Load Testing Our Event Pipeline 2019

Author: Vidhu Bhatnagar

Claps: --

Date: Jul 15, 2020

Black Friday and Cyber Monday (BFCM) are the biggest days in the ecommerce space. As a consumer, this means browsing the internet and scrolling through an endless stream of emails looking for those sweet deals.

On the flip side, this weekend is arguably the most important one for online stores and retailers. Over $80b is spent by Americans alone, and these businesses need to be on top of their marketing to make the most of this. Therefore, they rely on SaaS platforms like Klaviyo to run like well-oiled machines during this time.

Klaviyo is made up of several ingress, egress, analytics, and storage systems. From sending emails and SMS to processing events and providing analytics to our customers, we deal with a lot of data on an enormous scale. Here weâ€™ll discuss how we load tested our events (ingress) pipeline.

A lot of events are generated when people open and click links in emails, browse online stores, view products, and purchase items (just to name a few). We need to reliably ingest these events and process them to provide our customers with visibility into how their store is performing, how they can target their customers better, trigger flows, and so on. Every year on BFCM, we see a sustained ~10x increase in the number of events we receive per second compared to a regular day. In 2018, we saw ~75,000 events per second on BFCM, and in 2019 that number went up to ~250,000 per second.

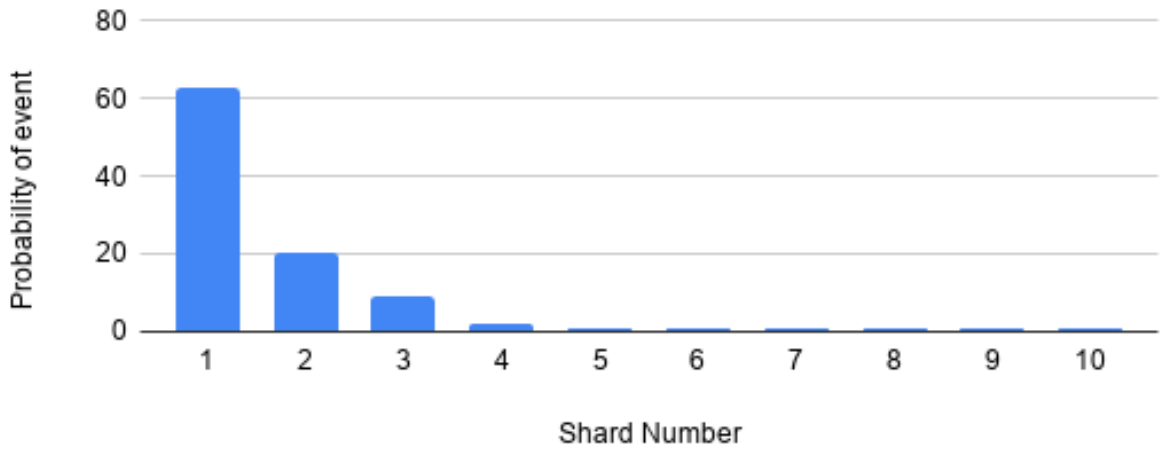
Planning the load tests

The idea of a load test is to see how your systems will perform in the real world. Simulating the real world is difficult. In data-centric systems, your test data must be representative of what your production systems are receiving.

For us, there were 3 factors which determined how representative our data is when compared to live data:

Distribution of events across accounts

Many of our databases are sharded by Account ID. This is why the distribution of events across accounts is important. As a result of this strategy, there is a very uneven â€œhockey stickâ€ distribution on the histogram of numbers of events by shard. A really large and active company can dominate and â€œheat upâ€ a shard. Therefore, generating our data with such a distribution was quite critical in testing for bottlenecks.



Histogram showing distribution of events across shards

Ratio of event types

We have two classes of data: email and API events. Email events are opens, deliveries, and clicks in emails. API events are interactions on our customersâ€™ ecommerce stores. Making sure the ratio is representative is crucial for representativeness.

Event payload sizes

Email events are pretty straightforward. We receive them in batches and they are roughly the same size and shape consisting of the event type (open/click), the recipient, and customer ID.

API events, on the other hand, can widely vary from a few bytes to over a megabyte. Trying to generate this type of data is difficult. The good news is we were able to scrape API event from production logs to reproduce representative payloads

The test driver

A bird eye view of our load test infrastructure and system looks like this.