
AWS
re:Invent

DAT317

Design for success with Amazon ElastiCache best practices

Damon LaCaille

Sr. Solutions Architect

AWS

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Agenda

Modern application requirements

Scalability

High availability

Security

Resources



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Modern application requirements

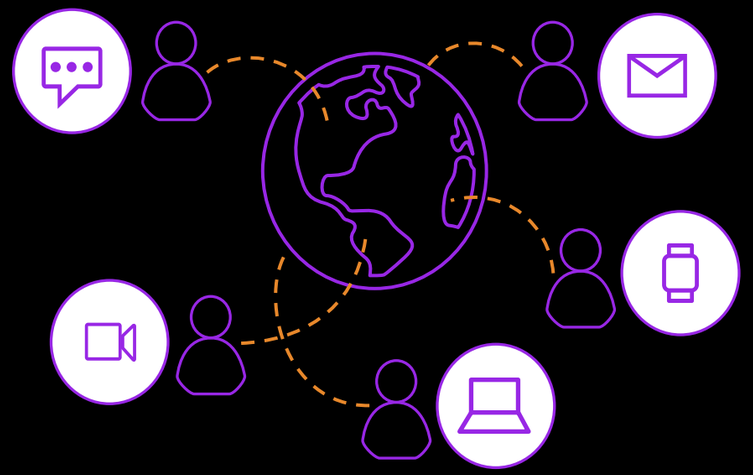


AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Modern application requirements

REQUIRES MORE PERFORMANCE, SCALE, AND AVAILABILITY





E-commerce



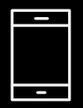
Media streaming



Social media



Online gaming



Shared economy

Users	1M+
Data volume	Terabytes–petabytes
Locality	Global
Performance	Microsecond latency
Request rate	Millions per second
Access	Mobile, IoT, devices
Scale	Up, down, out, or in
Economics	Pay-as-you-go
Developer access	Open API

Amazon ElastiCache

MANAGED, REDIS OR MEMCACHED-COMPATIBLE IN-MEMORY DATA STORE



Scalable

Write and memory scaling with sharding.
Non-disruptive scaling.
Read scaling with replicas (Redis).



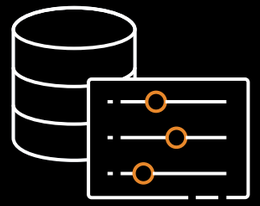
Extreme performance

In-memory data store and cache for sub-millisecond response times.



Fully managed

AWS manages all hardware and software setup, configuration, monitoring.



OSS compatible

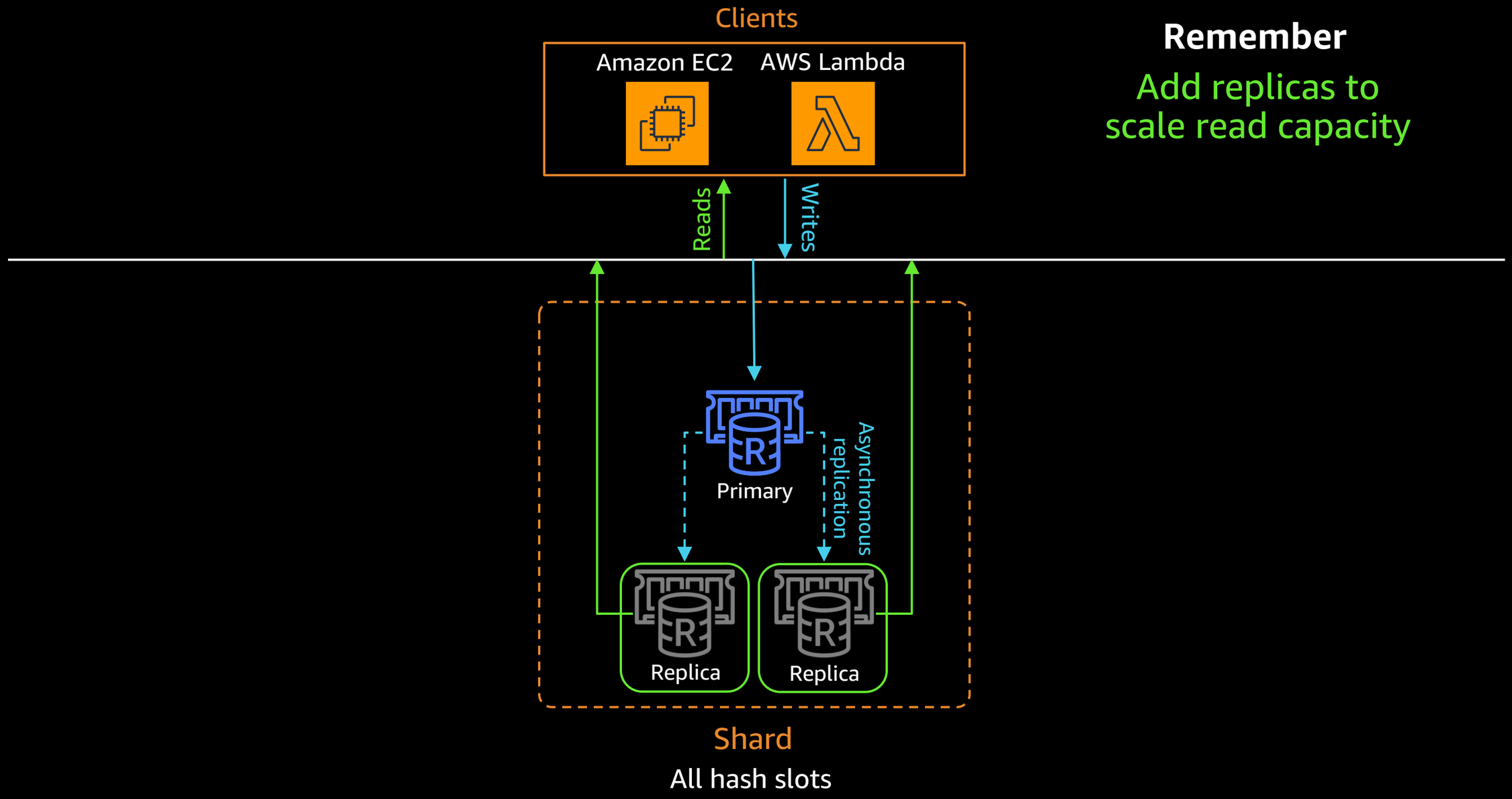
Fully compatible with open source Redis and Memcached engines.

