Automated Out-of-Band management with Ansible and Redfish

Jose Delarosa – Senior Linux Engineer at Dell EMC
Jake Jackson – Product Field Engineer at Ansible by Red Hat
May 2\textsuperscript{nd}, 2017
Who are we

• Jose De la Rosa (@jdelaros1)
  – Linux Engineer at Dell EMC.
  – Systems engineer, emerging technologies evangelist.
• Jake Jackson (@thedoubI3j)
  – Product Field Engineer at Ansible by Red Hat.
  – Works with the Getting Started team helping customers with standing up and getting started with Ansible.
Before we start

1. Thank you for coming.
2. Please ask questions at any time.
3. If time runs out, we will be happy to talk to you after the session.
Content

1. Out-of-band management with PowerEdge iDRAC
2. Scalable out-of-band management with Redfish
3. Automated out-of-band management with Ansible
iDRAC Overview
Integrated Dell Remote Access Controller (iDRAC)

- Embedded chip on a PowerEdge server used to run tasks that are independent of the host OS and the rest of the server.
  - Detects hardware failure
  - Manage power: turn off, on, hard reset
  - System event and lifecycle logs
- Has its own ethernet port, usually connected to separate management network.
- Referred to as “out-of-band” management, as opposed to “in-band” management which is provided by the OS.
Web UI Login
Storage controller status

- **Overview**
  - Server
  - Logs
  - Power / Thermal
  - Virtual Console
  - Alerts
  - Setup
  - Troubleshooting
  - Licenses
  - Intrusion
  - iDRAC Settings
  - Hardware
- **Storage**
  - Physical Disks
  - Virtual Disks
  - Controllers
- **Health and Properties**
  - Status: PERC H730 Adapter (PCI Slot 8)
  - Device Description: RAID Controller in Slot 8
  - PCI Slot: 8
  - Firmware Version: 25.5.0.0018
- **Controller Battery**
  - Status: Battery
  - Battery Name: Battery
  - Device Description: Battery on RAID Controller in Slot 8
  - State: Ready
### Power Supply Unit Readings

<table>
<thead>
<tr>
<th>Name</th>
<th>Amps</th>
<th>Volts</th>
</tr>
</thead>
<tbody>
<tr>
<td>PS1</td>
<td>0.3</td>
<td>122</td>
</tr>
</tbody>
</table>

### Cumulative Reading

<table>
<thead>
<tr>
<th>Time</th>
<th>Total Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Since Thu May 21 04:15:35 2015</td>
<td>1983.818 kWh</td>
</tr>
</tbody>
</table>

### Historical Peaks

<table>
<thead>
<tr>
<th>Time</th>
<th>Peak Watts</th>
<th>Peak Watts Time</th>
<th>Peak Amps</th>
</tr>
</thead>
<tbody>
<tr>
<td>Since Thu May 21 04:15:35 2015</td>
<td>317 W</td>
<td>Fri Feb 17 05:30:50 2017</td>
<td>2.6 A</td>
</tr>
</tbody>
</table>

### Historical Trends

<table>
<thead>
<tr>
<th>Time</th>
<th>Average Usage</th>
<th>Max Peak</th>
<th>Max Peak Time</th>
<th>Min Peak</th>
</tr>
</thead>
<tbody>
<tr>
<td>Last Hour</td>
<td>152 W</td>
<td>519 BTU/hr</td>
<td>168 W</td>
<td>573 BTU/hr</td>
</tr>
<tr>
<td>Last Day</td>
<td>155 W</td>
<td>529 BTU/hr</td>
<td>195 W</td>
<td>666 BTU/hr</td>
</tr>
<tr>
<td>Last Week</td>
<td>153 W</td>
<td>522 BTU/hr</td>
<td>195 W</td>
<td>666 BTU/hr</td>
</tr>
</tbody>
</table>
## System event logs

### System Event Log

#### System Event Log Table

<table>
<thead>
<tr>
<th>Severity</th>
<th>Date/Time</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Tue Feb 14 2017 00:55:11</td>
<td>Drive 4 is installed in disk drive bay 1.</td>
</tr>
<tr>
<td></td>
<td>Tue Feb 14 2017 00:55:01</td>
<td>Drive 4 in disk drive bay 1 is operating normally.</td>
</tr>
<tr>
<td>🔄</td>
<td>Tue Feb 14 2017 00:55:01</td>
<td>Drive 4 is removed from disk drive bay 1.</td>
</tr>
<tr>
<td>🔄</td>
<td>Sun Feb 12 2017 01:01:55</td>
<td>Fault detected on drive 4 in disk drive bay 1.</td>
</tr>
</tbody>
</table>

*Instructions: The System Event Log contains information about the managed system. To sort the log by column, click a column header.*
Simple Out-of-band management
Redfish Overview
What is Redfish?

