A better client ecosystem for MySQL at LinkedIn

Sundar Raman Ganesh

```sql
SELECT about FROM SREs WHERE
first_name = 'Sundar Raman' AND
last_name = 'Ganesh' AND
job_title = 'Sr. Engineer, Site Reliability';
```

about
● SRE working on relational databases ● Passionate about building automations that scale ● Plays on his Xbox in free time

Agenda

Introduction

Observability

Availability

Security

Conclusion

MySQL... We choose you!

- Open source
- Time tested
- Extensible
- Rich community

Powered by MySQL®

- Multi tenant
- Self-serve

Over the years

- Increased adoption
- Diverse implementation

Journey so far..

- Query latency (95th percentile)

  < 10 ms

- Fewer incidents year over year

- Increased availability of 99.99%

https://lnkd.in/MySQLHA

So…What's the problem?

Agenda

Introduction

Observability

Availability

Security

Conclusion

Problem 1

- High MTTD for incidents originating from clients

The Reason

- Lack of observability made it difficult to debug

Root Causes

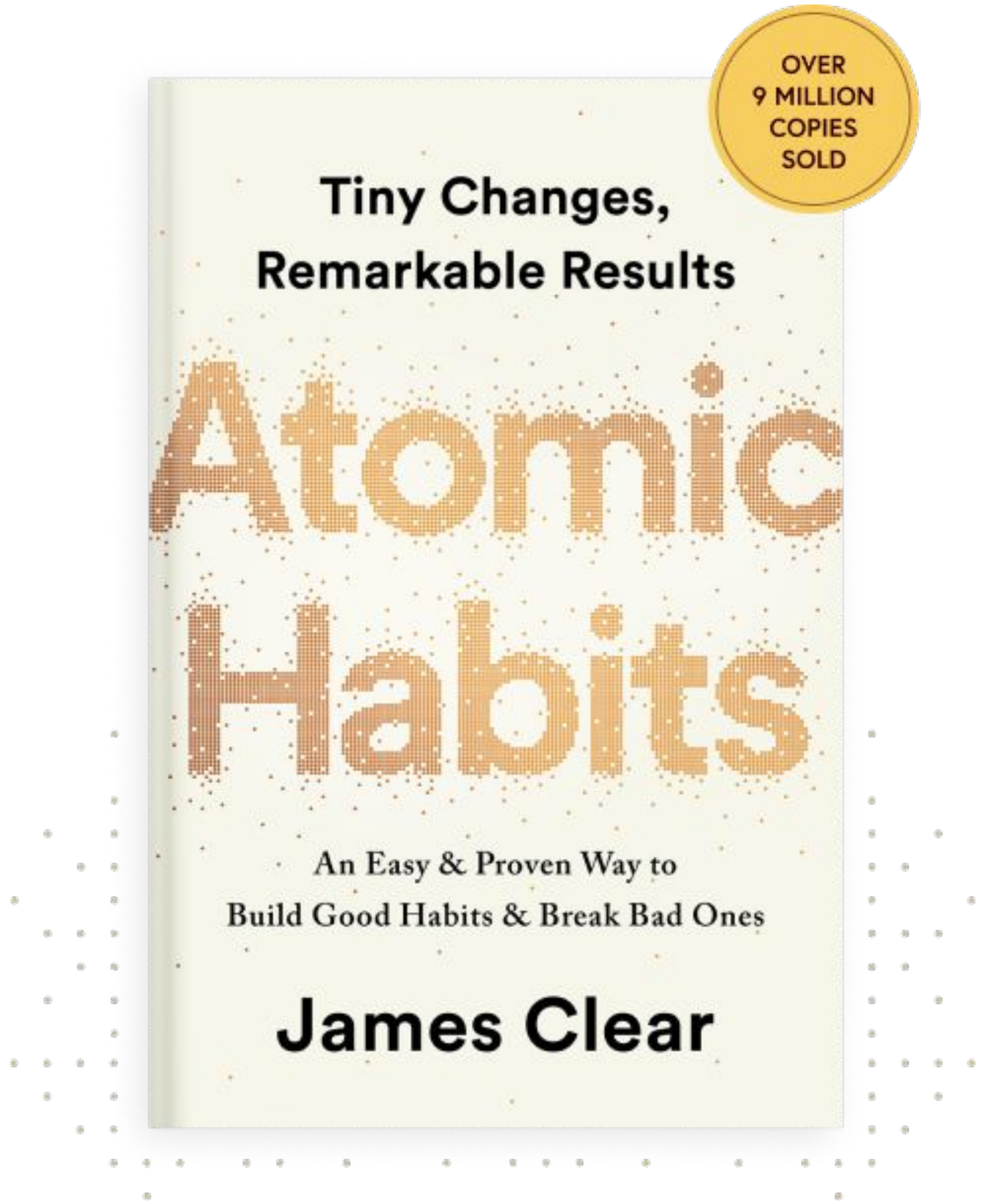
- Application bottlenecks
- Bugs in older version of client tools
- Misconfigurations
- Bad implementation

A costly bargain

- Application teams own the client code
- Database SREs owns the server infrastructure

- Standardization
- Client side observability

Design Philosophy


OVER 9 MILLION COPIES SOLD
Tiny Changes, Remarkable Results
Atomic Habits
An Easy & Proven Way to Build Good Habits & Break Bad Ones
James Clear

- Make it easy
- Make it obvious
- Make it attractive
- Make it satisfying

Challenges

- Diverse set of client tools used to interact with databases
- Version drift of client tools across applications

Client Tools for MySQL

.

- mysqlclient
- PyMySQL
- MySQLdb v2
- MySQL connector/Python
- mysql-connector-java

.

- Apache DBCP
- Hikari CP
- Hibernate
- JOOQ
- EBean
- ...

Programming languages of the clients

Java

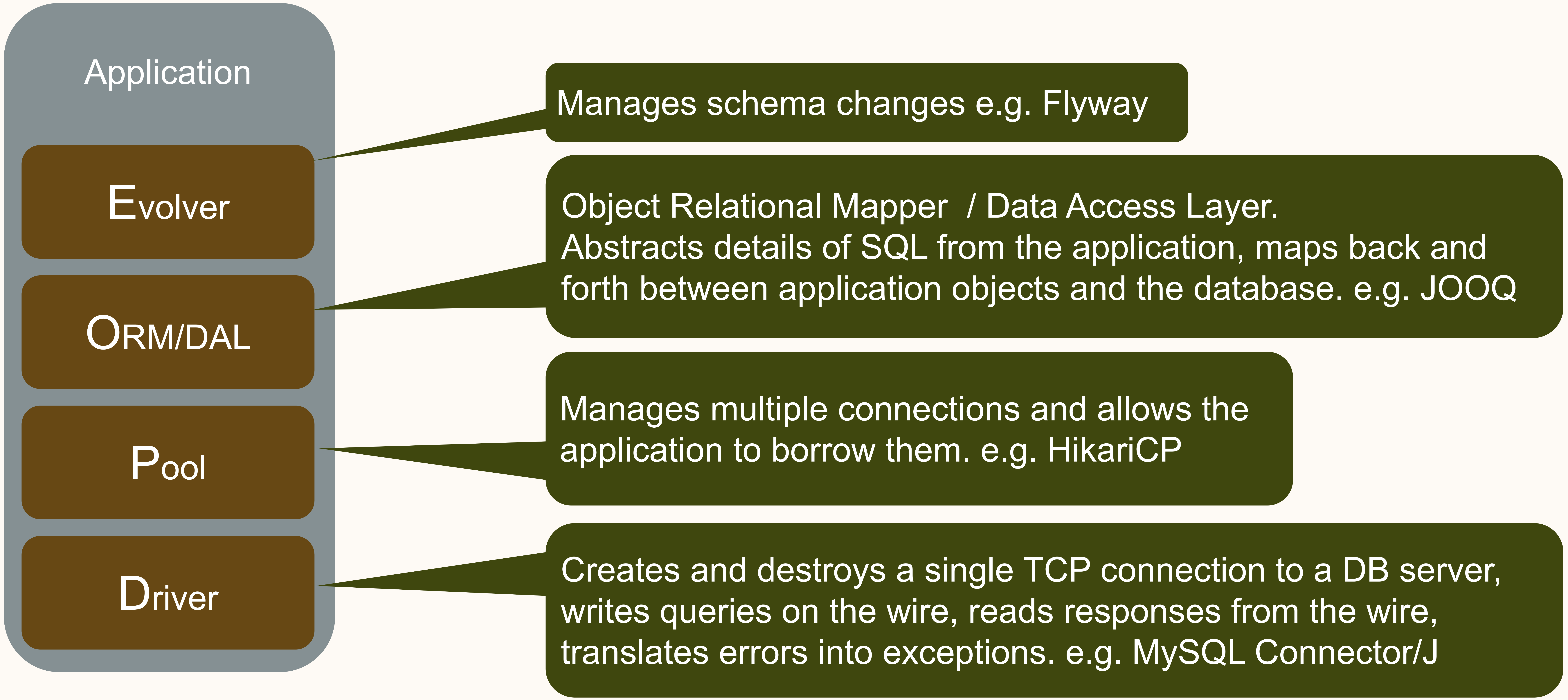
Python

Go

Deconstructing “The Client”

