# Redfish Interoperability Profiles

DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. Members and non-members may reproduce DMTF specifications and documents, provided that correct attribution is given. As DMTF specifications may be revised from time to time, the particular version and release date should always be noted.

Implementation of certain elements of this standard or proposed standard may be subject to third party patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose, or identify any or all such third party patent right, owners or claimants, nor for any incomplete or inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize, disclose, or identify any such third party patent rights, or for such party's reliance on the standard or incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any party implementing such standard, whether such implementation is foreseeable or not, nor to any patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a patent owner for such implementations.

For information about patents held by third-parties which have notified the DMTF that, in their opinion, such patent may relate to or impact implementations of DMTF standards, visit http://www.dmtf.org/about/policies/disclosures.php.

This document's normative language is English. Translation into other languages is permitted.

# Redfish Interoperability Profiles

# 1. Introduction

Because the Redfish Schemas are designed to provide signifcant flexibility, and allow conforming implementations on a wide variety of products, very few properties within the Schemas are required by the Specification. But consumers and software developers need a more rigidly defined set of required properties (features) in order to accomplish management tasks. This set allows users to compare implementations, specify needs to vendors, and allows software to rely on the availability of data. To provide that "common ground", a Redfish Interoperabilty Profile allows the definition of a set of schemas and property requirements, which meet the needs of a particluar class of product or service.

The Redfish Interoperability Profile is a JSON document which contains Schema-level, Property-level, and Registry-level requirements. At the property level, these requirements can include a variety of ConditionalRequirements under which the requirement applies.

# 2. Design Tenets

All profile entries (at the Profile, Resource, or Property level) are "additive". That is, each requirement can only apply more rigid requirements which override less rigid requirements.

The profile document is a JSON document designed to minimize the work necessary to define a profile, by defining default values that allow the majority of requirements to be stated with minimal effort.

The JSON document structure is intended to align easily with JSON payloads retreived from Redfish Service implementations, to allow for easy comparisons and conformance testing.

Profile requirements do not allow for exclusions of data. Implementations are able to provide more data in their resources than required by a profile, as an implementation likely addresses multiple use cases or Profiles. This include both standard properties and OEM extensions.

# 3. Profile Definition

A Redfish Interoperability Profile is specified in a JSON document. The JSON objects and properties contained in the document are described in this specification, and are also available in a json-schema form (RedfishProfile.v1_x_x.json) from the DMTF's Redfish Schema repository at http://redfish.dmtf.org/schemas for download. The json-schema can be used to validate a Profile document to ensure compatibility with automated conformance tools or utilities.

# 3.1. Basic functions

At the top level of the JSON document are the basic properties which describe the profile, including authorship and contact information, versioning, and other profiles to include in order to build upon previous work.

| property | type | description |
|---|---|---|
| ProfileName | string | The name of this Redfish Profile. |
| Author | string | The author(s) of this Redfish Profile. |
| Version | string | The version of this Redfish Profile. |
| Purpose | string | A description of the purpose of this Redfish Profile, such as its intended target audience, product segments, etc. |
| ContactInfo | string | An email address that can be used to provide feedback about this Redfish Profile. |
| RequiredProfiles | object | A set of Redfish Profiles which serve as a basis for this Profile. The requirements set forth in these Profiles are included in this Profile. |

## 3.1.1. Required Profiles

The RequiredProfiles object contains properties (of type object) that are named to match the name of the profile to be included. Each of these sub-objects contains the properties listed below.

| property | type | description |
|---|---|---|
| Repository | string | A URI providing the location of the repository which contains the file(s) to be included. If absent, the location shall be the Redfish Schema Repository at redfish.dmtf.org |
| OwningEntity | string | Indicates whether this resource is defined by schema published by a standards body or an OEM. If this property is absent, the value shall be 'DMTF'.The author(s) of this Redfish Profile. |
| OwningEntityName | string | Name of the owning entity, when used with 'Other', follows 'Oem Property Naming' in the Redfish Specification |
| MinVersion | string | The minimum version required by this Redfish Profile. If this property is absent, the minimum value shall be '1.0.0'. |