- RESTful API specification defined by the DMTF for hardware management.
- Aims to replace IPMI and vendor-specific interfaces like WSMAN.
- Schema-based but human-readable.
- Clients send URI requests over https to iDRAC, so clients can be any OS or application on a server, workstation or mobile device.
What can Redfish do?

- Monitor server health status
- Alert on server health status changes
- View server hardware inventory and firmware versions
- Reset, reboot, and power control servers
- Access system logs
Example: Get system health

```bash
{
  "Health": "OK",
  "HealthRollUp": "OK",
  "State": "Enabled"
}
```
Example: Get storage controller health

```
$ curl -s https://<idrac_ip>/redfish/v1/Systems/System.Embedded.1/Storage/Controllers/RAID.Slot.8-1 -k -u root:password | python -m json.tool | jq .Name

"PERC H730 Adapter"
```

```
$ curl -s https://<idrac_ip>/redfish/v1/Systems/System.Embedded.1/Storage/Controllers/RAID.Slot.8-1 -k -u root:password | python

{
  "Health": "OK",
  "HealthRollUp": "OK"
}
```
Example: Get power consumption during last hour

```
{
  "AverageConsumedWatts": 152,
  "IntervalInMin": 60,
  "MaxConsumedWatts": 168,
  "MinConsumedWatts": 148
}
```
Example: Get system event logs
What else can Redfish do?

Retrieve Telemetry
- Basic server identification and asset information
- Health state
- Temperature sensors and fans
- Power consumption and thresholds

Discovery
- Service endpoint (network-based discovery)
- System topology (rack, chassis, server, node)

Basic I/O Infrastructure Data
- Host NIC MAC addresses for LOM devices
- Simple hard drive status / fault reporting

Security
- Session-based leveraging HTTPS

Common Management Actions
- Reboot / power cycle
- Change boot order
- Configure BMC network settings
- Manage user accounts

Access and Notification
- Serial console access via SSH
- Alert / event notification
- Event log access

More Coming Soon!
https://www.dmtf.org/standards/redfish
Redfish Roadmap

- Version 1 focused on servers. Expand over time to cover rest of IT infrastructure.
- Will add devices over time to cover new technologies (i.e. NVDIMMs, Multifunction Adapters)
- SNIA is developing Swordfish, which builds upon Redfish’s local storage management to address advanced storage devices.
- Open source efforts:
  - Client libraries (Python, Java, PowerShell)
  - Command line utility redfishtool (similar to ipmitool)
Scalable Out-of-band management

https://<idrac-ip>/redfish/v1/Managers/iDRAC.Embedded.1/Logs/Sel

Management Network

More than 5 servers

Server data
Ansible Overview
What is Ansible?

It’s a simple automation language that can perfectly describe an IT application infrastructure in Ansible Playbooks.

It’s an automation engine that runs Ansible Playbooks.

Ansible Tower is an enterprise framework for controlling, securing and managing your Ansible automation with a UI and restful API.
How Ansible works

ANSIBLE’S AUTOMATION ENGINE

PUBLIC / PRIVATE CLOUD

CMDB

HOSTS

NETWORKING

INVENTORY

API

MODULES

PLUGINS

ANSIBLE PLAYBOOK

USERS
More about Ansible

- Agentless → minimum footprint
- Make repetitive tasks easy
- Defines a desired state, so OK to run playbook more than once
- Easier to use than writing shell scripts
Ansible use cases

● Infrastructure Automation
  – Networking
  – Containers
  – Code Deployment
  – Server / Bare Metal

