SwordfishJS
- A Swordfish JS Library

Vinod Eswaraprasad, Sowmya B
Wipro Technologies
What we want to talk?

- Manageability at HyperScale
- Redfish and Swordfish – For Better Health
- A look inside Swordfish – Usage Model
- The Reusable Recipe – JS Library
- Easy SRM Dev - Demo
- Questions/Comments?
Manageability for HyperScale
Digital Infrastructure == Hyper Scale

- Digital Infrastructure of today is large set of common hardware.

- Current Infrastructure management suffers with scale
  - Performance
  - Reliability
  - Security

- Modelling difficulty in a multi-vendor environment
- Non-standard tools and frameworks
Solving the Web-scale Manageability Problems

- Web-scale is best managed by **Web interface based protocol**
- Less Chatty
  - More information in fewer transactions
- Common APIs – Restful
- Internet standards and tool chains
  - Language Support
- Simplify the manageability protocol
What Are The Choices?

- Well-known protocol – Common CRUD semantics
- Make the discovery easy
- Primarily Out-of-band (Host interface optional)
- Easily Extensible
- Supports – Compute, Network and Storage
Redfish and Swordfish
DMTF Redfish - Basics

Rest + HTTP(S) + JSON

- HTTP methods are used as protocol for common CRUD operations
- A Redfish interface shall be exposed through a web service endpoint
- Hypermedia API with a small set of defined URIs
Redfish – Hypermedia Based Protocol

- Protocols and a core set of data models and behaviors for the management of systems
- Redfish Interface
  - Restful
- Redfish Models
  - Common Models, extensible
  - Easy to Discover

- Secure
- Scalable
- Extensible
- Commonly Used

Redfish Interface
- HTTP(S)
- JSON

OData Described Schema
Redfish - Features

- OData convention
  - Resources modelled using OData, and translated to JSON
- Model Orientated
  - No dependency between Model and Protocol; can change
- Sync and Async operations
  - Time consuming tasks at the server side
- Event support
  - Time Critical State Change or Errors
- Actions Support
  - Like Reset operation
SNIA Swordfish – Storage Model over Redfish

- Extension to Redfish to support Storage
- Model for Scalable storage and associated data services
- Storage Services
  - snapshots, replication, mapping and masking, and provisioning
- Wide Range of Storage
  - Small Object Drive – to- RAID arrays – File Server – Converged Systems, Hyper converged Cloud scale storage
Swordfish – Data Model Overview

/redfish/v1
Root of Resource

/redfish/v1/storageServices
Collection of storage services (block, file, object)

/redfish/v1/storageSystems
Collection of storage systems

/redfish/v1/storageServices/<id>
Storage Service Information
Class of Services

/redfish/v1/storageSystems/<id>
Storage System information
Collection of storage systems

/redfish/v1/Systems
Collection of Systems (Logical View)

/redfish/v1/Managers
Collection of storage services (block, file, object)

/redfish/v1/Managers/<id>
BMC

/redfish/v1/Chassis
Collection of storage services (block, file, object)

/redfish/v1/Chassis/<id>
Chassis – Tracking ID (block, file, object)

/redfish/v1/Systems
Server Information
Model#, Serial #, Boot Order, etc.

/redfish/v1/Managers
Services
Logs

/redfish/v1/Managers/<id>
BMC

/redfish/v1/Chassis
BMC

/redfish/v1/Chassis/<id>
BMC

Volume
Shares
Replicas
Disks
Controller
Interconnects

Processors
Disks
Nics

Power
Thermal
Reusable JS Library for clients?
SRM Using Swordfish

Business Goals to storage specific actions and requirements

- Common Storage Resource Management Tasks
  - Configuration and provisioning
  - Resource Monitoring
  - Event and log management
  - Performance assessment
  - Diagnostics, Fault detection and remediation
  - Accounting and resource consumption

The Management Application should talk Swordfish….
Reusable Objects - Framework

- Goal of the SwordfishJS
  - Provides an easy way to access redfish/swordfish resources within JS Clients
  - Set of JS APIs that wraps
    - GET, PATCH, PUT, POST and DELETE Operations
  - Provide a of pre-defined set of JSON objects - directly used by the Application
  - Abstracts the complexity of the protocol from application developers
- Leverage existing JS modules
  - ODataJS
Swordfish JS - Operation