Scalability



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

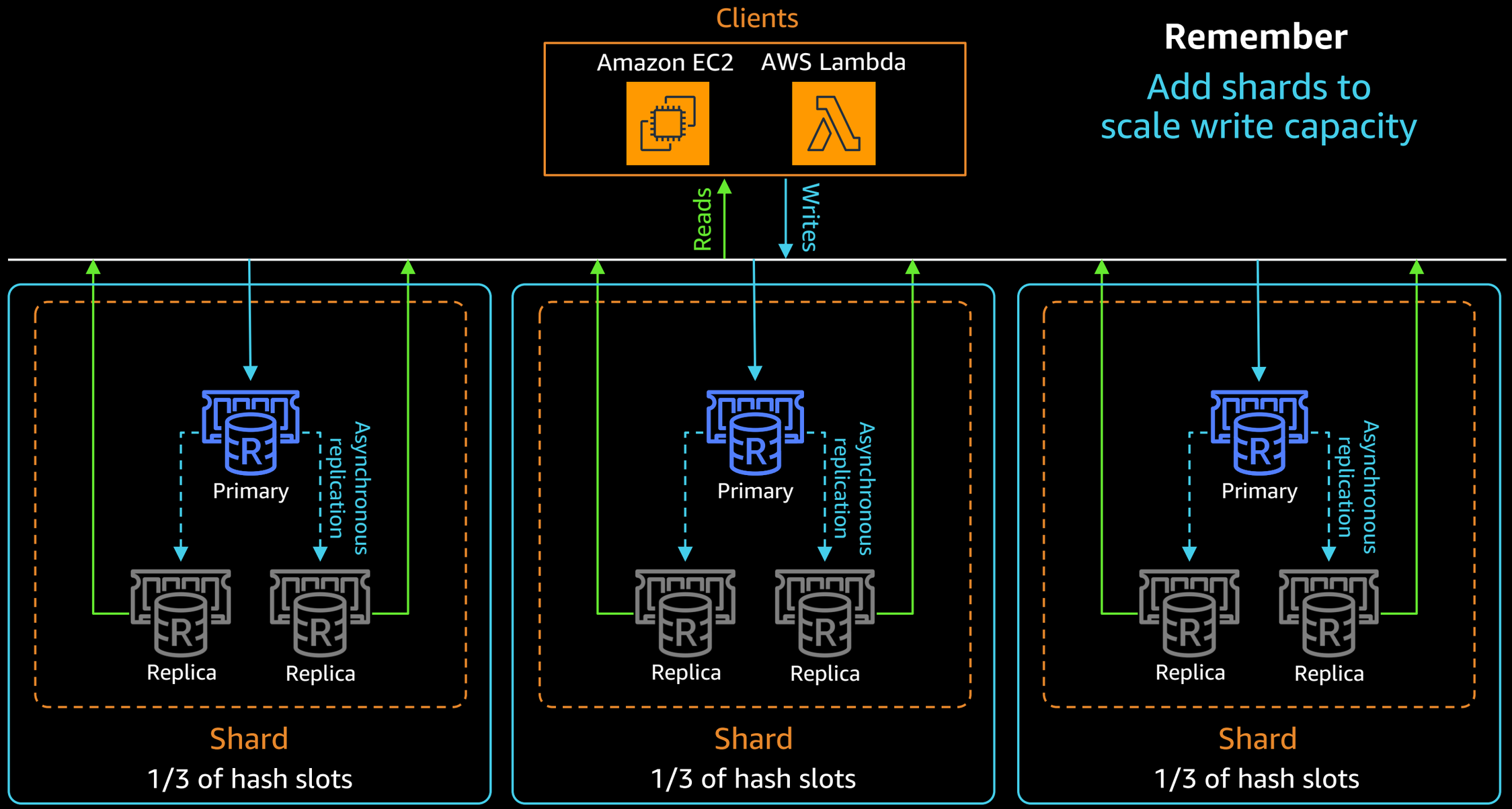
Scalability – Reads



Remember

Add replicas to scale read capacity

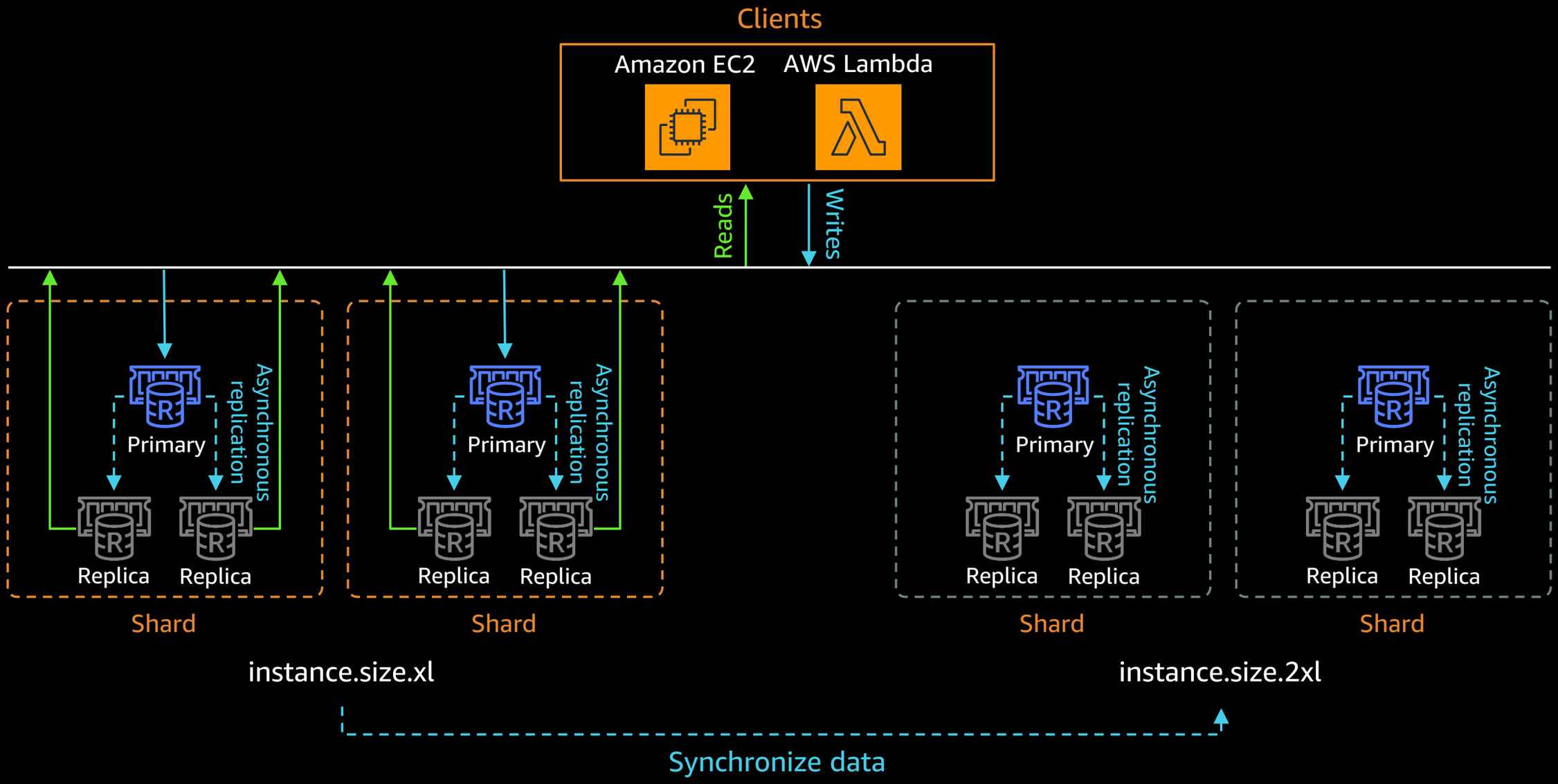
Scalability – Writes



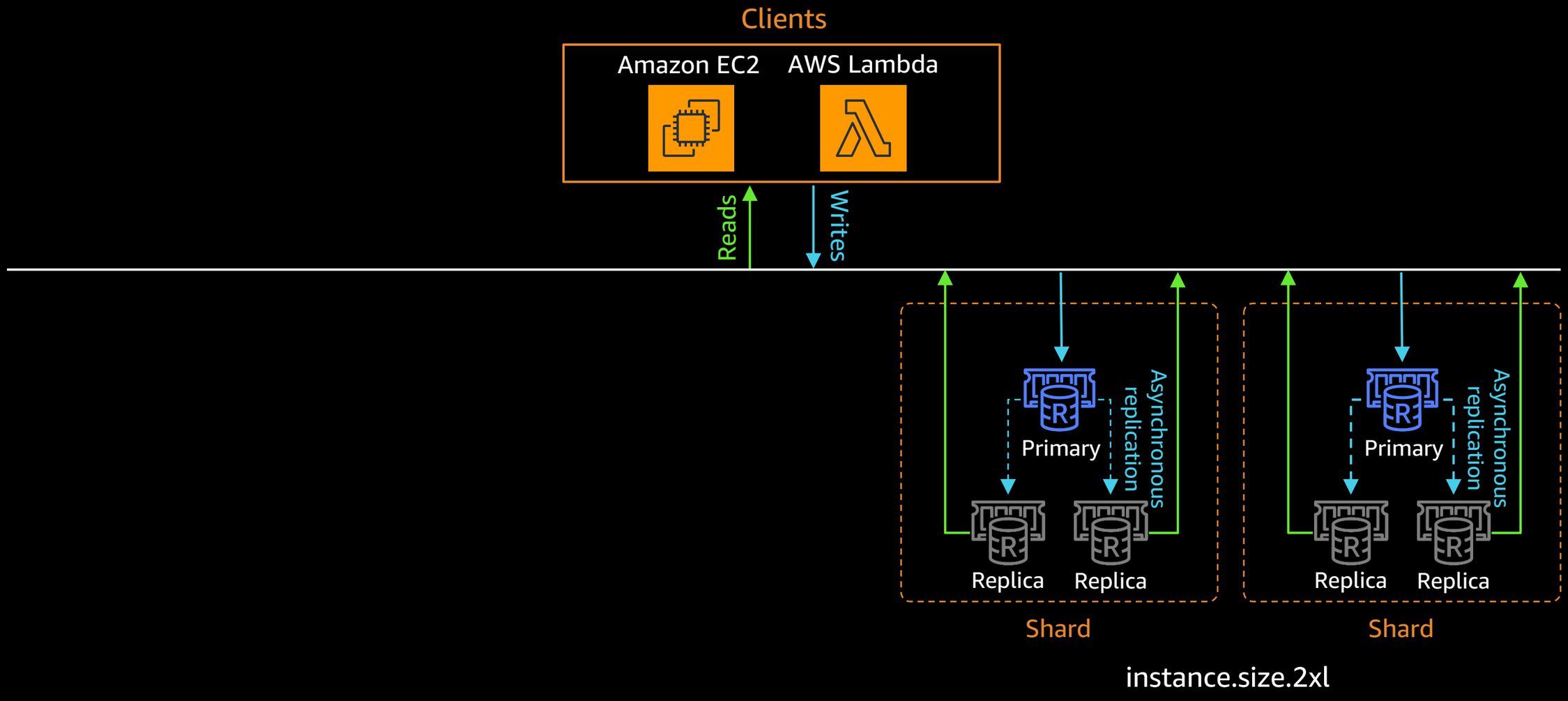
Remember

Add shards to scale write capacity

Scalability – Vertical



Scalability – Vertical



Scalability

	Vertical scaling	Horizontal scaling
Action	All instances are changed	Individual shards are added or removed