● Want more information? Visit www.ansible.com and docs.ansible.com
Ansible + Redfish + iDRAC together!
Scalable & Automated Out-of-band management

https://<idrac-ip>/redfish/v1/Managers/iDRAC.Embedded.1/Logs/Sel

Management
Network

More than 5 servers

Database
Server data
Key Lifecycle Management tasks

- Server Power On/Off; Reboot; Hard Reset
- Install BIOS, Configure BIOS, Reset to Default
- Configure iDRAC (CRUD operations):
  - User & Password Management
  - Certificate Management (import, export, delete)
  - Network Configuration
  - NTP and Time Zone settings
  - Storage (RAID, Virtual Disk, Physical Disks, PERC)
- System Inventory – H/W, Firmware, Sensor
- OS Deployment – remote file share, vMedia
- Import / Export SCP – remote file share, vMedia
- Backup and Restore
  - Server Profiles

- Upgrade using DSU (Dell Server Update) or DUEC (Dell Update Engine for Consoles)
  - Get list of available and applicable updates
  - Firmware Upgrade
  - BIOS Upgrade
  - OS Drivers Upgrade
- Job Management
  - Check JOB status
  - Create JOB
  - Delete JOB
  - Create JOB Queue
  - Delete JOB Queue
- Get Logs
  - Export LC logs
  - Export System Event Logs
Ansible module for iDRAC

- Manage your entire Dell EMC IT infrastructure (servers, routers, switches, storage) from your Ansible Controller.
- Automated monitoring, provisioning, firmware updates at scale.
- Open source, so you can write your own extensions as needed and contribute back to the community.
- Working with Red Hat to include as core Ansible module.
Implementation (playbook)

```yaml
---
- hosts: myhosts
  name: PowerEdge iDRAC
  gather_facts: False

  # Here we define global variables, but if some servers have different
  # credentials, then place these variables in /etc/ansible/hosts to override
  # for each host
  vars:
    idracuser: root
    idracpswd: calvin

  # Choices available:
  # Health          Get server health
  # Model           Get server model
  # BiosVersion     Get BIOS version
  # AssetTag        Get asset tag
  # Memory          Get system memory (GB)
  # CPU             Get CPU model
  # PowerRead       Get power consumed (watts)
  # Selog           Get SELogs
  tasks:
    - include: get_data.yml action={{item}}
      vars:
        json_file: /root/{{host}}-{{action}}
        with_items:
          - Health
          - Model
          - BiosVersion
          - AssetTag
          - Memory
          - CPU
          - PowerRead
          - Selog

---

idrac.yml

- name: "Get data: {{ action }}"
  local_action: >
    idrac choice={{ action }} idracuser={{ idracuser }}
    idracpswd={{ idracpswd }} idracip={{ idracip }}
    register: result

- name: Print inventory file
  local_action: copy content={{ result | to_nice_json }}
  dest={{ json_file }}

# Using simple JSON parser jq (https://stedolan.github.io/jq/) to further
# simplify output. Uncomment once you have it installed.
- local_action: shell jq .result {{ json_file }} > {{ json_file }},final
- local_action: file path={{ json_file }} state=absent
```
## Server inventory

