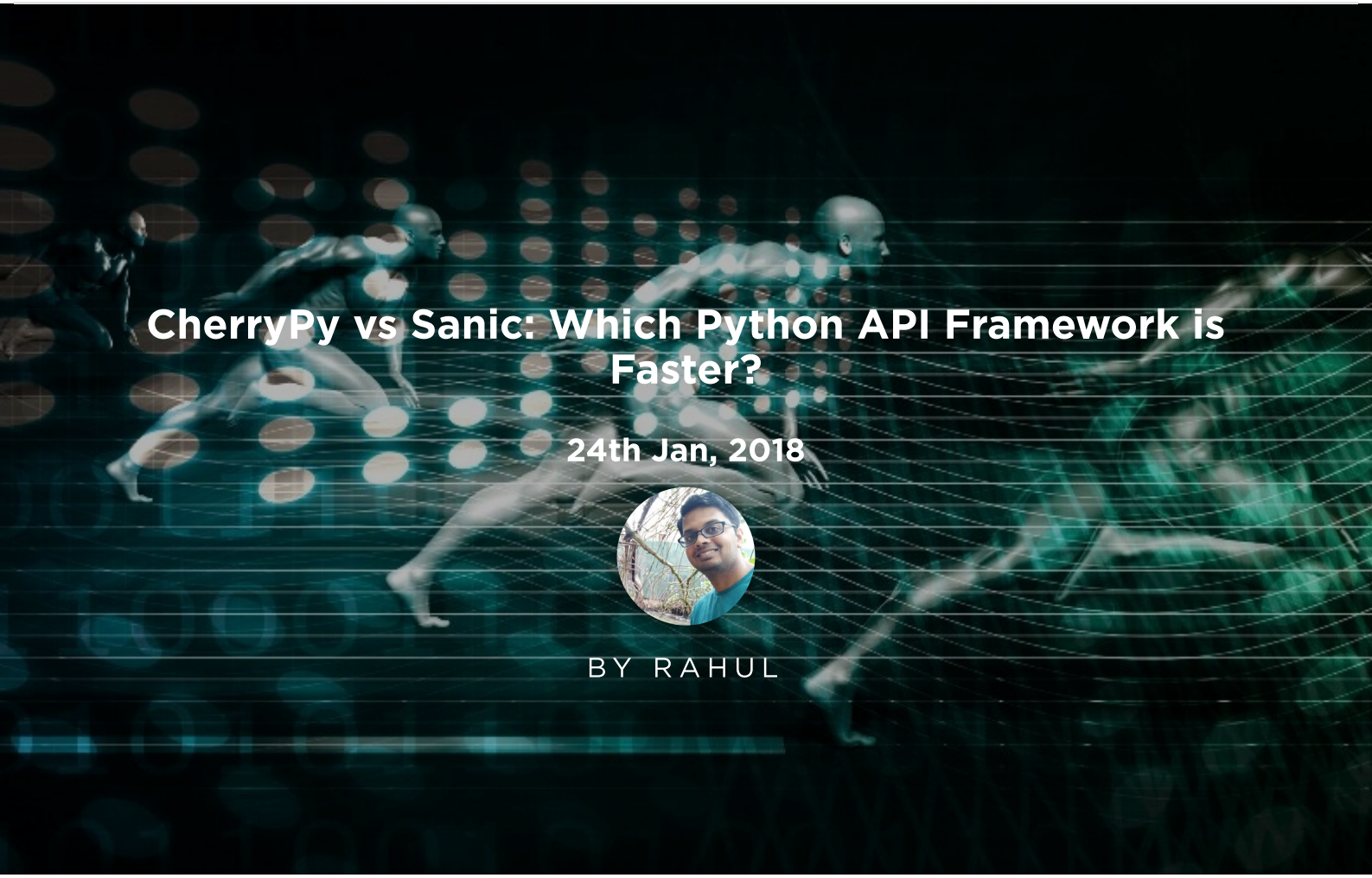


CherryPy vs Sanic: Which Python API Framework is Faster?

24th Jan, 2018

BY RAHUL

Rest APIs play a crucial role in the exchange of data between internal systems of an enterprise, or when connecting with external services.

