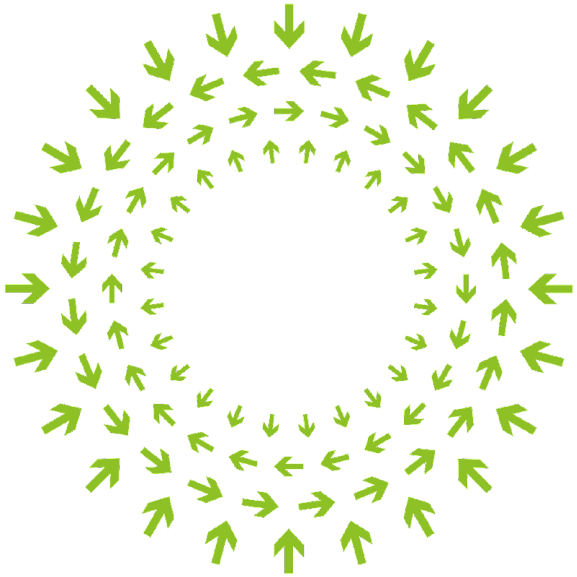


OPEN

Compute Project

Satellite Management Controller
(Auxiliary Management Controller)

Revision 1.0, Version 1.0

Base Specification

Author: Chad Yoshikawa
Author: Ed Tanous
Author: Jeff Hilland
Author: Gregg Shick
Author: Patrick Williams
Author: Changho Choi
Author: Kavin Lee
Author: BJ Kim

Author: Mike Allison

Table of Contents

Glossary of Terms

This section provides definitions for terms used in this document.

Server. Machine hardware that contains a Satellite plug-in. Satellite containers are typically Servers but are not required to be so. So we use the more general term Host for a Satellite container.

Host. A generalization of a Satellite container that includes Servers (for PCIe Plug-In Satellites) and motherboards (for SoC Satellites). A Host is managed by a logical Host Management Controller (HMC).

Satellite. A dependent group of hardware that is managed by a logical Satellite Management Controller (SMC). Satellites typically are smaller than their Host, are terminal points in the management graph, and contain a single power & thermal domain.

SMC. Satellite Management Controller provides a management API to Satellite hardware. SMCs may be backed by one or more discrete hardware components. SMC is typically a low-cost ARM microcontroller running a RTOS with no external DRAM, although this is not prescriptive and any CPU architecture and configuration is acceptable.

Terminal Hardware. Hardware that is an endpoint in the management graph. In other words, terminal hardware does not itself manage other hardware. Satellites are terminal hardware.

1. License

1.1. OPTION B: Open Web Foundation (OWF) CLA

Contributions to this Specification are made under the terms and conditions set forth in Modified OWF-CLA-1.0.2 (As of June 1, 2023) ("Contribution License") by:

Google

chadyoshikawa@google.com
edtanous@google.com

HPE

jeff.hilland@hpe.com
gregg.shick@hpe.com

Meta

patrickw3@fb.com

Samsung

changho.c@samsung.com
kavin.lee@samsung.com
bj20.kim@samsung.com
mike.allison@samsung.com

Usage of this Specification is governed by the terms and conditions set forth in **Modified OWFa1.0.2 Final Specification Agreement (FSA) (As of June 1, 2023) ("Specification License").**

You can review the applicable Specification License(s) referenced above by the contributors to this Specification on the OCP website at http://www.opencompute.org/participate/legal-documents/. For actual executed copies of either agreement, please contact OCP directly.

Notes:

1) The above license does not apply to the Appendix or Appendices. The information in the Appendix or Appendices is for reference only and non-normative in nature.

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.

