Device Management and Monitoring Through Redfish

An introduction to Redfish Data Importer, based on Edge core OLT implementation for REDFISH API
Agenda

• Intro to Redfish
• Edge’s Redfish Importer
• Q&A
Redfish

- A set of specifications, delivering a standard protocol to manage
  - Servers
  - Storage
  - Networking
  - Any converged infrastructure

- Management through
  - Logs
  - Events
  - Status (config)
Problems?

- Keeping track of devices
- Early warnings signs
- Preventive actions
Importer Block Diagram

Importer supports any standard Redfish implementation!

Edgecore XGS PON vOLT
Edgecore GPON OLT
Edgecore Switch
Edgecore Device OpenBMC
Edgecore Device
Storage Device

Redfish implementations without BMC (x86 based)
Redfish implementations on BMC
Overall Architecture

* Running on vOLT x86 processor or dedicated BMC HW
Overall Architecture

Device

Redfish Agent

SDN Server (e.g. SEBA POD)

Redfish Importer

Redfish Exporter

Communication Bus (e.g. Kafka)

Prometheus

Grafana

SDN Controller (e.g. NEM)

Redfish Dashbord
Additional Architectural Considerations

- Not restricted to SEBA environment

- Support for multiple SDN Controllers (such as NEM) for device information and configuration.

- Support for publishing data on any communication bus (such as Kafka.)

* For simplicity, the rest of this document will use SEBA, NEMA, and Kafka to indicate major components in the system.
PSME & Importer

- Pooled System Management Engine
  - Intel RSD based open source component
  - Implements REDFISH server
  - Supports Redfish RESFUL API
  - Queries ONLP to collect data from device

- Importer
  - Collects data from PSME and publishes to Kafka bus.
  - Receives events from PSME
Importer will run as a separate container and will be implemented in Go.
**Importer Functional Blocks**

**Global structure server, contains**
- Device map
- Kafka producer handle
- Mutex
- Channel for gRPC API
- Key for device map is IP address

**Grpc API, contains**
- IP address of device
- Type of device
- Name of device
- Data collection frequency
Sets up the environment and creates necessary data structures

- Creates
  - Server structure instance
  - gRPC server
  - HTTP server
  - Data producer (e.g. out to Kafka bus)
- Restores previous config from persistent storage
Importer: Event Subscriber

Manages subscriptions

- Subscribes/unsubscribes to events
- Notified when a device is added
  - Receives IP address of device
  - Creates event structure for event type(s).
- Subscribes to device events by registering an event URL with the Redfish server
Importer: Scheduler

Manages the scheduled data collection activities

- Scans the device list
- Periodically collects data from all devices using RESTFUL APIs
- Posts the data collected, as is, on Kafka bus

* Supports OCP Baseline Hardware Management Profile Version 0.2.1
An HTTP handler listening on HTTP path, specified when an event is registered.

- Retrieves the body of message from the HTTP POST
- Create Kafka record from the response, including the event data as is
- Posts the event data to Kafka bus
<table>
<thead>
<tr>
<th>Events Types</th>
<th>Resource Added</th>
<th>Resource Removed</th>
<th>Alert</th>
</tr>
</thead>
<tbody>
<tr>
<td>PSU Module Plug-In</td>
<td>![Green Check]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>PSU Module Plug-Out</td>
<td></td>
<td>![Green Check]</td>
<td></td>
</tr>
<tr>
<td>FAN Module Plug-In</td>
<td>![Green Check]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>FAN Module Plug-Out</td>
<td></td>
<td>![Green Check]</td>
<td></td>
</tr>
<tr>
<td>FAN Module No Spin</td>
<td></td>
<td></td>
<td>![Green Check]</td>
</tr>
<tr>
<td>CPU/Main Board Thermal Sensor over critical temp</td>
<td></td>
<td></td>
<td>![Green Check]</td>
</tr>
<tr>
<td>CPU/Main Board Thermal Sensor over fatal temp</td>
<td></td>
<td></td>
<td>![Green Check]</td>
</tr>
<tr>
<td>Transceiver Plug-In</td>
<td>![Green Check]</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Transceiver Plug-Out</td>
<td></td>
<td>![Green Check]</td>
<td></td>
</tr>
</tbody>
</table>
OpenOLT Driver \( \rightarrow \) PSME * \( \rightarrow \) ONLP Stack \( \rightarrow \) AGG Switch

* Running on vOLT x86 processor or dedicated BMC HW
Thanks...