Redfish Release 2024.2

DMTF Redfish Forum
July 2024
Redfish Release 2024.2

- Redfish Schema Bundle 2024.1
  - DSP8010 contains all released Redfish schemas
- 6 Updated schemas (highlights)
  - See release notes in DSP8010 for errata details
  - Release driven by industry need to synchronize support for NVM Configuration Lock functionality
  - Added `TargetConfigurationLockLevel`, `NVMe`, and `BlockSecurityIDEnabled` to Drive
  - Added `TargetConfigurationLockLevel` to Storage
  - Added `SetControllerPassword` to Storage
- Download all published material at:
  http://www.dmtf.org/standards/redfish
Drive and Storage Configuration Lock

- **ConfigurationLock** and **NEW TargetConfigurationLockLevel** were added to the **Drive** and **Storage** resources
  - Restricts the usage of *in-band* configurations of the drive or storage subsystems
- **ConfigurationLock** shows the current lock state and is used by clients to change the lock state
  - “Enabled” – In-band configurations are locked
  - “Disabled” – No locking applied
  - “Partial” – Only some of the desired locking configuration could be applied
    - This value is prohibited from PATCH/PUT operations and is only for reporting purposes
NEW Drive and Storage Configuration Lock

- **TargetConfigurationLockLevel** specifies the functions to lock
  - “Baseline” – Lock the ability to update firmware, manage security parameters including keys, or perform other hardware configurations
  - Other values can be added over time as organizations or environments have new use cases to restrict in-band configuration access of drives or storage subsystems
  - For NVMe devices, SNIA's Swordfish “NVMe Model Overview and Mapping Guide” defines the specific NVMe commands to lock based on the value

- For NVMe devices, the **ConfigurationLockState** property in both the Drive and Storage resources contains supplemental information
  - Shows each NVMe command with their locking status
  - Also shows if the command is supported or can be locked
Drive and Storage Configuration Lock

Example 1: Unlocked NVMe subsystem

```
{
    "ConfigurationLock": "Disabled",
    "TargetConfigurationLockLevel": "Baseline",
    "NVMeSubsystemProperties": {
        "ConfigurationLockState": {
            "FirmwareCommit": "Unlocked",
            "Lockdown": "Unlocked",
            "SecuritySend": "Unlocked",
            "FirmwareImageDownload": "Unlocked",
            "VPDWrite": "CommandUnsupported"
        }
    }
}
```

Example 2: Client requesting to lockdown an NVMe subsystem

```
{
    "ConfigurationLock": "Enabled",
    "TargetConfigurationLockLevel": "Baseline"
}
```
Drive and Storage Configuration Lock

Example 3: Fully locked NVMe subsystem

```
{
    "ConfigurationLock": "Enabled",
    "TargetConfigurationLockLevel": "Baseline",
    "NVMeSubsystemProperties": {
        "ConfigurationLockState": {
            "FirmwareCommit": "Locked",
            "Lockdown": "Locked",
            "SecuritySend": "Locked",
            "FirmwareImageDownload": "Locked",
            "VPDWrite": "CommandUnsupported"
        }
    }
}
```

Example 4: Partially locked NVMe subsystem; Security Send was not locked

```
{
    "ConfigurationLock": "Partial",
    "TargetConfigurationLockLevel": "Baseline",
    "NVMeSubsystemProperties": {
        "ConfigurationLockState": {
            "FirmwareCommit": "Locked",
            "Lockdown": "Locked",
            "SecuritySend": "Unlocked",
            "FirmwareImageDownload": "Locked",
            "VPDWrite": "CommandUnsupported"
        }
    }
}
```
Schema and Registry Guide documents

- **Redfish Data Model Specification (DSP0268)**
  - Document was previous titled “Redfish Schema Supplement”
  - Now includes normative statements (“LongDescription”) and informative description details from schema in a single document
  - Intended for both Redfish Service and client-side developers

  - Presents schema (data model) contents in a more friendly format for end users
  - Includes example payloads for each resource type

- **Redfish Message Registry Guide (DSP2065)**
  - Presents message registry definitions in a more human-readable format
  - Includes summary table and individual message details

- **Redfish Property Guide (DSP2053)**
  - Provides an alphabetical list of all properties defined in Redfish schema
  - Useful for schema writers to locate existing definitions or properties
  - Helps avoid re-defining property names already in use
Available Redfish conformance testing tools

- DMTF Redfish Forum provides open source tools for service developers to validate their conformance with the Redfish protocol, data model, and profiles

- **Redfish Protocol Validator**
  - Tests a live service for conformance to the Redfish HTTP protocol, including response headers and status codes
  - [https://github.com/DMTF/Redfish-Protocol-Validator](https://github.com/DMTF/Redfish-Protocol-Validator)

- **Redfish Service Validator**
  - Tests a live service for conformance with Redfish schemas, ensuring the returned JSON payloads validate against the standard data models
  - Recommend that developers run this tool first, as errors in payload are more likely to cause issues for end users and interoperability
  - [https://github.com/DMTF/Redfish-Service-Validator](https://github.com/DMTF/Redfish-Service-Validator)

- **Redfish Interop Validator**
  - Tests a service against a Redfish interoperability profile
  - [https://github.com/DMTF/Redfish-Interop-Validator](https://github.com/DMTF/Redfish-Interop-Validator)
Redfish-Publications repository

- Public GitHub repository contains an official read-only copy of the Redfish schemas and standard message registries
  - https://github.com/DMTF/Redfish-Publications
  - Creates public, durable locations for referencing specific schema or registry items in issue reports, forum postings, or other online references
  - Allows developers to automatically synchronize with new Redfish releases using normal GitHub tools and processes

- Repository will be updated as each Redfish release become public
- Contains materials published as DSP8010 and DSP8011
  - /csdl – Redfish schemas in OData CSDL XML format
  - /json-schema – Redfish schemas in JSON Schema format
  - /openapi – Redfish schemas in OpenAPI YAML format
  - /dictionaries – RDE dictionaries
  - /registries – Redfish standard message registries
Redfish Schema

MINOR REVISION DETAILS
Redfish Schema Minor Revisions

- **Drive v1.20.0**
  - Added `ConfigurationLockLevel`, `NVMe`, and `BlockSecurityIDEnabled`
- **PCleDevice v1.15.0**
  - Added `BadTLPCount` and `BadDLLPCount` to `PCleErrors`
- **PhysicalContext**
  - Added “Filter”, “Reservoir”, “Switch”, and “Manager” physical contexts
- **Port v1.13.0**
  - Added `Version` and `VendorOUI` to `SFP`
- **Sensor v1.10.0**
  - Added `Enabled`
- **Storage v1.17.0**
  - Added `ConfigurationLockLevel`
  - Added `ConfigurationLockState` to `NVMeSubsystemProperties`
  - Added `SetControllerPassword` action