- “Client” = more than just a driver
- An app has a stack of multiple database-related components
- Components competing for roles

Component Roles - Production



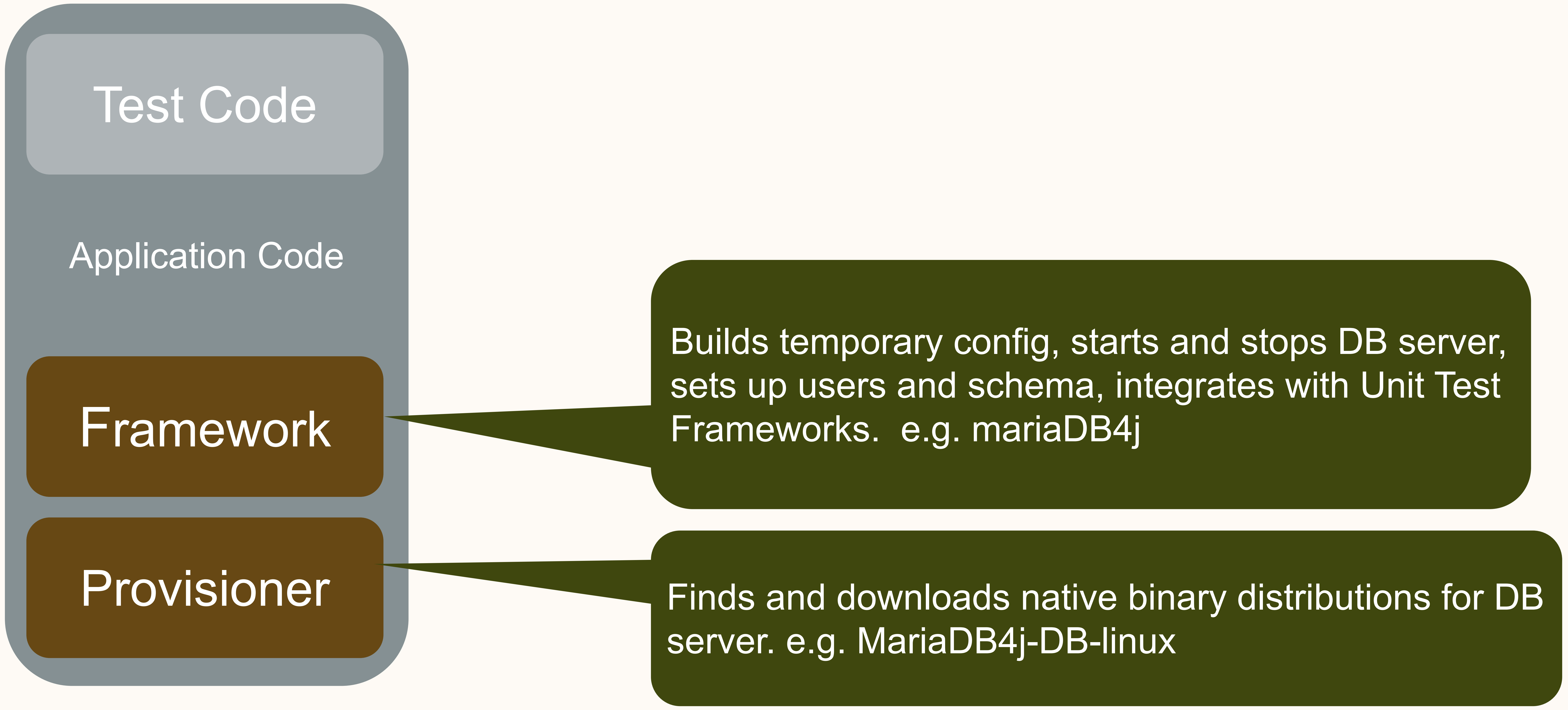
Manages schema changes e.g. Flyway

Object Relational Mapper / Data Access Layer.
Abstracts details of SQL from the application, maps back and forth between application objects and the database. e.g. JOOQ

Manages multiple connections and allows the application to borrow them. e.g. HikariCP

Creates and destroys a single TCP connection to a DB server, writes queries on the wire, reads responses from the wire, translates errors into exceptions. e.g. MySQL Connector/J

Component Roles - Testing



Builds temporary config, starts and stops DB server, sets up users and schema, integrates with Unit Test Frameworks. e.g. mariaDB4j

Finds and downloads native binary distributions for DB server. e.g. MariaDB4j-DB-linux

Let's Marie Kondo the Clients

- Choose one client from each component/lang combination
- Do the work to make it supported
- Replace the unsupported clients

What is a supported client?

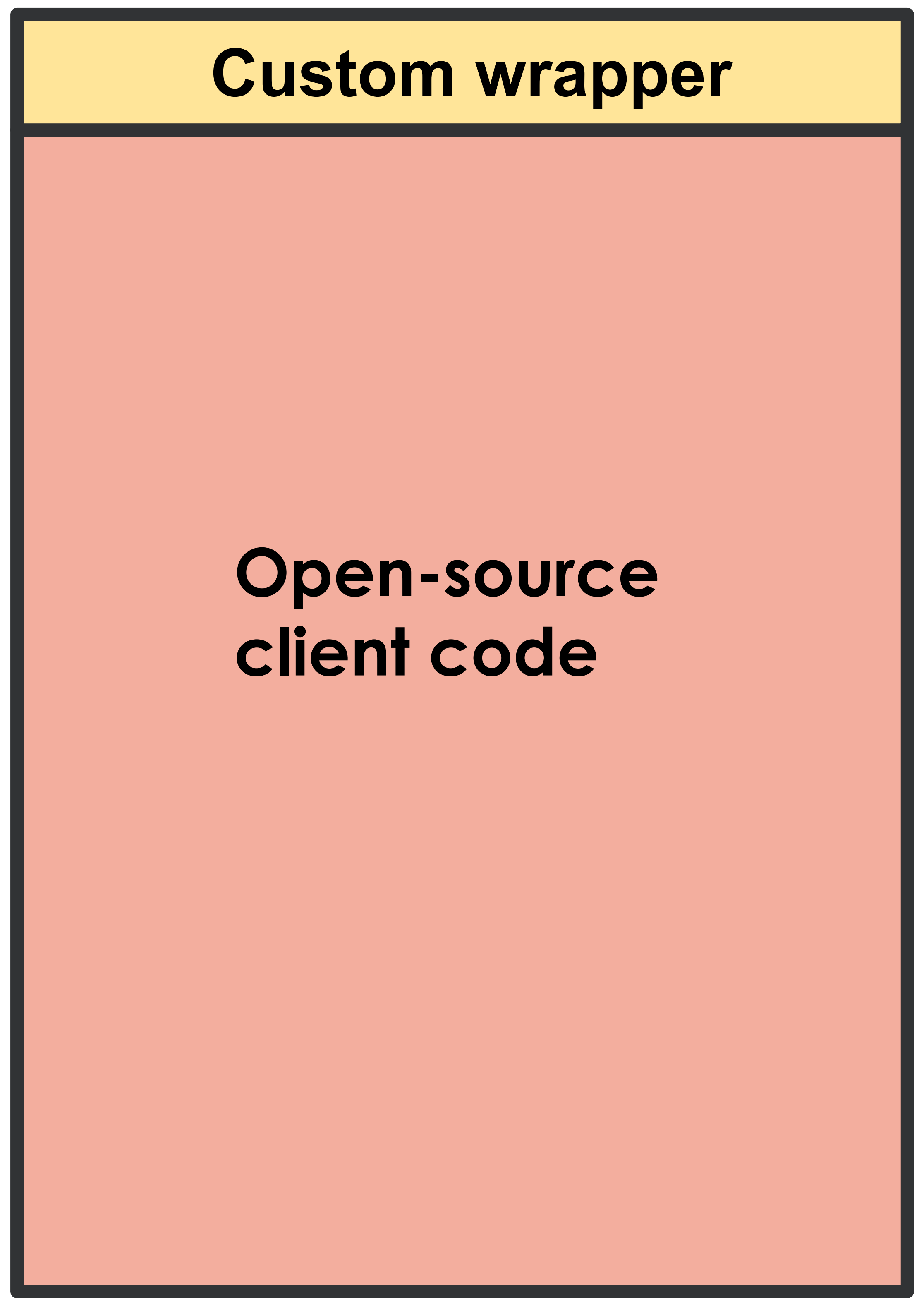
- Reliable
- Observable
- Easy to use (Dev, SRE)

How to build a supported client?