When an organization relies on APIs to deliver a service to its clients, the APIs' performance is crucial, and can make or break the success of the service. It is, therefore, essential to consider and choose an appropriate API framework during the design phase of development. Benefits of choosing the right API framework include the ability to deploy applications at scale, ensuring agility of performance, and future-proofing front-end technologies.

At DataWeave, we provide Competitive Intelligence as a Service to retailers and consumer brands by aggregating Web data at scale and distilling them to produce actionable competitive insights. To this end, our proprietary data aggregation and analysis platform captures and compiles over a hundred million data points from the Web each day. Sure enough, our platform relies on APIs to deliver data and insights to our customers, as well as for communication between internal subsystems.

Some Python REST API frameworks we use are:

- Tornado — which supports asynchronous requests
- CherryPy — which is multi-threaded
- Flask-Gunicorn — which enables easy worker management

It is essential to evaluate API frameworks depending on the demands of your tech platforms and your objectives. At DataWeave, we assess them based on their speed and their ability to support high concurrency. So far, we've been using CherryPy, a widely used framework, which has served us well.

CherryPy

An easy to use API framework, Cherrypy does not require complex customizations, runs out of the box, and supports concurrency. At DataWeave, we rely on CherryPy to access configurations, serve data to and from different datastores, and deliver customized insights to our customers. So far, this framework has displayed very impressive performance.

However, a couple of months ago, we were in the process of migrating to python 3 (from python 2), opening doors to a new API framework written exclusively for python 3 — Sanic.

Sanic

Sanic uses the same framework that libuv uses, and hence is a good contender for being fast.

(Libuv is an asynchronous event handler, and one of the reasons for its agility is its ability to handle asynchronous events through callbacks. More info on libuv can be found **here**)

In fact, **Sanic is reported** to be one of the fastest API frameworks in the world today, and uses the same event handler framework as nodejs, which is known to serve fast APIs. More information on Sanic can be found **here**.

So we asked ourselves, should we move from CherryPy to Sanic?

Before jumping on the hype bandwagon, we looked to first benchmark Sanic with CherryPy.

CherryPy vs Sanic

Objective

Benchmark CherryPy and Sanic to process **500 concurrent requests**, at a rate of **3500 requests per second.**