## 3.2. Example

The following is an example of the top-level properties in a Profile, with two Required profiles included.

```
"@odata.type": "RedfishProfile.v1_0_0.RedfishProfile",
"ProfileName": "Anchovy",
"Version": "1.0.2",
"Author": "Pizza Box Project",
"Purpose": "This is a sample Redfish Interoperability profile.",
"ContactInfo": "pizza@contoso.com",
"RequiredProfiles": {
    "DMTFBasic": {
        "MinVersion": "1.0.0"
    },
    "ContosoPizza": {
        "OwningEntity": "Other",
        "OwningEntityName": "Contoso",
        "Repository": "contoso.com/profiles",
        "MinVersion": "1.0.0"
    }
}
```

## 3.3. Protocol requirements

An object named 'Protocol' contains properties which describe Redfish protocol functionality that is not related to the supported schemas or properties. Therefore, these functions cannot be validated by comparing retreived JSON payloads.

| property | type | description |
|---|---|---|
| MinVersion | string | The minimum version of the Redfish Specification protocol support required by this Profile. This version shall be reported by the Redfish Service in the ServiceRoot property 'RedfishVersion'. If this property is absent, the minimum value shall be '1.0.0'. |
| DiscoveryRequired | boolean | Indicates that support of the Redfish SSDP Discovery protocol is required for this Profile. If this property is absent, the value shall be false. |

## 3.4. Example

```
"Protocol": {
    "MinVersion": "1.2",
    "DiscoveryRequired": true
}
```

# 4. Resource (Schema) requirements

The primary content in a Redfish Profile is the set of supported property requirements. As Redfish is organized and defined by schema-backed JSON resources, these requirements are also organized by schema.

For each schema, an object is created in the JSON document, named to match the schema's name. Within this object, properties describe the location of the schema file, and schema-level requirements. Within each schema-level object is a "PropertyRequirements" object that describes the property-level requirements for that schema. The definition of both the schema/resource-level and property-level requirements are accomplished using the same mechanisms, which are described in the next section.

The structure of the resource and property requirements is:

```
{
    <Schema Name>: {
        "MinVersion": "<version>"
        "PropertyRequirements": {
            <Property Name>: {
                <Requirements for this property>
            },
            <Property Name>: {
            }
        },
        "ActionRequirements": {
            <Action Name>: {
                <Requirements for this action>
            }
        }
    },
    <Additional Schemas...>
```

# 4.1. Schema level functions

The following options are available at the schema level:

| property | type | description |
|---|---|---|
| Repository | string | A URI providing the location of the repository which contains the file(s) to be included. If absent, the location shall be the Redfish Schema Repository at redfish.dmtf.org |
| OwningEntity | string | Indicates whether this resource is defined by schema published by a standards body or an OEM. If this property is absent, the value shall be 'DMTF'.The author(s) of this Redfish Profile. |
| OwningEntityName | string | Name of the owning entity, when used with 'Other', follows 'Oem Property Naming' in the Redfish Specification |
| MinVersion | string | The minimum version required by this Redfish Profile. If this property is absent, the minimum value shall be '1.0.0'. |
| Requirement | string | Resource-level requirement for this schema, see Requirement section. |
| ConditionalRequirements | object | Resource-level conditional requirements that apply to instances of this schema, see Conditional Requirements section. |

## 4.1.1. Example

This example shows a simple required schema

```
"ComputerSystem": {
    "MinVersion": "1.2.0",
    "PropertyRequirements": {
        "SerialNumber": {},
        "Manufacturer": {},
        "Model": {
            "Requirement": "Recommended"
        },
```

## 4.2. Property level functions

Within the 'PropertyRequirements' object are additional objects which are named to match the property name in the parent object's schema definition. This object then contains the property-level requirements, which account for the bulk of a Profile's definition. One additional level of JSON objects may be embedded, essentially nesting a 'PropertyRequirements' object.