“Code reuse is the holy grail of software engineering”

Douglas Crockford

Hello wrappers!



- Reuse existing code
- Collect and send metrics
- Additional logging
- Standardized configurations

Why wrappers?

- Need for custom configuration management
- Need for talking to internal services at LinkedIn
- Make changes that are not needed by the community

Client-side metrics

- Query metrics
- Connection metrics
- Connection pool

  Statistics

Query Metrics

- Call time averages and percentiles
- Average/Percentiles of query latency
- Success/failure count

Connection Metrics

- Avg. connection wait time
- Avg. connection start time
- Connection failure count

Connection Pool Statistics

- Total connections in pool
- Available connections
- Borrowed connections

Better logging

- Enable logging everywhere
- Standardize logging format

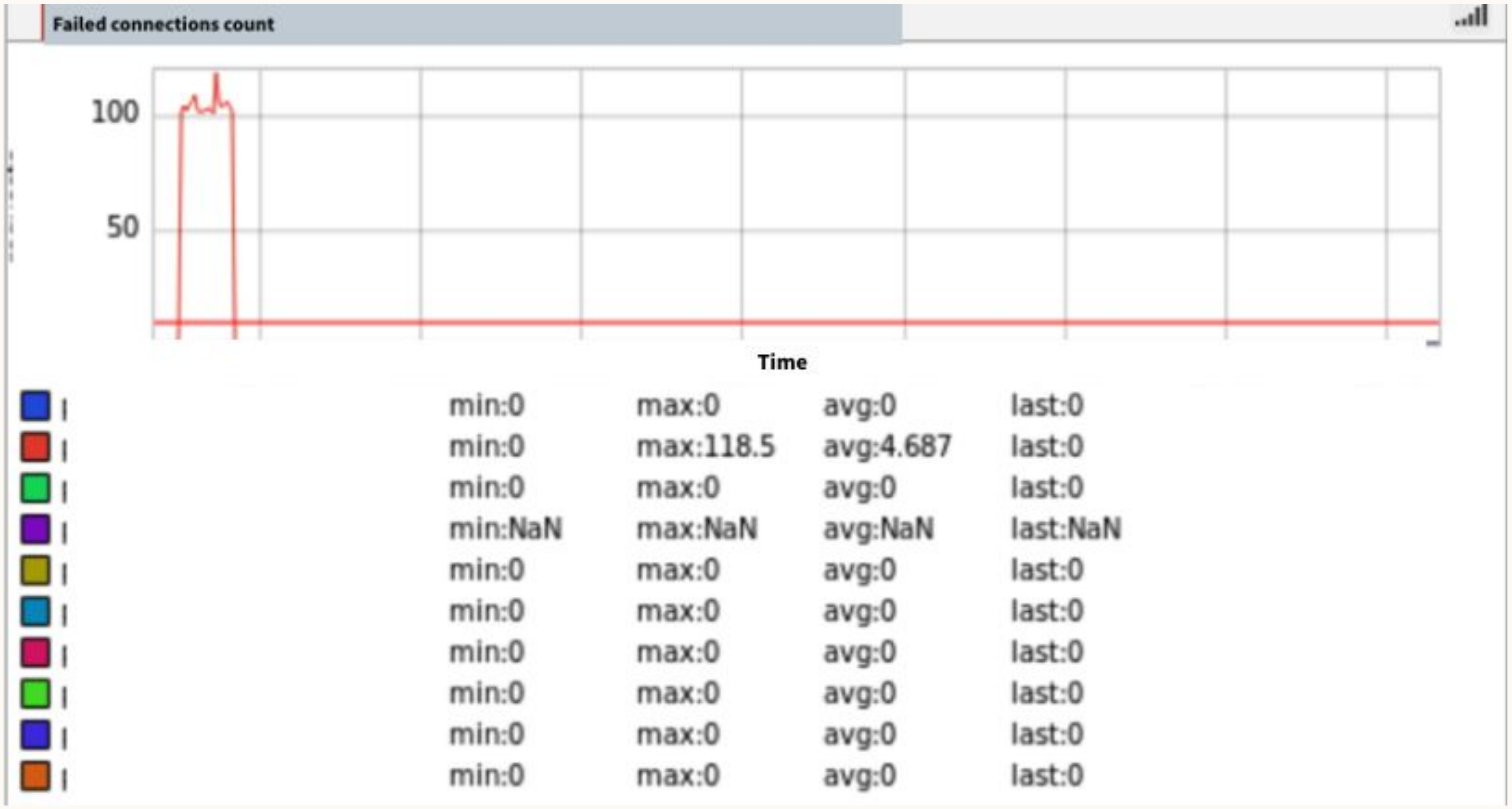
Additional things to log

- Results of DNS lookups
- Connection parameters
- Connection pool info

How has it helped?

Scenario 1

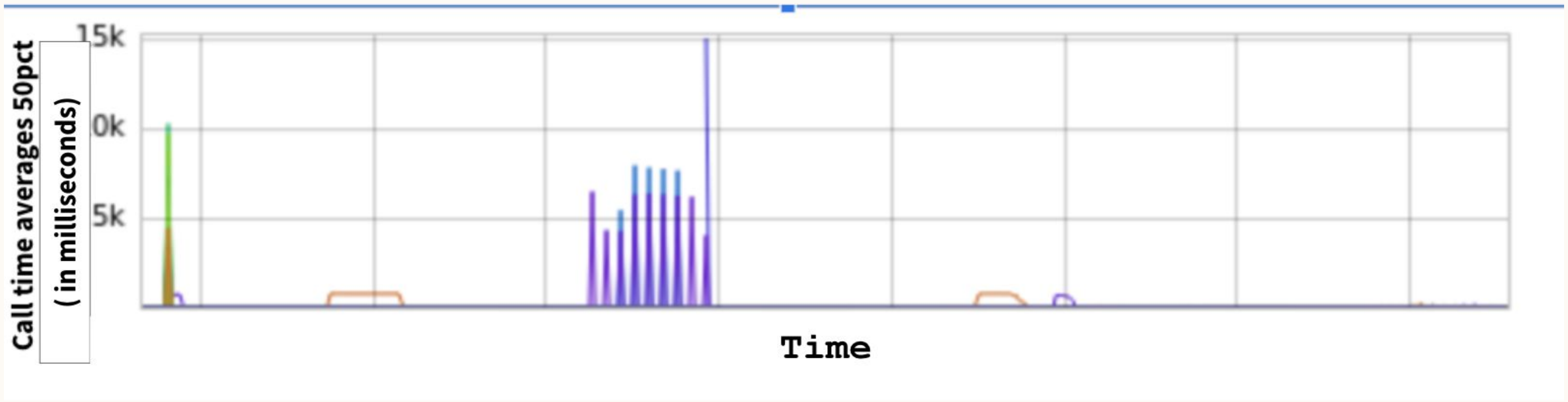
- Single client host issues



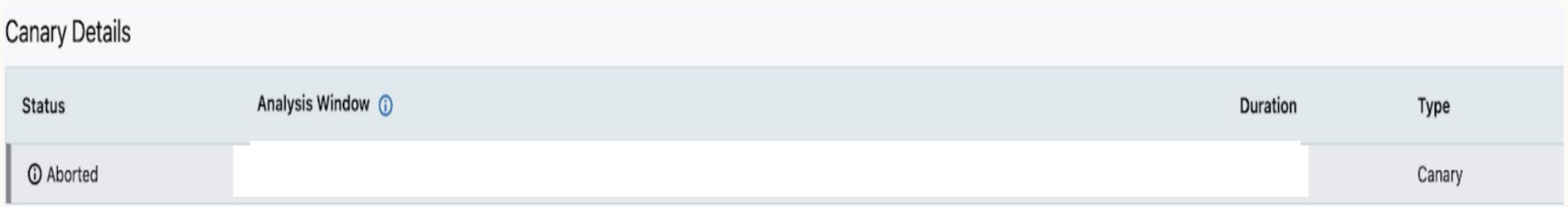
🟦	min:0	max:0	avg:0	last:0
🟥	min:0	max:118.5	avg:4.687	last:0
🟩	min:0	max:0	avg:0	last:0
🟪	min:NaN	max:NaN	avg:NaN	last:NaN
🟨	min:0	max:0	avg:0	last:0
🟦	min:0	max:0	avg:0	last:0
🟥	min:0	max:0	avg:0	last:0
🟩	min:0	max:0	avg:0	last:0
🟦	min:0	max:0	avg:0	last:0
🟧	min:0	max:0	avg:0	last:0