Cost	Cluster-wide change	Incremental change
Impact	Clients reconnect	Clients stay connected
Supported	Cluster mode enabled and disabled	Cluster mode enabled

Scalability – Recap

Best practices summary

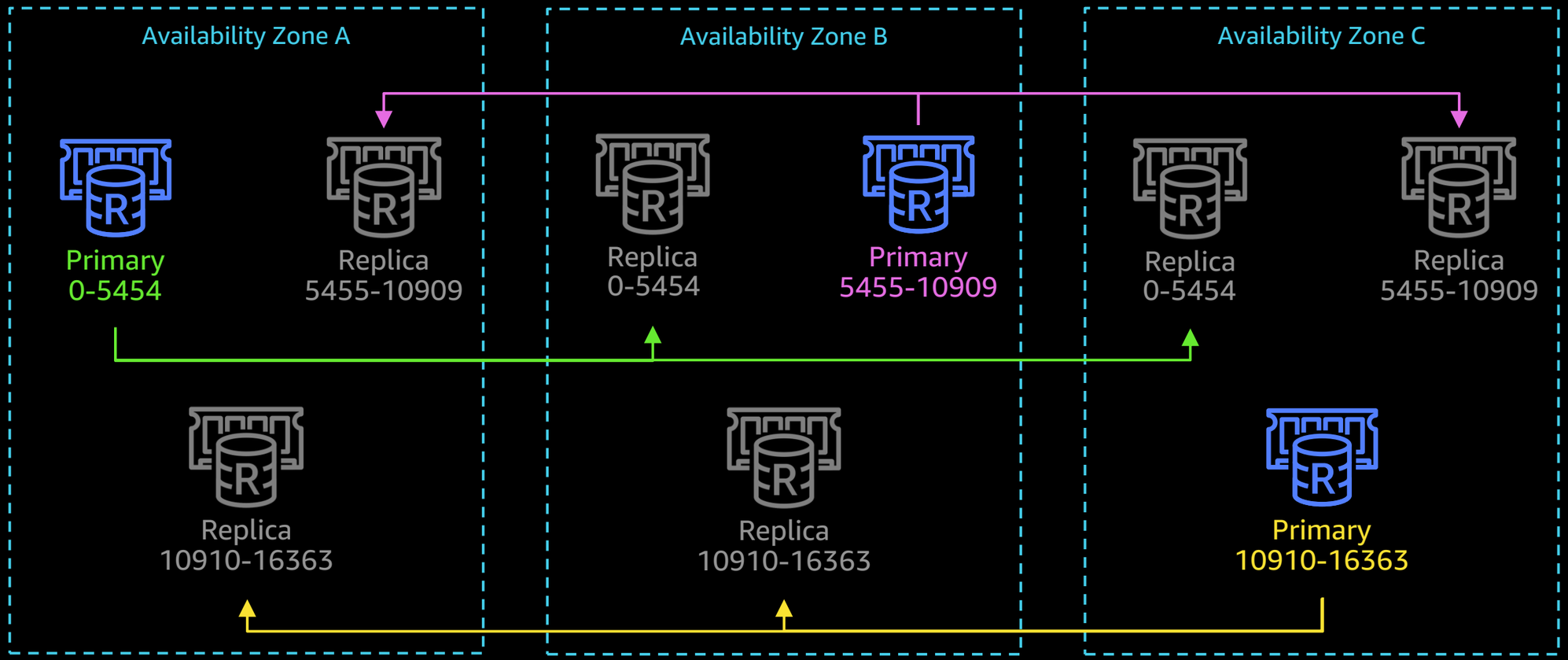
- Cluster-mode-enabled cluster
- Add replicas to increase read capacity
- Add shards to increase write capacity
- Amazon CloudWatch metrics and alarms
- Scale before you reach capacity
- Amazon Simple Notification Service (Amazon SNS) notifications

