



Hewlett Packard
Enterprise

TECHNOLOGY AND SOLUTIONS SUMMIT 2020



François Donzé
March 2020





**Hewlett Packard
Enterprise**

REDFISH LIVE DEMO EXPLAINED



B624



PLEASE GIVE US YOUR FEEDBACK



iOS



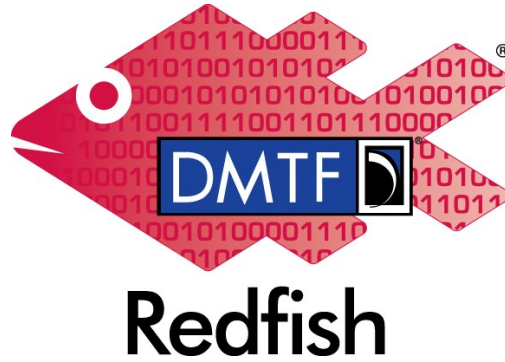
Android



CONFIDENTIAL DISCLOSURE AGREEMENT

- The information contained in this presentation is proprietary to Hewlett Packard enterprise and is offered in confidence, subject to the terms and conditions of a binding Confidential Disclosure Agreement (CDA)
- HPE requires customers and partners to have signed a CDA in order to view this training
- The information contained in this training is HPE confidential
- This presentation is NOT to be used as a 'leave behind' for customers and information may only be shared verbally with HPE external customers under NDA
- This presentation may be shared with Partners under NDA in hard-copy or electronic format for internal training purposes only
- Do not remove any classification labels, warnings or disclaimers on any slide or modify this presentation to change the classification level
- Do not remove this slide from the presentation
- HPE does not warrant or represent that it will introduce any product to which the information relates
- The information contained herein is subject to change without notice
- HPE makes no warranties regarding the accuracy of this information
- The only warranties for HPE products and services are set forth in the express warranty statements accompanying such products and services
- Nothing herein should be construed as constituting an additional warranty
- HPE shall not be liable for technical or editorial errors or omissions contained herein
- Strict adherence to the HPE Standards of Business Conduct regarding this classification level is critical

JUPYTER BASED SESSION



Basic use of the Redfish API

Version 0.28

What is Redfish ?

As per the [Redfish](https://www.dmtf.org/standards/redfish) (<https://www.dmtf.org/standards/redfish>) home page, "DMTF's Redfish® is a standard designed to deliver **simple and secure management** for converged, hybrid IT and the Software Defined Data Center (SDDC)".

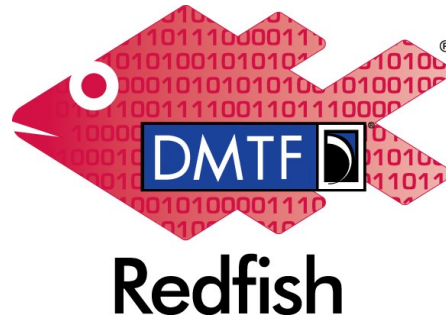
Workshop Goal

At the end of the workshop, you will be able to

- explain the basic architecture of the Redfish tree
- explain the session based authentication mechanism
- explain the Redfish Event Service
- explain the use of the OData `$expand` and `only` operators

Handouts

You can freely copy the Jupyter notebooks present in your environment, including their output in order to practice at your own place and pace.



Version 0.25

Redfish Session authentication and Event Service

Introduction

This Jupyter notebook explains the Redfish session authentication mechanism as well as the [Event Service](https://developer.hpe.com/blog/the-redfish-event-service) (<https://developer.hpe.com/blog/the-redfish-event-service>) using [Bash](https://www.gnu.org/software/bash/) (<https://www.gnu.org/software/bash/>) and the [cURL](https://curl.haxx.se/) (<https://curl.haxx.se/>) tool against an HPE iLO 5. For didactic reasons, commands presented in this notebook may be not optimized and don't follow the recommended [best practises](https://developer.hpe.com/blog/getting-started-with-the-redfish-api-part-2) (<https://developer.hpe.com/blog/getting-started-with-the-redfish-api-part-2>).

More details are in the [HPE Redfish API Reference document](https://hewlettpackard.github.io/ilo-rest-api-docs/ilo5/#introduction) (<https://hewlettpackard.github.io/ilo-rest-api-docs/ilo5/#introduction>).