1.2 Acknowledgements

The Contributors of this Specification would like to acknowledge the following companies for their feedback:

Hewlett Packard Enterprise
Google
Samsung
Dell
Meta

1.3 References

- DMTF DSP0218, Platform Level Data Model (PLDM) for Redfish Device Enablement, https://www.dmtf.org/sites/default/files/standards/documents/DSP0218_1.1.2.pdf
- DMTF DSP0233, Management Component Transport Protocol (MCTP) I3C Transport Binding Specification, https://www.dmtf.org/documents/pmci/management-component-transport-protocol-mctp-i3c-transport-binding-specification-100
- DMTF DSP0236, Management Component Transport Protocol (MCTP) Base Specification, https://www.dmtf.org/documents/pmci/management-component-transport-protocol-mctp-base-specification-131
- DMTF DSP0237, Management Component Transport Protocol (MCTP) SMBus/I2C Transport Binding Specification,

https://www.dmtf.org/documents/pmci/management-component-transport-protocol-mctp-smbusi2c-transport-binding-specification

- DMTF DSP0238, Management Component Transport Protocol (MCTP) PCIe VDM Transport Binding Specification, https://www.dmtf.org/documents/pmci/management-component-transport-protocol-mctp-pcie-vdm-transport-binding-specification
- DMTF DSP0240, Platform Level Data Model (PLDM) Base Specification, https://www.dmtf.org/sites/default/files/standards/documents/DSP0240_1.1.0.pdf
- DMTF DSP0241, Platform Level Data Model (PLDM) over MCTP Binding Specification, http://www.dmtf.org/standards/published_documents/DSP0241_1.0.0.pdf
- DMTF DSP0248, Platform Level Data Model (PLDM) for Platform Monitoring and Control Specification, https://www.dmtf.org/sites/default/files/standards/documents/DSP0248_1.2.0.pdf
- DMTF DSP0267, Platform Level Data Model (PLDM) for Firmware Update Specification, https://www.dmtf.org/sites/default/files/standards/documents/DSP0267_1.1.0.pdf
- DMTF DSP0274, Security Protocol and Data Model (SPDM) Specification, https://www.dmtf.org/sites/default/files/standards/documents/DSP0274_1.3.0.pdf

2. Compliance with OCP Tenets

2.1. Openness

The SMC V1.0 specification proposes an ecosystem-enabling set of requirements for peripherals to enable management compatibility between open systems. This allows interoperability between various device classes and host systems.

2.2. Efficiency

OEMs invest time to create specifications for industry Independent Hardware Vendors (IHVs) which must be implemented in order to support proper management by the host. IHVs invest time working with multiple OEMs to implement those requirements. The goal of the SMC specification is to standardize those various work streams into a single public OCP specification where both OEM and IHV can more effectively promulgate these requirements.

Additionally, multi-vendor customer environments will benefit from the efficiencies achieved through increased device interoperability and the utilization of a common code base for system management.

2.3 Impact

The SMC represents a single set of OCP device manageability requirements allowing for IHV ease of development, time to market, and effective use of engineering resources. Device management ASICs could be developed allowing multiple IHVs to leverage a standardized SMC component providing consistent management across device classes.

2.4. Scale

Large scale deployments benefit from the standardization of management capability across multiple device classes, server and device vendors which this specification provides.

Redfish, RDE and PLDM DMTF standards for management are utilized allowing for a common set of APIs and management tools regardless of hardware or software environment or size of server deployment.

2.5. Sustainability

Between customers' sustainability initiatives and demands to control energy consumption and costs, the ability to report, analyze and actuate server power usage data has become a key initiative.

The creation of a truly interoperable telemetry environment will allow businesses to datacenters, no matter the size, to more effectively meet sustainability targets. SMC thermal and power management capabilities can be utilized to enable this goal of minimizing power requirements and overall energy usage.

3. Change Log

Date	Version #	Author	Description
7/5/2022	0.1	Chad Yoshikawa	Filled in Title, Authors, Contributors and sections 1-3
10/1/2022	0.2	Ed Tanous	Major reorganization. Rewrites to many sections
4/27/2023	0.3	Ed Tanous	Removal of old text, formatting cleanups
9/15/2023	1.0	Chad Yoshikawa	Used recent

			Template, document clean-up and resolved outstanding comments.

