- ... five years later, a dozen data systems
“Everyday” Data Problems?

What the hell have you built.

- Did you just pick things at random?
- Why is Redis talking to MongoDB?
- Why do you even *use* MongoDB?

Goddamnit | Nevermind

https://twitter.com/coda

@ryanwrl
“Microservices”

- Can make this worse

@ryanworl
Why is this a problem?

- Operational costs
  - Administrative costs
  - Duplicated data
- Development costs
  - Atomicity mostly ignored in the real world
  - Corrupted data extremely common
Why is this a problem?

- Security costs
  - More systems = More risk
- Error handling never exercised
  - “De-coupled”, “redundant”, “fault tolerant” services mostly a myth
Why is this a problem?

- “Managed cloud services”
  - They will never pick up the pieces
  - They will reboot the machine for you…
  - A weak system run by someone else is still weak
    - e.g. data loss from async replication

@ryanwor1
Why is FoundationDB a solution?

- Build anything you want or need
- Multiple systems in one cluster
- Eventually consistent models easier to build too
  - OLTP ➔ Change Log ➔ OLAP
ClickFunnels’ Recent Data Problem
The Everyday Data Problem

- “Smart Lists” based on user-defined rules
- Running against billions of rows in an OLTP database
- Both user-facing and automated (100s of QPS)
The Everyday Data Problem

SELECT
    contacts.id from contacts
LEFT JOIN
    emails on emails.contact_id = contacts.id
LEFT JOIN
    templates on templates.id = emails.template_id
WHERE ...
Breaking it down

- Data volume = 100s of GB
- Complex joins and row-oriented storage
- Indexes can’t satisfy every query efficiently
- Aurora = single threaded queries
- Really just set operations on integers at the core...
Bitmap indexes!
Bitmap Indexes 101

- Roaring Bitmap Library (roaringbitmap.org)
- Space usage proportional to number of set bits
- Billions of operations per second (SIMD)
- Easily parallelizable (multi-core and distributed)
Bitmap Indexes 101

- Multi-minute evaluation times of rules in Aurora
- Under 100ms with bitmaps
New Possibilities

- Evaluating rules on every customer website page view
- “How many people will this rule match?” in real time
- Stats and analytics can adapt to this format
- E.g. unique email opens per hour with rules applied for fancy charts
- Pages load instantly even for the largest customers

@ryanworl
How to get there - Step One

- Replicate Aurora binlog into FoundationDB
- Write volume not high enough to worry about sharding
- Example of a log structure

["binlog", VersionStamp] => MySQL Binlog as JSON

@ryanworl
How to get there - Step Two

- Chunk bitmaps into small segments ($2^{18}$ is fine)
- Evaluate rules, set bits where rules match
- One writer at a time for low contention
- Example of storing a large object among many keys

```
[“bitmaps”, rule_id, chunk_id] => Bitmap Chunk
```
How to get there - Step Three

- Do a range read for every chunk for each rule
- Parallelize by evaluating different ranges
- Classic fork-join pattern

CORE 1

[“bitmaps”, rule_1, chunk_1] => Chunk
[“bitmaps”, rule_2, chunk_1] => Chunk

CORE 2

[“bitmaps”, rule_1, chunk_N] => Chunk
[“bitmaps”, rule_2, chunk_N] => Chunk
Experimental Results

- Real-world queries take 100ms
- One large box today
- Distributed later with little extra work
- HA from auto-scaling group and load balancer
- < 3000 lines of JavaScript + RoaringBitmap

@ryanworl
YOUR Everyday Data Problems

- FoundationDB’s performance
  - Concurrency Potential
  - Coordination Avoidance
- Break down the transaction critical path

@ryanworl
~ 275 allocations / second

160ms latency and growing

https://www.activesphere.com/blog/2018/08/05/high-contention-allocator
> 3500 allocations / second

~ 13ms latency @ high concurrency

https://www.activesphere.com/blog/2018/08/05/high-contention-allocator
YOUR Everyday Data Problems

- Tables, logs, queues, secondary indexes
- Simple to implement with little code
- Freedom to build your exact solution
- … without the explosion of data systems
- One cluster to manage
Questions

- Email or tweet me if you have questions or want to talk about specific use cases for FoundationDB

@ryanworl        ryantworl@gmail.com