Create environment variables

The following `bash` code defines environment variables (i.e. IP address, username, password....) depending on your student ID number stored in variable `$stud`. It creates as well several `.json` files containing various HTTP workloads required to POST or PATCH the managed iLO.

```

In [1]: # Create BMC related variables
iLO5_IP=172.16.50.99
iLO5_URI="https://${iLO5_IP}"
RemoteHost_IP=172.16.50.100

# iLO 5 Administrator credentials
iLO5_User="student"
iLO5_Passwd='P@ssw0rd!'

# EventReceiver
EventReceiverIP=16.31.87.40

# Minimum required Redfish headers
HeaderODataVersion="OData-Version: 4.0"
HeaderContentType="Content-Type: application/json"

# Data files
ResponseHeaders="ResponseHeaders.txt" # Used to hold HTTP response headers
SessionData="./CreateSession-data.json" # Body/Workload used to create the Redfish session
EventSubscription="./EventSubscription-data.json" # Body/Workload used to subscribe to events
CpuThresholds="./CpuThresholds-data.json" # Body/Workload used to set CPU Utilization Thresholds
TestEvent="./TestEvent-data.json"

cat > ${SessionData} << __EOF__
{
    "UserName": "${iLO5_User}",
    "Password": "${iLO5_Passwd}"
}
__EOF__

cat > ${EventSubscription} << __EOF__
{
    "Destination": "https://${EventReceiverIP}/RedfishEvents/EventReceiver.php",
    "EventTypes": [
        "StatusChange",
        "ResourceUpdated",
        "ResourceAdded",
        "ResourceRemoved",
        "Alert"
    ],
    "Context": "Public"
}
__EOF__

cat > ${TestEvent} << __EOF__
{
    "EventType": "ResourceAdded",
    "EventID": "myEventId",
    "EventTimestamp": "top-of-the-hour",
    "Severity": "OK",
    "Message": "This is a test message",
    "MessageID": "iLOEvents.0.9.ResourceStatusChanged",
    "MessageArgs": [ "arg0", "arg1" ],
    "OriginOfCondition": "/redfish/v1/Chassis/1/FooBar"
}
__EOF__

```



```
EventReceiver is reachable via Ping

Ncat: Version 7.50 ( https://nmap.org/ncat )
Ncat: Connected to 16.31.87.40:443.
Ncat: 0 bytes sent, 0 bytes received in 0.02 seconds.
EventReceiver listens to HTTPS requests

iLO 5 is reachable via Ping

RemoteHost is reachable via Ping
```

Accessing the Redfish RootService

Accessing the Redfish root service does not need any authentication

List the Redfish version(s) and their location(s) implemented in the managed BMC (iLO 5)

```
In [2]: echo 'List the Redfish version(s) implemented in the managed BMC:'

curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
      --header "$HeaderContentType" --header "$HeaderODataVersion" \
      --request GET "${iLO5_URI}/redfish | jq
```

```
List the Redfish version(s) implemented in the managed BMC:
{
  "v1": "/redfish/v1/"
}
```

List the content of the root service

```
In [3]: echo 'List the Redfish services available in Redfish v1:'

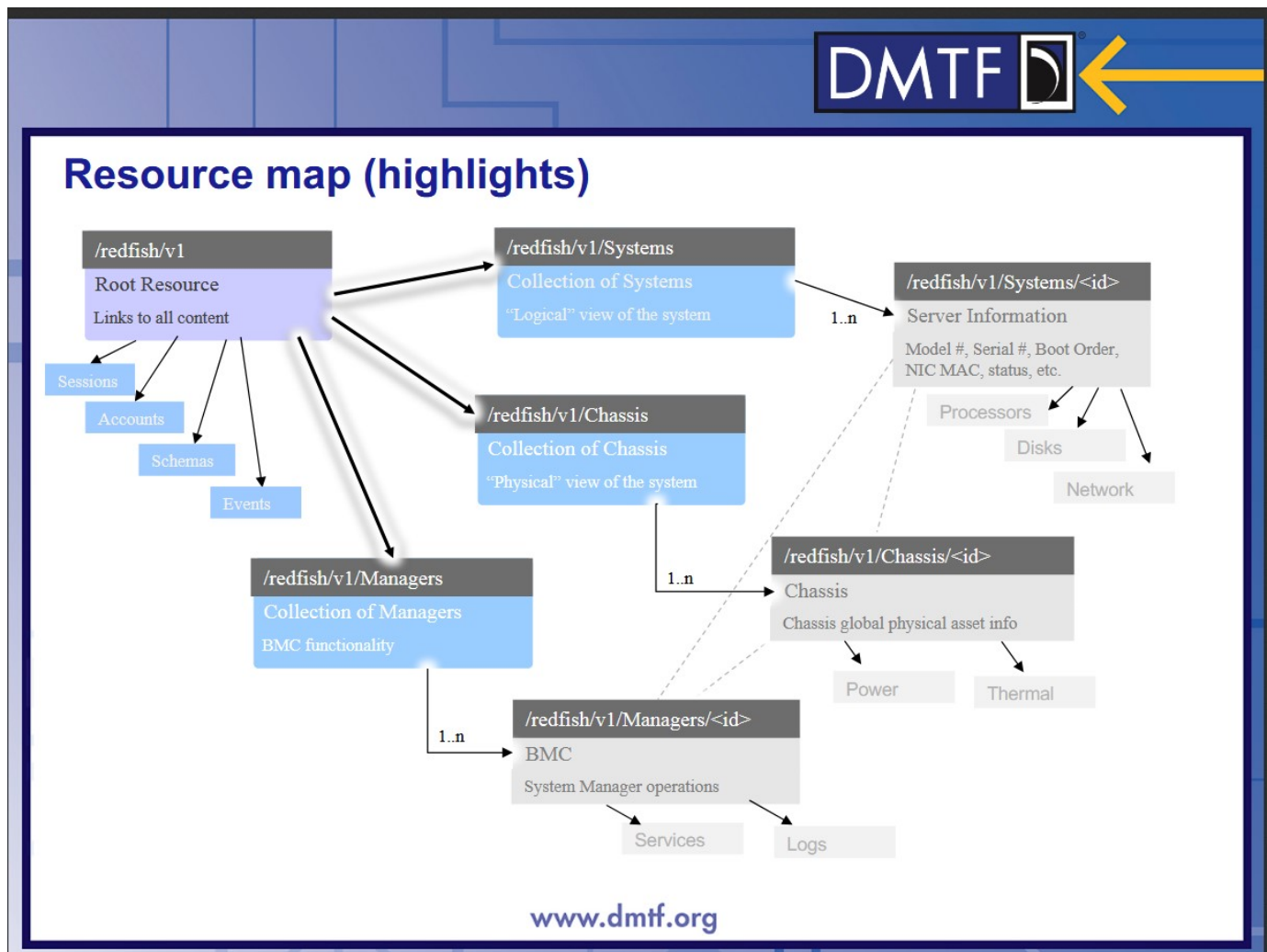
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--request GET "${iLO5_URI}/redfish/v1 | jq
```

