ABOUT THE LAST PICKLE

WORK WITH CLIENTS TO DELIVER AND IMPROVE APACHE CASSANDRA BASED SOLUTIONS

BASED IN USA, NEW ZEALAND, AUSTRALIA
microservices - devOps - distributed tracing
zipkin
zipkin & cassandra
Apache Cassandra the data platform de jure for the next evolution of software services

*an enterprise moving ever towards microservices and BASE architectures*

the missing piece for many is tracing and profiling difficult to reproduce problems
SCALING DATA

Apache Cassandra
data platform de jure
next evolution of software services
SCALING PEOPLE

an enterprise moving ever towards microservices and BASE architectures
Kibana

distributed tracing?

Grafana
ARCHITECTURAL SAFETY

- zero exceptions
- spot problems first
- stable master
- no user left out
- plug-n-play services
tracing and profiling difficult to reproduce problems
ZIPKIN

an implementation of Google's Dapper paper
### Analyse One Trace

**Duration:** 185,000 ms  
**Services:** 8  
**Depth:** 2  
**Total Spans:** 17

<table>
<thead>
<tr>
<th>Services</th>
<th>39,000 ms</th>
<th>78,000 ms</th>
<th>117,000 ms</th>
</tr>
</thead>
<tbody>
<tr>
<td>prod</td>
<td>185,000 ms</td>
<td>15,000 ms</td>
<td>5,000 ms</td>
</tr>
<tr>
<td>cms-lookup-server</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>statistics</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>oppdragserver</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>smjobberlookupservice</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>search-management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>search-management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>search-management</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>sybase: finnst</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Operations:**
- Hits: pr_myads_pagination_ordered
- Hits: AdCountService
- Hits: OppdragService
- Hits: SmjobberLookupService
- Hits: SearchManagementService
- Hits: pr_person_fetch
- Hits: pr_user_preferences_fetch
- Hits: pr_user_profile_fetch
- Hits: pr_person_update
- Hits: pr_user_preferences_ins_update
- Hits: pr_user_profile_ins_update
- Hits: pr_org_insertauxaddress
Outback grill til salgs

Elin Beate Forthun

Mobil

Kontakt selger på e-post

<table>
<thead>
<tr>
<th>Time</th>
<th>Trace ID</th>
<th>Zipkin URL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/14 - 10:12:42am CET</td>
<td>b18640d3e1ec000</td>
<td><a href="http://st.zipkin.finntech.no/traces/b18640d3e1ec000">http://st.zipkin.finntech.no/traces/b18640d3e1ec000</a></td>
</tr>
</tbody>
</table>
CLIENT    |    SERVER

CS -->

--> SR

<--- SS

CR <--
Result result = doRequest(request);

return result;
clientTracer.startNewSpan(request.getMethod() + " " + request.getNormalizedURI());
clientTracer.setCurrentClientServiceName(serviceName);
clientTracer.setClientSent();
Result result = doRequest(request);
clientTracer.setClientReceived();
return result;
ResultSet result = session.execute(statement);

return result;
SIMPLE C* CALL
SpanId spanId = clientTracer.startNewSpan(request.getMethod() + " " + request.getNormalizedURI());
request.setHeader(TraceId, Long.toString(spanId.getTraceId(), 16));
request.setHeader(SpanId, Long.toString(spanId.getSpanId(), 16));

clientTracer.setCurrentClientServiceName(serviceName);
clientTracer.setClientSent();
Result result = doRequest(request);
clientTracer.setClientReceived();
return result;
# TRACING IN C*

Cassandra / CASSANDRA-1123

Allow tracing query details

<table>
<thead>
<tr>
<th><strong>Details</strong></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type:</strong></td>
<td>![New Feature]</td>
<td></td>
<td></td>
<td>![Resolved]</td>
</tr>
<tr>
<td><strong>Priority:</strong></td>
<td>![Major]</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Component/s:</strong></td>
<td>Core</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Labels:</strong></td>
<td>None</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Status:</strong></td>
<td></td>
<td></td>
<td></td>
<td>![Resolved]</td>
</tr>
<tr>
<td><strong>Resolution:</strong></td>
<td></td>
<td></td>
<td></td>
<td>Fixed</td>
</tr>
<tr>
<td><strong>Fix Version/s:</strong></td>
<td></td>
<td></td>
<td></td>
<td>1.2.0 beta 1</td>
</tr>
</tbody>
</table>

**Description**

In the spirit of [CASSANDRA-511](CASSANDRA-511), it would be useful to tracing on queries to see where latency is coming from: how long did row cache lookup take? key search in the index? merging the data from the sstables? etc.

