Redfish Ecosystem For Storage

Jeff Hilland
Distinguished Technologist Manageability, Hewlett Packard Enterprise
President, DMTF
Disclaimer

- The information in this presentation represents a snapshot of work in progress within the DMTF.
- This information is subject to change without notice. The standard specifications remain the normative reference for all information.
- For additional information, see the DMTF website.
A Hybrid IT Management Solution

**Design Tenets**
- Leverage common Internet / Web Services standards, other standards where appropriate
- Represent modern hardware designs (standalone to scale-out, current silicon, OCP)
- Does not require a PhD to design or use.
- Separation of protocol from data model, allowing them to be revised independently

**Protocol Suite**
- HTTPS / SSL: Primary data transport
- SSDP from uPnP: Service Discovery
- HTTP-based alert subscription
- Leverage OData v4

**REST & JSON**
- Modern, standards-based
- Widely used for web services, software defined and public APIs
- Easy for IT professionals and amateurs to utilize

**Data Model**
- Schema-based, starting with CSDL & JSON Schema
- Prepare to add schema language definitions as market changes
- An easy to use data model that a human can just read
- Create new modeling tenants to facilitate ease of design (inheritance by copy, polymorphism by union)
What is Redfish?

- Industry Standard Software Defined Management for Converged, Hybrid IT
  - HTTPS in JSON format based on OData v4
  - Schema-backed but human-readable
  - Equally usable by Apps, GUIs and Scripts
  - Extensible, Secure, Interoperable

- Version 1.0 focused on Servers
  - A secure, multi-node capable replacement for IPMI
  - Represent full server category: Rackmount, Blades, HPC, Racks, Future
  - Intended to meet OCP Remote Machine Management requirement

- Expanded scope over time to rest of IT infrastructure
  - Additional features coming out approximately every 4 months
  - Working with SNIA to cover more advanced Storage (Swordfish)
  - Working with The Green Grid & ASHRAE to cover Facilities (Power/Cooling)
  - Work with the others to cover some level of Ethernet Switching
  - Working with fabric consortia to cover Data Fabrics
DMTF Redfish Forum
Co-Chairs: Jeff Autor (HPE), Mike Raineri (Dell)

Redfish Forum Leadership Companies

Redfish Supporting Companies

Redfish Industry Alliance Partners & efforts
OCP (Open Compute Project) – Collaborating on profile definition
UEFI – Collaborating on Firmware Update and Host Interface work
SNIA – Collaborating on Storage modeling / alignment between SNIA
SSM and Redfish
TGG – Pursuing relationship to work on Power/Cooling (existing DMTF Alliance Partner)
IETF – working on Switch modeling (no official alliance)

ASHRAE – American Society of Heating, Refrigerating and Air Conditioning Engineers
BBF – Broadband Forum
Gen-Z – Gen-Z Consortium
PICMG – Open Modular Computing for IIoT
NVMe – NVMe-MI
PCI SIG - PCIe
Timeline of Redfish® Specification

- The DMTF Redfish technology
  - Sep 2014: SPMF Formed in DMTF.
  - Aug 2015: Redfish Specification with base models (v1.0)
  - 2016.1: Models for BIOS, disk drives, memory, storage, volume (2016.1)
  - 2016.2: Models for endpoint, fabric, switch, PCIe device, zone, software/firmware inventory & update (2016.2)
  - 2016.3: Adv. communications devices (multi-function NICs), host interface (KCS replacement), privilege mapping (2016.3)
  - 2017.1: Composability (2017.1), WIP for Telemetry
  - 2017.2: Location, errata (2017.2), WIPs for Ethernet Switching, DCIM, OCP & Profiles
  - 2017.3: Profiles, Query parameters, errata (2017.3)
  - 2018.1: LDAP/AD, SSE, Assembly, minor enhancements & errata
  - 2018.2: OpenAPI, Telemetry, Jobs, Schedule, Compose II, Message II
  - 2018.3: Certificates, Sensor II (DCIM), FPGA
  - 2019.1: Spec Clean up; Additions to Certs, Telemetry, Console, Syslog
  - 2019.2: FW Update multipart, PCIe mods, Composition Registry

- Alignment with other standard organizations
  - Aug 2016: SNIA releases first model for network storage services (Swordfish)
  - Working open YANG Redfish mapping algorithm for Ethernet Switch
  - DMTF created work registers with UEFI, TGG, OCP, ASHRAE, Broadband Forum, ETSI-NFV, NVMe, PICMG, GenZ, ODCC for work on applying Redfish
Redfish Resource Map (simplified)

- **/redfish/v1**
  - Service Root
  - Tasks
  - Sessions
  - Accounts
  - Events
  - Registries
  - Schemas

- **/redfish/v1/Systems**
  - Collection Resource
    - "Logical view"
    - Collection of Systems
      - Computer System
        - /Systems/<id>
        - Tasks
        - Sessions
        - Accounts
        - Events
        - Registries
        - Schemas

- **/redfish/v1/Chassis**
  - Collection Resource
    - "Physical view"
    - Collection of Chassis
      - Chassis
        - /Chassis/<id>

