Performance Test Results for ScaleArc for MySQL on Aurora RDS

Nov 2016

ScaleArc for MySQL – Aurora RDS Testing

- ScaleArc has updated its ScaleArc for MySQL database load balancing software to be compatible with Aurora RDS – the upgrade involved joint effort between Aurora and ScaleArc Engineering Teams

- Sixgill (ScaleArc customer) ran performance tests of the ScaleArc code in their AWS environment

ScaleArc™

Sixgill™
More Mobility. More Control.

Amazon Aurora

ScaleArc and Aurora RDS – Testing Summary

Test Areas:

1. Latency: Validate that ScaleArc does not add any additional latency
2. Read/write split and load Balancing: Perform automatic read/write split, sending the writes to the primary and load balancing the reads across multiple replicas
3. Caching: Measure performance improvement by using ScaleArc caching

Test Results:

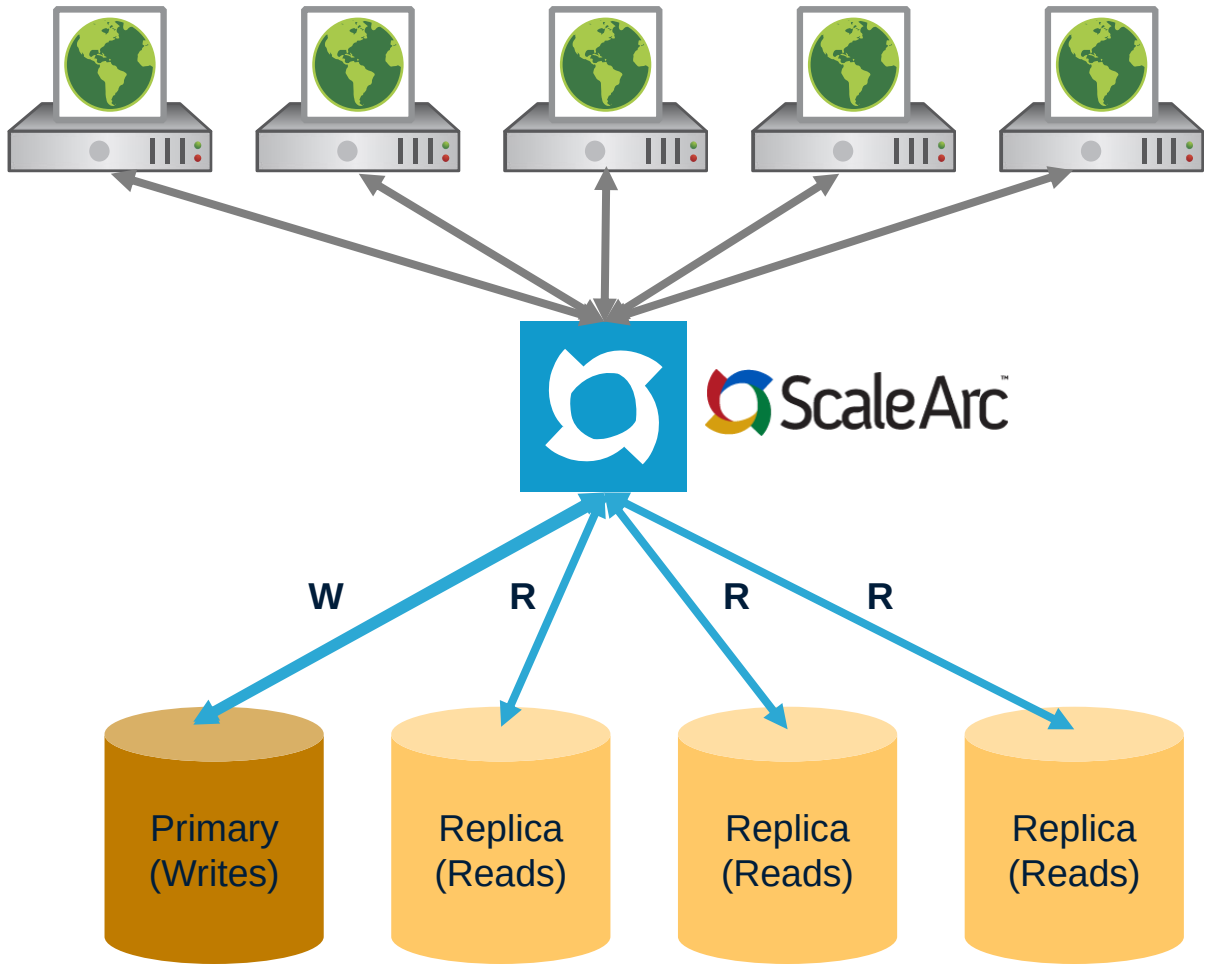
✓ Latency: No additional latency with ScaleArc inserted between the application and Aurora RDS instances