The following options are available at the property level:

| property | type | description |
|---|---|---|
| Requirement | string | Property-level requirement for this property, see [Requirement](#) section. |
| ConditionalRequirements | object | Property-level conditional requirements that apply to instances of this property, see [Conditional Requirements](#) section. |
| Writeable | boolean | True if the property is required to be writeable by the user. False or not present if the property may be read-only. |
| MinCount | integer | For array type properties, the minimum number of non-NULL instances within the array. |
| MinSupportValues | array | The minimum set of enumerations that must be supported for this writeable property. |
| Comparison | string | The condition used to compare the value of the property to 'Values'. See the [Comparison](#) section. |
| Values | array | The value(s) required for this property based on the 'Comparison'. If no 'Comparison' is present, the property must be equal to one of the values listed. |
| PropertyRequirements | object | For Redfish properties of type 'object', this object contains requirements for the properties contained within the specified object. This specification allows for only one level of nested objects and requirements. |

### 4.2.1. Example

This example shows property-level requirements, including one of type 'object' containing further requirements on that object's properties. For each 'Power' resource, the 'PowerSupplies' and 'Voltages' array properties are required. 'Voltages' has no further requirements (by default, this property is mandatory, and as an array type, must have at least one item in the array. The 'PowerSupplies' array

must contain at least two (object) items. Within the array, at least one item's 'PowerSupplyType' property must have a value of 'AC' or 'DC'.

```
"Power": {
    "PropertyRequirements": {
        "PowerSupplies": {
            "Requirement": "Mandatory",
            "MinCount": 2,
            "PropertyRequirements": {
                "Status": {},
                "PowerSupplyType": {
                    "Requirement": "AnyOf",
                    "Values": [ "AC", "DC" ]
                },
                "LineinputVoltage": {},
                "PowerCapacityWatts": {},
                "InputRanges": {
                    "Requirement": "Recommended"
                }
            }
        },
        "Voltages": {}
    }
},
```

## 4.2.2. Comparison

The Comparison function uses the following enumerations to represent the arithmetic comparisons available:

| value | description |
|---|---|
| AnyOf | An instance of the property in this resource must be equal to one of the values listed. |
| AllOf | At least one instance of the property in this resource must be equal to each of the values listed. |
| Equal | The value must be equal to the KeyValue. |
| NotEqual | The value of the property must not be equal to the value(s) listed. |
| GreaterThan | The value of the property must be greater than the Values. |
| GreaterThanEqual | The value of the property must be greater than or equal to the Values. |

| value | description |
|-------|-------------|
| LessThan | The value of the property must be less than to the Values. |
| LessThanEqual | The value of the property must be less than or equal to the Values. |

## 4.2.3. Requirement

This function specifies the level of requirement applied to the resource or property.

| value | description |
|-------|-------------|
| Mandatory | This property is required in all instances of this resource. For properties of type 'array', the property is required in all non-NULL array items. If 'Values' are listed, at least one instance of each enumeration value is required among instance(s) of this property. |
| Recommended | It is recommended, but not required, that this property be supported. |
| IfImplemented | This property is required if the underlying functionality is implemented. For properties of type 'object', requirements on embedded properties within the object will only apply if the object is present. |
| Conditional | This property is only required if 'ConditionalRequirements' items apply to this instance of the resource. |
| None | This property is not required by this profile. It is listed here for clarity. |

## 4.2.4. Conditional Requirements

The most flexible aspect of the Redfish Profile definition is the ability to make resource or property-level requirements that are dependent on one or more ConditionalRequirements within the resource and the parent resource(s) in the resource tree.

The 'ConditionalRequirements' array function specifies these conditional requirements, which add to any requirements also defined for the resource or property. Note that a condition cannot override or weaken a requirement already specified. For example, if a property requirement is marked as 'Mandatory', no conditional requirement could mark the property as 'None'. Instead, the property would be specified with a 'None' requirement, and with one or more ConditionalRequirements that would specify when the property requirement becomes 'Mandatory'.

