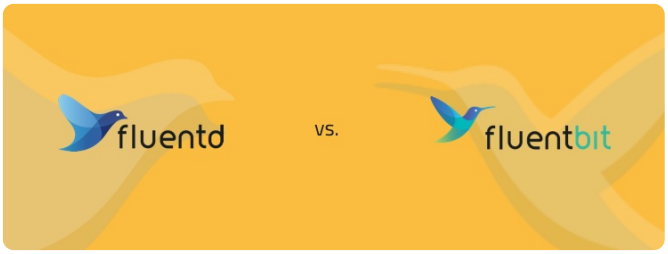
#Comparison #Fluentd #Kubernetes

# Fluentd vs. Fluent Bit: Side by Side Comparison

Daniel Berman
Updated on: Mar 25, 2020




We all like a pretty dashboard. For us data nerds, there's something extremely enticing about the colors and graphs depicting our environment in real-time. But while Kibana and Grafana bask in glory, there is a lot of heavy lifting being done behind the scenes to actually collect the data. This is where tools like Fluentd and Fluent Bit come in. This heavy lifting is performed by a variety of different tools called log forwarders, aggregators or shippers. These tools handle the tasks of pulling and receiving the data from multiple systems, transforming it into a meaningful set of fields, and eventually streaming the output to a defined destination for storage.

Fluentd is one of the most popular log aggregators used in ELK-based logging pipelines. In fact, it's so popular, that the "EFK Stack" (Elasticsearch, Fluentd, Kibana) has become an actual thing. A survey by Datadog lists Fluentd as the 8th most used Docker image. Fluent Bit is a relatively new player in town, but is also rising in popularity, especially in Docker and Kubernetes environments.

And so users are now wondering what part Fluent Bit should and can play in a logging pipeline. Is this a new and improved version of Fluentd? Should we retire Fluentd in favor of Fluent Bit? Should the two be used in tandem? In this article, I'll be providing a high-level comparison so users can understand the difference between the two and when to use them.

## What is Fluentd?

Fluentd is an open source log collector, processor, and aggregator that was created back in 2011 by the folks at Treasure Data. Written in Ruby, Fluentd was created to act as a unified logging layer — a one-stop component that can aggregate data from multiple sources, unify the differently

Design wise — performance, scalability, and reliability are some of Fluentd's outstanding features. A vanilla Fluentd deployment will run on ~40MB of memory and is capable of processing above 10,000 events per second. Adding new inputs or outputs is relatively simple and has little effect on performance. Fluentd uses disk or memory for buffering and queuing to handle transmission failures or data overload and supports multiple configuration options to ensure a more resilient data pipeline.

#### Fluentd, Kubernetes & Docker

Fluentd has been around for some time now and has developed a rich ecosystem consisting of more than 700 different plugins that extend its functionality. Fluentd is the de facto standard log aggregator used for logging in Kubernetes and as mentioned above, is one of the widely used Docker images.

The Fluentd Docker image includes tags `debian`, `armhf` for ARM base images, `onbuild` to build, and `edge` for testing.

Kubernetes utilizes daemonsets to ensure multiple nodes run copies of pods. There is a specific Kubernetes Fluentd daemonset for running Fluentd. You can clone the repository here:

```
$ git clone https://github.com/fluent/fluentd-kubernete
```

If you're an ELK user, all this sounds somewhat similar to what Logstash has to offer. There are of course some differences, and we cover some of these in our Fluentd vs. Logstash comparison.

## What is Fluent Bit?

Fluent Bit is an open source log collector and processor also created by the folks at Treasure Data in 2015. Written in C, Fluent Bit was created with a specific use case in mind — highly distributed environments where limited capacity and reduced overhead (memory and CPU) are a huge consideration.

To serve this purpose, Fluent Bit was designed for high performance and comes with a super light footprint, running on ~450KB only. An abstracted I/O handler allows asynchronous and event-driven read/write operations. For resiliency and reliability, various configuration option are available for defining retries and the buffer limit.

#### Fluent Bit, Kubernetes & Docker

Fluent Bit is also extensible, but has a smaller eco-system compared to Fluentd. Inputs include syslog, tcp, systemd/journald but also CPU, memory, and disk. Outputs include Elasticsearch, InfluxDB, file and http. For Kubernetes deployments, a dedicated filter plugin will add metadata to log data, such as the pod's name and namespace, and the containers name/ID.

annotations, and labels. The latter two and the pod ID can be found by querying the Kubernetes API Server. There are also configurations for its `regex_parser`, debugging (`tls.debug`), `buffer_size` as mentioned above, and more.