List the Redfish services available in Redfish v1:

```
{
  "@odata.context": "/redfish/v1/$metadata#ServiceRoot.ServiceRoot",
  "@odata.etag": "W/\"81F4ACAA\"",
  "@odata.id": "/redfish/v1",
  "@odata.type": "#ServiceRoot.v1_5_1.ServiceRoot",
  "Id": "RootService",
  "AccountService": {
    "@odata.id": "/redfish/v1/AccountService"
  },
  "Chassis": {
    "@odata.id": "/redfish/v1/Chassis"
  },
  "EventService": {
    "@odata.id": "/redfish/v1/EventService"
  },
  "JsonSchemas": {
    "@odata.id": "/redfish/v1/JsonSchemas"
  },
  "Links": {
    "Sessions": {
      "@odata.id": "/redfish/v1/SessionService/Sessions"
    }
  },
  "Managers": {
    "@odata.id": "/redfish/v1/Managers"
  },
  "Name": "HPE RESTful Root Service",
  "Oem": {
    "Hpe": {
      "@odata.context": "/redfish/v1/$metadata#HpeILOServiceExt.HpeILOServiceExt",
      "@odata.type": "#HpeILOServiceExt.v2_3_0.HpeILOServiceExt",
      "Links": {
        "ResourceDirectory": {
          "@odata.id": "/redfish/v1/ResourceDirectory"
        }
      },
      "Manager": [
        {
          "DefaultLanguage": "en",
          "FQDN": "ilo-hst360g10.b172.local",
          "HostName": "ilo-hst360g10",
          "Languages": [
            {
              "Language": "en",
              "TranslationName": "English",
              "Version": "2.10"
            }
          ],
          "ManagerFirmwareVersion": "2.10",
          "ManagerType": "iLO 5",
          "Status": {
            "Health": "OK"
          }
        }
      ],
      "Moniker": {
        "ADVPLIC": "iLO Advanced",
        "BMC": "iLO",
        "BSYS": "BladeSystem",
        "CLASS": "Baseboard Management Controller",
        "FEDGRP": "DEFAULT",
        "IPROV": "Intelligent Provisioning",
```


Redfish Resource map

Read the DMTF introduction to the [Redfish Architecture](https://www.dmtf.org/sites/default/files/Redfish_School-Redfish_Architecture_Sept_2016_0.pdf) (https://www.dmtf.org/sites/default/files/Redfish_School-Redfish_Architecture_Sept_2016_0.pdf).



Create the Redfish session

Redfish allows basic authentication and session authentication. With basic authentication you need to supply the required credentials at each and every HTTP request. Session oriented authentication is achieved by requesting a `Token` that will be sent in the headers of all requests until the removal of the session.

To get this `Token`, POST a session request with the remote BMC credentials in its body. The `Token` as well as the `session location` will be in the headers of the response.

In [4]: `echo 'Create iLO 5 Session'`

```
curl --dump-header $ResponseHeaders \  
--insecure --no-proxy "localhost, 127.0.0.1" --silent \  
--header "$HeaderContentType" --header "$HeaderODataVersion" \  
--request POST --data "@$SessionData" \  
${iLO5_URI}/redfish/v1/SessionService/Sessions | jq  
  
Token=$(awk '/X-Auth-Token/ {print $NF}' $ResponseHeaders | tr -d '\r')  
SessionLocation="${iLO5_URI}"$(awk '/^Loca.*Se/ {gsub("https://.*/red", "/red", $NF);  
print $NF}' $ResponseHeaders | tr -d '\r')  
  
echo  
echo "Token: $Token"  
echo -e "Session Location: $SessionLocation\n"
```

