LedgerDB: Alibaba’s Centralized Ledger Database

- Xinying(Derry) YANG
Terminologies

- DLT (Decentralized Ledger Technology)
- CLT (Centralized Ledger Technology)
  - CLD (Centralized Ledger Database): LedgerDB, QLDB, Oracle BC Table, ProvenDB, etc.
- Immutability: Any piece of data, once committed into the system, cannot be modified by subsequent operations and becomes permanently available.
- Verifiability: The capability of validating specific data integrity and operation proofs.
- Auditability: The capability of observing a serial of user actions and operation trails based on predefined audit rules.
  - Internal audit: an internal user of the ledger can observe and verify the authenticity of all actions.
  - External audit: an external third-party entity can observe and verify the authenticity of all actions.
Credibility for Traditional Database Applications

- Centralized DBMS
  - Oracle
  - IBM DB2

- Cloud (Distributed) DBMS
  - Amazon Aurora
  - PolarDB

- Bigdata & No-SQL
  - MongoDB
  - Apache HBase

Here comes ledger technique
DLT Dilemma

Permissionless blockchains: Bitcoin, Ethereum, etc.

Pros:
• Massive peers, widely spread, highly decentralized

Cons:
• Extremely low TPS (7 for Bitcoin)

Permissioned blockchains: Fabric, Corda, Quorum, etc.

Pros:
• Improved TPS, still cannot be compared with RDBMS or NoSQL

Cons:
• Few peers, consensus can be broken/manipulated by malicious nodes
Why CLD is important & valuable?

• Motivations
  • Decentralization is not proved to be indispensable for permissioned blockchain.
  • Conventional permissioned blockchain and CLD systems:
    • Low performance, storage overhead, regulatory issues, limited external auditability

• Gartner Forecast
  • Gartner Strategic Vision 2019
    Strategic Planning Assumption
    By 2021, at least 20% of projects envisioned to run on permissioned blockchains will instead run on centralized, auditable ledgers.

  • Gartner Strategic Vision 2020
    By 2021, most permissioned blockchain uses will be replaced by ledger DBMS products.
Highlight and Comparison

- LedgerDB – a ledger database that provides tamper-evidence and non-repudiation features in a centralized manner (CLD), which realizes strong auditability, high performance, and data removal support.

- Key comparisons between LedgerDB and other systems.

<table>
<thead>
<tr>
<th>System</th>
<th>Throughput (max TPS)</th>
<th>Auditability</th>
<th>Removal</th>
<th>Non-Repudiation</th>
<th>Provenance</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>external</td>
<td>third party</td>
<td>peg capability</td>
<td>purge</td>
</tr>
<tr>
<td>LedgerDB</td>
<td>100K+</td>
<td>✓</td>
<td>✓</td>
<td>strong</td>
<td>✓</td>
</tr>
<tr>
<td>QLDB [7]</td>
<td>1K+</td>
<td>✓</td>
<td>x</td>
<td>weak</td>
<td>x</td>
</tr>
<tr>
<td>Hyperledger [6]</td>
<td>1K+</td>
<td>x</td>
<td>x</td>
<td>weak</td>
<td>x</td>
</tr>
<tr>
<td>ProvenDB [40]</td>
<td>10K+</td>
<td>X</td>
<td>Bitcoin</td>
<td>medium</td>
<td>X</td>
</tr>
<tr>
<td>Factom [43]</td>
<td>10+</td>
<td>✓</td>
<td>Bitcoin</td>
<td>strong</td>
<td>x</td>
</tr>
</tbody>
</table>
How it works

You can use LedgerDB to store application data, such as financial data, electronic invoices and receipts, health records, and system logs.

1. Signatures from Ledger Users

2. Journal

3. Block

4. Signature from LedgerDB

5. Signature from the Third-party Time Authority

LedgerDB implements hashing on all the ledgers and creates a time anchor every hour. The created time anchors are stored in the time ledger.

The time ledger is a public ledger.

LedgerDB regularly sends requests to obtain trusted timestamp notaries by using the hash digests that are stored in the time ledger.

The returned results are written to the time ledger as time anchors.

No participants can tamper with or repudiate the data that is stored in LedgerDB. This ensures high reliability of the stored data.
LedgerDB system architecture.

**Ledger master** - manage the runtime metadata of the entire cluster (e.g., status of servers and ledgers) and coordinate cluster-level events (e.g., load balance, failure recovery).

**Ledger proxy** - receive client requests and preprocesses, and then dispatch them to the corresponding ledger server.

**Ledger server** - complete the final processing of requests, and interact with underlying storage layer that stores ledger data.
LedgerDB Operators and APIs.

**Append** - append user transaction or system-generated transaction to ledger.
**Retrieve** - get qualified journals from ledger.
**Verify** - verify integrity and authenticity of returned journals from journal proofs.
**Create** - create a new ledger with initial roles and members.
**Purge** - remove obsolete journals from ledger.
**Occult** - hide journal(s) from ledger.
**Recall** - rollback a purge (within a limited time window).
**Delete** - removes entities in the system, such as a ledger, a role, a member, or a clue.

