SMART DATA FAST.™

VOLTDB

MAXCDN DEPLOYS VOLTDB TO PROVIDE ITS CUSTOMERS WITH REAL-TIME ANALYTICS

CASE STUDY

MANAGING FAST DATA IN A CONTENT DELIVERY NETWORK

MaxCDN is a content delivery network (CDN) provider that emphasizes reducing the latency and increasing the reliability of its rich-content delivery. It provides CDN services to digital advertisers, ad networks, publishers, hosting providers, gaming companies, and mobile providers.

Facing the business challenge of providing customers with real-time analytics of their content delivery as well as having a reliable and accurate billing engine, MaxCDN had to deploy a solution that could process the CDN logs in real-time so data would be visible to the user in less than 15 seconds from the time the log was generated by the web server when serving the content. MaxCDN also needed a solution that would guarantee no data loss for billing purposes and that could scale horizontally to support ever-increasing content loads.

MaxCDN selected VoltDB over other streaming aggregation offerings for its SQL interface, ability to support real-time ad-hoc queries, its computational efficiency, and simple design. With VoltDB, MaxCDN uses roughly 1/10th of the CPU cycles that would have been required with alternative solutions, a conservative estimate the company reached after investigating alternative solutions that promised to deliver comparable real-time performance. MaxCDN was faced with the challenge of real-time aggregation of over 32 TB of daily web server log data.

The real-time analytic engine needed to keep close to one billion distinct aggregation counters in a ring-buffer design. The system had to have the power of updating these counters at the speed of incoming peak data in real-time. The system is currently peaking at around 300,000 log records per second, which translates into being able to do roughly 300,000 aggregation UPSERTs per second (or more if one log results in more than one UPSERT). The system also keeps the raw logs for close to one hour. This means 300,000 additional peak INSERTs per second.

"Most NoSQL solutions do not fit well with our real-time streaming aggregation design," said Behzad Pirvali, Performance Architect for MaxCDN. "For our real-time streaming aggregation requirements, we need to maintain the 'exactly-once' semantic as well as be able to guarantee no data loss. This can be achieved either by having old-fashioned Distributed Transaction Coordinators or by making the system idempotent and having some level of atomic WRITE support."

Many developers look to a combination of Apache projects, such as Zookeper, Kafka, Storm, or Spark and Cassandra, or to the Lambda Architecture, to solve fast data challenges. However, these approaches require a massive development effort to patch together a solution, often sacrificing performance, ACID guarantees, and ease of use. VoltDB helps customers like MaxCDN concentrate development resources on their applications rather than on the data processing infrastructure. It is purpose-built for companies like MaxCDN that need a fast, scalable database capable of processing streaming data and making in-transaction decisions with sub-millisecond latency.

SIMPLIFYING THE LAMBDA ARCHITECTURE

MaxCDN decided on the Lambda Architecture, a data-processing framework that introduces a hybrid of short-term data storage with fast query response times and long-term data storage with slower query response times. It also introduces atomic micro-batches, which could be used to implement the exactly-once semantic that aggregation engines need.

MaxCDN selected VoltDB as the short-term storage in their implementation of the Lambda Architecture. VoltDB is an in-memory SQL database that combines streaming analytics with transaction processing in a single, horizontal scale-out platform. It runs on commodity hardware in a massively parallel processing, shared-nothing architecture, and is built to tap the value of fast data—or data in motion.

VoltDB is a key component in MaxCDN's real-time analytic solution, which helps fuel its content delivery intelligence. VoltDB provides the advantages promised by the Lambda Architecture in a much simpler architecture. In the standard Lambda Architecture, the inclusion of VoltDB greatly simplifies the speed layer by replacing both the streaming and the operational data store portions of the speed layer. VoltDB provides the ability to execute real-time, ad-hoc analytics, conduct real-time transactions, and make per-event decisions on data as it arrives.

MaxCDN evaluated Spark Streaming, Trident, and VoltDB when selecting its real-time aggregation solution. Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics.

