



OPEN
Compute Project

Edgecore's Device Management

v1.12

Author: Edgecore

Table of Contents

TABLE OF CONTENTS.....	2
1. LICENSE.....	3
2. SCOPE & OVERVIEW.....	4
3. ARCHITECTURE.....	5
4. API DETAILS.....	7
4.1 Authentication.....	7
4.1.1 Enable/Disable Session Management.....	7
4.1.2 Create a New Session.....	7
4.1.3 Delete an Existing Session.....	8
4.2 Account Service.....	8
4.2.1 View Accounts.....	8
4.2.2 Create a New Account.....	9
4.2.3 Delete an Account.....	9
4.3 Event Service.....	10
4.3.1 Add Subscriptions.....	10
4.3.2 View Subscriptions.....	11
4.3.3 Delete Subscriptions.....	11
4.4 Hardware Information.....	12
4.4.1 Chassis information.....	12
4.4.2 Thermal Sensor Information.....	13
4.4.3 PSU Power Information.....	14
4.4.4 QSFP/XFP Information.....	15
4.4.4.1 Collection of Ports.....	15
4.4.4.2 Port Information.....	15
4.5 Enable/Disable Log Service.....	16
4.5.1 Enable/Disable Log Service.....	16
4.5.2 Get Log Entries of the Target Device.....	17
4.5.3 Clear Log Entries of the Target Device.....	17
4.6 System Management.....	17
4.6.1 System Reset.....	17
4.6.2 Configure Boot Source.....	18
4.6.3 Configure Default OS in GRUB.....	18
4.6.4 Update System BIOS.....	19
4.6.5 Check the BIOS reflash status.....	19
4.6.6 Update System Image.....	20
4.6.7 Check the System Update status.....	20
5. POTENTIAL FUTURE ENHANCEMENTS.....	21
5.1 In-band IP Management.....	21

1. License

Contributions to this proposal are made under the terms and conditions set forth in Open Compute Project Contribution License Agreement (“OCP CLA”) (“Contribution License”) by:

Edgecore Networks

Usage of this Specification is governed by the terms and conditions set forth in Open Compute Project Hardware License – Permissive (“OCPHL Permissive.”)

Note: The following clarifications, which distinguish technology licensed in the Contribution License and/or Specification License from those technologies merely referenced (but not licensed), were accepted by the Incubation Committee of the OCP:

[None].

NOTWITHSTANDING THE FOREGOING LICENSES, THIS SPECIFICATION IS PROVIDED BY OCP "AS IS" AND OCP EXPRESSLY DISCLAIMS ANY WARRANTIES (EXPRESS, IMPLIED, OR OTHERWISE), INCLUDING IMPLIED WARRANTIES OF MERCHANTABILITY, NON-INFRINGEMENT, FITNESS FOR A PARTICULAR PURPOSE, OR TITLE, RELATED TO THE SPECIFICATION. NOTICE IS HEREBY GIVEN, THAT OTHER RIGHTS NOT GRANTED AS SET FORTH ABOVE, INCLUDING WITHOUT LIMITATION, RIGHTS OF THIRD PARTIES WHO DID NOT EXECUTE THE ABOVE LICENSES, MAY BE IMPLICATED BY THE IMPLEMENTATION OF OR COMPLIANCE WITH THIS SPECIFICATION. OCP IS NOT RESPONSIBLE FOR IDENTIFYING RIGHTS FOR WHICH A LICENSE MAY BE REQUIRED IN ORDER TO IMPLEMENT THIS SPECIFICATION. THE ENTIRE RISK AS TO IMPLEMENTING OR OTHERWISE USING THE SPECIFICATION IS ASSUMED BY YOU. IN NO EVENT WILL OCP BE LIABLE TO YOU FOR ANY MONETARY DAMAGES WITH RESPECT TO ANY CLAIMS RELATED TO, OR ARISING OUT OF YOUR USE OF THIS SPECIFICATION, INCLUDING BUT NOT LIMITED TO ANY LIABILITY FOR LOST PROFITS OR ANY CONSEQUENTIAL, INCIDENTAL, INDIRECT, SPECIAL OR PUNITIVE DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND EVEN IF OCP HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

2. Scope & Overview

Scope:

Edgecore's Device Management can provide a method for identification and management of target devices. It can also provide data that can be used to analyze and detect possible future device health or out of state issues. This enables the entire infrastructure devices (including networking, storage, server) to be monitored to provide High Availability.