Scenario 2

- Call time average peaking



Scenario 2 (continued)

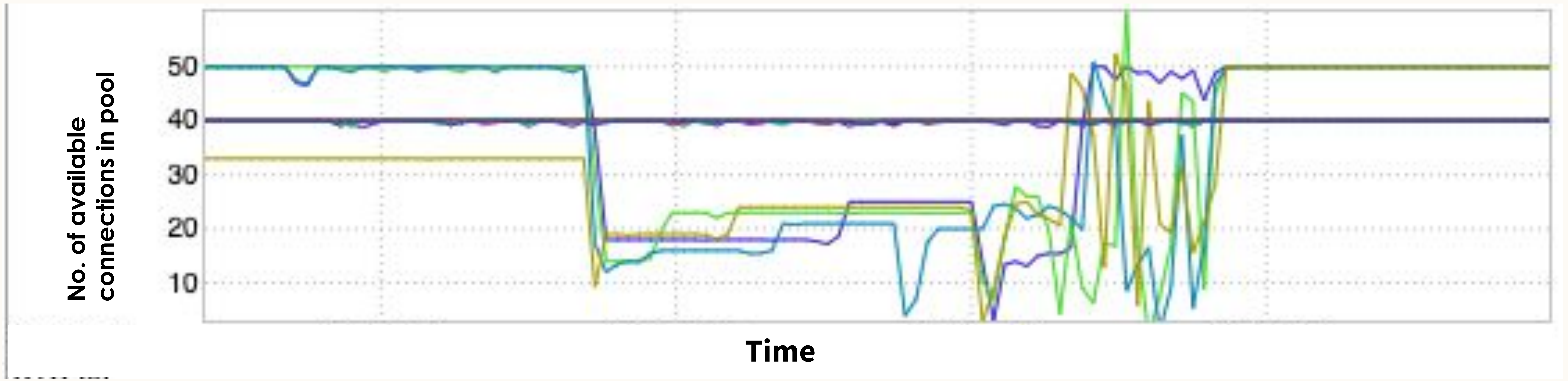


Canary Details

Status	Analysis Window ⓘ	Duration	Type
ⓘ Aborted			Canary

Scenario 3

- Blocked connections in the pool



Then ...

Now...