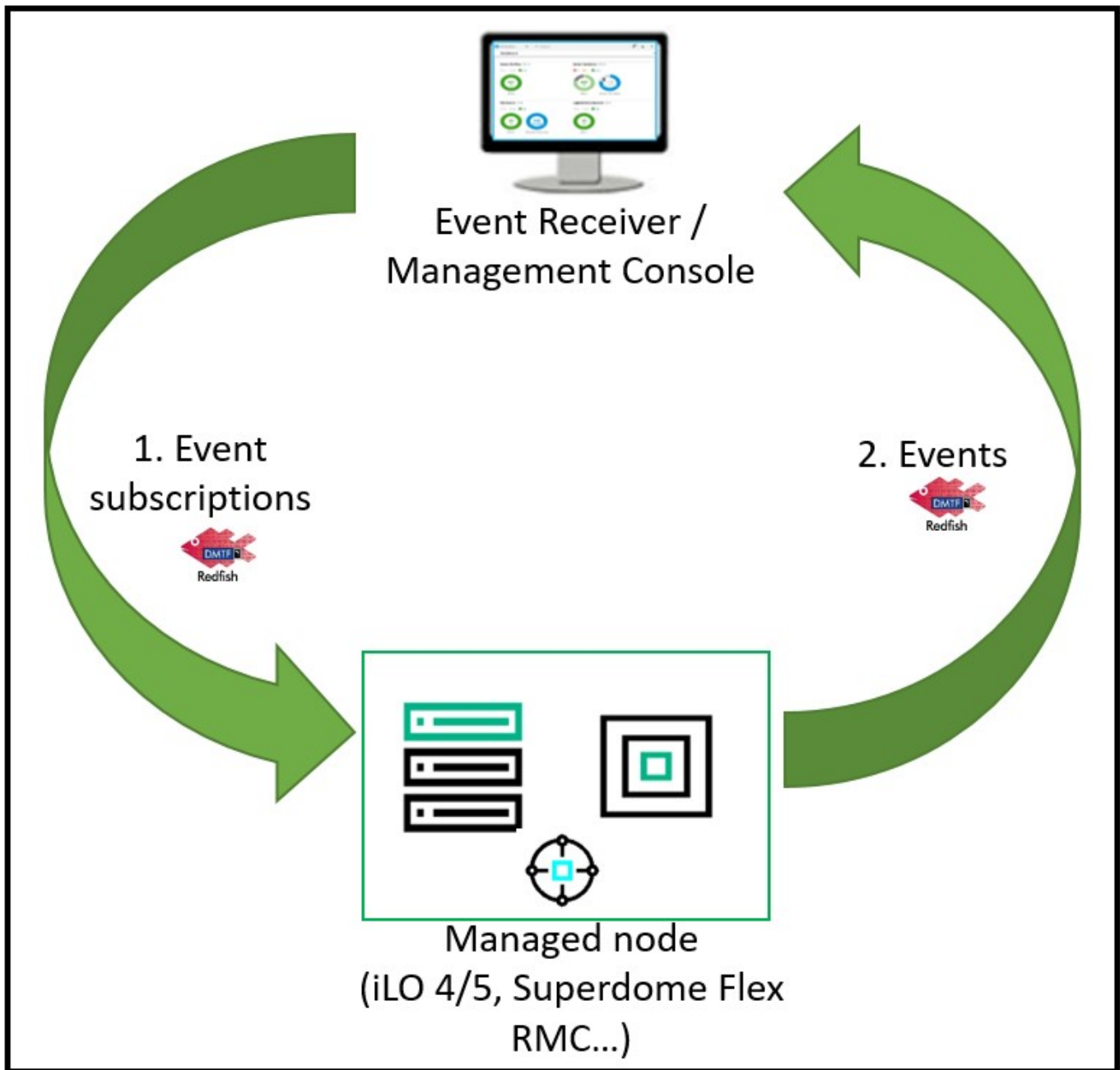
Create iLO 5 Session

```
{  
  "@odata.context": "/redfish/v1/$metadata#Session.Session",  
  "@odata.etag": "W/\\"4F32B6D5\\\"",  
  "@odata.id": "/redfish/v1/SessionService/Sessions/student000000005e66110ee0c49  
ba5",  
  "@odata.type": "#Session.v1_0_0.Session",  
  "Id": "student000000005e66110ee0c49ba5",  
  "Description": "Manager User Session",  
  "Name": "User Session",  
  "Oem": {  
    "Hpe": {  
      "@odata.context": "/redfish/v1/$metadata#HpeILOSession.HpeILOSession",  
      "@odata.type": "#HpeILOSession.v2_1_0.HpeILOSession",  
      "AccessTime": "2020-03-09T09:49:02Z",  
      "LoginTime": "2020-03-09T09:49:02Z",  
      "MySession": false,  
      "UserExpires": "2020-03-09T10:19:02Z",  
      "UserIP": "172.22.101.1",  
      "UserTag": "REST",  
      "UserType": "Local"  
    }  
  },  
  "UserName": "student"  
}
```

Token: 2f2c12a724b5f023cce354dc2733b97b

Session Location: https://172.16.50.99/redfish/v1/SessionService/Sessions/studen
t000000005e66110ee0c49ba5

Event subscription



Goal of the exercise

Generate an alert when the CPU utilization of the managed server **decreases** below a specified threshold during more than a defined dwell time.

