Data Collection and Monitoring Across Heterogeneous Workflows in Pegasus

Ryan Tanaka
tanaka@isi.edu

USC Viterbi
Information Sciences Institute

https://pegasus.isi.edu/
Background: Gravitational Waves, Workflows, LIGO

**Laser Interferometer Gravitational Wave Observatory:**

- Facility for gravitational wave research
- Methods:
  - PyCBC software package
  - Pegasus WMS workflows
  - Compute using OSG, XSEDE, etc.

What do these workflows look like...
Advanced PyCBC Workflows:
- 60,000 compute tasks
- 5,000 input files (10GB total)
- 60,000 output files (60GB total)
- Post run analysis

Challenges:
- Error analysis on workflows of this scale
- Monitoring across multiple runs, users, and submit machines
Advanced PyCBC Workflows:
- 60,000 compute tasks
- 5,000 input files (10GB total)
- 60,000 output files (60GB total)
- Post run analysis

Challenges:
- Error analysis on workflows of this scale
- Monitoring across multiple runs, users, and submit machines
Background: Empowering LIGO Researchers

CIF21 DIBBs: Domain-Aware Management of Heterogeneous Workflows
Active Data Management for Gravitational-Wave Science
PI: Duncan Brown¹, Co-PIs: Peter Couvares² Ewa Deelman³, Jian Qin¹ NSF Award ACI-1443047
¹ Syracuse University. 2 LIGO Caltech, 3 USC Information Sciences Institute.

Goals:

● Develop new data management techniques in Pegasus
● Improve data access for LIGO researchers
● Enhance Pegasus workflow monitoring capabilities
● Enable LIGO researchers to conduct analysis across multiple PyCBC pipeline runs
Background: Empowering LIGO Researchers

**Outcomes:**

- Developed Pegasus extensions to capture runtime provenance metadata
- Data storage Solutions
  - relational datastore linked to a single PyCBC run
  - Elasticsearch persisting data across multiple PyCBC runs
- Browser based monitoring/analytics solutions
  - Kibana: query/explore Elasticsearch data via a browser
  - Grafana: dashboard for viewing workflow runs at varying levels of granularity
Outline

- Background
- Data collection pipeline
- Getting started
- Demo
Data Collection Pipeline: Overview

1. A workflow job completes/fails and information is sent back to the submit host.
2. Workflow Event sent over AMQP
3. Consume messages from workflow-es.
4. Store messages into Elasticsearch
5. View data in Kibana
6. Store events in Elasticsearch
7. View data in Grafana
8. Workflow events can be conveniently viewed through the Kibana web interface.
9. Interactive visualizations can be viewed through the Grafana web interface.
Data Collection Pipeline: Data Flow

Sequence of Events:
1. Workflow job completes and information sent to pegasus-monitor
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-even-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
Data Collection Pipeline: Data Flow

Sequence of Events:

1. Workflow job completes and information sent to pegasus-monitor
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-even-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
Sequence of Events:

1. Workflow job completes and information sent to pegasus-monitor
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-even-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
**Data Collection Pipeline: Data Flow**

**Sequence of Events:**
1. Workflow job completes and information sent to pegasus-monitorrd
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-event-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
Sequence of Events:
1. Workflow job completes and information sent to pegasus-monitord
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-event-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
**Data Collection Pipeline: Data Flow**

**Sequence of Events:**

1. Workflow job completes and information sent to pegasus-monitord
2. Workflow event sent over AMQP to RabbitMQ message queue
3. Logstash consumes message from queue
4. Logstash inserts event under pegasus-composite-event-* index in Elasticsearch data store
5. Data Exploration/Visualization
   a. Workflow events viewable through Kibana
   b. Dashboard viewable through Grafana
Data Collection Pipeline: RabbitMQ

- Lightweight message broker
- Relevant advantages
  - Promotes **decoupling** between producer/consumers
    - You can swap out/augment Elasticsearch with another data storage solution
  - Can be **deployed in distributed environments** to meet high availability requirements

https://www.rabbitmq.com/
Data Collection Pipeline: ELK Stack

**Elasticsearch:**
- distributed, RESTful search and analytics engine
- Log analytics, application performance monitoring, infrastructure metrics