Test Setup

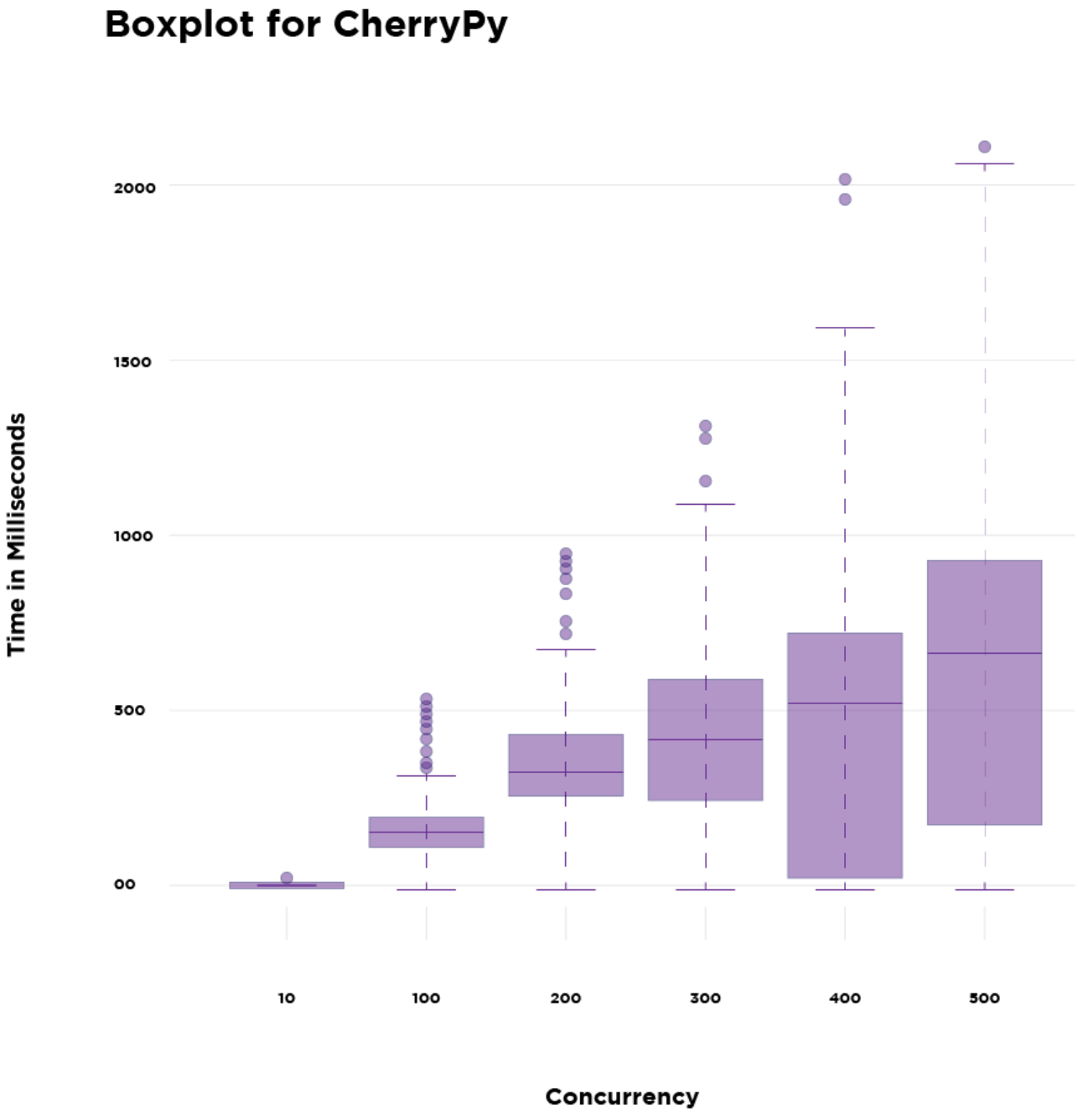
```
Machine configuration: 4 VCPUs/ 8GB RAM.
Network Cloud: GCE
Number of Cherrypy/Sanic APIs: 3 (inserting data into 3 topics of a Kafka cluster)
Testing tool : apache benchmarking (ab)
Payload size: All requests are POST requests with 2.1KB of payload.
```