Agenda

Introduction

Observability

Availability

Security

Conclusion

Problem 2

- High MTTR for failovers

orchestrator

https://github.com/openark/orchestrator

- Open source
- Tested extensively
- Supports complex topologies

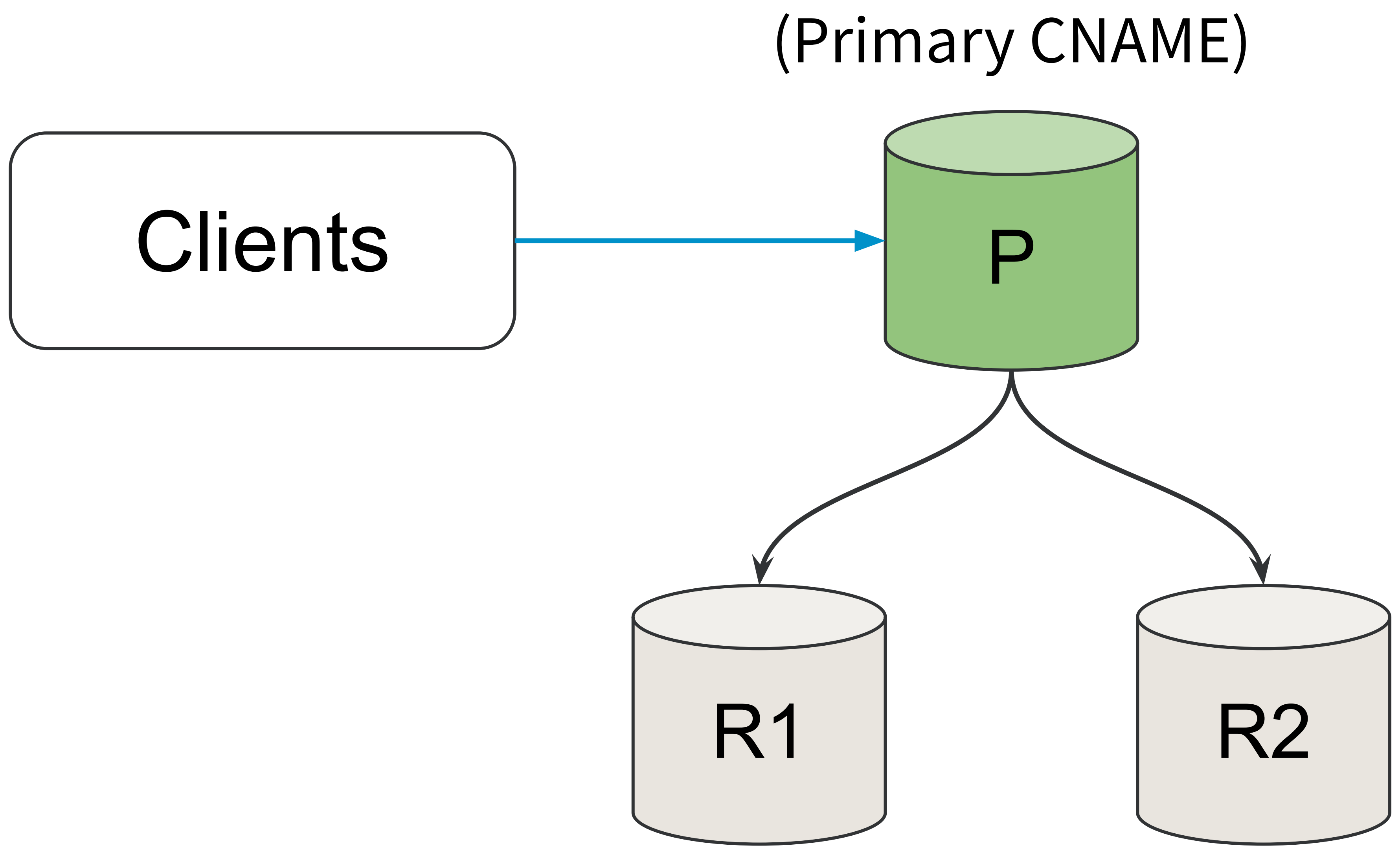
Benefits

Detect failure in Real time

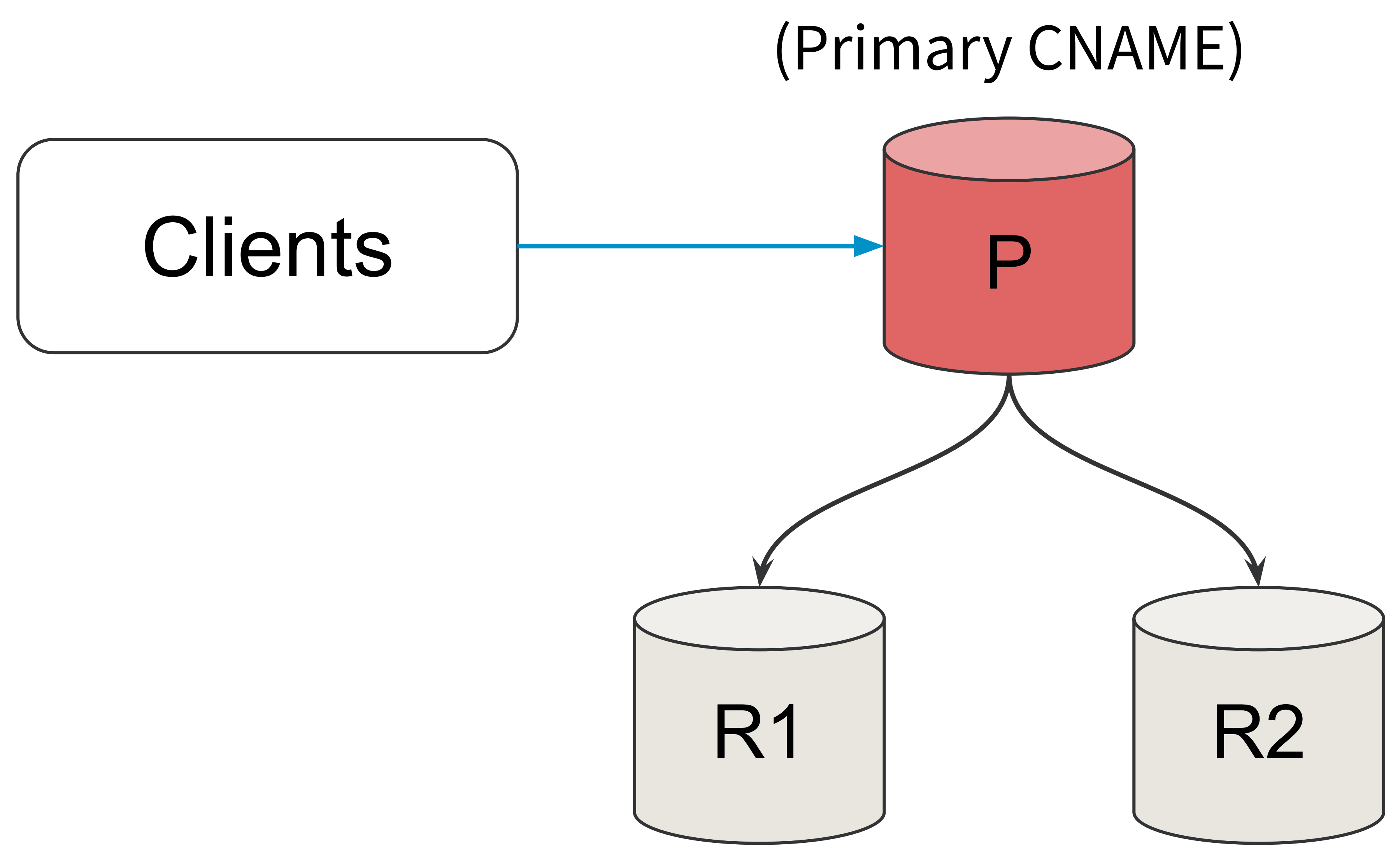
Topology Healing

Replica Promotion

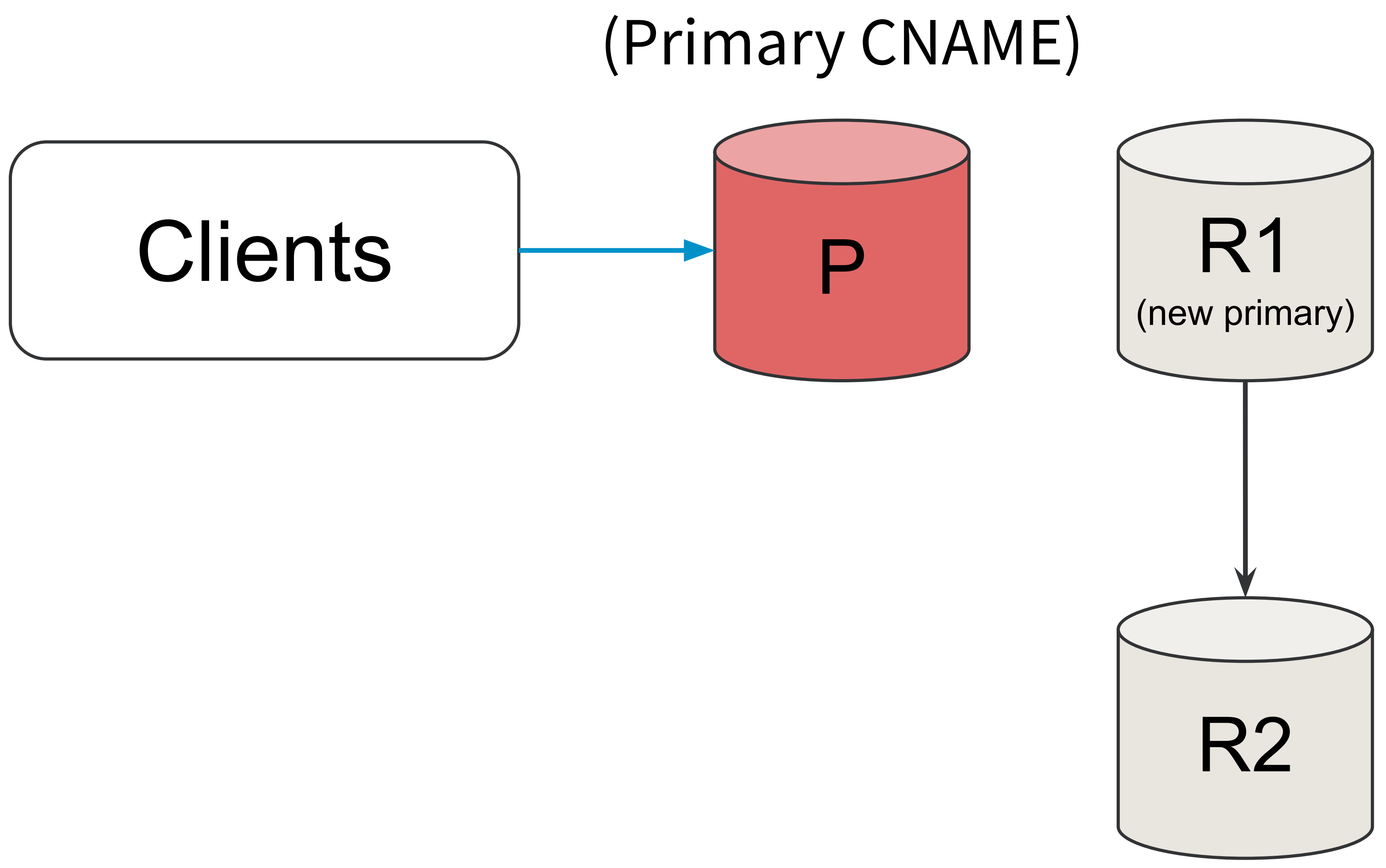
Consistent Emerging Topology



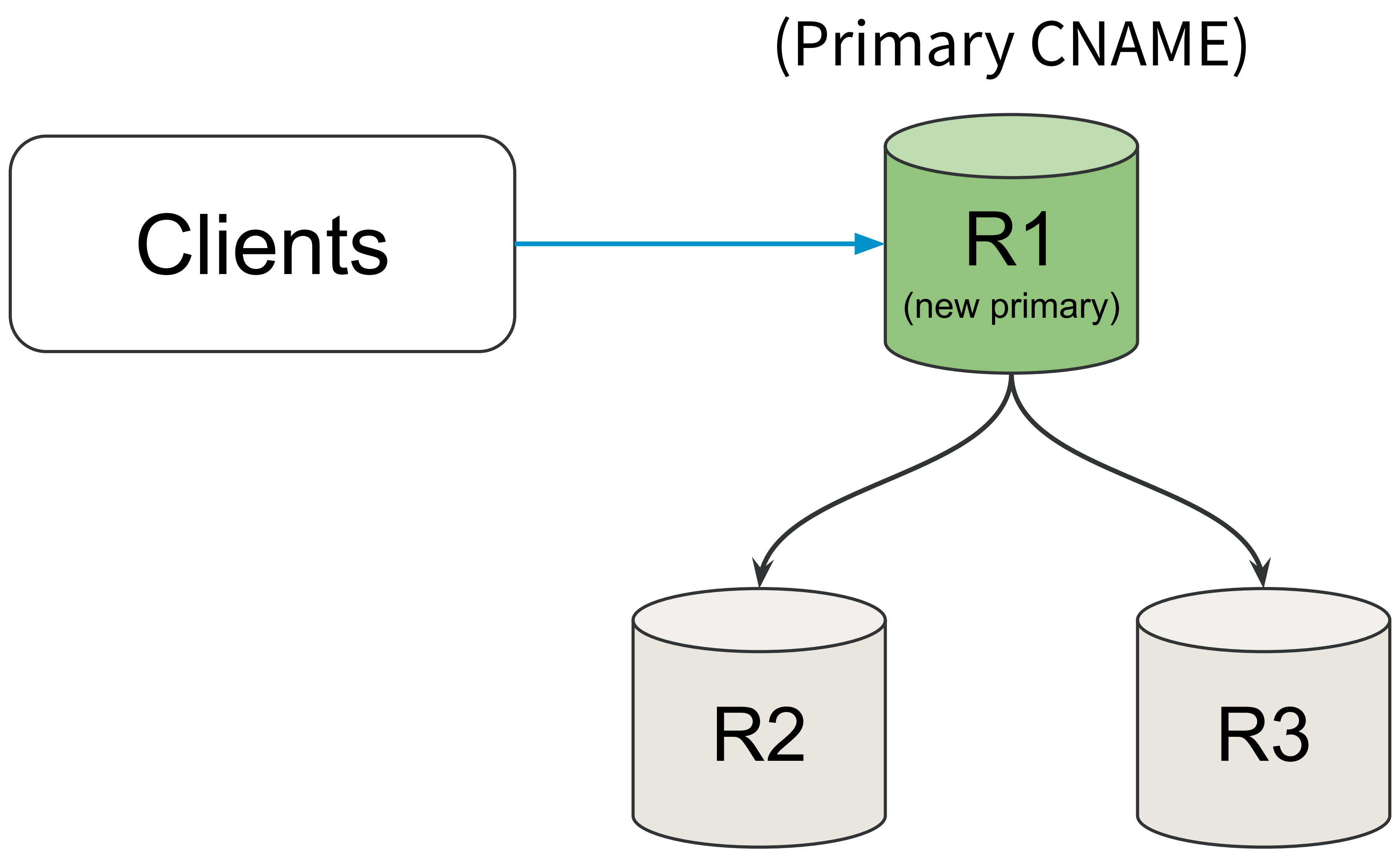
- Ideal Topology: 1 primary, 2 or more replicas
- Clients connect to the read-write Primary using primary cname
- MySQL asynchronous replication in action



- Orchestrator is used for automating HA
- Failover is performed when Primary is unavailable



- A suitable replica is promoted as the new primary.
- But, the clients still connect to the old Primary as the CNAME has not moved yet.