SRM JS Client

Predefined Objects

- System
- Logs
- Storage Pool
- Replica
- Volume
- Snapshots
- Drive
- File Share
- File System
- Logs

SRM Client

2017 Storage Developer Conference. © Wipro Technologies. All Rights Reserved.
Storage Object and Swordfish - Interactions

JS Application

SwordfishJS-Library

Redfish Wrapper

Swordfish Server

getStoragePool()

Pool -1 Object

Volume -1 Object

Volume 2 Object

Volume -1 Object

Volume 2 Object

Subscription resource

GET - /redfish/v1/StoargeServices/StoragePool/1/

GET - /redfish/v1/StoargeServices/Volume/1/

GET - /redfish/v1/StoargeServices/Volume/2/

Async Event on Volume -1

Subscription resource

2017 Storage Developer Conference. © Wipro Technologies. All Rights Reserved.
The JS Storage Resource – Sample Layout

<table>
<thead>
<tr>
<th>Storage System</th>
<th>GetStorageSystem()</th>
<th>ResetStorageSystem()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>IP address</td>
<td>Status</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Storage Pool</th>
<th>GetStoragePool()</th>
<th>CreateStoragePool()</th>
<th>DeleteStoragePool()</th>
<th>UpdateStoragePool()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>Status</td>
<td>Capacity</td>
<td>Allocated Capacity</td>
<td>Remaining Capacity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Volume</th>
<th>GetVolume()</th>
<th>CreateVolume()</th>
<th>DeleteVolume()</th>
<th>UpdateVolume()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>Status</td>
<td>Capacity</td>
<td>Allocated Capacity</td>
<td>Remaining Capacity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Snapshot</th>
<th>GetSnapshot()</th>
<th>CreateSnapshot()</th>
<th>DeleteSnapshot()</th>
<th>UpdateSnapshot()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>Status</td>
<td>Capacity</td>
<td>Allocated Capacity</td>
<td>Remaining Capacity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Drives</th>
<th>GetDrives()</th>
<th>SetDrives()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Size</td>
<td>Status</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>File Share</th>
<th>GetFileShare()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>Size</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Capacity</td>
<td>Allocated Capacity</td>
<td>Remaining Capacity</td>
<td>Shares</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Volume</th>
<th>GetVolume()</th>
<th>CreateVolume()</th>
<th>DeleteVolume()</th>
<th>UpdateVolume()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name/Description</td>
<td>Status</td>
<td>Capacity</td>
<td>Allocated Capacity</td>
<td>Remaining Capacity</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Replica Info</th>
<th>GetReplicaInfo()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replica Role</td>
<td>Source</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Log Entry</th>
<th>GetLogEntry()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log Entry Code</td>
<td>Log Entry Type</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Replicas Info</th>
<th>SetReplicaInfo()</th>
</tr>
</thead>
<tbody>
<tr>
<td>Replica Role</td>
<td>Source</td>
</tr>
</tbody>
</table>
What we have today and way forward?

- Fully compliant Redfish Wrapper
- JS Object wrapper (with GET/PUT/POST/DELETE) APIs
  - System
  - Storage Pool
  - Volume
  - Drives
  - File Share
  - File system
  - Snapshot
  - Log
- Support Event based Object State Update
- Support Actions on Objects
Reusable JS – Easy SRM Demo
The SwordfishJS Usage – Setup and Demo

**Setup**
- Demonstration of the SwordfishJS usage in sample Management Application
- Ability to quickly develop management actions
  - Sample Grommet JS Application
  - Swordfish Mockup schema and objects
  - Nginx webserver

**Demo**
- Storage System Status
  - Pools and Volume Information
- Pools and Volume Data gathering
  - Utilization
  - Health
- Volume Creation
- Event Handling
Learnings and Shortcomings…

- Mapping high level storage resources to the Swordfish Schema
  - Aggregation
  - Discovery process by navigating the GET response from Service Root
  - Handling ASYNC operations to update object status – special case
    - No direct way to identify snapshot volumes
    - Unavailability of performance statistics data in the current swordfish data model
    - Very less Diagnostic actions support
Questions ?
Thank You.