Cloudius Systems presents:

NoSQL goes NATIVE

Don Marti   Tzach Livyatan
@dmarti      @TzachL
Scylla: A new Open Source NoSQL Database

Capable of 1,800,000 operations per second
PER NODE

With predictable, low latencies

Compatible with Apache Cassandra
**BACKGROUND**

**SQL:**
- Structured,
- no scale

**Document store:**
- No structure
- Some scale

**Column store:**
- Some structure
- Scale out
- Awesome HA/DR

**Key-value:**
- Simple
- Scale
- Not a real DB
SOLUTION: SCYLLA DB
THE POWER OF CASSANDRA AT THE SPEED OF REDIS
AWESOME REDUNDANCY & HA

+ Multi DC
+ Spark
+ CQL
+ Auto sharding
+ Wide rows
RESULTS: THROUGHPUT/SCALE UP
Benchmark configuration

- **Server type:** Rackspace Bare Metal IO Class v1
- **CPU:** Dual 2.8 GHz, 10 core Intel® Xeon® E5-2680 v2
- **RAM:** 128 GB
- **Networking:** Redundant 10 Gb/s connections in a high availability bond
- **Data Disks:** 2 * 1.6 TB PCIe flash cards
- **OS:** CentOS 7.2.1511, Kernel version: 3.10.0-327.10.1.el7.x86_64
- **Java**
  - Cassandra - Oracle jdk-8u65
  - Scylla - Open JDK 1.8 (used only for scylla-jmx)

LATENCY - AVG

Read Latency Comparison

- Scylla 3 Nodes
- Cassandra 3 Nodes
- Cassandra 9 Nodes
- Cassandra 15 Nodes

Latency (microsecond)

OPS Actual

- 0
- 100,000
- 200,000
- 300,000
- 400,000
- 500,000
- 600,000
LATENCY - P99

99 Read Latency Comparison

Latency (microsecond)

- 35,000
- 30,000
- 25,000
- 20,000
- 15,000
- 10,000
- 5,000
- 0

Actual OPS

- 100,000
- 200,000
- 300,000
- 400,000
- 500,000
- 600,000

Scylla 3 Nodes
Cassandra 3 Nodes
Cassandra 9 Nodes
Cassandra 15 Nodes
FULLY COMPATIBLE
WHAT WOULD YOU DO WITH 1 MILLION TPS?

- Shrink your cluster by a factor of X10
- Handle 10X traffic spikes on Black Friday
- Faster repairs, faster scale out.
- Get the most out of your data - Run more queries
- Administration operations while serving
- Stop using caches in front of the database
TECHNOLOGY: HOW IT WORKS
SCYLLA IS QUITE DIFFERENT

Shard-per-core, no locks, no threads, zero-copy
Reactor programing with C++14
Our own efficient, DB-aware cache, not using Linux page cache
Better storage engine
Max out all HW resources - NUMA friendly, multiqueue NICs, etc
Userspace I/O scheduler
Based on Seastar project
SCYLLA ARCHITECTURE COMPARISON

- KVM was invented by Avi in 2006, development was managed by Dor.
- It was a new hypervisor after VMW, Xen had dominated the market.
- By smart design choices and leveraging Linux and the hardware, it became the most performing hypervisor.
  - KVM holds SPECvirt performance record.
  - KVM holds max IOPS record.
- The Open Virtualization Alliance includes hundreds of companies, including HP, IBM, Intel, AMD, Red Hat, etc.
- KVM is the engine behind many clouds such as OpenStack, IBM, NTT, Fujitsu, HP, Google, DigitalOcean, etc.

Traditional stack

- Cassandra
- Scheduler
- Memory
- TCP/IP
- Kernel
- NIC
- Queues

Lock contention
Cache contention
NUMA unfriendly

Seastar’s sharded stack

- Core Database
- Task Scheduler
- Userspace
- NIC
- Queue
- smp queue

No contention
Linear scaling
NUMA friendly
Scylla has its own task scheduler

**Traditional stack**
- **Thread** is a function pointer
- **Stack** is a byte array from 64k to megabytes

**Scylla’s stack**
- **Promise** is a pointer to eventually computed value
- **Task** is a pointer to a lambda function

Context switch cost is high. Large stacks pollutes the caches.

No sharing, millions of parallel events.
Unified cache

Cassandra
- Key cache
- Row cache
  - Linux page cache
    - SSTables

Scylla
- Unified cache

Page faults
Parasitic rows
Tuning
Scylla has an I/O scheduler.

Max useful disk concurrency

Scylla has an I/O scheduler

<table>
<thead>
<tr>
<th>Shares distribution</th>
<th>Throughput (KB/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>C1</td>
</tr>
<tr>
<td>10, 10, 10, 10</td>
<td>137506</td>
</tr>
<tr>
<td>100, 100, 100, 100</td>
<td>137504</td>
</tr>
<tr>
<td>10, 20, 40, 80</td>
<td>37333</td>
</tr>
<tr>
<td>100, 10, 10, 10</td>
<td>421211</td>
</tr>
</tbody>
</table>
Compatibility (and speed): Repair

Jan 13 17:33:44 server-02.localdomain scylla[7730]: [shard 27] stream_session - [Stream #cbbf8dc1-ba1b-11e5-a51a-00000000000b ID#0]
Creating new streaming plan for repair-in
Jan 13 17:33:44 server-02.localdomain scylla[7730]: [shard 27] stream_session - [Stream #cbbf8dc1-ba1b-11e5-a51a-00000000000b ID#0] Received streaming plan for repair-in
Jan 13 17:33:45 server-02.localdomain scylla[7730]: [shard 27] stream_session - [Stream #cbc02a01-ba1b-11e5-a51a-00000000000b ID#0] Creating new streaming plan for repair-in
Jan 13 17:33:45 server-02.localdomain scylla[7730]: [shard 27] stream_session - [Stream #cbc02a01-ba1b-11e5-a51a-00000000000b ID#0] Received streaming plan for repair-in
SCYLLA as an INFRASTRUCTURE

Scale up with the number of cores
Kernel bypass for direct networking and block I/O
Good match for upcoming Non Volatile Memory technology
High availability with gossip and flexible replication
Runs everywhere: Physical, virtual, containers
Can be integrated with microservices with its own httpd
Monitoring Scylla
Connections to Apache Ecosystem: today
Connections to Apache Ecosystem: Soon

kafka  Zeppelin  Zipkin  presto  Spark  KairosDB  Cassandra  Scylla
WHAT’S NEXT?

- Build a community
- Core database improvements
- VERTICAL: Spark, Solr, distributed SQL engines
- HORIZONTAL: Microservice integration, more protocols
- Upcoming releases: Scylla 1.0.3, Scylla 1.1
SCYLLA, NoSQL GOES NATIVE
Thank you.