4. Overview

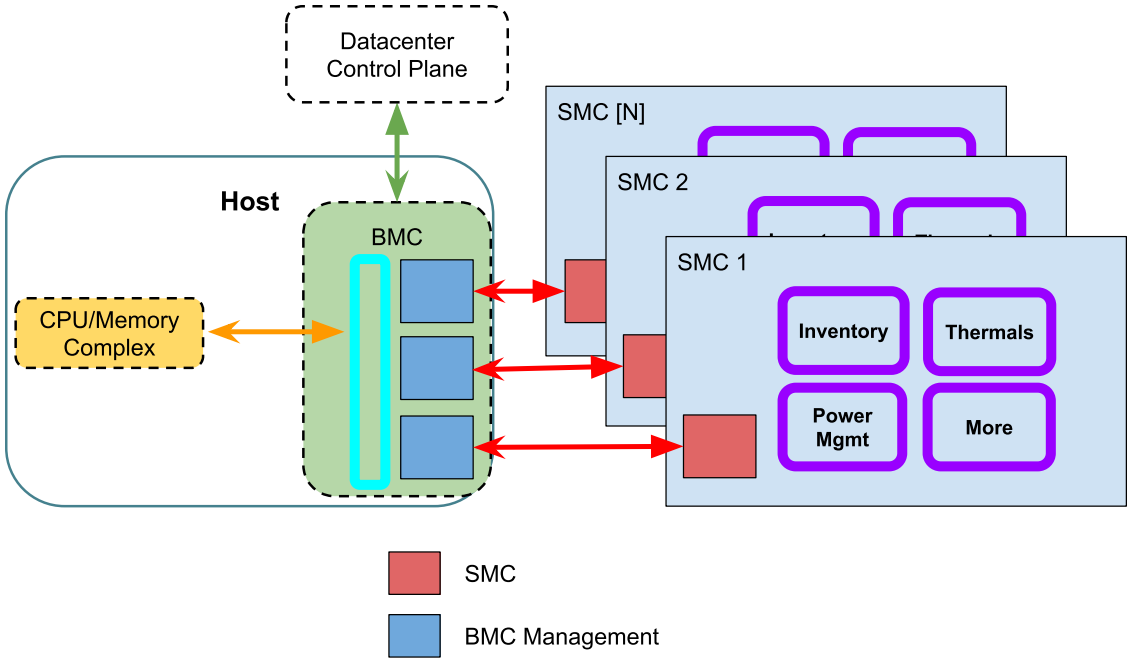
The Satellite Management Controller specification defines a validatable management interface between a satellite container (“server” or more generally a “host”) and simple hardware plug-ins (“satellites”). For example, for a server and PCIe plug-in card, the *host* server manages the *satellite* plug-in.

Satellites are simple: they do not manage other devices and typically contain a single thermal, power and security domain. Note that the satellite and host may not be discrete hardware; host and satellite may be integrated onto the same board in the case of a tray and its SoC.

Satellite conformance can be validated using software tools, which enables independent hardware development and bring-up. To provide validation, the SMC specification defines a compatibility test suite (CTS) to ensure conformance to specified functional requirements.

The management interface additionally specifies SLOs for operations such as firmware update and power management operations. These SLOs may impose constraints on the underlying hardware. For example, timely firmware update may require i3c (vs. i2c) or higher-bandwidth management links.

4.1 Architectural Example



SMC includes all API definitions required for managing a peripheral device from an out of band management controller (BMC) in the most common configuration. While other configurations may exist that this specification fulfills, the above diagram is considered the baseline. This specification may reference elements of the baseline configuration as examples. Other configurations may exist.