Steps

- Generate CPU load on the managed system
- Subscribe to Redfish events by providing the IP of the Event Receiver
- Verify the Event Receiver is working properly
- Subscribe and review the subscription
- Kill the load and wait until event reaches the Event Receiver

Get event subscription collection

The following command retrieves the event subscription collection using the `Token` obtained above.

```
In [5]: echo "Retrieve Event Subscription collection:"
curl --insecure --silent --no-proxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET "${iLO5_URI}/redfish/v1/EventService/Subscriptions" | jq

Retrieve Event Subscription collection:
{
  "@odata.context": "/redfish/v1/$metadata#EventDestinationCollection.EventDesti
nationCollection",
  "@odata.etag": "W/\"75983E8D\"",
  "@odata.id": "/redfish/v1/EventService/Subscriptions",
  "@odata.type": "#EventDestinationCollection.EventDestinationCollection",
  "Description": "iLO User Event Subscriptions",
  "Name": "EventSubscriptions",
  "Members": [],
  "Members@odata.count": 0
}
```

Remove event subscription if any

```
In [6]: Subscriptions=$(curl --insecure --silent --no-proxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET "${iLO5_URI}/redfish/v1/EventService/Subscriptions" | jq -r '.Members[] | ."@odata.id"')

for s in $Subscriptions ; do
curl --insecure --silent --no-proxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request DELETE "${iLO5_URI}${s}"
done
```

Prepare Event Receiver

```
In [7]: echo "EventReceiver source file"
ssh $EventReceiverIP "cat /opt/hpe/RedfishEventService/EventReceiver.php"

echo "Cleanup the Event Receiver log file:"
ssh $EventReceiverIP "cat /dev/null > /opt/hpe/RedfishEventService/Redfish_events.txt"
```

```
EventReceiver source file
<?php
// Version 0.9999
/** This PHP script receives RESTful POST events from an iLO or a Superdome Flex
RMC.
*   It reformats the JSON message with indentations and sends
*   it to a file in the current directory
**/

/*
* The JSON format functions.php comes from:
* https://github.com/GerHobbelt/nicejson-php
*/
include 'functions.php';

// iLO events will be written to $out_file
$out_file = "Redfish_events.txt" ;

// Read the Content of the POST message:
$body = file_get_contents("php://input");

// Read the headers values:
$headers = getallheaders() ;

// Get IP address of managed node
$IP_MANAGED = getenv ('REMOTE_ADDR') ;

// Write IP_MANAGED in $outfile:
file_put_contents($out_file, "IP Address of Managed node: $IP_MANAGED \n", FILE_
APPEND) ;

// Display headers and values
foreach ($headers as $header => $value) {
    file_put_contents($out_file, "$header: $value \n", FILE_APPEND) ;
}

//Insert new line to separate headers from body
file_put_contents($out_file, "\n", FILE_APPEND);

// Format message in nice and human readable format
file_put_contents($out_file, json_format($body) . "\n\n", FILE_APPEND);

?>
Cleanup the Event Receiver log file:
```

Subscribe to events

```
In [8]: echo "POST a Subscription"

curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request POST --data "@${EventSubscription}" \
${iLO5_URI}/redfish/v1/EventService/Subscriptions | jq

POST a Subscription
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "Base.1.4.Created"
      }
    ]
  }
}
```

Test Event subscription

```
In [9]: echo "Generate a fake event"

curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request POST --data "@${TestEvent}" \
${iLO5_URI}/redfish/v1/EventService/Actions/EventService.SubmitTestEvent | jq

Generate a fake event
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "Base.1.4.Success"
      }
    ]
  }
}
```

Verify test event reached the Event Receiver


```
In [10]: ssh $EventReceiverIP "tail -30 /opt/hpe/RedfishEventService/Redfish_events.txt"
```

IP Address of Managed node: 15.186.54.125

Host: 16.31.87.40

Transfer-Encoding: chunked

Content-Type: application/json

Cache-Control: no-cache

Date: Mon, 09 Mar 2020 09:49:58 GMT

Connection: keep-alive

```
{
  "@odata.context": "/redfish/v1/$metadata#Event.Event",
  "@odata.type": "#Event.v1_0_0.Event",
  "Events": [
    {
      "EventId": "myEventId",
      "EventTimestamp": "1970-01-01T00:00:00Z",
      "EventType": "ResourceAdded",
      "Message": "This is a test message",
      "MessageArgs": [
        "arg0",
        "arg1"
      ],
      "MessageId": "iLOEvents.0.9.ResourceStatusChanged",
      "OriginOfCondition": "/redfish/v1/Chassis/1/FooBar",
      "Severity": "OK"
    }
  ],
  "Name": "Events"
}
```

Read CPU Utilization

```
In [15]: echo "Read CPU Utilization:"
curl --insecure --silent --noproxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET "${iLO5_URI}/redfish/v1/TelemetryService/MetricReports/CPUUtilCus
tom1 | jq -r '[".MetricValues]"'
```

Read CPU Utilization:

```
[
  [
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:44:12Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:44:32Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:44:52Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:45:12Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:45:32Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:45:52Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:46:12Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
```

Start fake load on RemoteHost

```
In [12]: ssh $RemoteHost_IP "/usr/kits/fake_loader.sh " &  
[1] 26212
```

Read CPU Utilization

```
In [16]: echo "Read CPU Utilization:"
curl --insecure --silent --noproxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET "${iLO5_URI}/redfish/v1/TelemetryService/MetricReports/CPUUtilCus
tom1 | jq -r '[".MetricValues]"'
```


Read CPU Utilization:

```
[
  [
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:45:32Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:45:52Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:46:12Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:46:32Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:46:52Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:47:12Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
      "MetricValue": "0",
      "Timestamp": "2020-03-09T09:47:32Z"
    },
    {
      "MetricDefinition": {
        "@odata.id": "/redfish/v1/TelemetryService/MetricDefinitions/CPUUtil"
      },
      "MetricId": "CPUUtil",
```