Use case: collect inventory data, maintain in spreadsheet or database

<table>
<thead>
<tr>
<th>Server</th>
<th>iDRAC IP</th>
<th>Model</th>
<th>IP address</th>
<th>BIOS</th>
<th>CPU</th>
<th>Type</th>
<th>RAM</th>
<th>Service Tag</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>webserver-1</td>
<td>192.168.2.10</td>
<td>PowerEdge R630</td>
<td>10.0.1.30</td>
<td>2.3.4</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz</td>
<td></td>
<td>128</td>
<td>5WT4Q47</td>
<td>OK</td>
</tr>
<tr>
<td>webserver-2</td>
<td>192.168.2.11</td>
<td>PowerEdge R630</td>
<td>10.0.1.31</td>
<td>2.3.4</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz</td>
<td></td>
<td>128</td>
<td>5XR7Q32</td>
<td>OK</td>
</tr>
<tr>
<td>webserver-3</td>
<td>192.168.2.12</td>
<td>PowerEdge R630</td>
<td>10.0.1.33</td>
<td>2.3.2</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.40GHz</td>
<td></td>
<td>128</td>
<td>5XR7QYY</td>
<td>OK</td>
</tr>
<tr>
<td>appserver-1</td>
<td>192.168.2.13</td>
<td>PowerEdge R830</td>
<td>10.0.1.34</td>
<td>2.3.2</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.60GHz</td>
<td></td>
<td>512</td>
<td>5XR7QYY</td>
<td>OK</td>
</tr>
<tr>
<td>dbserver-1</td>
<td>192.168.3.10</td>
<td>PowerEdge R730</td>
<td>10.0.2.30</td>
<td>2.1.2</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.33GHz</td>
<td></td>
<td>256</td>
<td>5XR7Q67</td>
<td>OK</td>
</tr>
<tr>
<td>dbserver-2</td>
<td>192.168.3.11</td>
<td>PowerEdge R730</td>
<td>10.0.2.31</td>
<td>2.3.4</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.33GHz</td>
<td></td>
<td>256</td>
<td>5WT4Q37</td>
<td>OK</td>
</tr>
<tr>
<td>dbserver-3</td>
<td>192.168.3.12</td>
<td>PowerEdge R730</td>
<td>10.0.2.32</td>
<td>2.3.4</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.33GHz</td>
<td></td>
<td>256</td>
<td>5WR4Q12</td>
<td>OK</td>
</tr>
<tr>
<td>dbserver-4</td>
<td>192.168.3.13</td>
<td>PowerEdge R730</td>
<td>10.0.2.33</td>
<td>2.3.4</td>
<td>Intel(R) Xeon(R) CPU E5-2630 v3 @ 2.33GHz</td>
<td></td>
<td>256</td>
<td>5TT1Q44</td>
<td>OK</td>
</tr>
</tbody>
</table>
Source code repository

- https://github.com/dell/idrac-ansible-module

Will continue adding features in coming weeks and months. Pull requests and contributions are welcome and encouraged!
Resources

- iDRAC with Lifecycle Controller: http://dell.to/2qdBd0y
- Redfish API specification: https://www.dmtf.org/standards/redfish
- Dell EMC PowerEdge Redfish API Overview: http://dell.to/2odsH1p
- iDRAC Redfish API Reference Guide: http://dell.to/2oyjMTy
- Getting started with Ansible: http://docs.ansible.com/ansible/intro_getting_started.html
Q & A
Backup
## iDRAC operation APIs

<table>
<thead>
<tr>
<th>Dell Redfish API URLs</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>/redfish/v1/Managers</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/Actions/Manager.Reset</td>
<td>Used to perform iDRAC reset</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/NetworkProtocol</td>
<td>Reports information about iDRAC's network services. Includes Web server, SNMP, vMedia, Telnet, SSH, IPMI &amp; KVM.</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/SerialInterfaces</td>
<td>iDRAC BMC serial interface</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/SerialInterfaces/&lt;Serial-key&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/LogServices</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/LogServices/Sel</td>
<td>Access to server System Event Log</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/LogServices/Lclog</td>
<td>Access to Lifecycle Controller Log</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/LogServices/Sel/Actions/LogService.ClearLog</td>
<td>Used to clear LC Log</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia</td>
<td>Status of iDRAC virtual media</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/VirtualMedia/&lt;media-type&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces</td>
<td>iDRAC network interface</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/EthernetInterfaces/&lt;FQDD&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/AccountService</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/Accounts</td>
<td>iDRAC user accounts</td>
</tr>
<tr>
<td>/redfish/v1/Managers/iDRAC.Embedded.1/Accounts/&lt;Account-Id&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### Dell Redfish API URLs