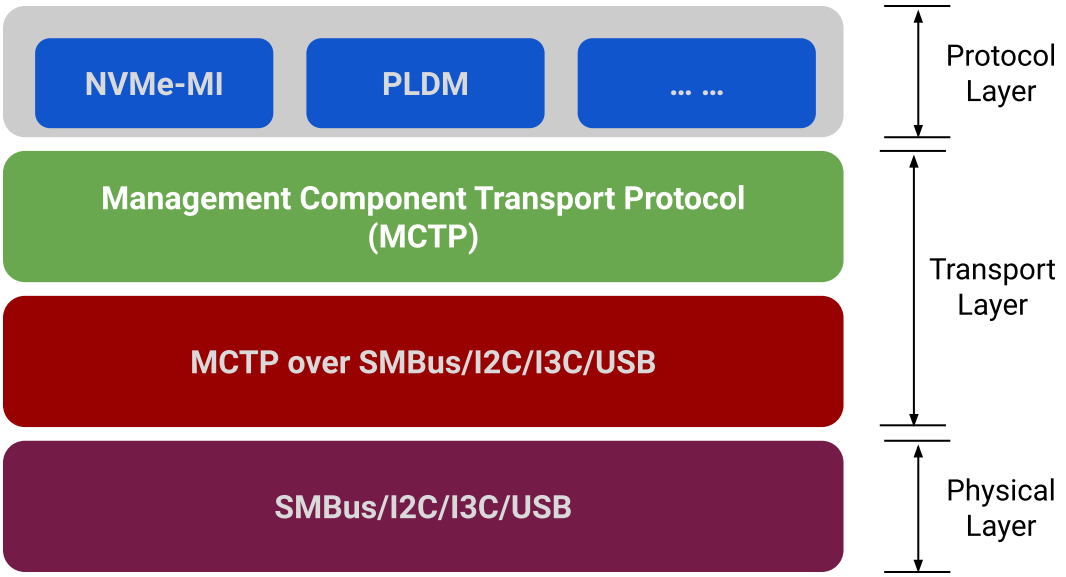
The SMC specification explicitly does not define any physical connectors, physical form factors, or specifications for non SMC components, although other specifications (such as the Modular Hardware Management DC-SCM) may define connector specifications that *may* be used in deployments. For some examples, this specification may assume a PCIe 16X connector, and a single socket server with BMC.

5. Protocols

SMC devices *shall* implement DSP0236 (Management Component Transport Protocol (MCTP) Base Specification).

5.1 Physical bindings supported

Devices meeting this specification shall implement DSP0233 ((MCTP) I3C Transport Binding Specification) OR DSP0237 (MCTP) SMBus/I2C Transport Binding Specification OR DSP0238 (MCTP) PCIe VDM Transport Binding Specification. Other physical bindings, such as USB, that have MCTP bindings are allowed. Devices shall implement these protocols without requiring an on-board i2c mux in the protocol path.



5.2 Protocol layers supported

5.2.1 MCTP

- Compliance with MCTP Base Protocol Specification

  i. The device will comply to all Endpoint requirements as specified in this section

  ii. For SMBus (Binding) Protocol, the device shall not be the MCTP Bus Owner

  iii. "The device shall support receiving an MCTP Set Endpoint ID command at any time (Standby or Main) power is applied. For example, the BMC firmware could be updated resulting in a BMC reset. The BMC initialization sequence will go out and dynamically discover devices and assign Endpoint IDs. While in most cases, the same Endpoint ID will be assigned, this is not guaranteed. The BMC will query the device to see if an Endpoint ID has been previously assigned and is not in conflict with any other assigned Endpoint.