- CNAME is moved in 2-5 mins, then the clients start connecting to the new primary.
- The old primary is added back to spares and another replica is added to the cluster for topology sanitisation.

Metrics

Time to recovery

- Mean TTR: **03:34**
- Median TTR: **02:42**
- Max TTR: **05:40**

CNAME propagation time

- Mean Propagation Time: **03:25**
- Median Propagation Time: **02:14**
- Max Propagation Time: **05:00**

The limiting factor

- DNS propagation times
- DNS caching

Configuring failover mechanism in client drivers

- Some connection drivers like MySQL Connector/J already support connection failovers

```
jdbc:mysql://[primary_cname][:port],[candidate_cname][port]...[/[database]]»

[?propertyName1=propertyValue1[&propertyName2=propertyValue2]...]
```

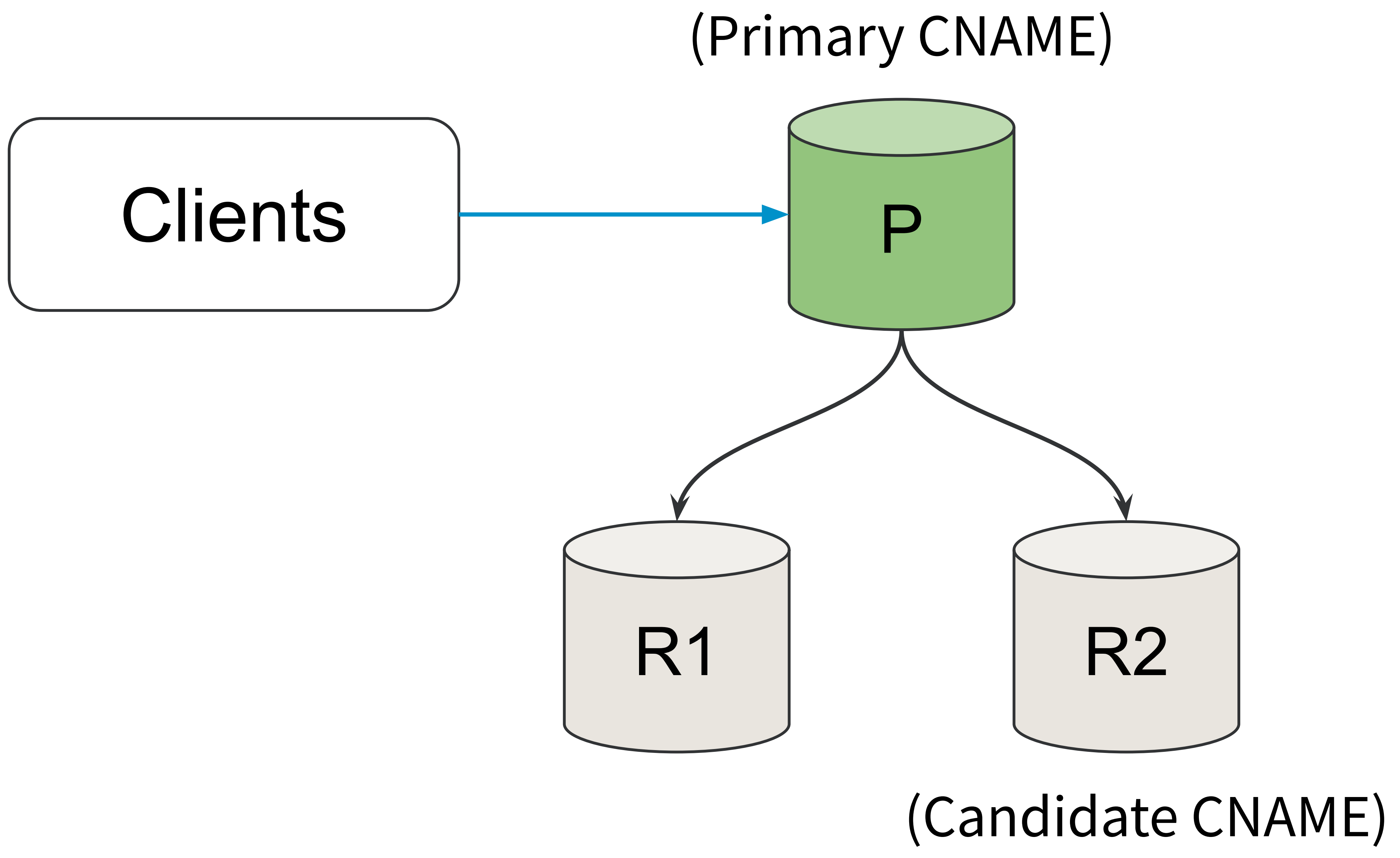
The cue for faster failover

Change of read/write state

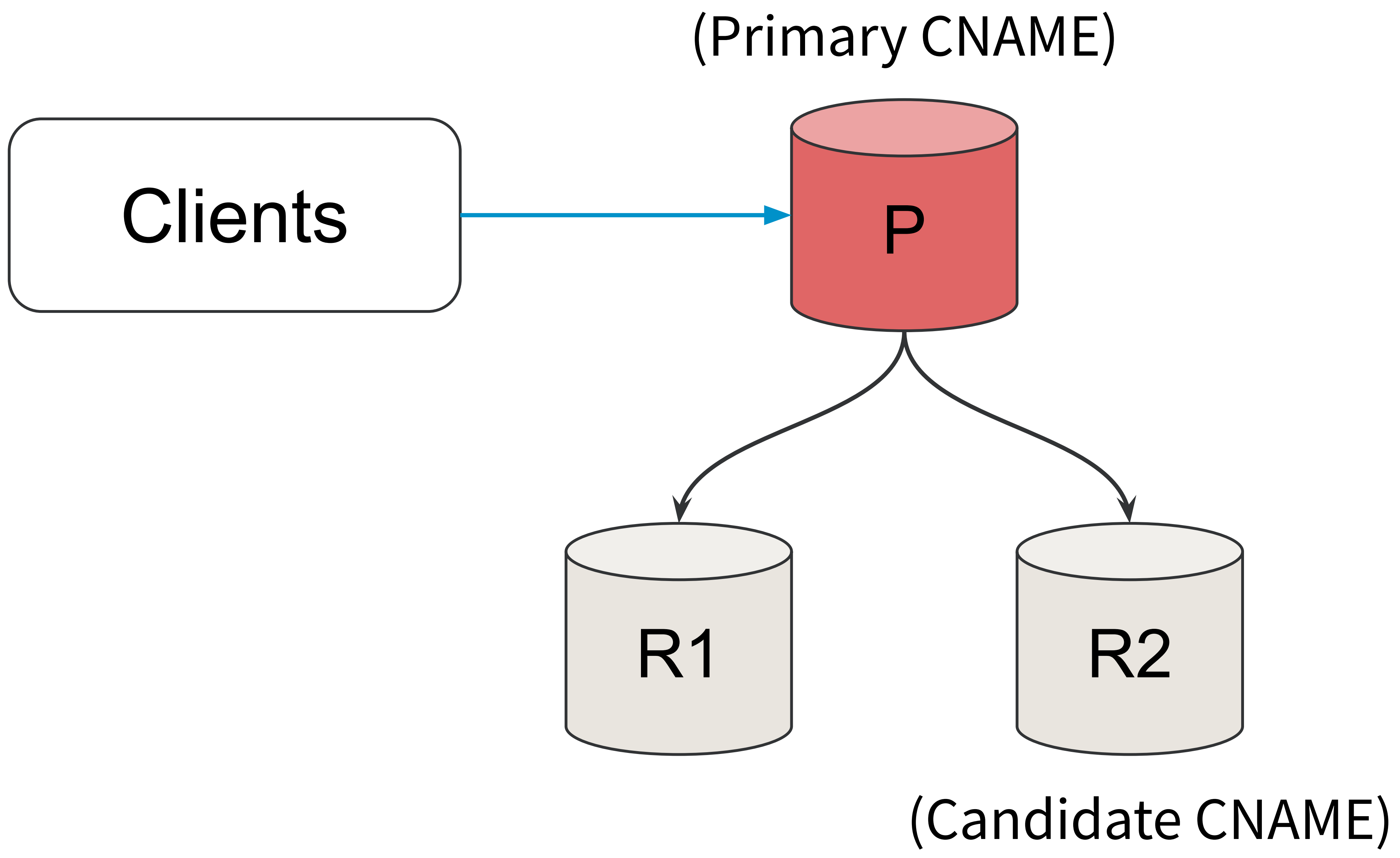
of source and replica

Failover related properties

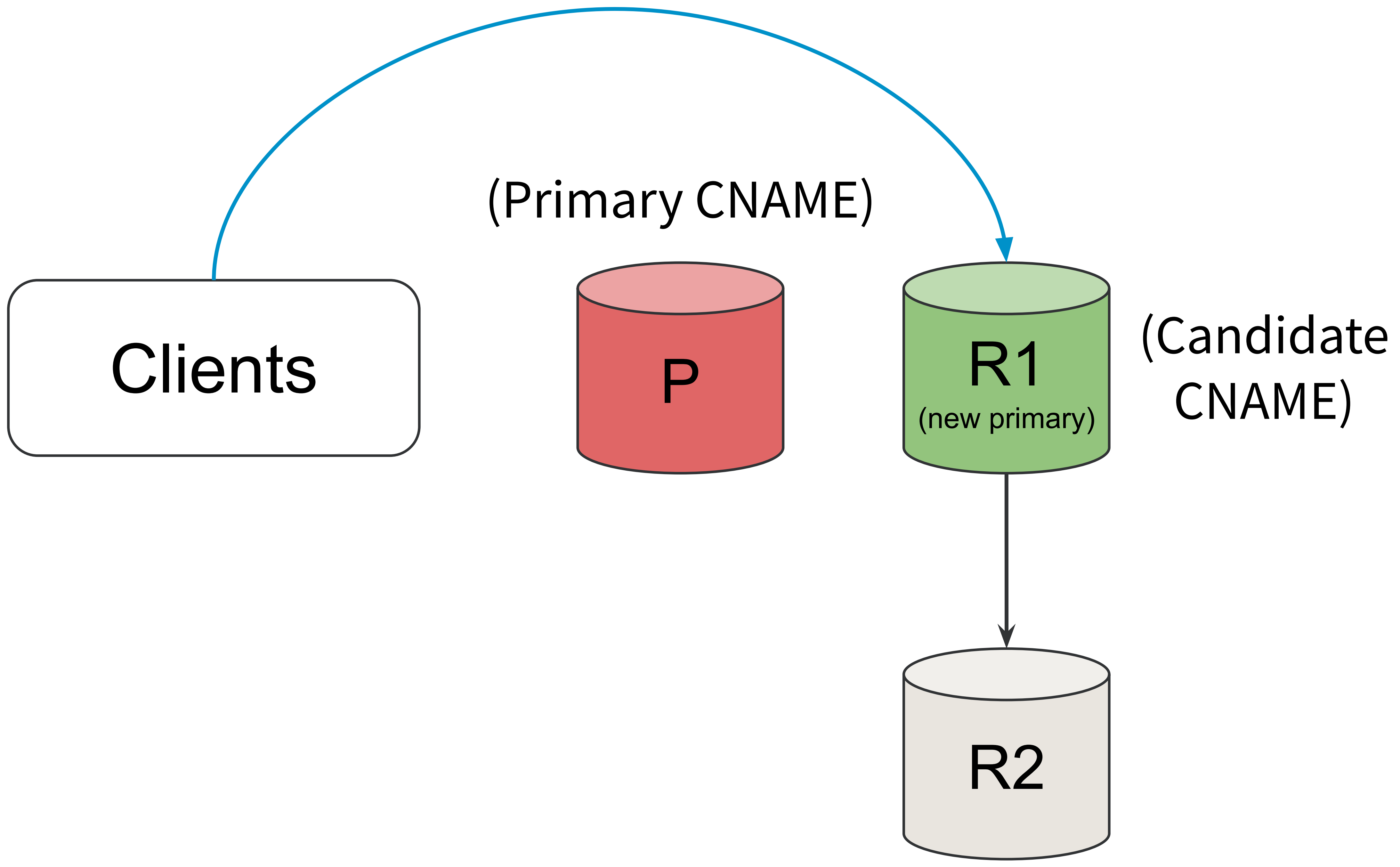
- failovertoReadWrite = true (**Newly introduced into driver code**)
- failOverReadOnly = false
- secondsBeforeRetrySource = 300
- queriesBeforeRetrySource = 0
- autoReconnect = true



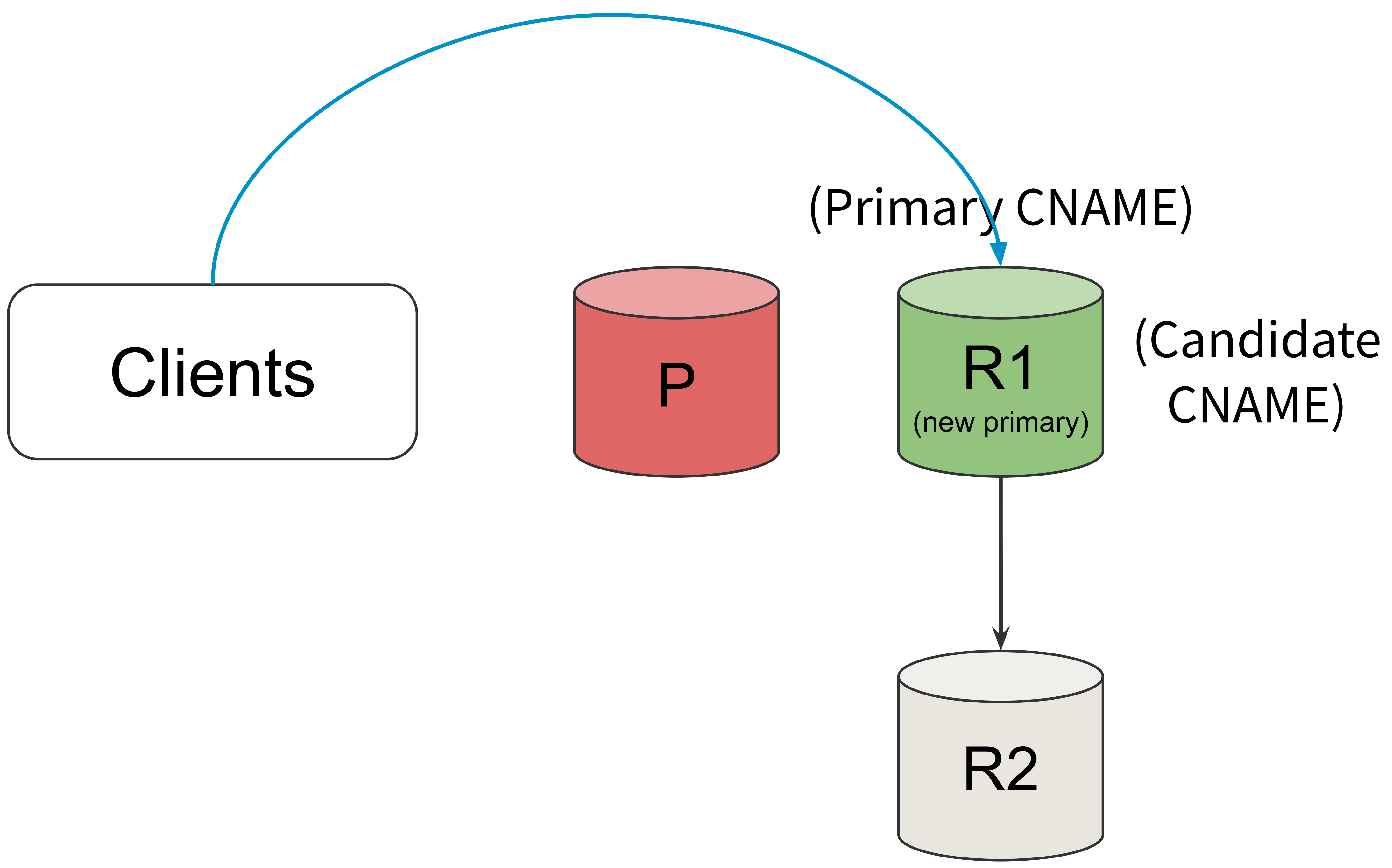
There will be a candidate CNAME along with a primary CNAME in the cluster pointing to a replica.