<table>
<thead>
<tr>
<th>URL</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>/redfish/v1/Chassis</td>
<td>Top-level URI for server chassis</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1</td>
<td>Reports fan status for server and FX2 chassis</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Thermal</td>
<td>Reports thermal data for server and FX2 chassis</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Fans</td>
<td>Reports fan status for server and FX2 chassis</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Fans/&lt;Fan-FQDD&gt;</td>
<td>&lt;Fan-FQDD&gt; addresses each fan probe</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Temperatures</td>
<td>Reports thermal data for server and FX2 chassis</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Temperatures/&lt;Sensor-FQDD&gt;</td>
<td>&lt;Sensor-FQDD&gt; addresses each temperature probe</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Power</td>
<td>Power consumption and supply status</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Power/PowerControl</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Voltages</td>
<td>&lt;Voltage-FQDD&gt; addresses each voltage output</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Sensors/Voltages/&lt;Voltage-FQDD&gt;</td>
<td>&lt;Voltage-FQDD&gt; addresses each voltage output</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Power/PowerSupplies</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Power/PowerSupplies/&lt;PSU-FQDD&gt;</td>
<td>&lt;PSU-FQDD&gt; addresses each power supply</td>
</tr>
<tr>
<td>/redfish/v1/Chassis/System.Embedded.1/Power/Redundancy/&lt;PSRedundancy-FQDD&gt;</td>
<td>&lt;PSRedundancy-FQDD&gt; addresses power supply redundancy</td>
</tr>
</tbody>
</table>
## System status APIs

<table>
<thead>
<tr>
<th>Dell Redfish API URLs</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>/redfish/v1</td>
<td>Top-level API access</td>
</tr>
<tr>
<td>/redfish/v1/Systems</td>
<td>Server inventory and status information access</td>
</tr>
<tr>
<td>/redfish/v1/Systems/&lt;ServiceTag+nodeid&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/Actions/ComputerSystem.Reset</td>
<td>Server reset operation</td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/Processors</td>
<td>Details on CPUs</td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/Processors/&lt;Processor-FQDD&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/EthernetInterfaces</td>
<td>Reports NIC IP address, DHCP and DNS information.</td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/EthernetInterfaces/&lt;EthernetInterface-FQDD&gt;</td>
<td>Example &lt;EthernetInterface-FQDD&gt; = NIC.Embedded.1-1-1</td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/EthernetInterfaces/&lt;EthernetInterface-FQDD&gt;/Vlans</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/EthernetInterfaces/&lt;EthernetInterface-FQDD&gt;/Vlans/&lt;Vlan-FQDD&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/Storage/Controllers</td>
<td>Typical &lt;Controller-FQDD&gt;=RAID.Slot.N-1; describes details of controller, backplane, enclosure, attached drives</td>
</tr>
<tr>
<td>/redfish/v1/Systems/System.Embedded.1/Storage/Controllers/&lt;Controller-FQDD&gt;</td>
<td></td>
</tr>
</tbody>
</table>
### Dell Redfish API URLs

<table>
<thead>
<tr>
<th>URL</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>/redfish/v1/Registries/Messages/En</td>
<td>PowerEdge message registry</td>
</tr>
<tr>
<td>/redfish/v1/odata</td>
<td>Enables OData clients to navigate iDRAC Redfish resources</td>
</tr>
<tr>
<td>/redfish/v1/$metadata</td>
<td>Provides a metadata document describing the resources and collections that are available at the iDRAC Redfish service root URI</td>
</tr>
<tr>
<td>/redfish/v1/$metadata#&lt;Collection or a single resource&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/JSONSchemas</td>
<td>Schema descriptions for all supplied data</td>
</tr>
<tr>
<td>/redfish/v1/JSONSchemas/&lt;file&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/SessionService</td>
<td>Redfish session management</td>
</tr>
<tr>
<td>/redfish/v1/Sessions</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/Sessions/&lt;SessionId&gt;</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/TaskService</td>
<td>Redfish internal task management</td>
</tr>
<tr>
<td>/redfish/v1/TaskService/Actions/EventService.SubmitTestEvent</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/EventService</td>
<td>Redfish event management</td>
</tr>
<tr>
<td>/redfish/v1/EventSubscriptions</td>
<td></td>
</tr>
<tr>
<td>/redfish/v1/EventSubscriptions/&lt;Subscription ID&gt;</td>
<td></td>
</tr>
</tbody>
</table>
THANK YOU

plus.google.com/+RedHat
linkedin.com/company/red-hat
youtube.com/user/RedHatVideos
facebook.com/redhatinc
twitter.com/RedHatNews

#redhat #rhsummit
LEARN. NETWORK. EXPERIENCE OPEN SOURCE.