- Specific compliance with Section “MCTP Message Types”
- Compliance with Section “MCTP Control Protocol”
- Compliance with Section “MCTP Control Messages”
- Support Section “MCTP Control Message Command Codes”, “MCTP Control Command Numbers”
- Support Section “Prepare for Endpoint Discovery”
- Support Section “Endpoint Discovery”

MCTP Control Command (DSP0236)	**Implementation**
0x01 Set Endpoint ID	Required
0x02 Get Endpoint ID	Required
0x03 Get Endpoint UUID	Required
0x04 Get MCTP Version Support	Required
0x05 Get Message Type Support	Required
0x06=Get Vendor Defined Message Support	Required
0x0B=Prepare for Endpoint Discovery	Required (PCIe VDM)
0x0C=Endpoint Discovery	Required (PCIe VDM)
0x0D=Discovery Notify	Required (PCIe VDM)

5.2.2 PLDM

PLDM Messaging Control and Discovery Command Codes (DSP0240)	**Implementation**
0x01 SetTID	Required

0x02 GetTID	Required
0x03 GetPLDMVersion	Required
0x04 GetPLDMTypes	Required
0x05 GetPLDMCommands	Required
0x06 SelectPLDMVersion	Required
0x07 NegotiateTransferParameters	Required for PLDM FW Multipart
0x08 MultipartSend	Required
0x09 MultipartReceive	Required

5.2.3 Redfish Device Enablement

Devices shall implement Redfish Device Enablement DSP0218.

Devices shall support 1 and should support a minimum of 4 concurrent RDE tree operations in parallel.

Devices should be capable of returning the requested portions of the Redfish tree in less than 2 seconds.

RDE Command (DSP0218)		**Implementation**
0x01 NegotiateRedfishParameters		Required
	DeviceCapabilities	Support atomic resource read to enable consistent reads
	DeviceFeatureSupport	Head and replace optional. Read mandatory. Other features are optional for inventory-only devices and mandatory for full support.

	DeviceConfigurationSignature	Required as documented
0x02 NegotiateMediumParameters		Required
0x03 GetSchemaDictionary		Required
0x04 GetSchemaURI		Required
0x05 GetResourceETag		Required
0x08 GetRegistryCount		Required for full event support
0x09 GetRegistryDetails		Required for full event support
0x0A SelectRegistryVersion		Required for full event support
0x0B GetMessageRegistry		Required for full event support
0x10 RDEOperationInit		Required
OperationType	0=OPERATION_HEAD	Required
	1=OPERATION_READ	Required
	2=OPERATION_CREATE	Conditional - Required for write support
	3=OPERATION_DELETE	Conditional - Required for write support
	4=OPERATION_UPDATE	Conditional - Required for write support
	6=OPRERATION_ACTION	Conditional - Required for write support
0x11 SupplyCustomRequestParameters		Required
0x12 RetrieveCustomResponseParameters		Required
0x13 RDEOperationComplete		Required
0x14 RDEOperationStatus		Required
0x16 RDEOperationEnumerate		Required
0x30 RDEMultipartSend		Required for write support
0x31 RDEMultipartReceive		Required

5.3 Device identification

For purposes of identification, devices meeting this specification shall expose an interface to a Platform Management FRU Information Storage Definition-compatible API. The SMC API does not specify where this FRU payload must be physically implemented within the card. Implementations may choose to implement in a physical eeprom device, or a virtual eeprom device. Care should be taken in platform design to ensure that the eeprom is available in the required power states consistent with the needs of the baseline server, but this specification does not specify which power states it will be available in. Devices shall implement the following minimum fields.

Product info area:

- Manufacturer Name
- Product name
- Product Serial Number

Other fields may be populated at the device's discretion.

5.4 Device Classes

The following device classes are supported: Accelerator, NIC, DPU (SmartNIC), Storage Tray, Memory (CXL).

6. API surface

The below table calls out for the various device classes, whether a given feature is Required (R) or Conditional (C) on existence of the feature. For example, an Accelerator device may not have a Fan and thus Fan Control is conditional. .

Subsystem	**Thermal**	**Inventory**	**Software Mgmt**	**Fan Control**	**Security**	**Power Mgmt**
Accelerator	R	R	R	C	R	C
NICs	C	R	R	C	R	C
DPUs	R	R	R	C	R	C
Storage Tray	C	R	R	C	R	C
Memory (CXL)	R	R	C	C	R	C

6.1 Thermal management

The thermal management subsystem within this specification is intended to allow SMC devices to be managed by a system closed loop thermal system. Devices required to implement the thermal management subsystem shall implement the following Redfish resources and Properties

Sensor:

An SMC SensorCollection shall implement one or more sensor resources representing the thermal temperature of the device. The sensor shall have the following properties supported.