The following options are available for each conditional requirement:

| property | type | description |
|---|---|---|
| Requirement | string | The requirement to apply to the resource or property if the condition is met. |
| Purpose | string | Text describing the purpose of this conditional requirement. |
| Writeable | boolean | Condition applies if the property is writeable. |
| SubordinateToResource | array | An ordered list (from top of heirarchy to bottom) of resources where this resource is linked as as subordinate resource. The conditional requirements listed for the resource apply only to instances which are subordinate to the listed parent resource list. See Parent and subordinate resources section. |
| CompareProperty | string | The name of the property in this resource whose value is used to test this condition. The property name will be evaluated at the current object level within the resource. If the property name is not found at the current level, upper levels will be searched until the root level is reached. See the Compare Property section. |
| CompareValues | array | Values of the CompareProperty used to test this condition. See the Compare Property section. |
| Comparison | string | The condition used to compare the value of the property named by 'CompareProperty' to the value of 'CompareValues'. If the comparison is true, then this conditional requirement applies. See the Compare Property section. |

### 4.2.4.1. Parent and subordinate resources

As there can be several instances of a particular Redfish schema in the resource tree, the requirements placed on those resources may vary depending on their usage. Since the Profile is schema-centric, the 'SubordinateToResource' function allows a Profile to specify requirements based a resource instance's placement in the resource tree.

'SubordinateToResource' allows specifying the schema (resource) path from parent resources to the resource to which the requirements apply. This property contains an array of schema names, in the top-down order that they appear in the path to the required resource.

#### 4.2.4.1.1. Example

For the property 'HostName' in the 'EthernetInterface' schema, the example shows it as 'Recommended'

property. But if an instance of 'EthernetInterface' is linked from a 'ComputerSystem' resource, through the 'EthernetInterfaceCollection', then the 'Condition' is met, which changes the 'HostName' property requirement to 'Mandatory'.

In the second part of the example, the 'IPv6Addresses' array property is required to have at least one item ('MinCount') in the array. But if, as above, the instance is subordinate to a 'ComputerSystem' (and 'EthernetInterfaceCollection') resource, then at least two items are required in the array.

```
    "EthernetInterface": {
        "PropertyRequirements": {
            "HostName": {
                "Requirement": "Recommended",
                "ConditionalRequirements": [{
                    "SubordinateToResource": ["ComputerSystem",
 "EthernetInterfaceCollection"],
                    "Requirement": "Mandatory"
                }]
            },
            "IPv6Addresses": {
                "Requirement": "Mandatory",
                "MinCount": 1,
                "ConditionalRequirements": [{
                    "SubordinateToResource": ["ComputerSystem",
 "EthernetInterfaceCollection"],
                    "MinCount": 2
                }]
            }
        }
    }
```

### 4.2.4.2. Compare Property

A typical need for a conditional requirement is a dependency on the value of another property within the resource. This type of dependency can be used when several different product variations share a common schema definition. In that case, Redfish schemas normally define a type-specifying property with enumerations (for a variety of product categories) that can be used to differentiate Profile requirements by product category.

To accomplish this, there are three Profile properties related to this function:

| property | type | description |
|---|---|---|
| CompareProperty | string | The name of the property in this resource whose value is used to test this condition. The property name will be evaluated at the current object level within the resource. If the property name is not |

| property | type | description |
|----------|------|-------------|
|  |  | found at the current level, upper levels will be searched until the root level is reached. |
| Comparison | string | The condition used to compare the value of the property named by 'CompareProperty' to the value of 'Values'. If the comparison is true, then this conditional requirement applies. |
| CompareValues | array | Values of the CompareProperty used to test this condition. |

### 4.2.4.3. Example

This example shows a CompareProperty condition applied to the 'IndicatorLED' property, which has a base 'Recommended' requirement, but becomes 'Mandatory' if the 'SystemType' property has a value of 'Physical' or 'Composed'.

```
    "IndicatorLED": {
        "Requirement": "Recommended",
        "ConditionalRequirements": [{
            "Purpose": "Physical and composed Systems must have a writable Indicator
 LED",
            "CompareProperty": "SystemType",
            "Comparison": "AnyOf",
            "CompareValues": ["Physical", "Composed"],
            "Requirement": "Mandatory",
            "Writeable": true
        }]
    },
```

## 4.3. Action Requirements

As several critical functions of a Redfish Service are implemented as 'Actions', the Profile may place requirements for support of these Actions. The requirements can which Parameters must be supported, and may specify Allowable Values for those parameters.