API Details

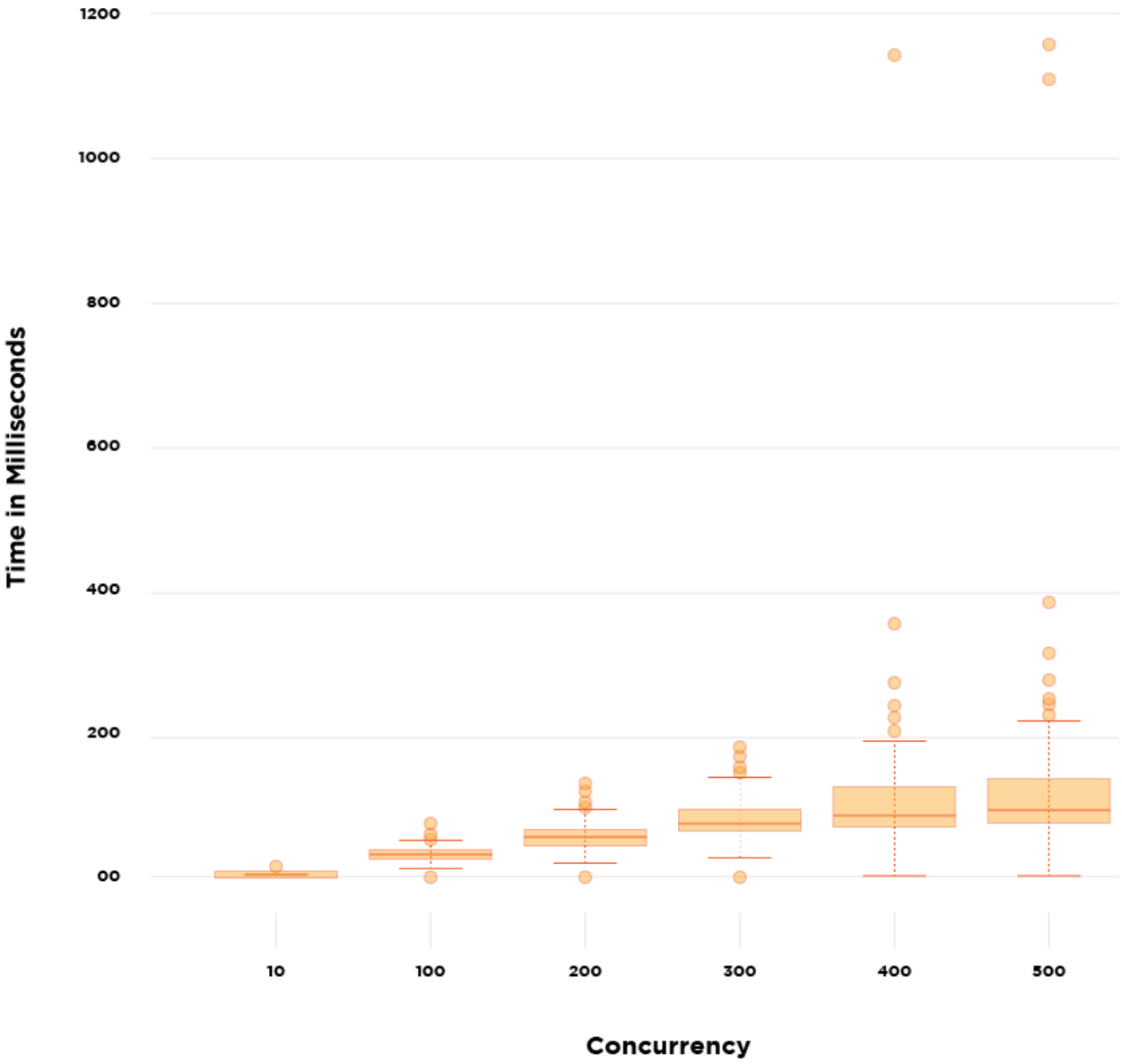
```
Sanic: In Async mode
Cherrypy: 10 concurrent threads in each API – a total of 30 concurrent threads
Concurrency: Tested APIs at various concurrency levels. The concurrency varied between
10 and 500
Number of requests: 1,00,000
```