The work, described in this paper, provides means to manage devices through a well-defined data model. It uses the Edgecore's Device Manager to collect and manage device status from all devices, and sends to an Output Bus, using RESTFUL APIS. The functionality also provides support for monitoring certain device states and sending event notifications.

All APIs and functionality set forth in this document will be available on both CPU and BMC sides, unless otherwise noted. The document also introduces Edgecore BMC daughter-card and motherboard reference design providing support to the new features, which is discussed in detail separately. At the current time, the proposed changes are only planned for vOLT devices.

Overview:

Scalability is one of the paramount problems in today's modern infrastructures. The growing needs of the end customers forces the infra operators to employ horizontal, scale-out solutions, often requiring them to deploy additional devices to their systems. This ever-growing infrastructure requires non-traditional thinking in which a more automated device management functionality must be employed.

Designed to manage large systems through a modern platform, DMTF's Redfish is an open industry standard specification enabling customers to monitor their infrastructure through a well-defined interface.

In its core, each device is monitored and managed through an agent, PSME running on OLT (or OpenBMC running on BMC daughter card, if available), to

- 1- retrieve specific parameters to be sent through its northbound interfaces.
- 2- receive notifications on certain device state changes.¹
- 3- manage device state and reboot options.

¹ Device Manager will not support notifications setup through BMC.

Edgecore's Device Manager uses the ONLP driver (or BMC when available) to manage and retrieve information about device, and makes data available at a configurable Output Bus. It uses a RESTFUL API to communicate with PSME at certain intervals to retrieve the data .

Other devices on the system, such as Aggregation switches and ONUs, can also use Redfish to report to Edgecore's Device Manager, the same way vOLT does.

To provide uniform functionality across an infrastructure employing a non-uniform device list provided by multiple vendors, Redfish (and OCP) exchanges information using a hardware management specification, employing a JSON schema. This proposal uses standard OCP baseline profile. If needed, we may enhance the baseline profile with attributes specific for TelCo in the future.

This will also work on BMC controller running OpenBMC standard interface.

The following is an example of OCP defined schema

```
Redfish >> v1 >> Chassis

  "@odata.type": "#ChassisCollection.ChassisCollection",
  "Name": "Chassis Collection",
  "Members@odata.count": 2,
  "Members": [
    {
      "@odata.id": "/redfish/v1/Chassis/Switch1"
    },
    {
      "@odata.id": "/redfish/v1/Chassis/Switch2"
    }
  ],
  "odata.context":
  "redfish/v1/$metadata#ChassisCollection.ChassisCollection",
  "odata.id": "redfish/v1/Chassis"
```