We don't need to be as sophisticated as the techniques discussed in the following
papers but they are interesting reading:

http://research.google.com/pubs/pub36356.html
http://www.usenix.org/events/osdi04/tech/full_papers/barham/barham_html/
http://www.usenix.org/event/nsdi07/tech/fonseca.html
org.apache.cassandra.tracing
  - ExpiredTraceState.java
  - TraceKeyspace.java
  - TraceState.java
  - Tracing.java
beginSession(..)
trace(..)
trace(..)

initialiseMessage(..)
trace(..)
trace(..)

endSession(..)
ZIPKIN IN C*

- visualisation
- detailed timings
- hierarchy and asynchronisity
- zero tracing overhead
<table>
<thead>
<tr>
<th>Services</th>
<th>Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>c*:Test Cluster:local...</td>
<td>317,000</td>
</tr>
<tr>
<td>1.000ms: Parsing select * from vortex_powervomil_test.envelopes limit 100; [SharedPool-Worker-1]</td>
<td>634,000</td>
</tr>
<tr>
<td>1.000ms: Preparing statement [SharedPool-Worker-1]</td>
<td></td>
</tr>
<tr>
<td>2.000ms: Computing ranges to query [SharedPool-Worker-1]</td>
<td></td>
</tr>
<tr>
<td>15.000ms: Submitting range requests on 257 ranges with a concurrency of 1 (24892.65 rows per range exec)</td>
<td></td>
</tr>
<tr>
<td>69.000ms: Executing seq scan across:7 sstables for (min(-9223372036854775808), min(-9223372036854775808),</td>
<td></td>
</tr>
<tr>
<td>531.000ms: Submitted 1 concurrent range requests covering 257 ranges [SharedPool-Worker-1]</td>
<td></td>
</tr>
<tr>
<td>2.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>8.000ms: Seeking to partition beginning in data file [SharedPool-Worker-2]</td>
<td></td>
</tr>
<tr>
<td>1.000ms: Read 12 live and 0 tombstone cells [SharedPool-Worker-2]</td>
<td></td>
</tr>
</tbody>
</table>
Two classes to override

```java
public class ZipkinTracing extends Tracing
{ .. }

public class ZipkinTraceState extends TraceState
{ .. }
```

then run enabling new tracing

```
bin/cassandra -Dcassandra.custom_tracing_class=..ZipkinTracing
```
ZIPKIN ACROSS C*
ZIPKIN ACROSS C*

<table>
<thead>
<tr>
<th>CO-ORDINATOR NODE</th>
<th>REPLICA NODE</th>
</tr>
</thead>
<tbody>
<tr>
<td>--&gt;</td>
<td></td>
</tr>
<tr>
<td>beginSession(..)</td>
<td></td>
</tr>
<tr>
<td>trace(..)</td>
<td></td>
</tr>
<tr>
<td>trace(..)</td>
<td></td>
</tr>
<tr>
<td>(zipkin headers)</td>
<td>--&gt; initialiseMessage(..)</td>
</tr>
<tr>
<td>trace(..)</td>
<td></td>
</tr>
<tr>
<td>trace(..)</td>
<td></td>
</tr>
<tr>
<td>&lt;--</td>
<td></td>
</tr>
<tr>
<td>trace(..)</td>
<td></td>
</tr>
<tr>
<td>endSession(..)</td>
<td></td>
</tr>
<tr>
<td>&lt;--</td>
<td></td>
</tr>
</tbody>
</table>
ZIPKIN ACROSS C*