- *ReadingUnits*: Shall be present and implemented as “Cel”
- *ReadingType*: Shall be present and implemented as “Temperature”
- Threshold properties of *UpperCritical*, *UpperFatal*, *LowerCritical*, and *LowerFatal*, shall be implemented and represent the design limits of the SMC device in question. Devices with unlimited thermal design limits shall omit these properties.

If an SMC device possess fans, the SMC SensorCollection shall implement a Sensor with the following properties:

- *ReadingUnits:* Shall be present and implemented as "{rev}/min"
- *ReadingType:* Shall be present and implemented as "Rotational"

Devices implementing a single replaceable component *may* implement PLDM type 2 for sensor readings.

PLDM Type 2 Sensor Requirements

Platform Level Data Model for Platform Monitoring and Control (DSP0248)		**Implementation**
Terminus Commands		
	0x04 SetEventReceiver	Required for RDE Alert
	0x05 GetEventReceiver	Required for RDE Alert
	0x0B PollForPlatformEventMessage	Required for RDE alert
	0x0C EventMessageSupported	Required
	0x0D EventMessageBufferSize	Required for RDE Alert if MSG size > 256 bytes
Numeric Sensor Commands		Required
	0x11 GetSensorReading	Required
	0x12 GenSensorThreshold	Required
	0x15 GetSensorHysteresis	Required
State Sensor Commands		
	0x21 GetStateSensorReadings	Required
PDR Repository Commands		
	0x50 GetPDRRepositoryInfo	Required
	0x51 GetPDR	Required

	0x53 GetPDRRepositorySignature	Required for RDE
PLDM Event Types		
	0x02 redfishTaskExecutedEvent	Required if implementation cannot complete commands quickly enough to avoid spawning RDE tasks
	0x03 redfishMessageEvent	Required for redfish eventing
	0x04 pldmPDRRepositoryChgEvent	Required for RDE
PDR Type Values		
	2 = Numeric Sensor PDR	Required
	4= State Sensor PDR	Required
	22 = Redfish Resource PDR	Required for RDE

Devices implementing multiple replaceable components shall implement ThermalSubsystem over RDE.

ThermalSubsystem:

An SMC Redfish ThermalSubsystem shall be implemented, with the following properties:

- *Fans*: With Fan resources representing the fans physically present on this device.
- Fan resources shall contain the following properties
    - *SpeedPercent*

6.2 Inventory management

Requirements within this section are intended to allow inventory management and control of a given device. SMC devices shall implement:

ChassisCollection:

The ChassisCollection in the device shall contain one or more Chassis Resources representing the device in question. Chassis resources shall implement the following properties:

- Model: The value of this property shall match the “Product Name” field present in the FRU identification from section 5.3

- Manufacturer:The value of this property shall match the “Manufacturer Name” field present in the FRU identification from section 5.3
- SerialNumber: The value of this property shall match the “Serial Number” field present in the FRU identification from section 5.3

SMC devices *may* implement more than one Chassis resource, for representing physical subsystems within the device. Within the SMC chassis collection, there shall be only one Chassis instance (referred to further as the “root”) that does not possess a ContainedBy attribute, and is intended to represent the overall containment of the device. All other devices shall have a ContainedBy Link, traceable to the root device. Root devices shall implement a “Contains” property representing the devices containment

6.3 Software management

SMC devices implementing multiple updatable components shall implement an UpdateService. The SMC UpdateService shall implement one or more of FirmwareInventory, or SoftwareInventory collections.