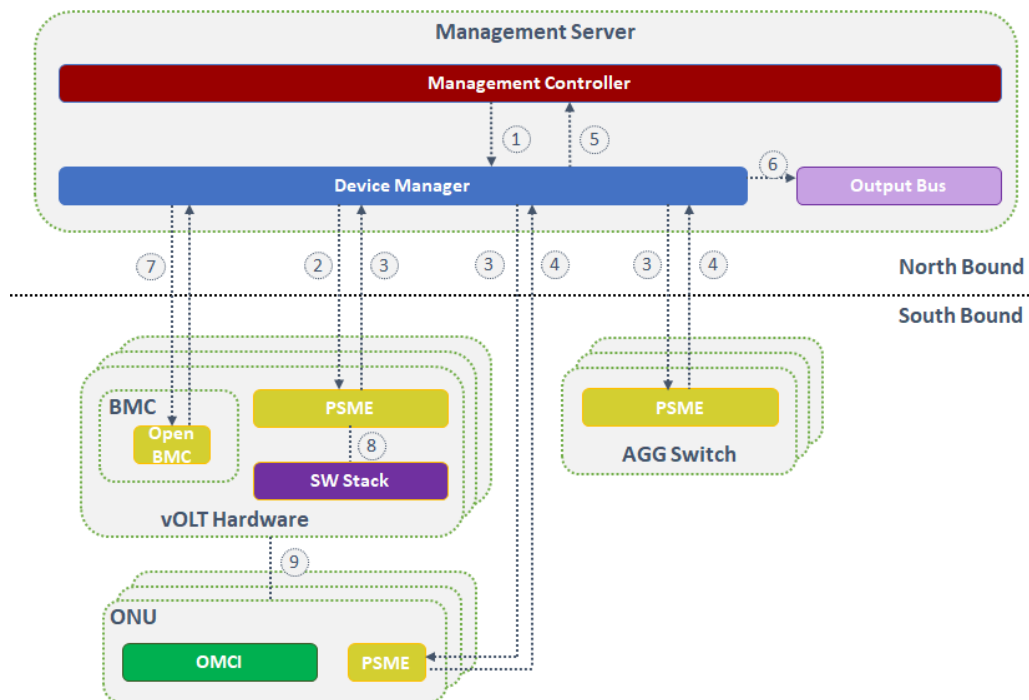
3. Architecture

Edgecore's Device Manager will use a standard interface and dataflow to manage and monitor for deployment in the field. Each device in the system will communicate North Bound with Device Manager to transmit device data and notifications, using PSME (or OpenBMC in BMC), adhering Redfish standards. Device Manager will collect device data and notifications from each device, and make the data available on a predetermined Output Bus for consumers. The consumers of this data could then use it for various purposes, including

- inventory management
- status monitoring
- device updates

- reboots

Following diagram illustrates the overall architecture and the interaction between the components at a high level.



- ① : The first step is the notification from Management Controller to add the IP address of the device to be monitored to the device list (Device Manager can also use Ref ⑤ to refresh all its data.) Device Manager can also subscribe to events from the device by sending a HTTP POST to the URL of the Resource Collection for "Subscriptions" in the Event Service. At this point, Device Manager can query status parameters with HTTP GET method to PSME (Pooled System Management Engine) periodically. The default polling interval is set to five seconds.
- ② : Device Manager is an alert Receiver. When the Redfish service interface is responding to Device Manager, it uses RESTful API. If PSME detects a change in one of the monitored hardware states, it will send an alert for the subscribed event to the Device Manager. A particular issue is reported only once until the state of the hardware changes.
- ③ : Device Manager retrieves data using HTTP GET method from devices periodically.
- ④ : Switch responds using HTTP RESPONSE to Device Manager.
- ⑤ : The Management Controller will provide the active device list to Device Manager which needs to be monitored. When the device becomes available, the Device Manager uses this IP address to connect PSME (or OpenBMC) using HTTP protocol.

- ⑥ : Device Manager publishes the device alarm/status into Output Bus.
- ⑦ : If exists, BMC can also be used to retrieve device status and manage the target device. However, BMC may not provide the full functionality offered by PSME, including periodic data collection or alarms.
- ⑧ : PSME inspects hardware status at regular intervals to detect hardware failure/status through Network OS's SW stack.

4. API Details

4.1 Authentication

Redfish Service uses session management to implement authentication.

4.1.1 Enable/Disable Session Management

Enable secure access to resources.

POST: /redfish/v1/SessionService

```
{
  "ServiceEnabled": "true",
  "SessionTimeout": 30
}
```

Property	Requirement	Value
ServiceEnabled	Mandatory	true, false
SessionTimeout	Mandatory	≥1 [in seconds]

Following is a mockup of the Session Service State

```
{
  "ServiceEnabled": "true",
  "SessionTimeout": 30
}
```

4.1.2 Create a New Session

Create a new session (Login) by using the login credential of an existing account (default username "admin" and pPassword "redfish"), enable Session Management, first. Include the token is generated in the https request header to perform any operations on the secured resources.

POST: /redfish/v1/SessionService/Sessions

```
{
  "UserName ":"admin",
  "Password": "redfish"
}
```

Property	Requirement	Value
UserName	Mandatory	
Password	Mandatory	

Following is a mockup of the session token generated

```
{
  "X-Auth-Token": "5HHYVsfNOb9Dm4X0PVmQcK9aJnWUMQXU"
}
```

4.1.3 Delete an Existing Session

Delete an existing session (Logout) by using default UserName "admin" and Password "redfish". Following example assumes a session with ID "1" has already been created.