✓ Read/write split and Load Balancing: ScaleArc performed automatic read/write split across the Aurora primary and replicas with no code changes needed, enabling the primary to process more write traffic and load balancing reads across the replicas

✓ Caching: Processed 4x the number of read queries with ScaleArc cache enabled

ScaleArc

Test Environment – Setup



Load Generators
Sysbench, OLTP
5 nodes, r3.2xlarge

ScaleArc
r3.8xlarge

Aurora Cluster
64 Tables,10K Records
4 nodes, db.r3.xlarge

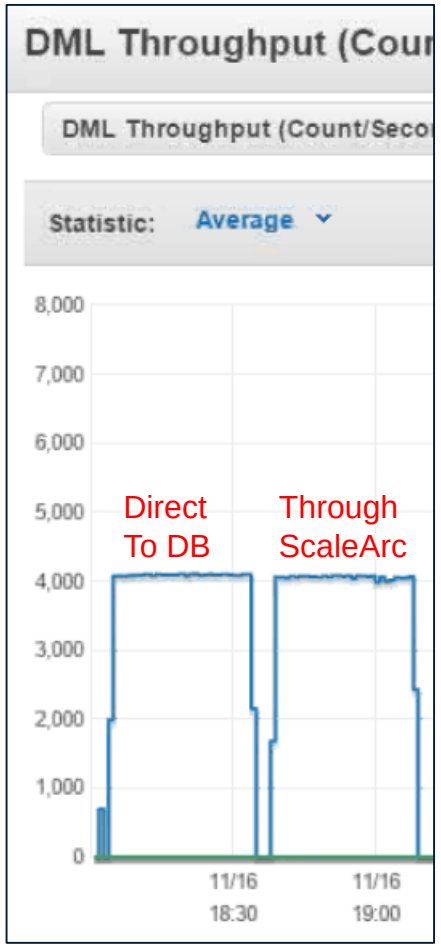
ScaleArc

Test #1: Latency – Methodology

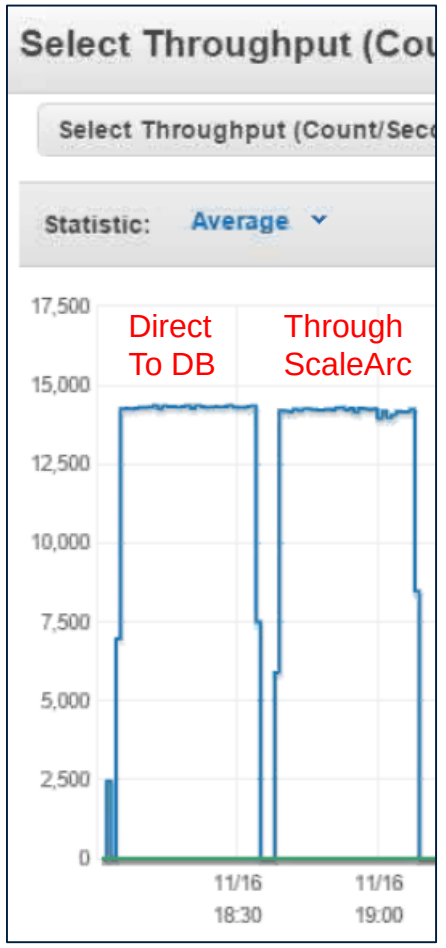
A mix of read and write queries would be executed directly on the Aurora cluster using only a single node to serve the traffic. Repeat the same test with ScaleArc software instance in the middle to measure latency, if any, ScaleArc adds in the path.

Test #1: Latency – Results

- Direct Database
  - DML Throughput – 4K QPS
  - Select Throughput – 14K QPS
  - Database CPU – 100% utilization
- Through ScaleArc
  - DML Throughput – 4K QPS
  - Select Throughput – 14K QPS
  - Database CPU – 100% utilization
  - ScaleArc CPU – 7% utilization
- **Result: ScaleArc does not add latency**



DML Query Tests



Select Query Tests

ScaleArc

Test #2: Read/write Split and Load Balancing – Methodology

A mix of read and write queries would be executed on Aurora cluster with ScaleArc inserted. Start with a single node and gradually increase the number of nodes to four in the Aurora cluster, demonstrating read/write split and load balancing features of ScaleArc.

