influxdata

Flood IO uses InfluxCloud to generate the load of millions of users in real time

AN INFLUXDATA CASE STUDY

Tim Koopmans
CTO and Founder, Flood IO

September 2017

Overview

Flood IO offers on-demand load testing service on open source tools such as JMeter, Gatling, and Selenium with no test launch delays, no limits on number of tests and users, and real-time visibility into test performance. They wanted to enable customers to quickly distribute their test plan across hundreds of servers in multiple AWS regions. Quick feedback loops were also critical for successful testing, so Flood IO wanted to show test results as soon as they got them, allowing testers to stop a test and make changes as necessary. Limits on performance, visibility or scalability would jeopardize testing success and limit company growth. Flood IO's on-demand testing service uses InfluxData to provide insights into customers' performance tests. In addition, Kapacitor is used to automatically spin up test environments for customers and provide a real-time view of the tests they run.

The result: Flood IO's choice of InfluxData as its Time Series Platform freed it to offer load testing with no limits on number of tests or users and no test launch delays plus providing real-time views into results as tests ran.

InfluxDB serves as the Time Series Database to store Flood IO's time series data, and metrics and events are collected and analyzed by Kapacitor using InfluxCloud. Flood IO's distributed grid of semi-autonomous, loosely coupled nodes hosted on AWS infrastructure runs different testing tools and collates results in near real time across multiple geographic regions including the US, EU and Asia Pacific.

About Flood.io

Flood IO, based in Melbourne (Australia), is bringing a new approach to the traditional load testing space. Built for testers, Flood IO allows you to run distributed load tests for millions of concurrent users across the globe with your choice of JMeter, Gatling, Selenium or Capybara. Customers can use Flood IO's On Demand infrastructure or Host Your Own nodes on AWS. Unlike traditional load testing services, Flood is designed to maximize testing success. It allows you to run as many users and tests as you need, for as long as you like, while it takes care of the servers and collecting the results.

Flood IO provides a real-time view of a test as it runs, with aggregates of all core metrics, as well as the ability to drill down into a single transaction to view captured traces and analyze errors. You can integrate with your own services using Flood IO's API or share results from Flood IO's reports with your team in real time.

> *"Our early design was a bit of a nightmare to maintain...it was difficult to scale horizontally due to cluster dependencies that would impact grid startup time."*
>
> ***Tim Koopmans,*** *CTO and Founder*

The Business Problem

Flood IO wanted to provide a high-performance load testing service with real-time metrics and no test launch delays. With their existing architecture, it would take 5–20 min to start a 30-node grid, which was unacceptable, especially for paid customers trying to launch big tests. They wanted their design to be scalable to accommodate more users and larger tests.

Since their launch, Flood IO had also been maintaining a single free grid node that let everyone run short 5-minute flood tests. They called this node the "shared grid" as it was shared by all and gave testers an idea of a single node's performance to help baseline their tests. Three years later, the team found that the single node would often, on a daily basis, get hammered beyond recognition due to a combination of free accounts running suspect-looking tests, or just normal wear and tear of a few hundred tests hitting it daily. Their support tickets started to feature shared-grid-type questions and issues more often, and they regularly found themselves replacing the shared grid with a fresh node. The shared grid also added artificial delay to customers waiting for tests behind other customers.

The Technical Problem

Flood IO's first-generation design used Elasticsearch. Each grid (infrastructure that the tests run on) would operate in a cluster of 30 nodes. Each node ran Elasticsearch, and

they would pull that information from those grids. This early design presented challenges that had to be overcome to solve the business problem:

- It was difficult to scale horizontally since cluster dependencies would impact grid node startup time.
- Flood IO was pulling data from Elasticsearch clusters in real time and presenting it to the user, but once the test was finished and they needed to retrieve data, they had code that would be serializing data back and forth, and data wouldn't be visible without significant delays.
- They had different storage mechanisms for hot and cold time series data, some coming from Elasticsearch and some from AWS S3.
- They were using the full-text search database Elasticsearch to store time series data, which is not that database's intended use.

When re-architecturing to eliminate cluster dependencies, they wanted the new design to be a:

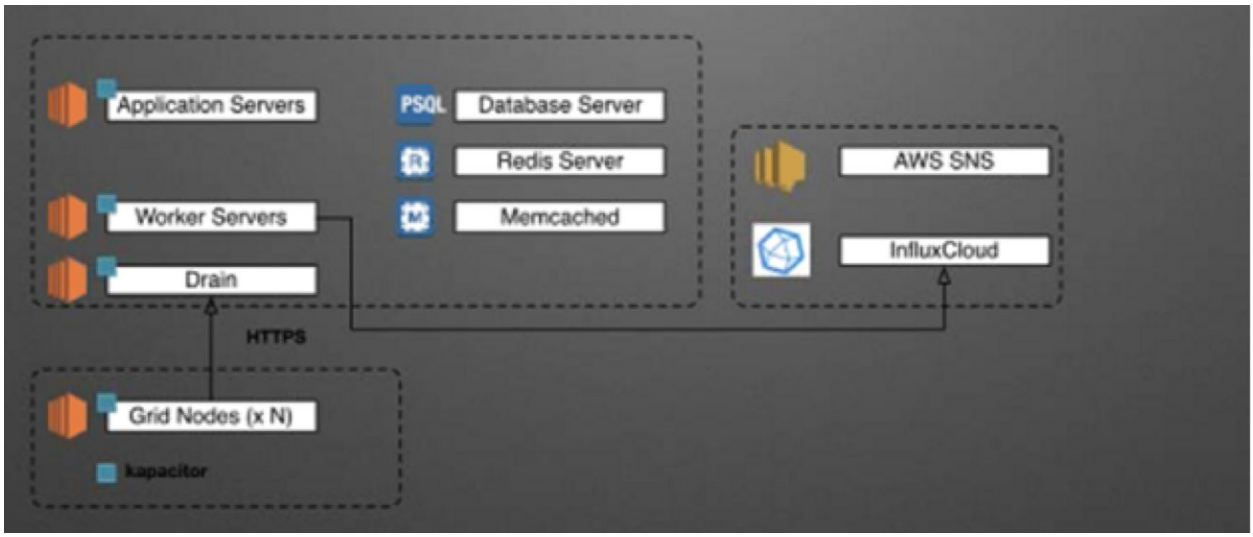
- Distributed system – a software system in which components located on networked computers communicate and coordinate their actions by passing messages to achieve a common goal.
- Loosely coupled system – one in which each of its components has, or makes use of, little or no knowledge of the definitions of other separate components.
- Shared Nothing architecture (SN) – a distributed computing architecture in which each node is independent and self-sufficient, and there is no single point of contention across the system. None of the nodes share memory or disk storage.

Upon discovering InfluxDB through online forums and looking into its early versions, they were happy with the throughput they were getting, the amount of disk space that was used and the low memory overhead. So they decided to adopt it to meet their design goals.

> *"If we hadn't adopted InfluxData, we wouldn't have been able to scale to the capacity or requirements of customers we have today. Running 900 nodes across 30-node clusters, Elasticsearch would have been extremely painful. We probably would have lost business."*

The Solution

In Flood IO's use case, InfluxDB and its native data processing engine Kapacitor work together to store, collect, and aggregate time series data in real time.



InfluxCloud and Kapacitor Power Flood IO's Platform

Flood IO's current solution achieves its design goals of being a distributed, loosely coupled, Shared Nothing Load Test Platform. Like their previous solution, their current one has application servers sitting in a central location, but instead of worker servers pulling information from the grid nodes themselves, the grid nodes are actually pushing information up to a proxy service that the team call "Drain" and that they've written in Go.

- Drain can cue up messages or points to be written by their Redis server.
- Then the system processes that cue asynchronously without delay and runs all the points up into InfluxCloud, which processes around 60 write points/sec.
- Kapacitor is used to collect and aggregate the results on the nodes themselves since it can talk to InfluxData.
- Each of Flood IO's grid nodes runs an instance of Kapacitor, and that's how they get the performance data from the tools into their system.
- Results are aggregated to the nearest 15-second interval.

> *"InfluxData keeps improving with every release, and things get faster."*

What's Next?

Flood IO plans to make the most of InfluxData to further improve its service and is considering:

- Developing the ability to alarm / callout via escalation procedures in the cloud solution
- Exploring aggregations with finer resolution (down to 1 second)
- Structuring schema/queries to support multiple percentiles
- Enforcing data retention in the future as their measurements expand

Results

Using InfluxData, Flood IO is now a distributed cloud-based load testing platform that offers truly flexible and affordable testing. Hundreds of companies today—in various industries including retail, finance, telecom, and government—use Flood IO to generate the load of millions of users around the globe. Flood IO even has customers launching in excess of 900 nodes in any one load test. Customers can simulate pretend users coming at their systems, observe how their systems react, where the system is under stress when the number of users is ramped up, and address hotspots accordingly.

Quick wins gained from adopting InfluxData include:

- There is no more clustering on grid nodes and no restrictions on testing, which enabled infinite horizontal scale—customers can run as many tests as they like within any timeframe, start, stop as they please, scale out or back with their grid nodes.
- Data is pushed centrally, so it's available as it arrives.
- Time series data is finally in a Time Series Database.

With these wins, Flood IO offers all the features customers need to scale their load tests on its infrastructure.

- Load Tests are now fully configurable, from test duration, to number of users to run on each node you want to simulate load for, and ramp up duration. Customers can select the Grids to distribute their test to, as well as the region to run grids in, from the 14 regions supported by AWS. Customers can choose how many nodes to start with, how long to run them, and can have hundreds of nodes distributed across multiple grids in different regions. They can view generic information about how

the grid is running such as CPU, Memory, IO Utilization and many other performance indicators.

- Load Tests now provide real-time visibility. They collect measurements which are stored in InfluxData, such as concurrency (number of active users), response time, transaction rate, network throughput, latency, passed transactions, failed transactions—reported to customers in real time, down to 15 second point resolution. Flood IO provides customers with further insight into each individual transaction through descriptive stats and error & status codes. Customers can zoom into the chart to find out what was going on in that test at that point in time and access actionable data to solve their performance problems.

Powered by InfluxData, Flood IO is helping developers perform continuous testing to avoid outages associated with excess load and to monitor performance under load to detect performance regressions and scaling bottlenecks before customers discover them.

What's Next?

InfluxDB Documentation, Downloads & Guides

- 1.3 Download
- 1.3 Installation Guide
- 1.3 Getting Started
- 1.3 Schema Design
- 1.3 Line Protocol Reference
- 1.3 Key Concepts

Have Questions or Need Help?

- InfluxData Community
- contact@influxdata.com
- Virtual Training