DELETE: /redfish/v1/SessionService/Sessions/<session_id>

4.2 Account Service

Account Service contains properties common to all user accounts, such as password requirements, and control features such as account lockout. It also contains links to the collections of Manager Accounts and Roles.

4.2.1 View Accounts

List default accounts.

GET: /redfish/v1/AccountService/Accounts

Following is a mockup of the accounts list

```
{
  "@odata.context": "/redfish/v1/$metadata#AccountService.AccountService",
  "@odata.id": "/redfish/v1/AccountService/Accounts",
  "@odata.type": "#ManagerAccountCollection.ManagerAccountCollection",
  "Name": "Accounts Collection",
  "Members@odata.count": 1,
  "Members": [
    {
      "@odata.id": "/redfish/v1/AccountService/Accounts/admin"
    }
  ]
}
```

```
}
}]
}
```

4.2.2 Create a New Account

Create a new account.

POST: /redfish/v1/AccountService/Accounts

```
{
  "Name": "User Account",
  "UserName": "new_user",
  "Password": "user password",
  "RoleId": "Administrator",
  "Enabled": true,
  "Locked": false
}
```

Property	Requirement	Value
UserName	Mandatory	
Password	Mandatory	
RoleId	Mandatory	Administrator, Operator, ReadOnlyUser
Enabled	Mandatory	true, false
Locked	Mandatory	true, false

Following is a mockup of the session token generated

```
{
  "@odata.context":
    "/redfish/v1/$metadata#AccountService/Accounts/Members/$entity",
  "@odata.id": "/redfish/v1/AccountService/Accounts/1",
  "@odata.type": "#ManagerAccount.v1_0_2.ManagerAccount",
  "Id": "1",
  "Name": "User Account",
  "Password": "user password",
  "UserName": "new user",
  "Links": {
    "Role": {
      "@odata.id":
        "/redfish/v1/AccountService/Roles/Administrator"
    }
  }
}
```

4.2.3 Delete an Account

Delete an existing account.

DELETE: /redfish/v1/AccountService/Accounts/<user_name>

4.3 Event Service

The event service is a new alert mechanism for Redfish. This alert will be sent out through HTTP or HTTPS to a web service that is subscribed to the service. All subscriptions added will remain on the device across reboots. 3 event types are supported as follows:

Event Types	Resource Added	Resource Removed	Alert
PSU Module Being Plugged In	x		
PSU Module Being Pulled Out		x	
FAN Module Being Plugged In	x		
FAN Module Being Pulled Out		x	
FAN Module Stopped Spinning			x
CPU/Main Board Thermal Sensor Exceeding Critical Temperature			x
CPU/Main Board Thermal Sensor Exceeding Fatal Temperature			x
QSFP or XFP Transceiver Being Plugged In	x		
QSFP or XFP Transceiver Being Pulled Out		x	

4.3.1 Add Subscriptions

Add subscriptions and inform Redfish intended destination for the events. Subscriptions to different events are added separately.

POST: /redfish/v1/EventService/Subscriptions

```
{
  "Name": "Event Subscription ResourceAdded TEST",
  "Destination": "https://10.10.10.10:8889",
  "EventTypes": [ "ResourceAdded" ],
  "Context": "THIS IS A TEST EVENT",
  "Protocol": "Redfish"
}
```

Property	Requirement	Value
Name	Mandatory	
Destination	Mandatory	
EventTypes	Mandatory	ResourceAdded, ResourceRemoved, Alert
Protocol	Mandatory	Redfish

Following is a mockup of an event triggered

```
{
  "@odata.context": "/rest/v1/$metadata#EventService/Members/Events/1",
  "@odata.id": "/rest/v1/EventService/Events/1",
  "@odata.type": "#EventService.1.0.0.Event",
  "Id": "1",
  "Name": "Event Subscription ResourceAdded TEST",
  "Description": "Events",
  "Events": [ {
    "EventType": "ResourceAdded",
    "EventId": "12f03ec2-714f-11e9-ba1f-ef16238bd344",
    "EventTimestamp": "2019-05-08T05:06:44+00:00",
    "Severity": "OK",
    "Message": "Port 1 Plug In.",
    "MessageId": "Base.1.0.0.Success",
    "MessageArgs": [],
    "OriginOfCondition": "",
    "Context": "THIS IS A TEST EVENT"
  }
]
```

4.3.2 View Subscriptions

List current subscriptions.