- **c*:zipkin:localhost**: Parsing select * from zipkin traces ; [SharedPool-Worker-1]
- **c*:zipkin:localhost**: 1.000ms : Preparing statement [SharedPool-Worker-1]
- **c*:zipkin:localhost**: 2.000ms : Computing ranges to query [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.2**: 292.000ms : Submitting range requests on 4 ranges with a concurrency of 1 (0.0 rows per range expected) [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.2**: 63.000ms : Message received from /127.0.0.1 [MessagingService-Incoming-/127.0.0.1]
- **c*:zipkin:127.0.0.2**: 3.000ms : Executing seq scan across 0 sstables for (min(-922372036854775808), max(-4611686018427387904)) [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.2**: 2.000ms : Scanned 0 rows and matched 0 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.2**: 34.000ms : Enqueuing response to /127.0.0.1 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.2**: 5.000ms : Sending message to /127.0.0.1 [MessagingService-Outgoing-/127.0.0.1]
- **c*:zipkin:localhost**: 7.000ms : Enqueuing request to /127.0.0.2 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 614.000ms : Submitted 1 concurrent range requests covering 1 ranges [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 139.000ms : Message received from /127.0.0.1 [MessagingService-Incoming-/127.0.0.1]
- **c*:zipkin:127.0.0.3**: 15.000ms : Executing seq scan across 0 sstables for (max(-4611686018427387904), max(0)) [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 4.000ms : Read 14 live and 0 tombstone cells [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 2.000ms : Read 3 live and 0 tombstone cells [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: : Scanned 2 rows and matched 2 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 8.000ms : Enqueuing response to /127.0.0.1 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.3**: 22.000ms : Sending message to /127.0.0.1 [MessagingService-Outgoing-/127.0.0.1]
- **c*:zipkin:127.0.0.4**: 440.000ms : Enqueuing request to /127.0.0.3 [SharedPool-Worker-1]
- **c*:zipkin:127.0.0.4**: 50.000ms : Message received from /127.0.0.1 [MessagingService-Incoming-/127.0.0.1]
<table>
<thead>
<tr>
<th>Event Type</th>
<th>Duration</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>c:zipkin:127.0.0.4</td>
<td>99.000ms</td>
<td>Message received from /127.0.0.1 [MessagingService-Incoming/127.0.0.1]</td>
</tr>
<tr>
<td>c:zipkin:127.0.0.4</td>
<td></td>
<td>Executing seq scan across 0 sstables for (max(0), max(4611686018427387904)) [SharedPool-Worker-3]</td>
</tr>
<tr>
<td>c:zipkin:127.0.0.4</td>
<td></td>
<td>Scanned 0 rows and matched 0 [SharedPool-Worker-3]</td>
</tr>
<tr>
<td>c:zipkin:127.0.0.4</td>
<td>2.000ms</td>
<td>Enqueuing response to /127.0.0.1 [SharedPool-Worker-3]</td>
</tr>
<tr>
<td>c:zipkin:127.0.0.4</td>
<td>11.000ms</td>
<td>Sending message to /127.0.0.1 [MessagingService-Outgoing-/127.0.0.1]</td>
</tr>
<tr>
<td>c:zipkin:localhost</td>
<td></td>
<td>Enqueuing request to /127.0.0.4 [SharedPool-Worker-1]</td>
</tr>
<tr>
<td>c:zipkin:localhost</td>
<td>9.931s</td>
<td>Submitted 3 concurrent range requests covering 3 ranges [SharedPool-Worker-1]</td>
</tr>
<tr>
<td>c:zipkin:localhost</td>
<td>115.000ms</td>
<td>Processing response from /127.0.0.3 [SharedPool-Worker-4]</td>
</tr>
</tbody>
</table>
ZIPKIN INTO C*
| Duration: 195.000ms | Services: 8 | Depth: 2 | Total Spans: 17 |

