Fast, Durable, Flexible Pub/Sub based on Segment-Oriented Architecture

PULSAR

演讲者 / streamlio 翟佳
What is Apache Pulsar?

- **Durability**: Data replicated and synced to disk
- **Geo-replication**: Out of box support for geographically distributed applications
- **Unified messaging model**: Support both Topic & Queue semantic in a single model
- **Ordering**: Guaranteed ordering
- **Multi-tenancy**: A single cluster can support many tenants and use cases
- **High throughput**: Can reach 1.8 M messages/s in a single partition
- **Delivery Guarantees**: At least once, at most once and effectively once
- **Low Latency**: Low publish latency of 5ms at 99pct
- **Highly scalable**: Can support millions of topics
Architecture
Architecture view

- Separate layers between brokers bookies

[Diagram showing the architecture of Apache Pulsar with separate layers between producers, consumers, brokers, and bookies.]
Brokers

Diagram showing the interaction between producers, brokers, topics, and consumers in a Pulsar ecosystem.
Bookies - Apache BookKeeper

- Durable and Consistent
- I/O Isolation
- High Throughput
- Low Latency
Bookies - Apache BookKeeper

[Diagram showing the flow of data from Application to BookKeeper ensemble, with quorum splits for write and acknowledgment, and a potential replacement of an unhealthy bookie.]
Architecture view

- Unbounded topic partition storage
- Instant scaling without data rebalance
- Independent scalability
A Compare

Apache Kafka

Logical partition view

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Partition (lead)</th>
<th>Partition (copy)</th>
<th>Partition (copy)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Partition 1</td>
<td>Partition 1</td>
<td>Partition 1</td>
</tr>
<tr>
<td>Broker 1</td>
<td>Broker 2</td>
<td>Broker 3</td>
<td>Broker 4</td>
</tr>
</tbody>
</table>

**Kafka Partitions** — All log segments are replicated in order across brokers (replication = 3 here).

Apache Pulsar/BookKeeper

Logical topic/partition view

<table>
<thead>
<tr>
<th>Segment distribution</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sample 1</td>
</tr>
<tr>
<td>Sample 2</td>
</tr>
<tr>
<td>Sample 3</td>
</tr>
<tr>
<td>Sample 4</td>
</tr>
<tr>
<td>Sample X</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Logical topic/partition view</th>
</tr>
</thead>
<tbody>
<tr>
<td>Segment 1</td>
</tr>
<tr>
<td>Segment 2</td>
</tr>
<tr>
<td>Segment 3</td>
</tr>
<tr>
<td>Segment 4</td>
</tr>
<tr>
<td>Sample X</td>
</tr>
</tbody>
</table>

**Pulsar/BookKeeper Stream** — All log segment are replicated to a configurable number of bookies (replication = 3 here) across N possible bookies (N = 4 here). Log segments are evenly distributed to achieve horizontal scalability with no rebalancing.
Benefits
Seamless - broker failure
Seamless - bookie failure
Seamless - cluster expand
Conclusion

• Unbounded topic partition storage

• Instant scaling without data rebalance
  • Seamless - broker failure recovery
  • Seamless - bookie failure recovery
  • Seamless - cluster expansion

• Independent scalability
Benchmark

https://github.com/openmessaging/openmessaging-benchmark
Throughput

![Graph showing throughput over time for Pulsar, Kafka, and Kafka-sync. The graph indicates the publish rate over time, with peaks and troughs representing the message rate per second.]
Latency

Publish latency quantiles

Latency (ms)

Percentile

Pulsar-sync
Pulsar-nosync
Kafka-sync
Kafka-nosync
Pulsar Functions
Pulsar Functions

- Lightweight stream processing

- New in Pulsar 2.0

- Currently supports Java and Python

**Java**

```java
import java.util.function.
Function;

public class Anon implements Function<String, String> {
    @Override
    public String apply(String input) {
        return input.replace("jia", "anonymous");
    }
}
```

**Python**

```python
def process(input):
    return input.replace("jia", "anonymous")
```
Pulsar Functions

**Python**

```bash
# pulsar-admin functions create \
-py anon.py --className anon \
--fqfn lc3-tenant/demo/anony \
--inputs persistent://lc3-tenant/demo/input \
--output persistent://lc3-tenant/demo/output
```

**Java**

```bash
# pulsar-admin functions create \
-jar anon.jar --className Anon \
--fqfn lc3-tenant/demo/anony \
--inputs persistent://lc3-tenant/demo/input \
--output persistent://lc3-tenant/demo/output
```
Curious to Get More

- Apache Pulsar: http://pulsar.incubator.apache.org
- Apache BookKeeper: http://bookkeeper.apache.org
- Technical Blog: https://streaml.io/blog/
- Twitter: @apache_pulsar @asfbookkeeper
- slack:
  - https://apache-pulsar.herokuapp.com/
  - https://apachebookkeeper.herokuapp.com/
Thanks!