POST: URL: /redfish/v1/EventService/Subscriptions

Following is a mockup of list of events registered

```
{
  "@odata.context": "/redfish/v1/$metadata#EventService.EventService",
  "@odata.id": "/redfish/v1/EventService/Subscriptions",
  "@odata.type": "#EventDestinationCollection.EventDestinationCollection",
  "Name": "Event Subscriptions Collection",
  "Description": "Collection of Event Subscriptions",
  "Members@odata.count": 3,
  "Members": [
    { "@odata.id": "/redfish/v1/EventService/Subscriptions/1" },
    { "@odata.id": "/redfish/v1/EventService/Subscriptions/2" },
    { "@odata.id": "/redfish/v1/EventService/Subscriptions/3" }
  ]
}
```

4.3.3 Delete Subscriptions

Delete a subscription from Redfish service.

DELETE /redfish/v1/EventService/Subscriptions/<subscription_id>

4.4 Hardware Information

Hardware inventory and health state information, such as information about temperature sensors, fans, PSU, HDD, NIC, can be obtained using Redfish API's.

4.4.1 Chassis information

Retrieve chassis information.

GET: /redfish/v1/Chassis/<chassis_id>

Following is a mockup of chassis info retrieved for Edgecore's XGSPON device x86-64-accton-asxvolt16-r0

```
{
  "@odata.context": "/redfish/v1/$metadata#Chassis.Chassis",
  "@odata.id": "/redfish/v1/Chassis/1",
  "@odata.type": "#Chassis.v1_3_0.Chassis",
  "Id": "1",
  "ChassisType": "Drawer",
  "Name": "Chassis",
  "Description": "Chassis of switch devices.",
  "PowerState": "On",
  "Manufacturer": "Accton",
  "Model": "FN1EC0816400Z",
  "SKU": null,
  "SerialNumber": "EC1840000200",
  "PartNumber": "FN1EC0816400Z",
  "AssetTag": "N/A",
  "IndicatorLED": null,
  "Status": {
    "State": "Enabled",
    "Health": "OK",
    "HealthRollup": "OK"
  },
  "Oem": {
    "Intel_RackScale": {
      "@odata.type": "#Intel.Oem.Chassis",
      "Location": {
        "Id": null,
        "ParentId": null
      },
    },
  },
  "Links": {
    "@odata.type": "#Chassis.v1_2_0.Links",
    "Contains": [],
    "ComputerSystems": [ {
      "@odata.id": "/redfish/v1/Systems/1",
      "Actions": {
        "#Chassis.Reset": {
          "target":
            "/redfish/v1/Chassis/1/Actions/Chassis.Reset",
          "ResetType@Redfish.AllowableValues":
            [ "ForceOff", "iK", "Thermal" ],
          "@odata.id":
            "/redfish/v1/Chassis/1/Thermal",
          "Power": {
            "@odata.id":
              "/redfish/v1/Chassis/1/Power"
          }
        }
      }
    }
  ]
}
```

```

}

}

```

4.4.2 Thermal Sensor Information

Fan speeds can be read from "Reading" property in payload of Fans object. PSU power thermal sensor can be read from "ReadingCelsius" property in payload of Temperatures object.