High availability

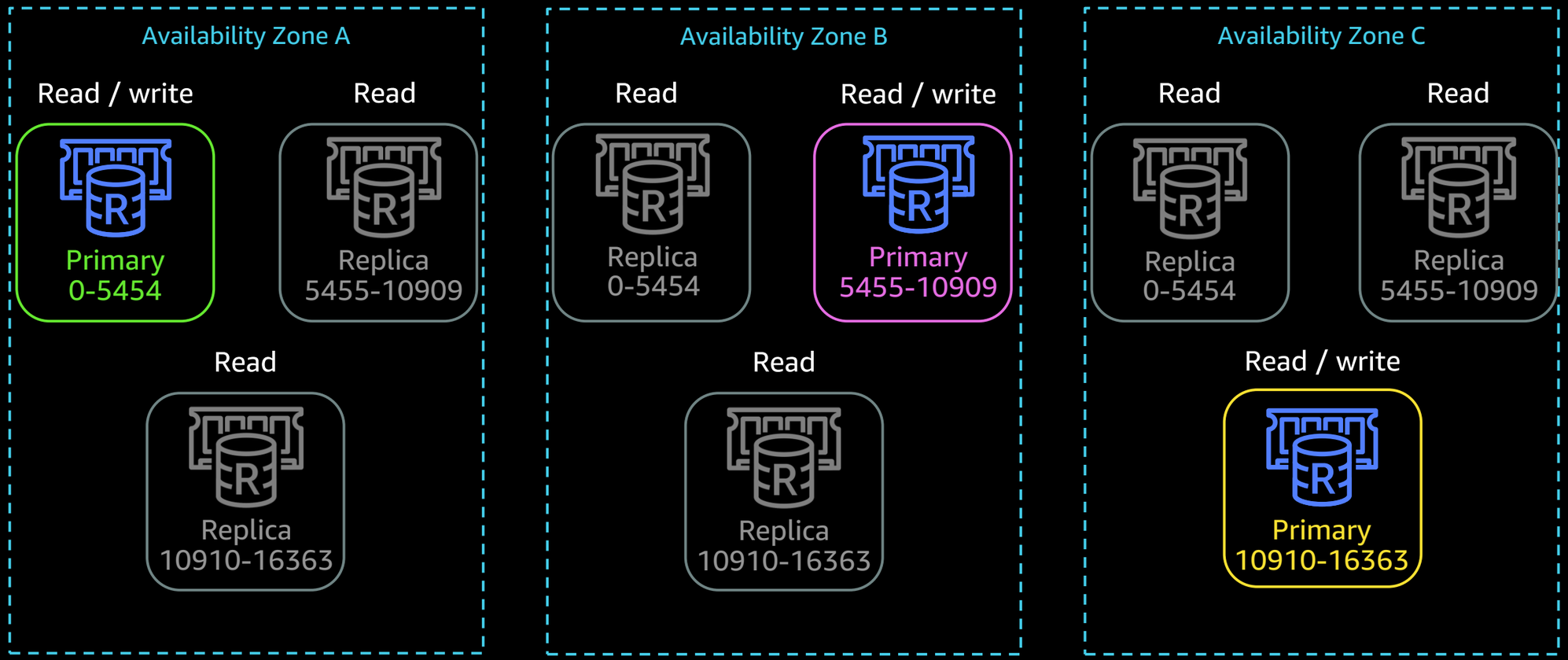


© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

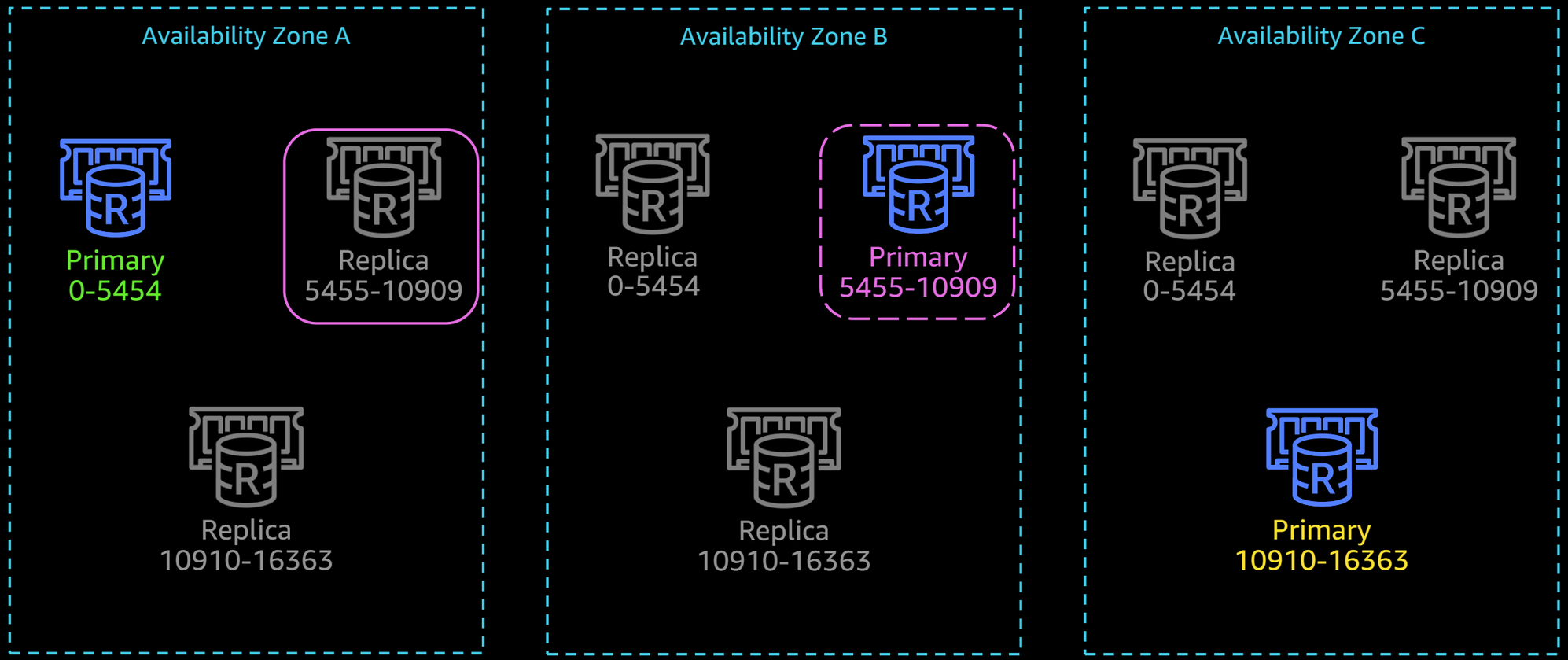
High availability – Replication



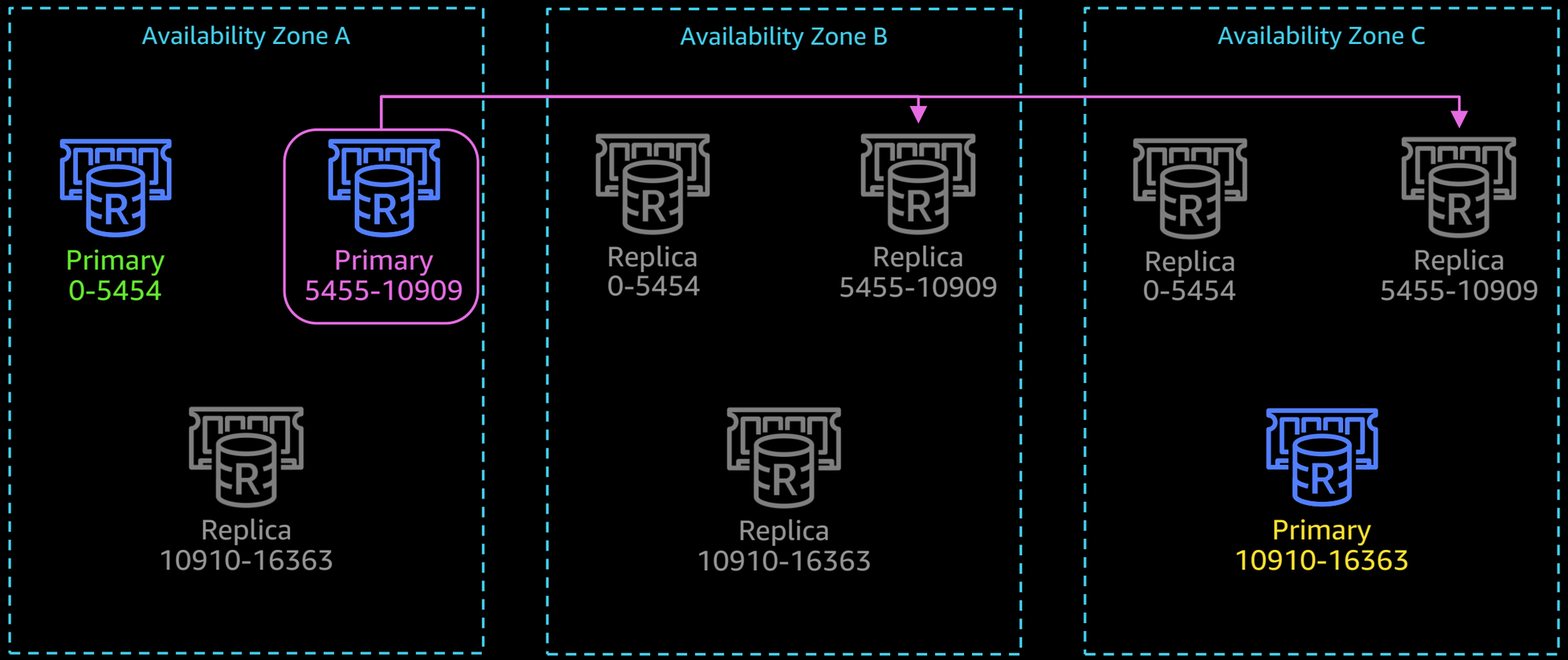