- **prod**
  - 195.000ms /finn/
    - 15.000ms: CmsLookupService
    - 5.000ms: pr_myads_pagination_ordered
    - 2.000ms: AdCountService
    - 6.000ms: OppdragService
    - 6.000ms: SmajorberLookupService
    - 2.000ms: SearchManagementService
    - 3.000ms: SearchManagementService
    - 9.000ms: SearchManagementService
    - Parsing select * from [SharedPool-Worker-1]
    - 1.000ms: Preparing statement [SharedPool-Worker-1]
    - 2.000ms: Computing ranges to query [SharedPool-Worker-1]
    - 292.000ms: Submitting range requests on 4 ranges with a cost
    - 63.000ms: Message received from /127.0.0.1 [MessageReceived]
    - 3.000ms: Executing seq scan across 0 stables for (min)
    - 2.000ms: Scanned 0 rows and matched 0 [SharedPool-Worker-1]
TO THE RESCUE
**Add a key-value payload for third party usage**

<table>
<thead>
<tr>
<th>Details</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Type:</td>
<td>Sub-task</td>
<td>Status:</td>
<td>RESOLVED</td>
</tr>
<tr>
<td>Priority:</td>
<td>Major</td>
<td>Resolution:</td>
<td>Fixed</td>
</tr>
<tr>
<td>Component/s:</td>
<td>None</td>
<td>Fix Version/s:</td>
<td>2.2.0 beta 1</td>
</tr>
<tr>
<td>Labels:</td>
<td>client-impacting</td>
<td>protocolv4</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

An useful improvement would be to include a generic key-value payload, so that developers implementing a custom `QueryHandler` could leverage that to move custom data back and forth.
SpanId spanId = clientTracer.startNewSpan(request.getMethod()
        + " " + request.getNormalizedURI());
request.setHeader(TraceId, Long.toString(spanId.getTraceId(), 16));
request.setHeader(SpanId, Long.toString(spanId.getSpanId(), 16));

clientTracer.setCurrentClientServiceName(serviceName);
clientTracer.setClientSent();
Result result = doRequest(request);
clientTracer.setClientReceived();
return result;
SpanId spanId = clientTracer.startNewSpan(statement.toString());

ByteBuffer traceHeaders = ByteBuffer.allocate(16);
traceHeaders.putLong(spanId.getTraceId());
traceHeaders.putLong(spanId.getSpanId());

statement.setOutgoingPayload(singletonMap("zipkin", traceHeaders.array()));

clientTracer.setCurrentClientServiceName(serviceName);
clientTracer.setClientSent();
ResultSet result = session.execute(statement);
clientTracer.setClientReceived();
return result;
enable zipkin tracing and the custom payload handler

bin/cassandra
-Dcassandra.custom_tracing_class=..ZipkinTracing
-Dcassandra.custom_query_handler_class=..CustomPayloadMirroringQueryHandler
the patch?

src/java/org/apache/cassandra/net/MessageOut.java          |   7 +-------
src/java/org/apache/cassandra/net/OutboundTcpConnection.java |   4 +++-
src/java/org/apache/cassandra/service/QueryState.java     |  12 ++++++++++-
src/java/org/apache/cassandra/transport/messages/ExecuteMessage.java|  2 +-
4 file changed, 15 insertion(+), 10 deletion(-)
WHAT ELSE?

- anti-entropy repair
- compaction
THANKS

- Zipkin  – https://github.com/openzipkin/zipkin

- Brave (zipkin java instrumentation)  
  - https://github.com/openzipkin/brave

- C* patch for pluggable tracing  
  - https://issues.apache.org/jira/browse/CASSANDRA-10392

- Zipkin Cassandra implementation  
  - https://github.com/thelastpickle/cassandra-zipkin-tracing

- Google's Dapper paper  

- C* custom payloads  
  - https://issues.apache.org/jira/browse/CASSANDRA-8553  
  - https://datastax.github.io/java-driver/2.2.0-rc2/features/custom_payloads/
Mick Semb Wever

TWITTER  @mck_sw
mick@thelastpickle.com

THE LAST PICKLE
APACHE CASSANDRA CONSULTING & SERVICES