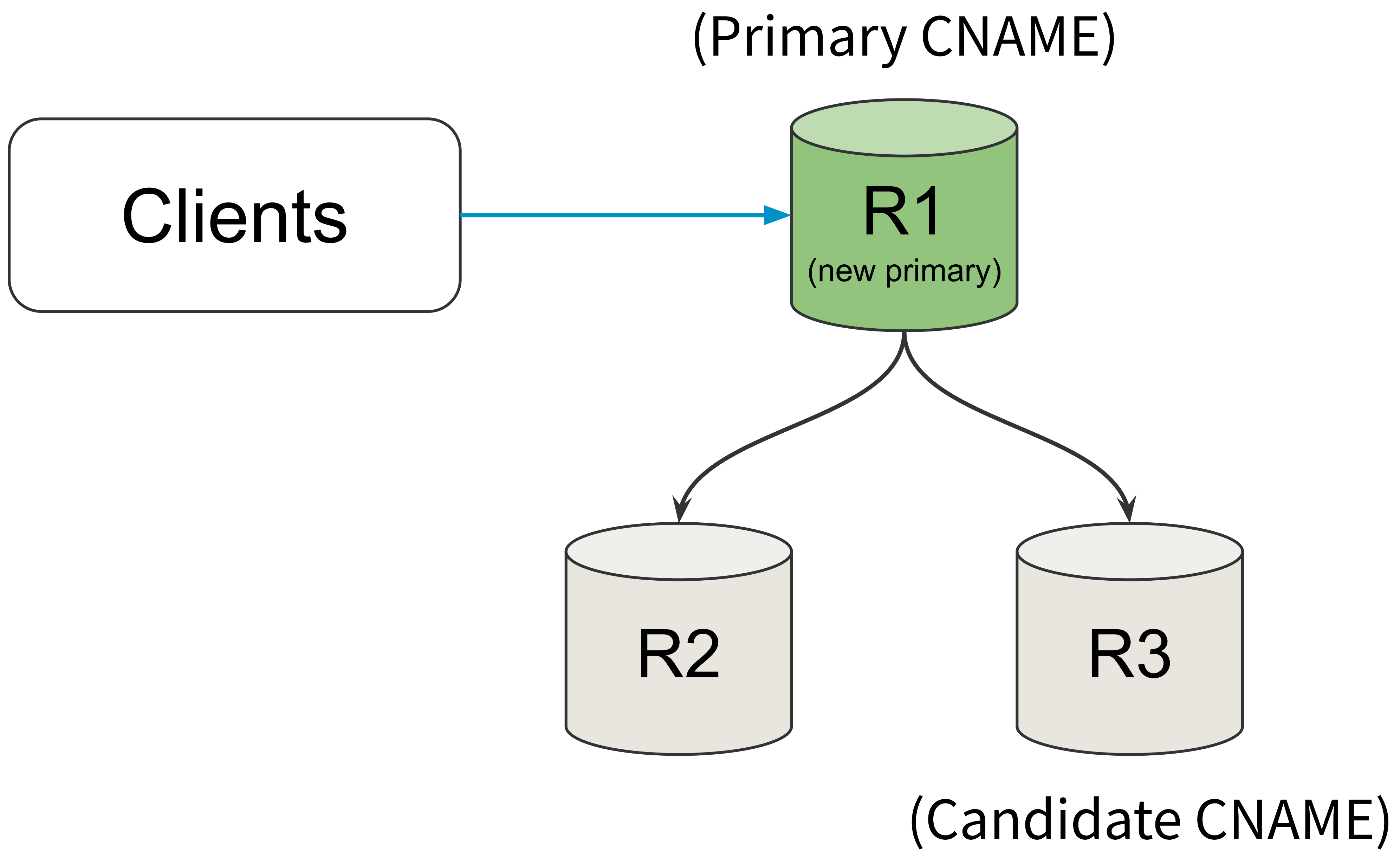
- Now the failover case is similar to the previous scenario.
- A candidate is promoted by Orchestrator on Primary unavailability.



- Based on new connection failover configuration, clients start connecting to the Candidate CNAME in this Scenario.



- After DNS propagation completes, the primary CNAME is also moved to the new primary host.
- The clients start using the primary CNAME now.



- Old Primary is taken out of rotation
- A new replica is built in the cluster and the Candidate CNAME is moved to it

Before

2-5 min

Before connection failover

After

10-20 secs

After connection failover

Agenda

Introduction

Observability

Availability

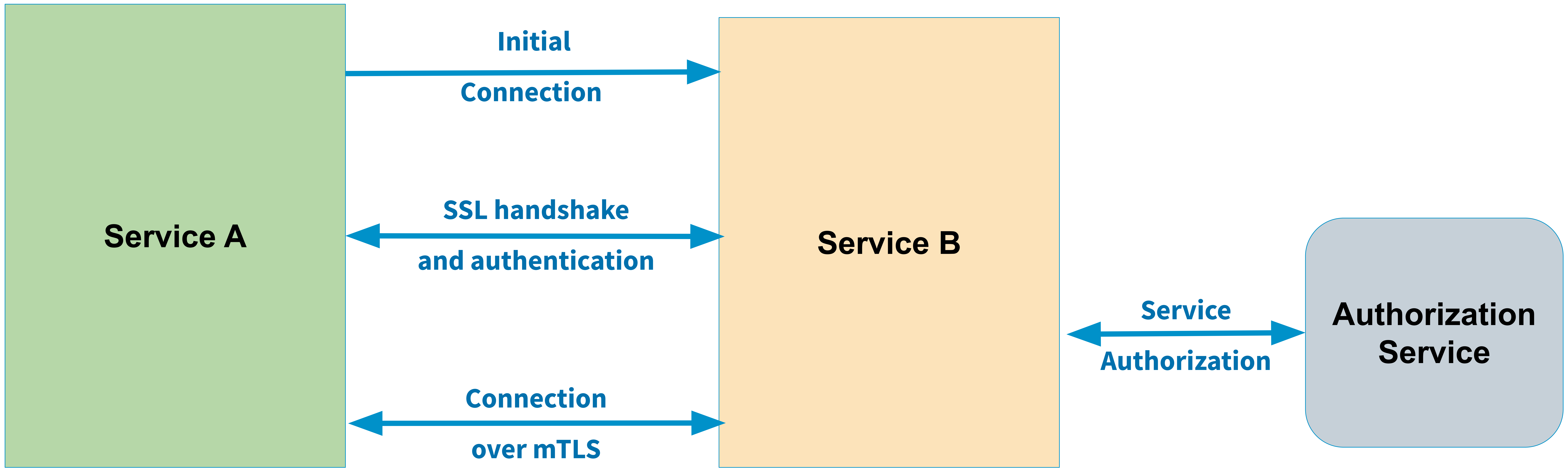
Security

Conclusion

Making security easy

1-click TLS

Authentication and Authorization @ LinkedIn



Authentication Workflow in MySQL

Initial connection
SSL exchange
Authentication method switch
Continue authentication exchange
Authentication response OK/ERR

Cert based Authentication with native MySQL

```
CREATE USER 'jeffrey'@'localhost'
  REQUIRE SUBJECT
'/C=SE/ST=Stockholm/L=Stockholm/
    O=MySQL demo client certificate/
    CN=client/emailAddress=client@example.com'
  AND ISSUER '/C=SE/ST=Stockholm/L=Stockholm/
    O=MySQL/CN=CA/emailAddress=ca@example.com'
  AND CIPHER 'EDH-RSA-DES-CBC3-SHA';
```

Limitation of mTLS with MySQL

- Cannot authenticate against SAN field of the x509 certificate

New authentication plugin for MySQL servers

- Allows us the define the expected SAN field of the certificate for a user
- Retrieves the X509 certificate information provided by client
- Authenticates against the information in the SAN field of the certificate

```
CREATE USER `user_name`@`%` IDENTIFIED WITH
'mysql_san_auth' AS '<service_name>';
```

Agenda

Introduction

Observability

Availability

Security

Conclusion

Conclusion

- Ownership
- Standardization
- Simplification for wider adoption

Quick Recap

- Client side observability with wrappers
- Faster failover for connections
- Making security easy with 1-Click-TLS and cert based authentication

Thank you

Q&A