Retrieve CPU Utilization Thresholds

```
In [17]: echo "CPU Utilization Thresholds:"
curl --dump-header $ResponseHeaders \
--insecure --silent --noproxy "localhost, 127.0.0.1" \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET ${iLO5_URI}/redfish/v1/TelemetryService/Triggers/CPUUtilTriggers
| jq

echo -e "\n\nResponse Headers:"
grep Allow $ResponseHeaders
echo
```

```
CPU Utilization Thresholds:
{
  "@odata.context": "/redfish/v1/$metadata#Triggers.Triggers",
  "@odata.etag": "W/\"BFAAE441\"",
  "@odata.id": "/redfish/v1/TelemetryService/Triggers/CPUUtilTriggers",
  "@odata.type": "#Triggers.v1_0_0.Triggers",
  "Id": "CPUUtilTriggers",
  "Description": "Triggers for CPU Utilization",
  "MetricProperties": [
    "/redfish/v1/Systems/1#SystemUsage/CPUUtil"
  ],
  "MetricType": "Numeric",
  "Name": "Triggers for CPU Utilization",
  "NumericThresholds": {
    "LowerCritical": {
      "Activation": "Decreasing",
      "DwellTime": "PT0S",
      "Reading": 0
    },
    "UpperCritical": {
      "Activation": "Increasing",
      "DwellTime": "PT0S",
      "Reading": 0
    }
  },
  "Status": {
    "Health": "OK",
    "State": "Enabled"
  },
  "TriggerActions": [
    "LogToLogService"
  ]
}
```

```
Response Headers:
Allow: GET, HEAD, PATCH
```

Modify CPU Utilization thresholds (part of the Telemetry Service)

```
In [18]: echo "Patching CPU Utilization Thresholds"
         curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
           --header "$HeaderContentType" --header "$HeaderODataVersion" \
           --header "X-Auth-Token: $Token" \
           --request PATCH --data "@$CpuThresholds" \
           ${iLO5_URI}/redfish/v1/TelemetryService/Triggers/CPUUtilTriggers | jq
```

Patching CPU Utilization Thresholds

```
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "Base.1.4.Success"
      }
    ]
  }
}
```

Kill load on managed system and wait for event

```

In [23]: ssh $RemoteHost_IP "pkill fake_loader"
        sleep 20

        ssh $EventReceiverIP "tail -30 /opt/hpe/RedfishEventService/Redfish_events.txt"

        "Events": [
            {
                "EventId": "91f31abd-0f13-bafc-de4f-825dd1d891ec",
                "EventTimestamp": "2020-03-09T09:58:12Z",
                "EventType": "Alert",
                "MemberId": "0",
                "MessageArgs": [
                    "CPU Utilization"
                ],
                "MessageId": "iLOEvents.2.1.MetricValueBelowLowerThresho
ld",
                "Oem": {
                    "Hpe": {
                        "@odata.context": "/redfish/v1/$metadata
#HpeEvent.HpeEvent",
                        "@odata.type": "#HpeEvent.v2_1_0.HpeEven
t",
                        "CorrelatedEventNumber": 1048,
                        "CorrelatedEventTimeStamp": "2020-03-09T
09:58:12Z",
                        "CorrelatedEventType": "Hpe-IML",
                        "CorrelatedIndications": [
                            "HP:SNMP:1.3.6.1.4.1.232:6:2018:
3755959344"
                        ],
                        "Resource": "/redfish/v1/TelemetryServic
e/Triggers/CPUUtilTriggers"
                    }
                },
                "OriginOfCondition": "/redfish/v1/TelemetryService/Trigg
ers/CPUUtilTriggers",
                "Severity": "Warning"
            }
        ],
        "Name": "Events"
    }

```

Remove Event subscription

```
In [21]: echo "Retrieve Event subscription URIs: "

EventLocations=$(curl --insecure --silent --noproxy "localhost, 127.0.0.1" \
    --header "$HeaderContentType" --header "$HeaderODataVersion" \
    --header "X-Auth-Token: $Token" \
    --request GET ${iLO5_URI}/redfish/v1/EventService/Subscriptions | jq -r '.Members[] | ."@odata.id"')

echo -e "Event Locations : ${EventLocations}\n"

echo "Remove Event(s)"
for s in $EventLocations ; do
    echo "Processing $s"
    curl --insecure --silent --noproxy "localhost, 127.0.0.1" \
        --header "$HeaderContentType" --header "$HeaderODataVersion" \
        --header "X-Auth-Token: $Token" \
        --request DELETE ${iLO5_URI}${s} | jq
done

echo -e "\nVerify event(s) have been removed:"
curl --insecure --silent --noproxy "localhost, 127.0.0.1" \
    --header "$HeaderContentType" --header "$HeaderODataVersion" \
    --header "X-Auth-Token: $Token" \
    --request GET ${iLO5_URI}/redfish/v1/EventService/Subscriptions | jq
```

Retrieve Event subscription URIs:

Event Locations : /redfish/v1/EventService/Subscriptions/26

Remove Event(s)

