OpenSearch Data Prepper

OpenSearch Data Prepper is a server-side data collector capable of filtering, enriching, transforming, normalizing, and aggregating data for downstream analysis and visualization. Data Prepper is the preferred data ingestion tool for OpenSearch. It is recommended for most data ingestion use cases in OpenSearch and for processing large, complex datasets.

With Data Prepper you can build custom pipelines to improve the operational view of applications. Two common use cases for Data Prepp are trace analytics and log analytics. Trace analytics can help you visualize

event flows and identify performance problems. Log analytics equips you with tools to enhance your search capabilities, conduct comprehensive analysis, and gain insights into your applications' performance and behavior.

Key concepts and fundamentals

Data Prepper ingests data through customizable pipelines. These pipelines consist of pluggable components that you can customize to fit your needs, even allowing you to plug in your own implementations. A Data Prepper pipeline consists of the following components:

- One source
- One or more sinks
- (Optional) One buffer
- (Optional) One or more processors

Each pipeline contains two required components: `source` and `sink`. If a `buffer`, a `processor`, or both are missing from the pipeline, then Data Prepper uses the default `bounded_blocking` buffer and a no-op processor. Note that a single instance of Data Prepper can have one or more pipelines.

Basic pipeline configurations

To understand how the pipeline components function within a Data Prepper configuration, see the following examples. Each pipeline configuration uses a `yaml` file format. For more information, see Pipelines for more information and examples.

Minimal configuration

The following minimal pipeline configuration reads from the file source and writes the data to another file on the same path. It uses the default options for the `buffer` and `processor` components.

```
sample-pipeline:
  source:
```

```
    file:
        path: <path/to/input-file>
  sink:
    - file:
        path: <path/to/output-file>
```

Comprehensive configuration

The following comprehensive pipeline configuration uses both required and optional components:

```
sample-pipeline:
  workers: 4 # Number of workers
  delay: 100 # in milliseconds, how often the workers should run
  source:
    file:
        path: <path/to/input-file>
  buffer:
    bounded_blocking:
      buffer_size: 1024 # max number of events the buffer will accept
      batch_size: 256 # max number of events the buffer will drain for
  processor:
    - string_converter:
       upper_case: true
  sink:
    - file:
       path: <path/to/output-file>
```

In the given pipeline configuration, the `source` component reads string events from the `input-file` and pushes the data to a bounded buffer with a maximum size of `1024`. The `workers` component specifies `4` concurrent threads that will process events from the buffer, each reading a maximum of `256` events from the buffer every `100` milliseconds. Each `workers` component runs the `string_converter` processor, which converts the strings to uppercase and writes the processed output to the `output-file`.

Next steps

- Getting started with OpenSearch Data Prepper.

- Get familiar with Data Prepper pipelines.
- Explore common use cases.