GET: /redfish/v1/Chassis/<chassis_id>/Thermal

Following is a mockup of thermal snsr info retrieved for Edgecore's XGSPON device x86-64-accton-asxvolt16-r0

```

{
  "Description" : "Collection of Thermal sensors",
  "Temperatures" : [ {
    "PhysicalContext" : "CPU",
    "UpperThresholdCritical" : 92,
    "ReadingCelsius" : 34,
    "RelatedItem" : [ { "@odata.id" : "/redfish/v1/Chassis/1" } ],
    "LowerThresholdFatal" : null,
    "LowerThresholdNonCritical" : null,
    "Status" : {
      "Health" : "OK",
      "HealthRollup" : null,
      "State" : "Enabled"
    },
    "@odata.id" : "/redfish/v1/Chassis/1/Thermal",
    "UpperThresholdNonCritical" : 82,
    "MemberId" : "1",
    "Name" : "System CPU Thermal Sensor Temperature",
    "LowerThresholdCritical" : null,
    "UpperThresholdFatal" : 104,
    "SensorNumber" : 1
  },
  {
    "UpperThresholdFatal" : 60,
    "SensorNumber" : 2,
    "Name" : "Chassis Thermal Sensor Temperature",
    "LowerThresholdCritical" : null,
    "MemberId" : "2",
    "UpperThresholdNonCritical" : 45,
    "@odata.id" : "/redfish/v1/Chassis/1/Thermal", "Status" : {
      "HealthRollup" : null,
      "State" : "Enabled",
      "Health" : "OK"
    },
    "LowerThresholdNonCritical" : null,
  }
]
}

```

```

        "ReadingCelsius" : 27.5,
        "RelatedItem" : [ { "@odata.id" : "/redfish/v1/Chassis/1" } ],
        "LowerThresholdFatal" : null,
        "UpperThresholdCritical" : 55,
        "PhysicalContext" : "SystemBoard"
    },
    ...
}

```

4.4.3 PSU Power Information

Power consumption information can be read from the "PowerConsumedWatts" property in the payload of the PowerControl object. PSU presence information can be read from the "State" property in the payload of "Status" object.

GET: /redfish/v1/Chassis/<chassis_id>/Power

```

{
    {
        "Description" : "Collection of Power",
        "@odata.type" : "#Power.v1_1_0.Power",
        "@odata.id" : "/redfish/v1/Chassis/1/Power",
        "Voltages" : [
            {
                "RelatedItem" : [ { "@odata.id" : "/redfish/v1/Chassis/1" } ],
                "@odata.id" : "/redfish/v1/Chassis/1/Power",
                "LowerThresholdFatal" : null,
                "Name" : "Voltage",
                "LowerThresholdCritical" : null,
                "Status" : { "State" : "Enabled", "Health" : "OK" },
                "UpperThresholdFatal" : null,
                "PhysicalContext" : "VoltageRegulator",
                "MemberId" : "1",
                "LowerThresholdNonCritical" : null,
                "ReadingVolts" : 12.046,
                "UpperThresholdNonCritical" : null,
                "SensorNumber" : 1,
                "UpperThresholdCritical" : null
            },
            {
                "UpperThresholdNonCritical" : null,
                "SensorNumber" : 2,
                "UpperThresholdCritical" : null,
                "PhysicalContext" : "VoltageRegulator",
                "MemberId" : "2",
                "UpperThresholdFatal" : null,
                "LowerThresholdNonCritical" : null,
                "ReadingVolts" : 12.015,
                "Status" : { "State" : "Enabled", "Health" : "OK" },
                "RelatedItem" : [ { "@odata.id" : "/redfish/v1/Chassis/1" } ],
                "@odata.id" : "/redfish/v1/Chassis/1/Power",
                "LowerThresholdFatal" : null,
                "LowerThresholdCritical" : null,
                "Name" : "Voltage"
            }
        ]
    }
}

```

```

    "Name" : "Power Collection",
    "@odata.context" : "/redfish/v1/$metadata#Power.Power",
    ...
  }

```

4.4.4 QSFP/XFP Information

4.4.4.1 Collection of Ports

Retrieve the list of ports

GET: /redfish/v1/EthernetSwitches/<ethernetswitch_id>/Ports

Following is a mockup of list of ports

```

{
  "@odata.context":
    "/redfish/v1/$metadata#EthernetSwitchPortCollection.EthernetSwitchPortCollection",
  "@odata.id": "/redfish/v1/EthernetSwitches/1/Ports",
  "@odata.type":
    "#EthernetSwitchPortCollection.EthernetSwitchPortCollection",
  "Name": "Ethernet Switch Port Collection",
  "Description": "Collection of Ethernet Switch Ports",
  "Members@odata.count": 4,
  "Members":[
    {"@odata.id": "/redfish/v1/EthernetSwitches/1/Ports/1"},
    {"@odata.id": "/redfish/v1/EthernetSwitches/1/Ports/2"},
    {"@odata.id": "/redfish/v1/EthernetSwitches/1/Ports/3"},
    {"@odata.id": "/redfish/v1/EthernetSwitches/1/Ports/4"}
  ]
}

```

4.4.4.2 Port Information

Retrieve detailed information about a port, if present, and if supported transceivers are detected, their detailed information.