High availability – Replication



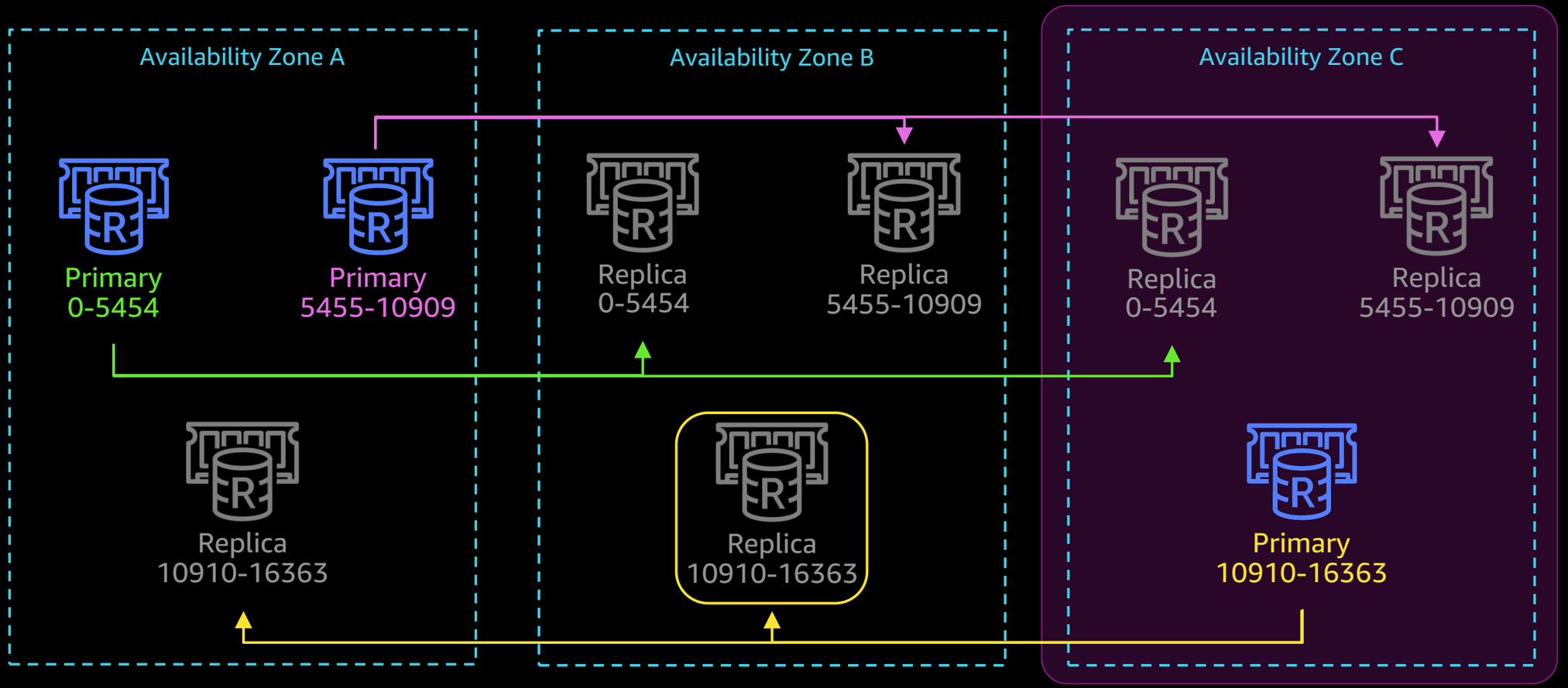
High availability – Primary node failure



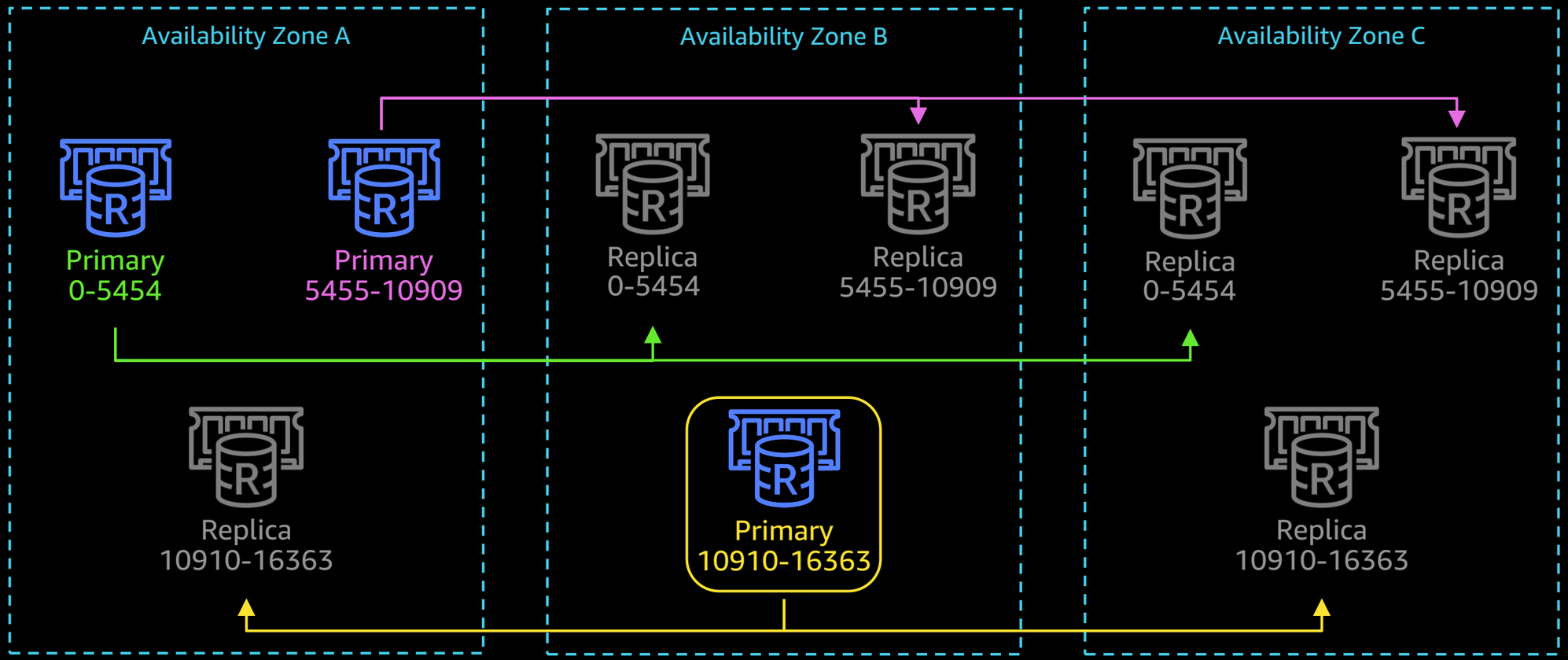
High availability – Primary node failure