Processing /redfish/v1/EventService/Subscriptions/26

```
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "iLO.2.13.EventSubscriptionRemoved"
      }
    ]
  }
}
```

Verify event(s) have been removed:

```
{
  "@odata.context": "/redfish/v1/$metadata#EventDestinationCollection.EventDestinationCollection",
  "@odata.etag": "W/\"75983E8D\"",
  "@odata.id": "/redfish/v1/EventService/Subscriptions",
  "@odata.type": "#EventDestinationCollection.EventDestinationCollection",
  "Description": "iLO User Event Subscriptions",
  "Name": "EventSubscriptions",
  "Members": [],
  "Members@odata.count": 0
}
```

Reset CPU thresholds

```
In [22]: cat > ${CpuThresholds} << __EOF__
{
  "NumericThresholds": {
    "LowerCritical": {
      "DwellTime": "PT0S",
      "Reading": 0
    },
    "UpperCritical": {
      "DwellTime": "PT0S",
      "Reading": 0
    }
  }
}
__EOF__

echo "Reseting CPU Utilization Thresholds"
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request PATCH --data "@${CpuThresholds}" \
${iLO5_URI}/redfish/v1/TelemetryService/Triggers/CPUUtilTriggers | jq
```

```
Reseting CPU Utilization Thresholds
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "Base.1.4.Success"
      }
    ]
  }
}
```

Delete sessions

It is extremely important to delete Redfish sessions to avoid reaching the maximum number of opened sessions in a BMC, preventing any access to it. Read this [article \(https://developer.hpe.com/blog/managing-ilo-sessions-with-redfish\)](https://developer.hpe.com/blog/managing-ilo-sessions-with-redfish) for more detail.

```
In [24]: echo "Body response of a session deletion:"

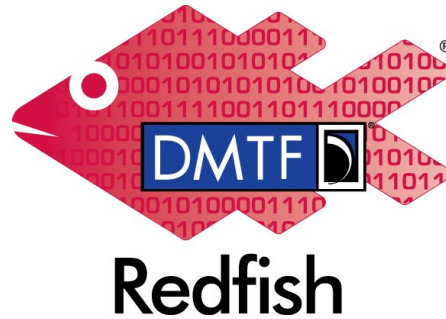
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request DELETE $SessionLocation | jq
```

```
Body response of a session deletion:
{
  "error": {
    "code": "iLO.0.10.ExtendedInfo",
    "message": "See @Message.ExtendedInfo for more information.",
    "@Message.ExtendedInfo": [
      {
        "MessageId": "Base.1.4.Success"
      }
    ]
  }
}
```


Wrap up

In this notebook you performed the following actions:

- Discover the standard Redfish tree
- Create a Redfish session
- Subscribe to events
- Generate a fake event
- Modify a telemetry thresholds
- Generate a telemetry alert



Version 0.17

OData operations

Introduction

This Jupyter notebook demonstrates the `$expand` and `only` Odata queries using [Bash](https://www.gnu.org/software/bash/) (<https://www.gnu.org/software/bash/>) and the [cURL](https://curl.haxx.se/) (<https://curl.haxx.se/>) tool against an HPE iLO 5. For didactic reasons, commands presented in this notebook may be not optimized and don't follow the recommended [best practises](https://developer.hpe.com/blog/getting-started-with-the-redfish-api-part-2) (<https://developer.hpe.com/blog/getting-started-with-the-redfish-api-part-2>).

[OData](https://www.odata.org/) (<https://www.odata.org/>):

An open protocol to allow the creation and consumption of queryable and interoperable RESTful APIs in a simple and standard way.

More details are in the [HPE Redfish API Reference document](https://hewlettpackard.github.io/ilo-rest-api-docs/ilo5/#introduction) (<https://hewlettpackard.github.io/ilo-rest-api-docs/ilo5/#introduction>).

`$expand` query parameter

iLO 5 `$expand` Support

Using the rules above, iLO 5 supports `$expand` in this way:

`$expand` is applicable to HTTP GET only.

`$expand=.`, `$expand=*`, `$expand=($levels=n)` result in the same behavior: * Expands all links in both root and Oem/Hpe sections not inside the `Links` sections. * Levels is always interpreted as 1, regardless of n. This is to avoid the potential for expanding recursively for interlinked resources. * The `Links` section is never expanded. This is to avoid expanding the Chassis and Manager related links on GET operations to System.

NOTES: * The root resource at `/redfish/v1/` is available without authentication and has navigational links that can be expanded. An `$expand` request does not result in expansion unless valid authentication credentials are supplied. * There might be other links that do not support expand.

only Query parameter

iLO 5 “only” Query Parameter

iLO 5 1.40 and later supports the `only` query parameter documented in the Redfish API specification. This query parameter is ignored except on collections with only one member. Examples include the `ComputerSystemCollection`, `ChassisCollection`, and `ManagerCollection`.