- **/redfish/v1/Managers**
  - Collection Resource
    - "Management"
    - Collection of Managers
      - BMC
        - Managed By
        - Managed By

- **Subordinate Resources & Collections**
  - Processors
  - Disks
  - NICs
  - Power
  - Thermal
  - LogService
  - NW protocol

**GET** http://<ip-addr>/redfish/v1/Systems/{id}/Processors/{id}

Use the Redfish Resource Explorer (redfish.dmtf.org) to explore the resource map.
Host Interface

- Replacement for IPMI KCS, etc.
- Exposes a NIC from Management Controller to OS
  - SMBIOS records provide information to allow kernel access
- Same access in-band as out-of-band
  - Kernel mode or user mode accessible
  - Encouraging OS vendors to begin consuming Redfish data.
  - This means you can get to the BMC homepage from the OS.
  - This means you can write your tools for the BMC homepage or Redfish and run them in the host OS.
  - Anything that accesses the out of band can be run on the host OS to access to local management subsystem.
Interoperability Profiles: Goals

- An “Interoperability Profile” provides a common ground for Service implementers, client software developers, and users
  - A profile would apply to a particular category or class of product (e.g. “Front-end web server”, “NAS”, “Enterprise-class database server”)
  - It specifies Redfish implementation requirements, but **is not** intended to mandate underlying hardware/software features of a product
  - Provides a target for implementers to meet customer requirements
  - Provide baseline expectations for client software developers utilizing Redfish
  - Enable customers to easily specify Redfish functionality / conformance in RFQs

- Create a machine-readable Profile definition
  - Document must be human-readable
  - Can be created by dev/ops personnel and non-CS professionals

- Enable authoring of Profiles by DMTF, partner organizations, and others
- Create open source tools to document and test conformance
Redfish Storage Model

a.k.a.:

- Local Storage
- Server Storage
- Redfish Storage
Storage Resource Overview

- Storage: A representation of a storage sub-system
  - Contains sets of Volumes, Drives, and Storage Controllers
  - Storage Controller information is an array of objects in the Storage resource
    - Describes the protocols supported by the controller, the speed of the controller interface, and manufacturer information about the controller
- Drive: The physical media for the data
  - Manufacturer information about the drive (part number, serial number, etc.)
  - Capability information about the drive (size, protocol, encryption, etc.)
  - Contains control aspects (secure erase and LED setting)
- Volume: The logical construct used by the OS/hypervisor
  - Contains status about a volume (what drives contribute to the volume, size information, identifier information, etc.)
  - Allows a client to control the volume (initialization, encryption settings, etc.)
Storage in Redfish

Note that the Volumes are in Collections off of the Storage resource, drives are in arrays off of the storage resource and optionally the Chassis.
Limitations of Redfish Storage

- Redfish Storage Model is best aligned with local server storage, particularly HW/SW Raid.
  - Storage resource
    - Controllers are an array
    - Disks are an array
    - Volumes are a collection
- This can make it difficult for large scale
  - Disks array can quickly get large such as in JBOD environments
  - Not optimized for shared resources like shared sparing
  - Other storage technologies not represented (only Disks)
- There is a need for preservation of Redfish client code
  - Clients should be able to tell when they have left a Redfish implementations and started down the Swordfish path without complete re-write
  - Clients should be able to implement a migration path as more storage features get added to an implementation (represent real world solutions).
  - Swordfish 1.1 will hopefully address this.
Redfish Device Enablement (RDE)

Or “How you can fill all that storage stuff out without creating a lock step firmware dependency between the management controller firmware and the storage firmware”
PMCI WG developing a standard to enable a server Management Controller to present a Redfish-conformant management of I/O Adapters without building in code specific to each adapter family/vendor/model.

- Support adapter “self-contained, self-describing” including value-add (OEM) properties
- New managed devices (and device classes) do not require Management Controller firmware updates
- Support a range of capabilities from primitive to advanced devices (lightweight/low bandwidth options)
- Leveraging PLDM, a provider architecture is being specified that can binary encode the data in a small enough format for devices to understand and support.
- MC acts as a broker to encode/decode the data to/from the provider
- PLDM works over I2C & PCIeVDM. Additional mappings under consideration.
RDE Discovery & Registration

Discovery is PLDM based
- Devices are discovered using PLDM (which uses MCTP) and determines that it supports RDE (PLDM Type 6)
- MC uses RDE to negotiate parameters with the device.
  - Concurrency, Operations Supported, Provider Name
- MC uses RDE to negotiate channel parameters on each channel
  - Asynchrony, Max Chunk Size

Registration leverages Platform Data Record
- MC queries device for PDRs and any Associations and Actions
  - PDR = Platform Data Record. Equates to one or more Redfish Resources
- And gets Schema Identities and versions for the PDR
  - Can get a Resource ETag
- And Retrieves any Dictionary for the PDR
  - Dictionaries may be truncated to only what device supports
RDE BEJ – Translating JSON to binary using dictionaries