Test #2: Read/write Split and Load Balancing – Results

Using query-level load balancing

# of Nodes	DML Throughput	Select Throughput	Total Throughput	ScaleArc CPU Utilization
1	4,000	14,000	18,000	7%
2	6,000	21,500	27,500	18%
3	7,000	24,000	31,500	20%
4	7,500	26,500	34,000	23%

Using query routing

# of Nodes	DML Throughput	Select Throughput	Total Throughput	ScaleArc CPU Utilization
1	4,000	14,000	18,000	7%
2	6,000	20,000	26,000	13%
3	9,000	31,000	40,000	15%
4	10,000	36,000	46,000	16%

- ScaleArc load balanced read traffic across multiple replicas
- Because of offloading read traffic to replica nodes, primary node was able to process more write traffic
- CPU utilization on ScaleArc is very minimal while load balancing reads across replicas
- No code changes are required to achieve read/write split or load balancing of read traffic

ScaleArc

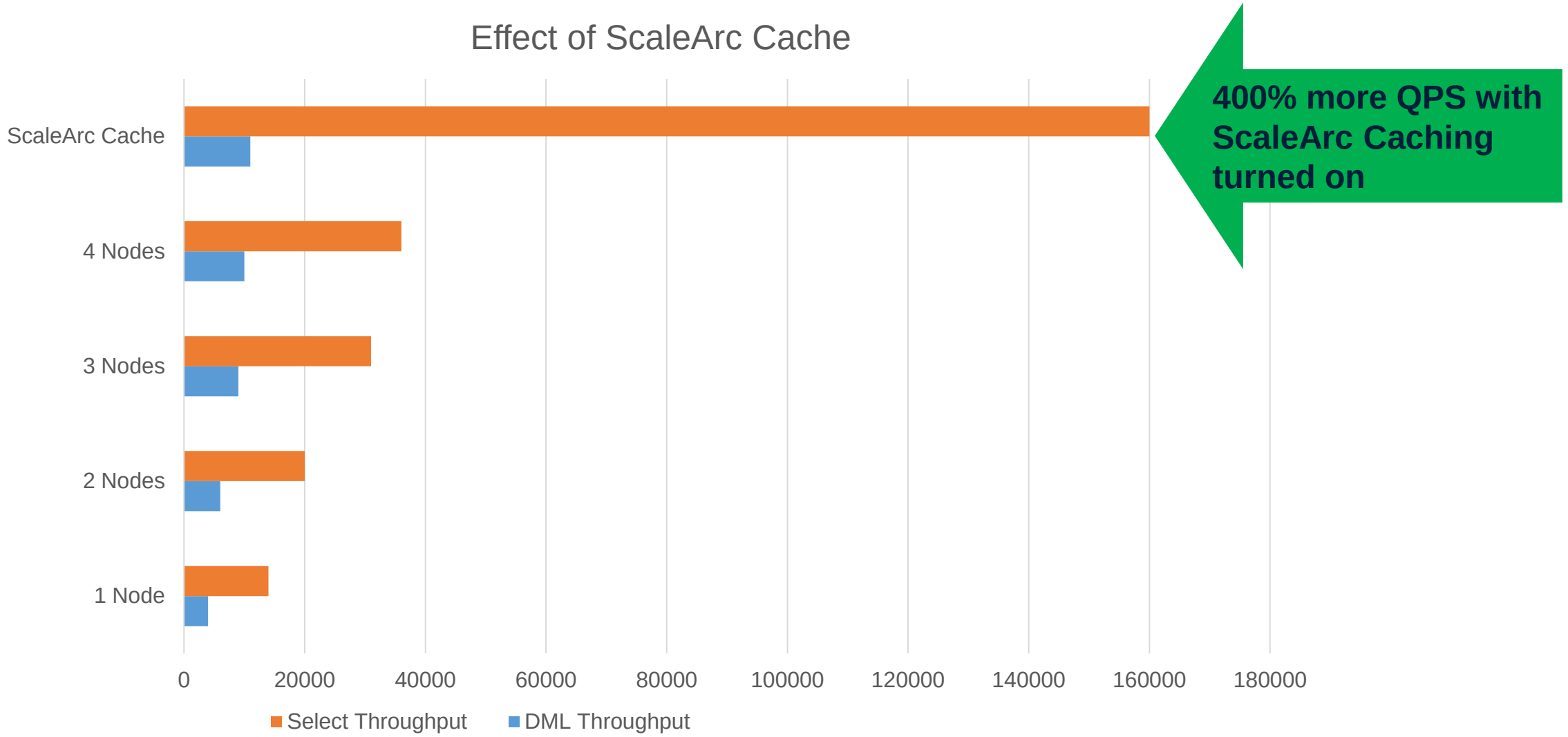
Test #3: Caching – Methodology

Test A: Execute only read queries with 100% cache hit rate to show the performance improvement by using ScaleArc caching

Test B: Execute read and write queries and use ScaleArc to cache read queries

ScaleArc

Test #3: Caching – Results



ScaleArc