Trident is a high-level abstraction for doing real-time aggregation on top of Apache Storm; it supports the exactly-once semantic but does not include storage capabilities. "Trident is an aggregation engine without a storage engine. Trident's micro-batching and idempotency requirement at the micro-batch level would benefit much more from MULTI-WRITE atomic storage like VoltDB or Redis than SINGLE-WRITE atomic storage like HBase and Cassandra," Pirvali explained.

THE ADVANTAGE OF MULTI-WRITE ATOMICITY

	Trident with Cassandra or HBase	VoltDB
Number of Environments to Manage	At least 3	1
Atomicity	Single-write	Multi-write
Unit of Atomicity	A single row	A single partition
Indexed Look-Up Requirements Per Micro-Batch	150,000	336
Transaction ID Space Requirements Per Micro-Batch	18 GBs	0.000012 GB

Implementing VoltDB and micro-batching with multi-write atomicity within the Lambda framework simplified management and improved performance, storage, scalability, and operations.

Pirvali explained, "Currently, our micro-batch size is 70,000 web server logs. This translates into 70,000 INSERTs of raw logs and up to 80,000 aggregation UPSERTs, a total of 150,000 INSERTs/UPDATEs. These 150,000 INSERTs/UPDATEs are pushed more or less evenly through 336 (7 servers x 48 active partitions) VoltDB active partitions in the cluster. So, each partition does around 450 (150,000/336) INSERTs/UPDATEs in one single atomic unit. Each partition adds one row into a transaction table with a single transaction-id for this atomic operation. By recording the transaction-id, the system can behave in an idempotent way and avoid double counting if it sees the same data again."

"Now let's see what would have happened if we had deployed Trident with Cassandra or HBase," said Pirvali. "Because of the limitation of single-write atomicity in HBase and Cassandra, we would have had to assign this transaction ID to each of our 150,000 INSERT/UPDATE rows. So we would have had to write this transaction-id 150,000 times instead of 336 times. As I have stated earlier, about 80,000 out of these 150,000 INSERTs/UPDATEs are aggregation UPSERTs. So, we would have had to do an index lookup on this row-level transaction-id 80,000 times in order to guarantee idempotency. Now contrast this to 336 lookups in the case of VoltDB. For every single index-lookup in VoltDB, we would have had to do about 238 (=80,000/336) index lookups in Cassandra or HBase to be able to guarantee no double counting. We therefore would have needed many more CPU cycles with Cassandra or HBase to get the same performance."

He added, "Also, please keep in mind that the bigger the micro-batch size, the more drastic the difference. So, for example, if we go with 140,000 logs instead of 70,000 as our micro-batch size, then we will have 476 (=2x238) index lookups for each VoltDB index lookup. In terms of space, one micro-batch requires about 5 MB of space just for one transaction-id (36 bytes in our case), which is written 150,000 times. In the case of VoltDB, this would be 336x32, or about 12 KB."

ENSURING PERFORMANCE, SCALABILITY, & RELIABILITY

"If you're trying to keep track of content and bill customers on how many GB of content have been streamed each month, you have to be accurate," said Pirvali. "If we under-count then we won't bill enough for our CDN services, and if we over-count we risk losing good customers. VoltDB is a key component in our design of real-time analytics and a completely accurate billing system."

According to Pirvali, "Our real-time analytic platform opens an ocean of cool ideas that no other CDN company provides. One would be 'elastic provisioning', something similar to what Amazon has been doing at the VM level. Elastic provision says that if we can detect in real-time that a customer's serving needs are suddenly increasing exponentially, we can dynamically assign more web servers to scale the web application under heavy load. 'Speculative pre-loading' is another cool idea that says that if requests for page1.html and pic-1.jpg, pic-2.jpg, and pic-3.jpg always come together, we could send back page-1.html and pic-1 through pic-3, together, but this would require HTTP2 or Speedy support. This way, the browser does not have to wait for pic-1 through pic-3 to render page-1.html."

Pirvali concluded, "But, even more important than these performance optimizations is the ability to monitor the quality of content delivery in real-time. For example, if a web-server is acting faulty and is sending back HTTP 5xx, we need to know about this in real-time and act on it. Otherwise, we would end up with lots of unhappy customers. Last but not least, we have to count 100% on the accuracy and reliability of our aggregation system to ensure billing accuracy."