- RDE’s BEJ (Binary Encoded JSON) is a binary representation of the JSON payload using the algorithm specified
- Dictionaries are the key
- So the dictionary is used to turn each of these into a compact form.
  - RDE uses the dictionary to take the JSON Body and turn it into binary
  - Includes how to nest objects, handle annotations and other Redfish nuances
  - Roughly a 10:1 compression in early analysis
- DMTF will publish Dictionaries for all DMTF Schema on the website
  - Program to generate dictionaries will be made open source
### RDE Operations

- **RDE has slightly different terms than HTTP in the spec**

<table>
<thead>
<tr>
<th>HTTP Operation</th>
<th>RDE Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>GET</td>
<td>Read</td>
</tr>
<tr>
<td>PUT</td>
<td>Replace</td>
</tr>
<tr>
<td>PATCH</td>
<td>Update</td>
</tr>
<tr>
<td>POST</td>
<td>Action or Create</td>
</tr>
<tr>
<td>DELETE</td>
<td>Delete</td>
</tr>
<tr>
<td>HEAD</td>
<td>Read Headers</td>
</tr>
</tbody>
</table>

- **RDE supports multiple outstanding operations, tasks and commands**

<table>
<thead>
<tr>
<th>Command</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>RDEOperationInit</td>
<td>Begin an Operation</td>
</tr>
<tr>
<td>SupplyCustomRequestParameters</td>
<td>Provide additional parameters for Operations</td>
</tr>
<tr>
<td>RetrieveCustomResponseParameters</td>
<td>Get additional response data</td>
</tr>
<tr>
<td>RDEOperationStatus</td>
<td>Check up on an active Operation</td>
</tr>
<tr>
<td>RDEOperationComplete</td>
<td>Finalize an Operation</td>
</tr>
<tr>
<td>RDEOperationKill</td>
<td>Cancel an Operation</td>
</tr>
<tr>
<td>RDEOperationEnumerate</td>
<td>See what Operations are active</td>
</tr>
<tr>
<td>MultipartSend, MultipartReceive</td>
<td>Bulk data transfer</td>
</tr>
</tbody>
</table>
So how does all this fit together?

- Before a Client ever Contacts the Redfish Service:
  - The Management Controller uses MCTP to enumerate the devices
    - This is dependent on the Medium (I2C, PCIe, etc.)
  - MC then uses PLDM for next phase of discovery
  - PLDM support for Type 6 (RDE) is discovered as being supported!
  - RDE Discovery takes place over PLDM
  - RDE Device Registration
    - Get all the Resources, Actions, and Entity Associations
    - MC Retrieves the Dictionaries
  - The MC is now ready to handle operations to and from the device
So how does all this fit together?

- Request comes in for client
- MC determines that the URI equates to an RDE provider.
- MC translates HTTP headers that need to be passed down
- The MC translates the JSON body into BEJ using the dictionaries
- MC initiates the operation to the device
- Device processes the request
- MC takes the response, reassembles it and translates from BEJ to JSON using the dictionaries
- MC formulates HTTP Response and sends to Client
Additional information about RDE

- RDE also handles Tasks, specifies how Events are handled, and has information on handling OEM sections
- There are state machine examples and tables
- Binary Format for dictionaries is specified
  - Standard dictionaries will be published each release
- There are examples of:
  - What a Redfish tree would look like,
  - How the PDR would look like for that tree,
  - Examples of what a dictionary would look like
  - Example of BEJ encode/decode
DMTF does more than Redfish

- SMBIOS is everywhere
- PMCI
  - Security Task Force SPDM Protocol
    - Based on the USB Authentication Protocol (and then expanded)
    - Expected to be leveraged by PCIe, OCP, JEDEC, HDBaseT and others
    - Provides Authentication, Attestation and Encryption Key Exchange
    - Includes mapping for MCTP & encrypted MCTP
  - FW Update, NC-SI, Monitoring & Control, FRU, RDE, MCTP, PLDM, Mappings and Bindings
  - Updated White Paper forthcoming
- CIM
  - Consolidated efforts under a single CIM Forum
In Summary

- Redfish, along with the other DMTF WGs and DMTF alliance partners, is working to define interoperable software defined hybrid IT management for servers, storage, networking, power/cooling, fabrics and more
- And is solving problems from composition to resource managers, aggregation engines to fabric management
- As well as plumbing the mechanisms inside the box to be self contained and self describing
- And enable a zero-trust model in the platform
Can’t Miss: SNIA Swordfish™ Events

- Hands-on Workshop, Tuesday, 9/24
  - Mezzanine, 5:00 – 7:00 p.m.
  - Visit interactive workstations and see actual implementations
  - Work with open source development tools
Redfish Developer Hub: redfish.dmtf.org

- Resources
  - Schema Index
  - Specifications
  - GitHub for Redfish Tools
  - Registries
  - Other Documentation
- Mockups
  - Simple Rack-mounted Server
  - Bladed System
  - Proposed OCP Redfish Profile
  - More being added
- Education/Community
  - Redfish User Forum
  - Whitepapers, Presentations
  - YouTube shorts & Webinars

2019 Storage Developer Conference. © DMTF. All Rights Reserved.
Thank you!