<table>
<thead>
<tr>
<th>Operator</th>
<th>Method</th>
</tr>
</thead>
<tbody>
<tr>
<td>Create</td>
<td>Create(ledger_uri, enum, op_metadata)</td>
</tr>
<tr>
<td>Append</td>
<td>AppendTx(ledger_uri, tx_data, clue, set) SetTrustedAnchor(ledger_uri, jsn, level) GrantRole(ledger_uri, member_id, role) GrantTime(ledger_uri, timestamp, proof)</td>
</tr>
<tr>
<td>Retrieve</td>
<td>GetTx(ledger_uri, jsn) ListTx(ledger_uri, ini_json, limit, clue) GetTrustedAnchor(ledger_uri, jsn, level) GetLastGrantTime(ledger_uri, timestamp)</td>
</tr>
<tr>
<td>Verify</td>
<td>Verify(ledger_uri, jsn</td>
</tr>
<tr>
<td>Purge</td>
<td>Purge(ledger_uri, block)</td>
</tr>
<tr>
<td>Occult</td>
<td>Occult(ledger_uri, jsn</td>
</tr>
<tr>
<td>Recall</td>
<td>Recall(ledger_uri, purged_point)</td>
</tr>
<tr>
<td>Delete</td>
<td>Delete(ledger_uri, enum, op_metadata)</td>
</tr>
</tbody>
</table>
Journal Management

LedgerDB adopts an `execute-commit-index` transaction management approach:

1. **execute** - a transaction first enters the execute phase based on its transaction type. It runs on ledger proxy for better scalability.
2. **commit** - collect multiple executed transactions, arranges them in a global order (jsn), and persist them to the storage system. It runs on ledger server.
3. **index** - start on ledger server to build indexes for subsequent data retrieval and verification.
Two-way peg TSA notary journals

- A TSA journal contains a ledger snapshot (i.e., a ledger digest) and a timestamp, signed by TSA in entirety. These journals are mutually entangled between each other, which provide external auditability for timestamps.

- Two-way peg protocol: 
  1. A ledger digest is first submitted and then signed by TSA;  
  2. TSA journal is recorded back on ledger as a TSA journal.

- We offer T-Ledger service on Alibaba Cloud LaaS+ (Ledger-as-a-Service).
Credibility guarded by multipart signatures
Verifiable Data Removals

• **Purge**

A purge operation deletes a set of contiguous (obsolete) journals starting from genesis to a designated jsn on ledger.

• **Occult**

An occult operation converts the original journal to a new one that only keeps its metadata, and retains its digest.

---

**Database Queries**

01 | `DELETE FROM ledger_uri WHERE jsn < pur_json;`

02 | `SET TS = na, cps = CONCAT(`

03 | `seqX, journal_hash, blanks) WHERE jsn = Seq`

04 | `OR cid = des_cid;`
Clue – Native lineage in LedgerDB

- A clue is a user-specified label (key) that carries on business logic for data lineage.
- Quick index is supported to fetch or verify through related events in chronological order.

- **Clue - Append**
- **Clue - Query**
- **Clue - Verify**

Clue: [Order_id] - 123456
- Clue: 123456
  - Data: *not paid yet*
- Clue: 123456
  - Data: *paid*
- Clue: 123456
  - Data: *delivered*
- Clue: 123456
  - Data: *received*
Evaluation – clue Skiplist (cSL) & batch accumulated Merkle-tree (bAMT)

**cSL vs. RocksDB**

**bAMT vs. Libra accumulator**

(a) bAMT root calculation

(b) bAMT vs. Libra

(c) Throughput comparison

(d) Latency comparison
LedgerDB is 80 times faster compared to Hyperledger Fabric in the same notarization application.
LedgerDB Solution Category

Mono Ledger

Federal Ledger

Delegated Ledger
LedgerDB in Production

Federated ledger vs. permissioned blockchain

<table>
<thead>
<tr>
<th>PEER1</th>
<th>PEER2</th>
<th>PEER3</th>
<th>PEER4</th>
</tr>
</thead>
<tbody>
<tr>
<td>P1 SIGN</td>
<td>P3 SIGN</td>
<td>P3 SIGN</td>
<td>P4 SIGN</td>
</tr>
<tr>
<td>LSP SIGN</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Ledger Service Provider (LSP)

(a)  

T-Ledger

Interval & Time

Batch & TSA Sig

(b)  

T-Ledger

LedgerDB customer use cases

- Finance (Bank, Insurance, SCF): 12%
- IP (copyright): 6%
- IoT: 6%
- Regtech: 6%
- Retail: 3%
- Internet: 15%
- MISC (Healthcare, Manufacture, Energy...): 45%
Decentralized vm-like exec is just an implementation, the soul of consensus in ledger technique is dancing with time and cryptographic theorem.

- LedgerDB

https://www.alibabacloud.com/product/ledgerdb

xinying.yang@alibaba-inc.com

Thanks!