



Near Real-Time Logging

Paul Nilsson, Shuwei Ye

October 15, 2021

NPPS Group Meeting

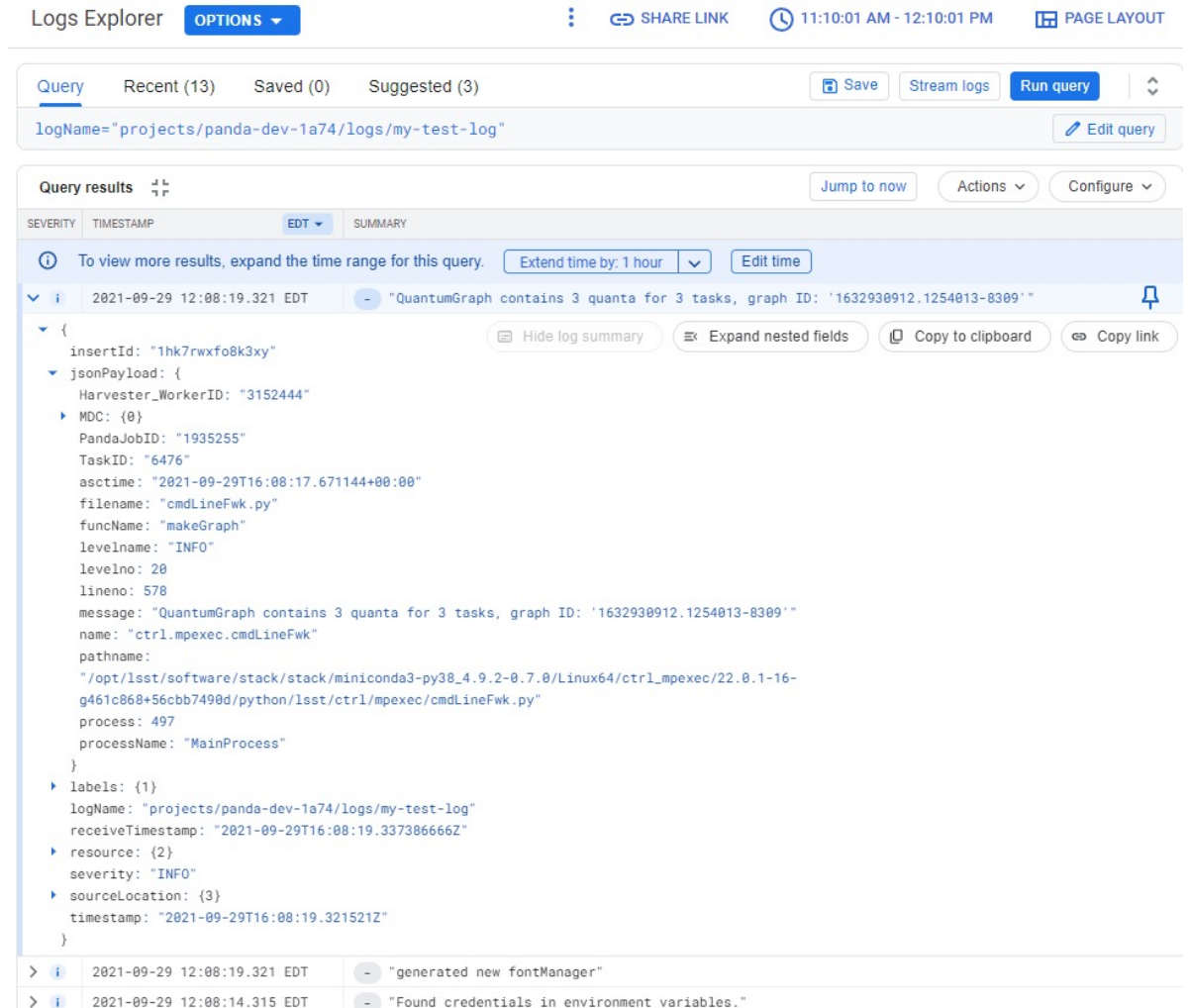


Introduction

- In PanDA, the payload stdout/stderr are part of the job log and can be reached via the job page in the PanDA monitor – after the job has finished
- Ideally, a user should be able to follow the progress of the payload as it runs
- Since a long time, it is possible to turn on debug mode on a running job
 - Instructs the pilot to send tails of the payload stdout back to the server
 - Server, in turn, makes the tail available on the job monitor
 - Primarily used by power users debugging production tasks
 - Update frequency hardcoded in the pilot – one update every five minutes
- Recently, a more advanced debug mode was introduced
 - E.g. the pilot can now find any log file of interest to the user
 - But still with infrequent updates that are not practical to increase due to the load on the PanDA server
- Rubin developers suggested to use log collectors tailormade for rapid logging
 - E.g. Fluentd, Google Cloud Logging, Logstash
 - Note: requires special JSON format of log messages which makes downstream data processing easier and makes it possible to query many jobs (of the same task or multiple tasks) at the same time based on log fields

The Rubin Implementation

- Rubin payload already produces stdout in JSON
- When there is a change, tail is picked up by a thread in the Pilot - checked once every five seconds
- Pilot enhances the JSON tail with job info and sends it to an external logging service
- Logging service puts the tail in desired place
- User can follow the progress of the payload as it runs in near real time



The screenshot displays the Logs Explorer interface. At the top, there are navigation options: "LogName='projects/panda-dev-1a74/logs/my-test-log'", "Save", "Stream logs", "Run query", and "Edit query". Below this, the "Query results" section shows a table with columns for "SEVERITY", "TIMESTAMP", and "SUMMARY". The first entry is expanded, showing a detailed JSON log entry:

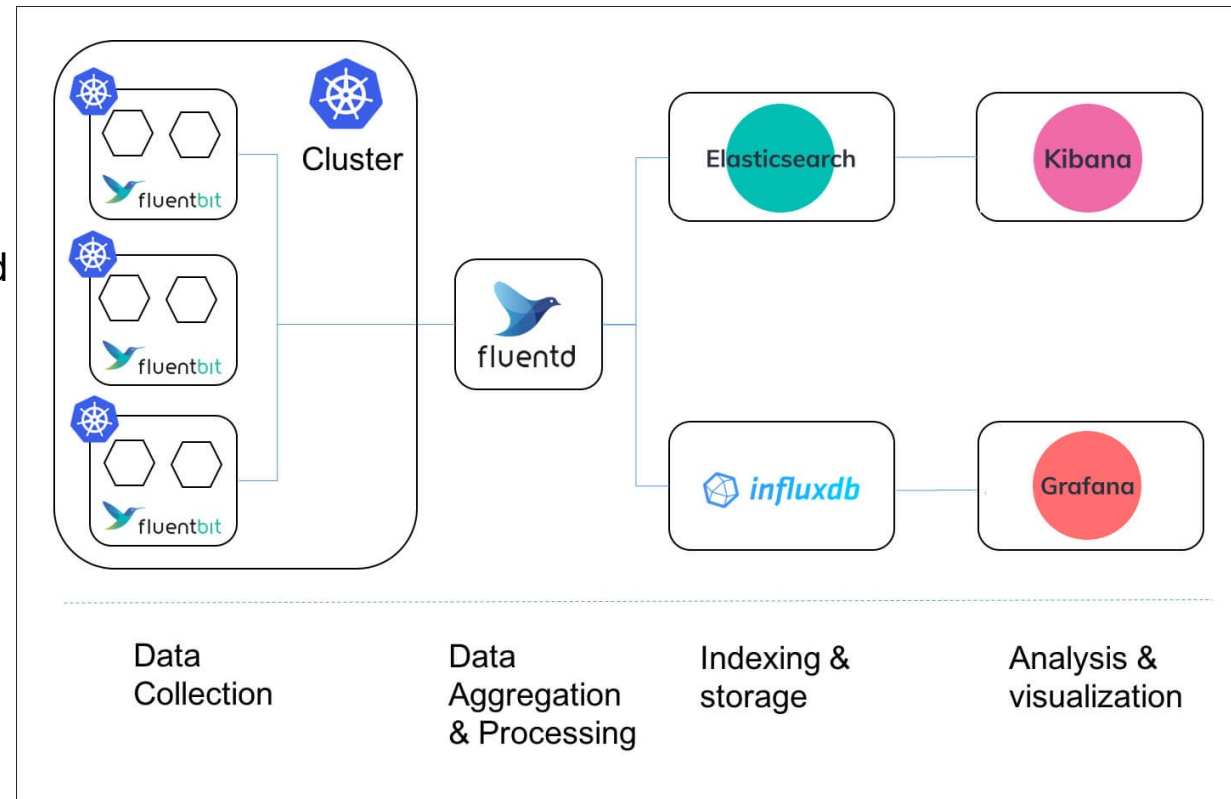
```
{
  insertId: "1hk7rxfo8k3xy"
  jsonPayload: {
    Harvester_WorkerID: "3152444"
    MDC: {}
    PandaJobID: "1935255"
    TaskID: "6476"
    asctime: "2021-09-29T16:08:17.671144+00:00"
    filename: "cmdLineFwk.py"
    funcName: "makeGraph"
    levelName: "INFO"
    levelNo: 20
    lineno: 578
    message: "QuantumGraph contains 3 quanta for 3 tasks, graph ID: '1632930912.1254013-8309'"
    name: "ctrl.mpexec.cmdLineFwk"
    pathname:
      "/opt/lstt/software/stack/stack/miniconda3-py38_4.9.2-0.7.0/Linux64/ctrl_mpexec/22.0.1-16-g461c868+56cbb7490d/python/lstt/ctrl/mpexec/cmdLineFwk.py"
    process: 497
    processName: "MainProcess"
  }
  labels: {1}
  logName: "projects/panda-dev-1a74/logs/my-test-log"
  receiveTimestamp: "2021-09-29T16:08:19.337386666Z"
  resource: {2}
  severity: "INFO"
  sourceLocation: {3}
  timestamp: "2021-09-29T16:08:19.321521Z"
}
```

Below the main entry, there are two more log entries visible in a table format:

SEVERITY	TIMESTAMP	SUMMARY
> i	2021-09-29 12:08:19.321 EDT	"generated new fontManager"
> i	2021-09-29 12:08:14.315 EDT	"Found credentials in environment variables."

A few more Rub details

- Preliminary implementation in Pilot 2 already used in Rubin production
 - Pilot 3 implementation in progress - probably won't be ready for first Pilot 3 release (within days)
- Rubin has implemented support in Pilot for both fluentd and google-cloud-logging
 - Currently using google-cloud-logging
 - Fluentd tests under way
 - Log messages seen on server
 - Fluent-bit services deployed on cluster as log forwarders
 - Central fluentd server as log aggregator
 - Logstash test planned as well
 - Logging switched for all jobs
 - GCL keeps logs for 30 days (default)
 - Better to have user request this service via job option
 - Fluentd+Elasticsearch+Kibana possible solution + other services



ATLAS: Preliminary plans

- Discussions are under way
- Central installation of fluentd server
 - “Fluentd’s performance has been put to the test at many large services; in fact, a regular PC box can handle 18,000 messages/second with a single process ” - <https://www.fluentd.org/faqs>
 - Testing likely to start with Logstash
- Pilot would only send log messages e.g. for jobs in debug mode / when logging has been switched on by user
 - If service scales well, it could be switched on for all user jobs (and logs could be purged after a week or so)
- ATLAS payloads create output in text - Pilot will need to format tail in JSON
- Send frequency to be tested / evaluated
- Fluentd needs to send logs somewhere (it’s just the collector)
 - Again, Fluentd+Elasticsearch+Kibana possible solution
- Logs to be made available on PanDA Monitor