Collections shall contain at least one member of type SoftwareInventory, implementing the following properties

- **Version**: A string representing the software version running on the SMC device.
- **Updateable**: A property that conveys whether or not the device supports update. For SMC devices, this shall be set to True.
- **SoftwareId:** A property to uniquely identify this devices firmware type.
- **AdditionalVersions:** Devices that fit the Redfish descriptions in these properties shall implement AdditionalVersions, with the appropriate subproperties. Devices that do not fit the description shall omit the AdditionalVersions Property.

Devices implementing a single updatable component should implement updates of their firmware through PLDM type 5 (DSP0267 Section 6).

PLDM For Firmware Update DSP0267	**Implementation**
0x01 QueryDeviceIdentifiers	Required
0x02 GetFirmwareParameters	Required
0x03 QueryDownstreamDevices	Required

0x04 QueryDowstreamIdentifiers	Required
0x05 GetDownstreamFirmwareParameters	Required
0x10 RequestUpdate	Required
0x13 PassComponentTable	Required
0x14 UpdateComponent	Required
0x15 RequestFirmwareData	Required
0x16 TransferComplete	Required
0x17 VerifyComplete	Required
0x18 ApplyComplete	Required
0x1A ActivateFirmware	Required
0x1B GetStatus	Required
0x1C CancelUpdateComponent	Required
0x1D CancelUpdate	Required
0x20 RequestDownstreamDeviceUpdate	Required

SMC devices shall be required to be updated in 1 minute or less, measured in the time that the device is unavailable, and 5 minutes or less from the time the update is requested, including all data transfers to the device.

6.4 Fan control

SMC devices containing fans shall implement control and monitoring of those fans through the RDE interface. Devices shall support the Redfish Control schema for fan control within a system. SMC devices *may* run internal control loops in addition to the control loops presented on the RDE interface.

6.5 Power Management

If an SMC device supports reset, the Redfish Chassis.Reset action shall be supported.

If an SMC device captures power metrics, the SMC EnvironmentMetric and Sensor collection shall implement the following properties where supported:

- EnergykWh or EnergyJoules
- PowerWatts
- PowerLimitWatts
- ResetMetrics
- AverageReading
- AveragingInterval

6.6 Security

Sync with the OCP Security Group around these requirements is required.

Security Protocol & Data Model (SPDM) Specification DSP0274 - https://www.dmtf.org/dsp/DSP0274

SPDM Request Codes (DSP0274)		**Implementation**
0x81 GET_DIGESTS		Required
0x82 GET_CERTIFCATE		Required
0x83 CHALLENGE		Required
0x84 GET_VERSION		Required
0xE0 GET_MEASUREMENTS		Required
	MEAS_CAP=10b	Required
	DMTFSpecMeasurementValueType ● [00h] Immutable ROM ● [01h] Mutable FW	Required
0xE1 GET_CAPABILITIES		Required
	CERT_CAP	Required

	CHAL_CAP	Required
	MEAS_CAP	Required
0xE3 NEGOTIATE_ALGORITHMS		Required
	BaseAsymAlgo [Bit 2] TPM_ALG_RSASSA_3072 [CMA, CNSA, OCP] (Allowed) [Bit 4) TPM_ALG_ECDSA_ECC_NIST_P256 [CMA] (Allowed) [Bit 7] TPM_ALG_ECDSA_ECC_NIST_P384 [CMA, CNSA, OCP] (Preferred)	
	BaseHashAlgo [Bit 0] TPM_ALG_SHA_256 [CMA] (Allowed) [Bit 1] TPM_ALG_SHA_384 [CMA, CNSA, OCP] (Preferred)	
	MeasurementHashAlgo [Bit 1] TPM_ALG_SHA_256 [CMA] (Allowed) [Bit 2] TPM_ALG_SHA_384 [CMA, CNSA, OCP] (Preferred)	
0xFF RESPOND_IF_READY		Required

Security Requirements

SPDM 1.1 or later is required. The following attributes shall be supported:

- Authentication
- Identification
- Attestation