The following functions are available to specify requirements for an Action within a Resource requirement:

| property | type | description |
|----------|------|-------------|
| Requirement | string | The requirement to apply to this Action. |
| Parameters | object | Requirements for any parameter available for this Action. |

### 4.3.1. Parameters

The following functions are available to specify requirements for a parameter on a particular Action:

| property | type | description |
|---|---|---|
| Requirement | string | The requirement to apply to this parameter. |
| MinSupportValues | array | The minimum set of enumerations that must be supported for this parameter. |

### 4.3.2. Example

This exampls shows the 'Reset' action as required for this resource, along with the required parameter 'ResetType', which must support the values of 'ForceOff' and 'PowerCycle'.

```
"ActionRequirements": {
    "Reset": {
        "Requirement": "Mandatory",
        "Parameters": {
            "ResetType": {
                "MinSupportValues": ["ForceOff", "PowerCycle"],
                "Requirement": "Mandatory"
            }
        }
    }
}
```

# 5. Registry level requirements

While not normally part of the JSON resources, the Redfish-defined Message Registries are important for interoperability, as they indicate what functionality has been implemented for Events, and are also a useful method for setting expections on the use of Extended Info error messages when interacting with a Redfish Service implementation.

The following functions are available to specify Registry-level requiremenets:

| property | type | description |
|---|---|---|
| Repository | string | A URI providing the location of the repository which contains the file(s) to be included. If absent, the location shall be the Redfish Schema Repository at redfish.dmtf.org |

| property | type | description |
|---|---|---|
| OwningEntity | string | Indicates whether this resource is defined by schema published by a standards body or an OEM. If this property is absent, the value shall be 'DMTF'.The author(s) of this Redfish Profile. |
| OwningEntityName | string | Name of the owning entity, when used with 'Other', follows 'Oem Property Naming' in the Redfish Specification |
| MinVersion | string | The minimum version required by this Redfish Profile. If this property is absent, the minimum value shall be '1.0.0'. |
| Requirement | string | Resource-level requirement for this Registry, see Requirement section. |
| Messages | object | The Messages in this Registry which have support requirements for this Redfish Profile. If this property is absent, all Messages in this Registry follow the registry-level 'Requirement'. |

## 5.1. Messages

Within the Registry object are additional objects which are named to match the Message name in the Registry definition. This object then contains the message-level requirements.

The following options are available at the property level:

| property | type | description |
|---|---|---|
| Requirement | string | Message-level requirement for this Message, see Requirement section. |

## 5.2. Example

This example shows requirements for two Message Registries, including one OEM-defined registry. The 'Base' Registry is a DMTF standard Registry (by default since no 'OwningEntity' is listed) and therefor can be retreieved by default from the DMTF Repository. The 'Base' Registry lists only four Messages that are required.

In the case of the OEM-defined Registry 'ContosoPizzaMessages', the 'Mandatory' requirement set at the Registry level specifies that all Messages defined in that Registry are required.

```
    "Registries": {
        "Base": {
```

```
            "MinVersion": "1.1.0",
            "Messages": {
                "Success": {},
                "GeneralError": {},
                "Created": {},
                "PropertyDuplicate": {}
            }
        },
        "ContosoPizzaMessages": {
            "OwningEntity": "Other",
            "OwningEntityName": "Contoso",
            "Repository": "contoso.com/registries",
            "Requirement": "Mandatory"
        }
    }
```

# Change Log

| version | date | changes |
|---------|------|---------|
| v0.9 | 5-14-17 | Work In Progress release. |