GET: /redfish/v1/EthernetSwitches/<ethernetswitch_id>/Ports/<port_id>

Following is a mockup of port details

```

{
  "@odata.context":
    "/redfish/v1/$metadata#EthernetSwitchPort.EthernetSwitchPort",
  "@odata.id": "/redfish/v1/EthernetSwitches/1/Ports/1",
  "@odata.type": "#EthernetSwitchPort.v1_0_0.EthernetSwitchPort",

```

```

    "Id": "1",
    "Name": "Port1",
    "Description": "Ethernet Switch Port description",
    "PortId": "Port ID",
    "Status":{"State": "Enabled", "Health": "OK", "HealthRollup": "OK"},
    "LinkType": "Ethernet",
    "OperationalState": null,

    "AdministrativeState": null,
    "LinkSpeedMbps": null,
    "NeighborInfo":{"SwitchId": null, "PortId": null, "CableId": null},
    "NeighborMAC": null,
    "FrameSize": null,
    "Autosense": null,
    "FullDuplex": null,
    "MACAddress": null,
    "PortClass": null,
    "PortMode": null,
    "PortType": null,
    "Oem": {},
    "IPv4Addresses": [],
    "IPv6Addresses": [],
    "VLANs":{"@odata.id":"/redfish/v1/EthernetSwitches/1/Ports/1/VLANs"},
    "StaticMACs":{"@odata.id":"/redfish/v1/EthernetSwitches/1/Ports/1/StaticMACs"},
    "Links":{
    ...
  }

```

Transceiver	Vendor	Part No
QSFP	Precision	QSFP28AOC03
XFP (XGSPON)	Source Photonics	XPPXG2N1CDFA
XFP (XGSPON)	Hisense	LTH7226-PC+
SFP (GPON)	Hisense	LTE3680M-BH+

4.5 Enable/Disable Log Service

This resource represents the log service for the resource or service to which it is associated. It records all ONL peripheral Add/Remove/Alert events, such as, fan plugging-in and out, thermal sensor exceeding fatal critical threshold temperature, ..., etc.

4.5.1 Enable/Disable Log Service

PATCH: /redfish/v1/Managers/<manager_id>/LogServices/<log_id>

```

{
  "ServiceEnabled": "true",
}

```

Property	Requirement	Value
ServiceEnabled	Mandatory	true, false

4.5.2 Get Log Entries of the Target Device

GET /redfish/v1/Managers/<manager_id>/LogServices/<log_id>/Entries

Following is a mockup of logs retrieved from the device

```
{
  "@odata.context":
    "/redfish/v1/$metadata#LogEntryCollection.LogEntryCollection",
  "@odata.id": "/redfish/v1/Managers/1/LogServices/1/Entries",
  "@odata.type": "#LogEntryCollection.LogEntryCollection",
  "Name": "Log Service Collection",
  "Description": "Collection of Logs for this System",
  "Members@odata.count": 3,
  "Members": [
    {"@odata.id": "/redfish/v1/Managers/1/LogServices/1/Entries/0", "Id": "0"},
    {"@odata.id": "/redfish/v1/Managers/1/LogServices/1/Entries/1", "Id": "1"},
    {"@odata.id": "/redfish/v1/Managers/1/LogServices/1/Entries/2", "Id": "2"}
  ]
}
```

4.5.3 Clear Log Entries of the Target Device

POST:/redfish/v1/Managers/<manager_id>/LogServices/<log_id>/Actions/LogService.Reset

4.6 System Management

4.6.1 System Reset

Reset the system.

POST: URL: /redfish/v1/Systems/<system_id>²/Actions/ComputerSystem.Reset

```
{
  "ResetType": "On",
}
```

Property	Requirement	Value
ResetType	Mandatory	On, ForceOff, GracefulRestart

² system_id is a placeholder for the system resource identifier.