**Logstash:**
- Service that can aggregate and process data from various sources and insert it to elasticsearch

**Kibana:**
- Data visualization and exploration

https://www.elastic.co/what-is/elk-stack
Data Collection Pipeline: Grafana

- Customizable browser based dashboards
- Support for **multiple data sources**
  - Elasticsearch
  - MySql
  - Etc.
- Enable data sharing across teams without exposing database

https://grafana.com/
Data Collection Pipeline: How We Use It

**Deployment:**
- 3 node elasticsearch cluster on 3 VMs
- Single logstash instance in docker
- Single RabbitMQ instance

**Usage:**
- Also stores OSG data, system data
- Projects:
  - IRIS: Integrity Introspection For Scientific Workflows ([http://nrig.renci.org/project/iris-integrity-introspection-for-scientific-workflows/](http://nrig.renci.org/project/iris-integrity-introspection-for-scientific-workflows/))
Outline

- Background
- Data collection pipeline
  - Getting started
- Demo
Getting Started: Prerequisites

Requirements:

- Host with a static ip (**dibbs.isi.edu** in this example)
  - Following ports unused:
    - 5673 <- RabbitMQ
    - 15672 <- RabbitMQ
    - 9200 <- Elasticsearch
    - 9600 <- Logstash
    - 5601 <- Kibana
    - 3000 <- Grafana
- Docker v17.02+
- Docker Compose v3.5
Getting Started: Up and Running


```bash
```

Grant Read/Write Access Permissions:

- `dibbs-data-collection-setup/elasticsearch/data`
- `dibbs-data-collection-setup/grafana/data`
- `dibbs-data-collection-setup/kibana/data`
- `dibbs-data-collection-setup/rabbitmq/data`
Getting Started: Up and Running

Starting the pipeline:

```bash
cd dibbs-data-collection-setup
docker-compose up
```

Access:

- Kibana: dibbs.isi.edu:5602
- Grafana: dibbs.isi.edu:3000
Getting Started: Configuring Pegasus

Say that this setup is running on a host `dibbs.isi.edu`, then the following must be included in the pegasus configuration file used to run your workflows:

```plaintext
pegasus.monitord.encoding = json
pegasus.catalog.workflow.amqp.url = amqp://friend:donatedata@dibbs.isi.edu:5672/prod/workflows
```

*requires Pegasus 4.9.2+*
Getting Started: Sending Application Specific Metadata

* requires Pegasus 4.9.3+

```bash
1 # the start of the marker the monitord will look for in the stdout
2 echo '@@MONITORING_PAYLOAD - START@@@
3
4 # a json blurb describing the content following
5 # the actual content
6 cat <<EOF
7 {
8      "ts": 1437688574,
9      "monitoring_event": "metadata",
10      "payload": [
11          {
12              "name": "num_template_banks",
13              "value": 3
14          },
15          {
16              "name": "event_name",
17              "value": "binary start merger"
18          }
19      ]
20 }
21 EOF
22
23 # the end of the marker the monitord will look for in the stdout
24 echo '@@MONITORING_PAYLOAD - END@@@
```

[@@MONITORING_PAYLOAD - START@@@]

```
[
  "ts": <long>,
  "monitoring_event": "metadata",
  "payload": [
    [
      "name": <string>,
      "value": <scalar|string>
    ]
  ]
]
```

[@@MONITORING_PAYLOAD - END@@@]
Outline

- Background
- Data collection pipeline
- Getting started
  - Demo
Pegasus est. 2001
Automate, recover, and debug scientific computations

Get Started

Pegasus Online Office Hours
https://pegasus.isi.edu/blog/online-pegasus-office-hours

Bi-monthly basis on the second Friday of the month, where we address user questions and also apprise the community of new developments.

Pegasus Website
https://pegasus.isi.edu/

Users Mailing List
pegasus-users@isi.edu

Pegasus Website
pegasus-support@isi.edu