High availability – Availability Zone failure



High availability – Availability Zone failure



High availability – Recap

Best practices summary

- Two read replica nodes per shard
- Deploy primary nodes across Availability Zones
- Deploy replica nodes in different Availability Zones from their respective primary nodes
- Test your application with the failover primary feature

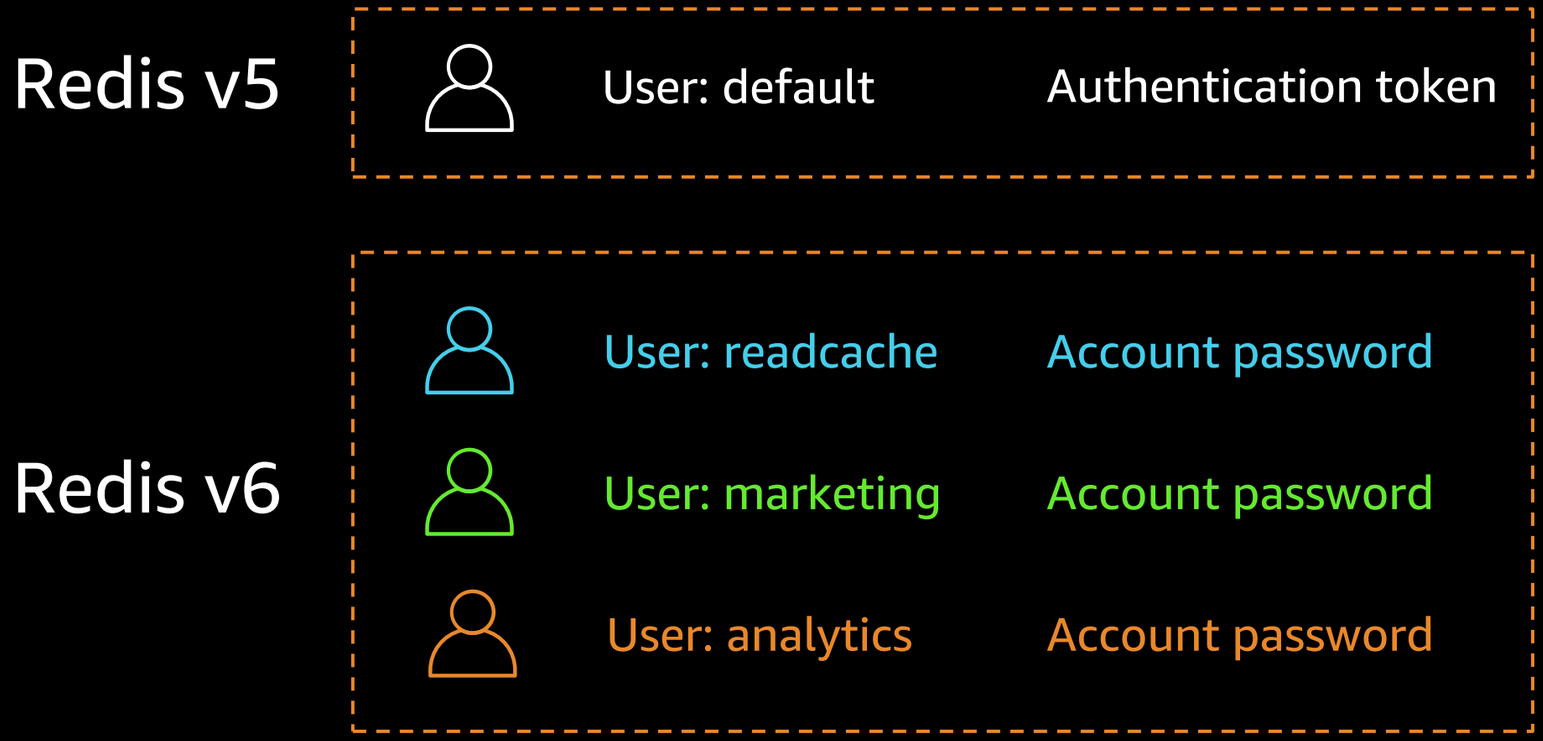
Security



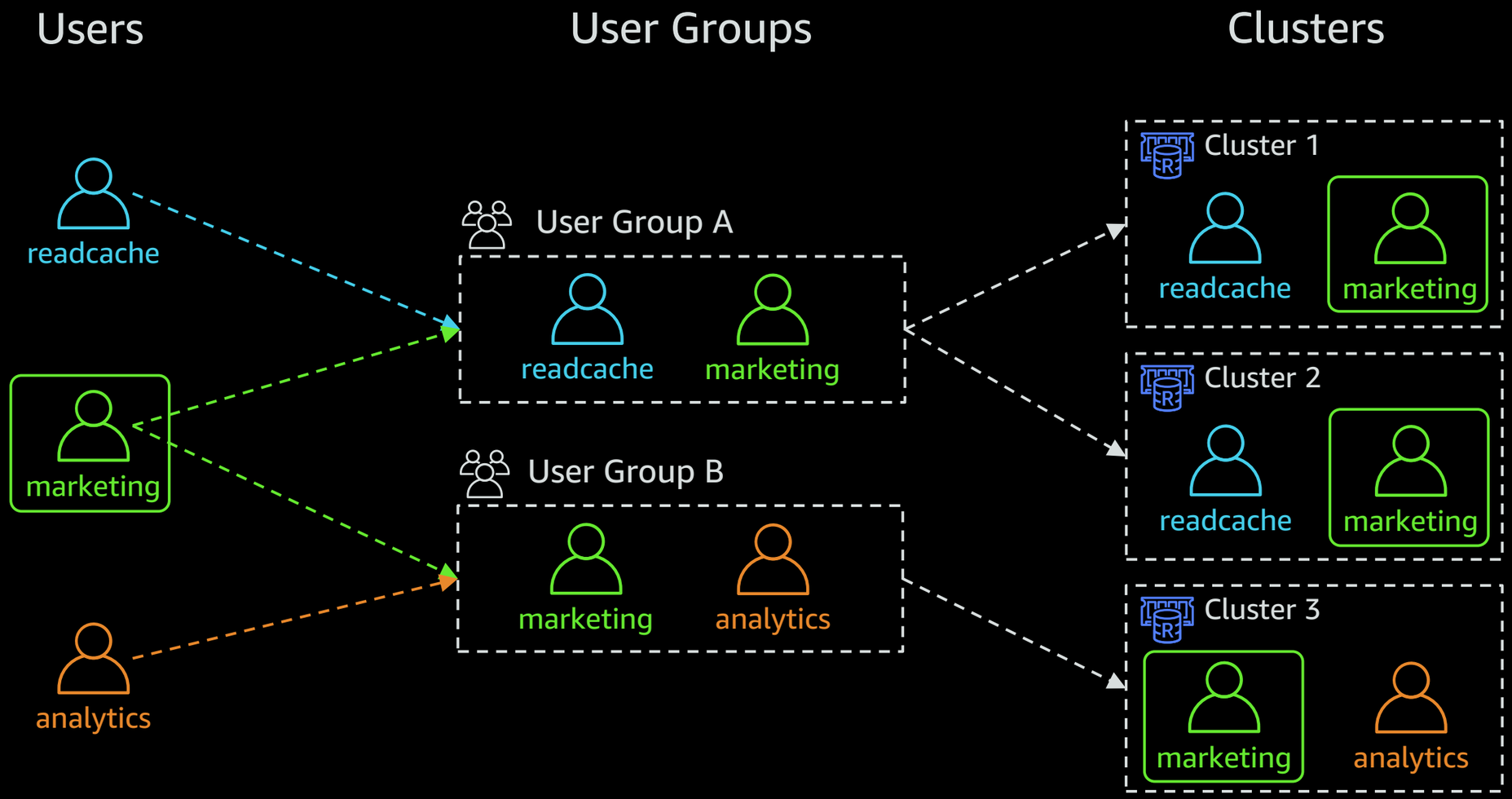
AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Security – Authentication



Security – Authentication



Security – Authorization

User	Authorized	Key space	Access string
 readcache	**GET** command	Cache keys	`on ~cache:* -@all +get`