Following is a mockup of the Session Service State

```
{
  "ResetType": "On",
}
```

4.6.2 Configure Boot Source

Change the boot source for the system

PATCH: /redfish/v1/Systems/<system_id>

```
{
  "BootSourceOverrideEnabled": "Once",
  "BootSourceOverrideTarget": "Pxe",
}
```

Property	Requirement	Value
BootSourceOverrideEnabled	Mandatory	Disabled, Once, Continuous
BootSourceOverrideTarget	Mandatory	Pxe, Hdd, BiosSetup

Following is a mockup of the reconfigured boot source

```
{
  "BootSourceOverrideEnabled": "Once",
  "BootSourceOverrideTarget": "Pxe"
}
```

4.6.3. Configure Default OS in GRUB

Set the default OS in grub. This resource is only available when the CPU is running. Please note that if multiple OS options are intended, the kernel and related files for each OS should reside in separate directories.

POST: /redfish/v1/Systems/<system_id>/Actions/Oem/GrubConfig.SetGrubDefault

```
{
  "GrubDefault": "/boot"
}
```

Property	Requirement	Value
GrubDefault	Mandatory	

Following is a mockup of the reconfigured grub source

```
{
  "GrubDefault": "/boot"
}
```

4.6.4. Update System BIOS

Starts reflashing system BIOS from the specified source.

POST: /redfish/v1/Systems/<system_id>/UpdateService/Actions/Oem/Bios.Update

```
{
  "ImagePath": "https:// 10.10.10.10/BIOS/bios.img",
  "Protocol": "HTTPS"
}
```

Property	Requirement	Value
ImagePath	Mandatory	
Protocol	Mandatory	TFTP, HTTPS

Following is a mockup of the response from BOIS being reflashed

```
{
  "ImagePath": "https:// 10.10.10.10/BIOS/bios.img",
  "Protocol": "HTTPS"
}
```

4.6.5. Check the BIOS reflash status

Retrieves the status of a reflash action started previously

GET: /redfish/v1/Systems/<system_id>/UpdateService/Oem/BIOS

Following is a mockup of the response from BOIS reflash status

```
{
  "@odata.context": "/redfish/v1/$metadata#Oem.Oem",
  "@odata.id": "/redfish/v1/UpdateService/Oem/BIOS ",
  "@odata.type": "#Oem.Oem",
  "Description": "BIOS reflash status",
  "Status": "Downloading",
}
```

Property	Requirement	Value
Status	Mandatory	None, Downloading, InProgress, LastCompletedAt, Error

4.6.6.Update System Image

Retrieves a compressed file from the specified source, and decompresses it to a target, preserving all file paths

POST:/redfish/v1/Systems/<system_id>/UpdateService/Actions/Oem/System.Update

```
{
  "SourceImagePath": "https:// 10.10.10.10/system.tar",
  "TargetImagePath": "/boot"
  "Protocol": "HTTPS"
}
```

Property	Requirement	Value
SourceImagePath	Mandatory	
TargetImagePath	Mandatory	
Protocol	Mandatory	TFTP, HTTPS

Following is a mockup of the response from BOIS being reflashed

```
{
  "ImagePath": "https:// 10.10.10.10/BIOS/system.tar",
  "TargetImagePath": "/boot"
  "Protocol": "HTTPS"
}
```

4.6.7.Check the System Update status

Retrieves the status of a system update action started previously

GET: /redfish/v1/Systems/<system_id>/UpdateService/Oem/System

Following is a mockup of the response from system update status

```
{
  "@odata.context": "/redfish/v1/$metadata#Oem.Oem",
  "@odata.id": "/redfish/v1/UpdateService/Oem/System ",
  "@odata.type": "#Oem.Oem",
  "Description": "System update status",
  "Status": "Downloading"
}
```

Property	Requirement	Value
Status	Mandatory	None, Downloading, InProgress, LastCompletedAt, Error

5. Potential Future Enhancements

5.1 In-band IP Management

Add a tiny switch between BMC and the switch chip connected using the ethernet interface on the motherboard to gather system information in the event CPU is not functioning.