## Comparing Fluentd and Fluent Bit

Both Fluentd and Fluent Bit were developed by Treasure Data to help users build centralized, reliable and efficient logging pipelines. The vision behind Fluentd, and later on, Fluent Bit, was to help overcome some of the challenges involved in logging production environments — formatting unstructured data, aggregation from multiple data sources, resiliency and security.

While there are architectural and design similarities between the two tools, there are also some core differences that should be taken into consideration when picking between the two.

Below is a table summing up the differences between the two tools:

Source: Fluent Bit documentation

### Performance

As seen in the table above, while Fluentd can boast efficiency and a relatively small footprint, Fluent Bit takes it up a notch or two. To gauge the difference, take a look at the recommended default specs for running the two tools in Kubernetes. You can do the math yourselves.

Fluentd:

```
resources:
  limits:
    memory: 500Mi
```

```
    memory: 200Mi
```

Fluent Bit:

```
resources:
  requests:
    cpu: 5m
    memory: 10Mi
  limits:
    cpu: 50m
    memory: 60Mi
```

In an environment consisting of hundreds of servers, the aggregated effect on CPU and memory utilization is substantial.

**Aggregation**

Fluent Bit acts as a collector and forwarder and was designed with performance in mind, as described above. Fluentd was designed to handle heavy throughput — aggregating from multiple inputs, processing data and routing to different outputs. Fluent Bit is not as pluggable and flexible as Fluentd, which can be integrated with a much larger amount of input and output sources.

**Monitoring**

Fluent Bit ships with native support for metric collection from the environment they are deployed on. A variety of input plugins, such as cpu and disk, will collect data on CPU and memory usage, and forward them to a selected output. Version 0.13 also ships with support for Prometheus metrics. Fluentd does not ship with this functionality and would most likely act as the aggregator for these metrics.

**Ecosystem**

While Fluentd and Fluent Bit are both pluggable by design, with various input, filter and output plugins available, Fluentd (with ~700 plugins) naturally has more plugins than Fluent Bit (with ~45 plugins), functioning as an aggregator in logging pipelines and being the older tool. Fluentd's history contributed to its adoption and large ecosystem, with the Fluentd Docker driver and Kubernetes Metadata Filter driving adoption in Dockerized and Kubernetes environments.

**Community**

Taking a look at the code repositories on GitHub provides some insight on how popular and active both these projects are.

Fluentd

- Stars: 6423
- Forks: 777
- Watch: 339

Fluent Bit

- Stars: 586
- Forks: 135
- Watch: 46
- Contributors: 40
- Commits: 3173

## So, when do I use Fluentd or Fluent Bit?

In a way, Fluent Bit is to Fluentd, what Beats are to Logstash — a lightweight shipper that can be installed as agents on edge hosts or devices in a distributed architecture

Fluentd instance deployed per cluster and acting as an aggregator — processing the data and routing it to different sources based on tags.

Same goes for an IoT architecture, where Fluent Bit is installed per device, sending data to a Fluentd instance.

Fluent Bit can be used on it own of course but has far less to offer in terms of aggregation capabilities and with a much smaller amount of plugins for integrating with other solutions.

## Summing it up

The difference between Fluentd and Fluent Bit can therefore be summed up simply to the difference between log forwarders and log aggregators. The former are installed on edge hosts to receive local events. Once received, the event is forwarded to the log aggregators. The latter are daemons that receive streams of events from the log forwarders, buffer them and periodically upload the data to a data store of some sorts.

The combination of Fluentd and Fluent Bit is becoming extremely popular in Kubernetes deployments because of the way they compliment each other — Fluent Bit acting as a lightweight shipper collecting data from the different nodes in the cluster and forwarding the data to Fluentd for aggregation, processing and routing to any of the supported output destinations.

The rise of Kubernetes will only help drive adoption of Fluent Bit and it would not surprise anyone if the ecosystem around this logging tool explodes with new plugins and features.

YOU MIGHT ALSO LIKE

DevOps

**A Fluent Bit Tutorial: Shipping to Elasticsearch**

ELK Stack

**Fluentd vs Logstash: A Comparison of Log Collectors**

DevOps

**A Fluentd Tutorial: Shipping Logs to Logz.io**

← Back to Blog