 marketing	All commands	Marketing keys	`on +@all ~mktg:* ~marketing:*`
 analytics	All read-only commands	All keys	`on ~* +@read`

Security – Example

```
ec2> redis-cli -h cluster_endpoint --user marketing --askpass
Please input password: ***********
> set mktg:a 1
OK
> get mktg:a
"1"
> get cache:a
(error) NOPERM this user has no permissions to access one of the keys used as arguments
```

```
ec2> redis-cli -h cluster_endpoint --user readcache --askpass
Please input password: ***********
> get cache:a
"123"
> set cache:a 345
(error) NOPERM this user has no permissions to run the 'set' command or its subcommand
```

Security – Monitoring

AuthenticationFailures

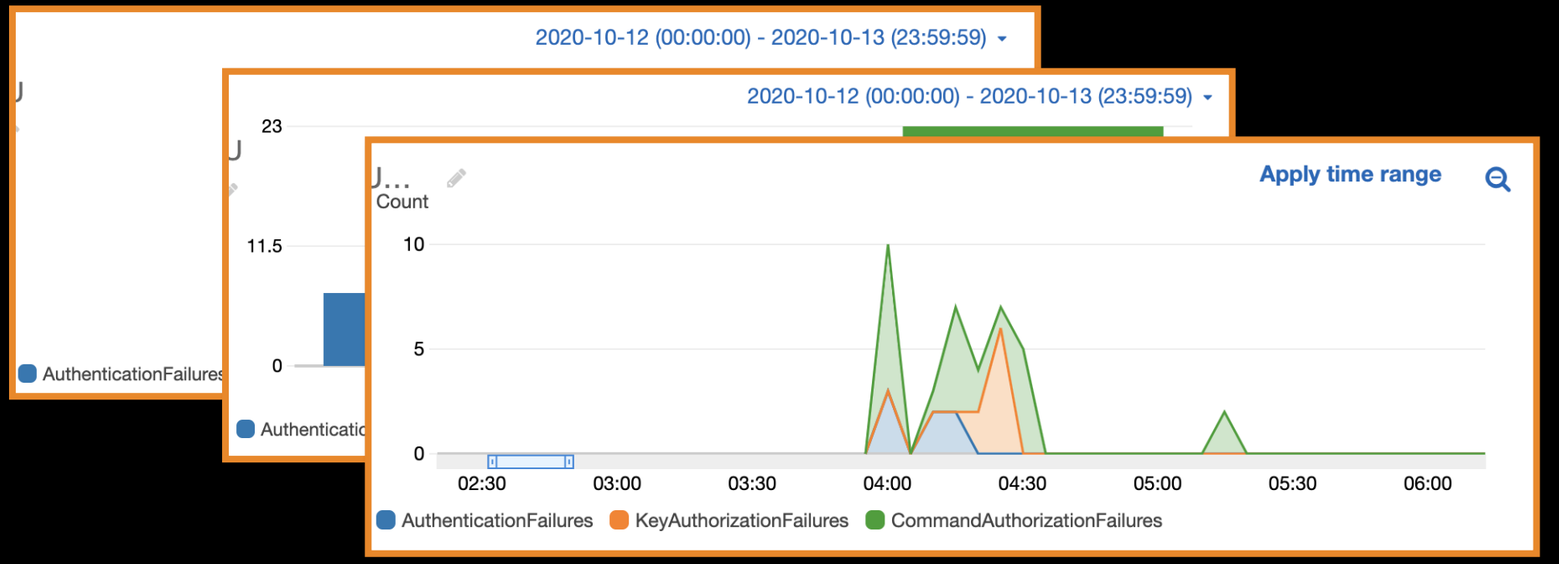
Failed attempts to authenticate to Redis

