Taking Omid to the Clouds:
Fast, Scalable Transactions for Real-Time Cloud Analytics
O. Shacham • Y. Gottesman • A. Bergman • E. Bortnikov • E. Hillel • I. Keidar
Yahoo Research

**Omid**
- Transactional API over NoSQL key value
- Client Library + Runtime Service
- Open source – Apache incubator
- Implemented over Apache HBase
- Snapshot Isolation consistency
- Highly Available
- Originally optimized for throughput

**Omid Design Choices**

<table>
<thead>
<tr>
<th>System</th>
<th>Validation</th>
<th>Commit entry writes</th>
<th>Multi tenancy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Percolator (2PC)</td>
<td>D</td>
<td>D</td>
<td>no</td>
</tr>
<tr>
<td>CockroachDB</td>
<td>D</td>
<td>D</td>
<td>yes</td>
</tr>
<tr>
<td>Omid</td>
<td>C</td>
<td>C</td>
<td>yes</td>
</tr>
<tr>
<td>Omid LL</td>
<td>C</td>
<td>D</td>
<td>yes</td>
</tr>
</tbody>
</table>

C – centralized, D – distributed

**Phoenix Integration**

Secondary index creation
When a new index is created, two coprocessors are installed:
A. Populates new index with history
B. Maintains index by augmenting write to table with a write to index

**Taking Omid To The Cloud**

- Optimized for low latency – Omid LL
- Redesign Omid to eliminate key bottleneck
- Distribute commit table updates among clients.

Novel Fast Path API – Omid FP
Many workloads have single key transactions
Eliminate transaction overhead for them

Direct access to region server without access to TM API:
- `brc(key)` • `bwc(key,value)` • `br(key)` + `wc(key,val)`

**Apache Phoenix Integration**

Integrate Omid as the transactional layer in Phoenix SQL engine

---

**Omid Architecture**

**Performance**