Results

Requests Completion: A lower mean and a lower spread indicate better performance



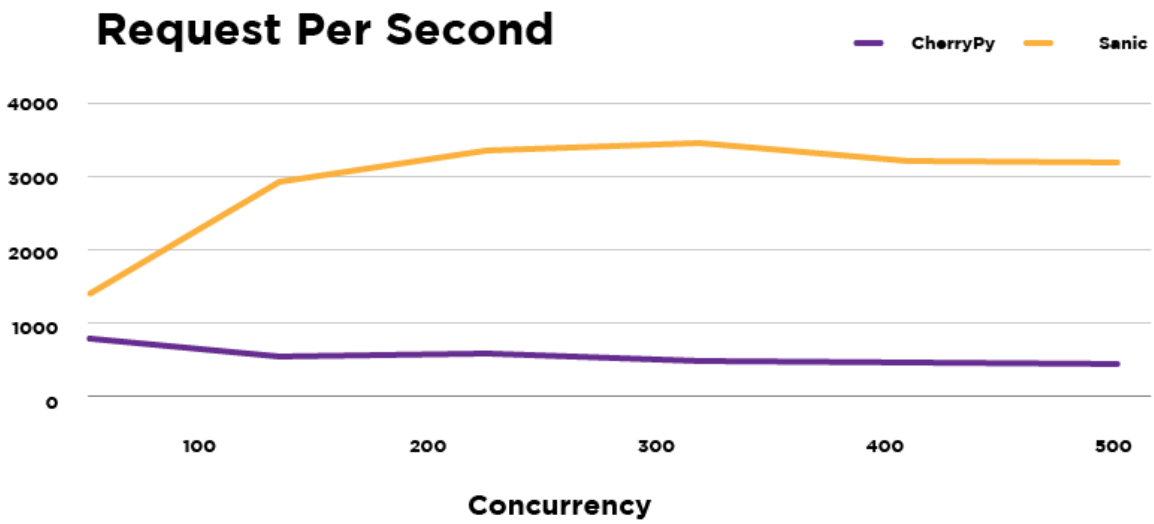
Boxplot for Sanic



Observation

When the concurrency is as low as 10, there is not much difference between the performance of the two API frameworks. However, as the concurrency increases, Sanic's performance becomes more predictable, and the API framework functions with lower response times.

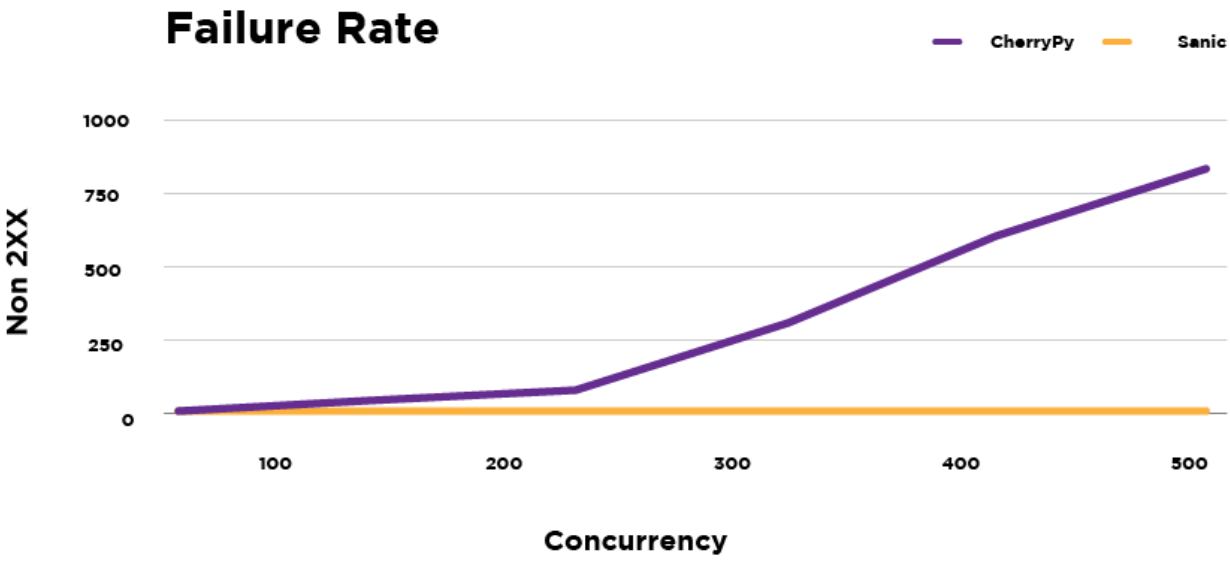
Requests / Second: Higher values indicate faster performance



Sanic clearly achieves higher requests/second because:

- Sanic is running in Async mode
- The mean response time for Sanic is much lower, compared to CherryPy

Failures: Lower values indicate better reliability



Number of non-2xx responses increased for CherryPy with increase in concurrency. In contrast, number of failed requests in Sanic were below 10, even at high concurrency values.

Conclusion

Sanic clearly outperformed CherryPy, and was much faster, while supporting higher concurrency and requests per second, and displaying significantly lower failure rates.

Following these results, we transitioned to Sanic for ingesting high volume data into our datastores, and started seeing much faster and reliable performance. We now aggregate much larger volumes of data from the Web, at faster rates.

Of course, as mentioned earlier in the article, it is important to evaluate your API framework based on the nuances of your setup and its relevant objectives. In our setup, Sanic definitely seems to perform better than CherryPy.

What do you think? Let me know your thoughts in the comments section below.

If you're curious to know more about DataWeave's technology platform, check out **our website**, and if you wish to join our team, check out our **jobs page!**

- Rahul Ramesh

Technical Architect at DataWeave, 24th Jan, 2018

DATA ENGINEERING