KeyAuthorizationFailures

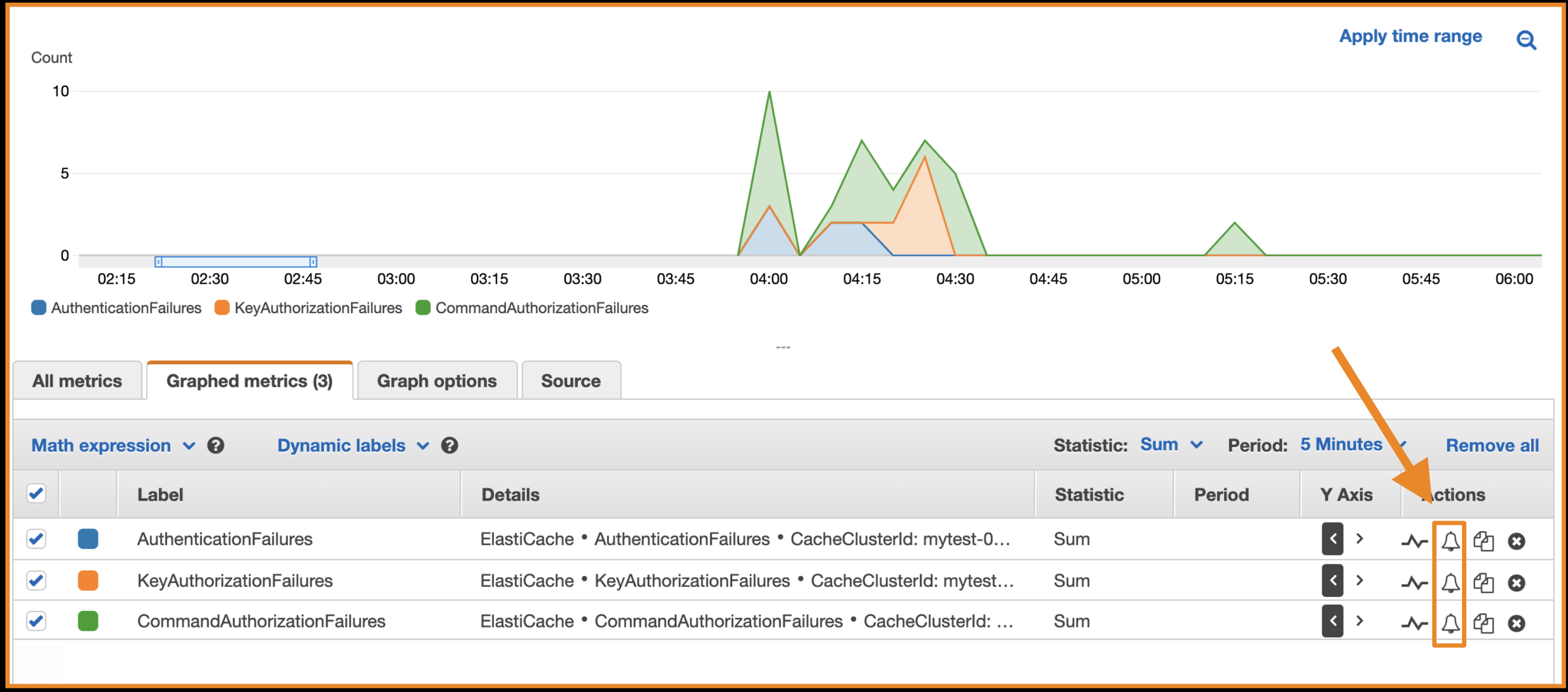
Failed attempts by users to access keys without permission

CommandAuthorizationFailures

Failed attempts by users to run commands without permission



Security – Notifications



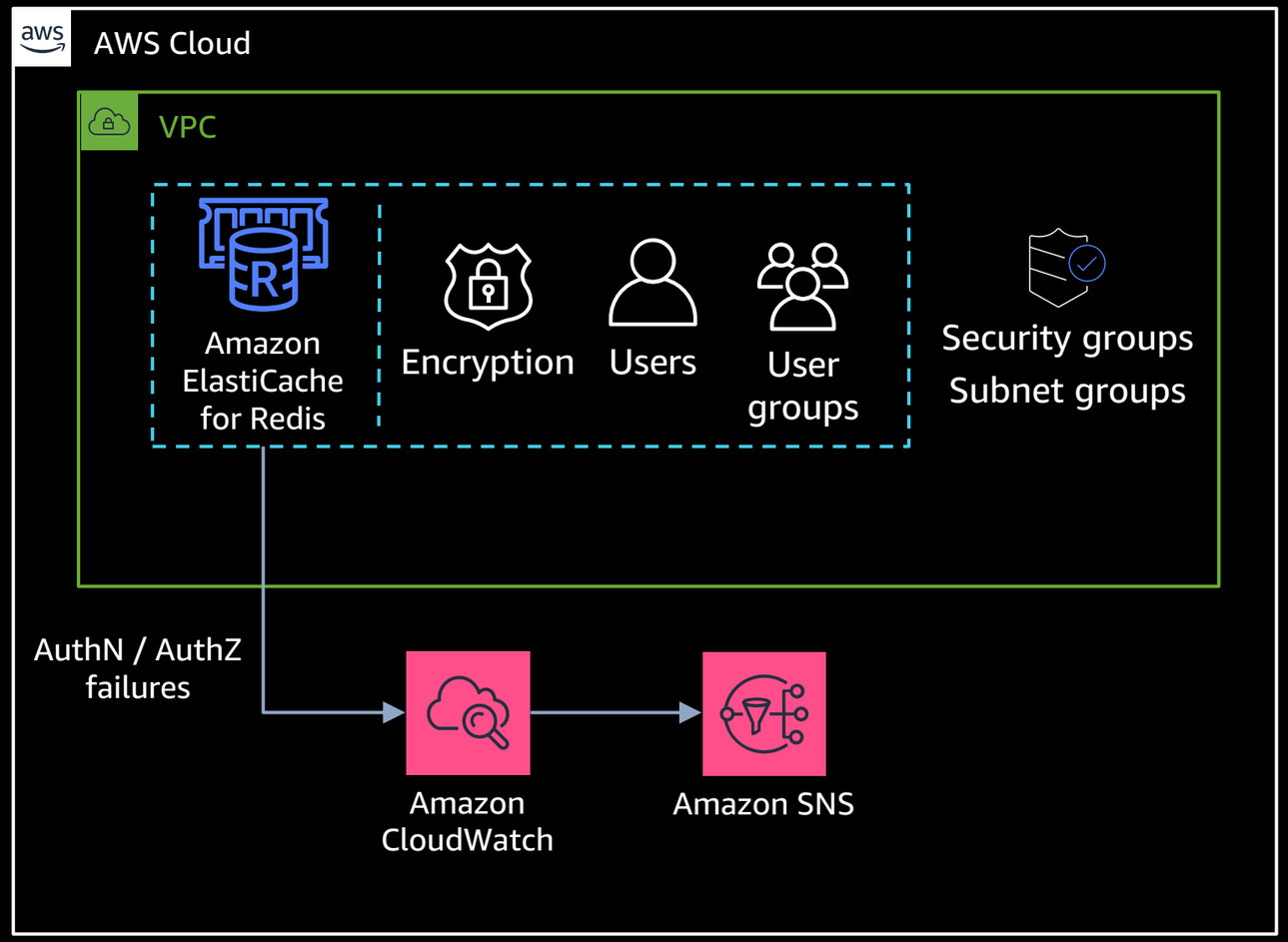
Security

Run the **ACL LOG** command to learn more

```
> acl log
```

```
localhost:6379> acl log
 1)  1) "count"
     2) (integer) 1
     3) "reason"
     4) "command"
     5) "context"
     6) "toplevel"
     7) "object"
     8) "set"
     9) "username"
    10) "cacheuser"
    11) "age-seconds"
    12) "462.910000000000003"
    13) "client-info"
    14) "id=5682 addr=172.31.43.79:60048 fd=9 name= age=3
        idle=0 flags=N db=0 sub=0 psub=0 multi=-1 qbuf=27
        qbuf-free=32741 obl=0 oll=0 omem=0 events=r
        cmd=set user=cacheuser"
```

Security – Recap



Best practices summary

- Use encryption in transit and at rest
- Use Role-Based Access Control (RBAC) and user passwords
- Configure strong security group controls
- Monitor AuthN / AuthZ failures
- Set up notifications
- Use ElastiCache for Redis 6

Resources



© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Resources

FAQ	https://aws.amazon.com/elasticache/faqs/
Blog	https://aws.amazon.com/blogs/database/tag/elasticache/
Forum	https://forums.aws.amazon.com/forum.jspa?forumID=127
Getting started	https://aws.amazon.com/elasticache/redis/resources/

Related sessions

DAT209

What's new in Amazon ElastiCache



DAT306

Beyond caching: Advanced design patterns in Redis



Thank you!



AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.



Please complete the session survey

AWS re:Invent

© 2020, Amazon Web Services, Inc. or its affiliates. All rights reserved.