Feature Project Proposal——Akraino Profiling

Helloway He
(helloway.wewe@gmail.com)
05/02/2019
Background - Requirement for Unified Metrics Exposition Format

**Requirement**
- Have the unified metrics exposition format standard at the edge

**Advantages**
- Reduce development complexity
- Vendors only use define metrics exposition format
- Develop one format conversion tool at most
- Gather more comprehensive metrics, do better scheduling

---

**too many monitoring systems**

**too many devices**

**too many metrics formats**

Hard to choose the optimal running nodes/devices
High development complexity
How to seamlessly connect all edge nodes/devices
Background – OpenMetrics Metrics Exposition Format Standard

- A Metrics Exposition Format Standard
- CNCF sandbox project launched by the Prometheus community
- Prometheus has become a de-facto standard in metric monitoring
- Take out Prometheus' metrics format alone to set up OpenMetrics project
- Main contributors
  - Google, Prometheus, InfluxData, SolarWinds, OpenCensus, Uber, Data Dog, etc.

```
# HELP http_requests_total The total number of HTTP requests.
# TYPE http_requests_total counter
http_requests_total{method="post",code="200"} 1027 1395066363000
http_requests_total{method="post",code="400"} 3 1395066363000

# Escaping in label values:
msdos_file_access_time_seconds{path="C:\DIR\FILE.TXT",error="Cannot find file: n"FILE.TXT""} 1.458

# Minimalistic line:
metric_without_timestamp_and_labels 12.47

# A weird metric from before the epoch:
something_weird{problem="division by zero"} +Inf -3982045

# A histogram, which has a pretty complex representation in the text format:
# HELP http_request_duration_seconds A histogram of the request duration.
# TYPE http_request_duration_seconds histogram
http_request_duration_seconds_bucket{le="0.05"} 24054
http_request_duration_seconds_bucket{le="0.1"} 33444
http_request_duration_seconds_bucket{le="0.2"} 100392
http_request_duration_seconds_bucket{le="0.5"} 129389
http_request_duration_seconds_bucket{le="1"} 139308
http_request_duration_seconds_bucket{le="+Inf"} 144320
http_request_duration_seconds_sum 53423
http_request_duration_seconds_count 144320
```

OpenMetrics Metrics Data Format
Akraino Profiling Feature Project & Framework

**Akraino Profiling Goals - Use OpenMetrics to unify the end-to-end metrics exposition format for the edge computing,**

1. **Requirement Doc**
   - Analyze which metrics data need to be monitored, such as: node_memory_total, response_latency, transmission_bandwidth.

2. **Exporter**
   - Collect metrics data from the monitored object
   - Expose those metrics in OpenMetrics format

3. **Adapter** *(optional, pluggable and dynamically loaded component)*
   - Do format conversion work
   - Analyze whether the BP monitoring systems need to develop the adapter or not
     - OpenMetrics compliance
     - OpenMetrics incompliance

4. **Integration**

---

**Monitored Objects (App, Container, VM, Hardware, …)**

- **ELIOT:** Edge Lightweight and IoT Blueprint - Prometheus
- **Integrated Edge Cloud** *(IEC)* Blueprint Family - Not sure
- **Network Cloud Blueprint - Prometheus**
- **Time-Critical Edge Compute Blueprint - Not sure**
- **BP Monitoring System Adapter**
- **Exporter** (required, pluggable)
- **Multiple Implementations**
- **Integration** (required, pluggable)

- **Requirement Doc** – Required;
- **Exporter** – Required;
- **Adapter** – Optional;
- **Integration** – Required;
The requirements document is roughly as follows.

**Exporter**
Add a new node exporter/update an existing node exporter to collect and expose metrics data which are compliant with OpenMetrics standard;

**Adapter (Eliot Not required)**
Eliot uses Prometheus as the monitoring system that can directly process the OpenMetrics metric data collected;

**Integration**
Develop scripts based on ansible/helm to integrate Profiling with Eliot.

---

### Monitoring Metric Data

<table>
<thead>
<tr>
<th>Metric Name</th>
<th>Metric Type</th>
<th>Metric Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>eliot_node1_sound</td>
<td>gauge</td>
<td>Sound (noise) level</td>
</tr>
<tr>
<td>eliot_node1_temperature</td>
<td>gauge</td>
<td>Temperature in C and F</td>
</tr>
</tbody>
</table>
Take Ganglia (on the cloud) and gmond (Ganglia collector daemon at the edge) as an example:

- **Pre-analysis**
  - Determine the monitoring metrics;

- **Exporter**
  - Redevelop `gmond_exporter` to serve OpenMetrics metrics for ganglia on the Cloud;

- **Adapter (Ganglia required)**
  - Develop `ganglia_adapter` for ganglia to convert the OpenMetrics metric to ganglia metrics as it can't directly process the OpenMetrics metric collected by `gmond_exporter`;

- **Integration**
  - Develop scripts based on ansible/helm to integrate Profiling which includes cloud part and edge part.

**Monitoring (developer/user tools)**
- Using Ganglia, Nagios, etc., and their own collector daemon to monitor resources

**Profiling**

1. **Edge Node** Add/redevelop exporters to collect and expose the metric data to the monitoring system on the Cloud Manager in the OpenMetrics format.
2. **Cloud Manager** Develop adapter to convert OpenMetrics metric data format to proprietary format that they can process.

---

![Diagram of Profiling Implementation - OpenMetrics Noncompliant]

5/9/19

The Linux Foundation Internal Use Only
Next Steps

- Preliminarily plan to focus on IOT scenario - Eliot
  - Complete Metrics Requirement Analysis
  - Add/Update a exporter at least
  - Develop scripts to integrate with Eliot
  - Expected to be released with Eliot in R2
- Future plans to extend to more Akraino Blueprint Families/BPs
# Akraino Profiling Summary

<table>
<thead>
<tr>
<th>Feature</th>
<th>Description</th>
<th>Companies Participating / Committers</th>
<th>Requested Release / Timeline</th>
<th>Informational</th>
</tr>
</thead>
</table>
| Akraino Profiling| **Akraino Profiling to use OpenMetrics to unify the end-to-end metrics exposition format for the edge computing, i.e., make all the edge nodes/devices/apps/services expose their metrics data in OpenMetrics format.**<br>1. Pre-analysis  
Analyze which metrics data require to be monitored, then aggregate them into the requirement document;<br>2. Exporter Development  
Analyze which exporters need to be add/redevelop/updated according to the metrics requirement document and develop them;<br>3. Adapter Development (**optional, pluggable and dynamically loaded component**)  
Analyze which BP monitoring systems need to develop the adapter and develop them;<br>4. Integration  
To develop scripts based on ansible/helm to integrate Profiling into each BP. | Huawei ARM Dell                     | R2                                | Impacted Blueprint Family – Applies to all BP Families and Blueprints  
See next slide for additional details |
Q & A
Thanks