Create environment variables and session

The following `bash` code defines environment variables (i.e. IP address, username, password....) depending on your student ID number stored in variable `$Stud`. It creates as well several `.json` files containing various HTTP workloads required to POST or PATCH the managed iLO.

```
In [1]: # Create BMC related variables
iLO5_IP=172.16.50.99
iLO5_URI="https://${iLO5_IP}"
RemoteHost_IP=172.16.50.100

# iLO 5 Administrator credentials
iLO5_User="student"
iLO5_Passwd='P@ssw0rd!'

# EventReceiver
#EventReceiverIP=192.168.0.99
EventReceiverIP=16.31.87.40

# Minimum required Redfish headers
HeaderODataVersion="OData-Version: 4.0"
HeaderContentType="Content-Type: application/json"

# Data files
ResponseHeaders="ResponseHeaders.txt" # Used to hold HTTP response headers
SessionData="./CreateSession-data.json" # Body/Workload used to create the Redfish session

cat > ${SessionData} << __EOF__
{
    "UserName": "${iLO5_User}",
    "Password": "${iLO5_Passwd}"
}
__EOF__
```

Create the Redfish session

Redfish allows basic authentication and session authentication. With basic authentication you need to supply the required credentials at each and every HTTP request. Session oriented authentication is achieved by requesting a `Token` that will be sent in the headers of all requests until the removal of the session.

To get this `Token`, POST a session request with the remote BMC credentials in its body. The `Token` as well as the session location will be in the headers of the response.

```
In [2]: echo 'Create iLO 5 Session'

curl --dump-header $ResponseHeaders \
      --insecure --no-proxy "localhost, 127.0.0.1" --silent \
      --header "$HeaderContentType" --header "$HeaderODataVersion" \
      --request POST --data "@$SessionData" \
      ${iLO5_URI}/redfish/v1/SessionService/Sessions > /dev/null 2>&1

Token=$(awk '/X-Auth-Token/ {print $NF}' $ResponseHeaders | tr -d '\r')
SessionLocation="${iLO5_URI}${(awk '/^Loca.*Se/ {gsub("https://.*/red", "/red", $NF); print $NF}' $ResponseHeaders | tr -d '\r')}"

echo "Token: $Token"
echo -e "Session Location: $SessionLocation\n"

Create iLO 5 Session
Token: 23f1292c002ebef68e5b7976d0a44858
Session Location: https://172.16.50.99/redfish/v1/SessionService/Sessions/student000000005e661416e72b020c
```

Retrieve iLO 5 supported OData protocol features

The following command retrieves the OData supported protocol features.

```
In [3]: echo "Retrieve supported OData features:"

curl --insecure --silent --no-proxy "localhost, 127.0.0.1" \
      --header "$HeaderContentType" --header "$HeaderODataVersion" \
      --header "X-Auth-Token: $Token" \
      --request GET ${iLO5_URI}/redfish/v1 | jq '.ProtocolFeaturesSupported'

Retrieve supported OData features:
{
  "ExpandQuery": {
    "ExpandAll": false,
    "Levels": true,
    "Links": false,
    "MaxLevels": 1,
    "NoLinks": true
  },
  "FilterQuery": true,
  "OnlyMemberQuery": true,
  "SelectQuery": false
}
```

GET collection without \$expand

In [4]: `echo "GET FirmwareInventory collection"`

```
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \  
--header "$HeaderContentType" --header "$HeaderODataVersion" \  
--header "X-Auth-Token: $Token" \  
--request GET ${iLO5_URI}/redfish/v1/UpdateService/FirmwareInventory | jq
```

GET FirmwareInventory collection

```
{
  "@odata.context": "/redfish/v1/$metadata#SoftwareInventoryCollection.SoftwareInventoryCollection",
  "@odata.etag": "W/\"D821FF8B\"",
  "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory",
  "@odata.type": "#SoftwareInventoryCollection.SoftwareInventoryCollection",
  "Description": "Firmware Inventory Collection",
  "Name": "Firmware Inventory Collection",
  "Members": [
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/1"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/2"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/3"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/4"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/5"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/6"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/7"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/8"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/9"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/10"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/11"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/12"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/13"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/14"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/15"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/16"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/17"
    },
    {
      "@odata.id": "/redfish/v1/UpdateService/FirmwareInventory/18"
    },
  ],
}
```


Expand collection with `$expand`

```
In [5]: echo "Retrieve ALL Firmware versions with a single GET"

curl --insecure --noproxy "localhost, 127.0.0.1" --silent \
      --header "$HeaderContentType" --header "$HeaderODataVersion" \
      --header "X-Auth-Token: $Token" \
      --request GET "${iLO5_URI}"/redfish/v1/UpdateService/FirmwareInventory?$expand
=. ' | \
      jq '.Members | .[] | {Name: .Name, Description: .Description, Version: .Version}'
```

Retrieve ALL Firmware versions with a single GET

```
{
  "Name": "iLO 5",
  "Description": "SystemBMC",
  "Version": "2.10 Oct 30 2019"
}
{
  "Name": "System ROM",
  "Description": "SystemRomActive",
  "Version": "U32 v2.22 (11/13/2019)"
}
{
  "Name": "Intelligent Platform Abstraction Data",
  "Description": "PlatformDefinitionTable",
  "Version": "9.8.0 Build 15"
}
{
  "Name": "System Programmable Logic Device",
  "Description": "SystemProgrammableLogicDevice",
  "Version": "0x2A"
}
{
  "Name": "Power Management Controller Firmware",
  "Description": "PowerManagementController",
  "Version": "1.0.7"
}
{
  "Name": "NVMe Backplane Firmware",
  "Description": "NVMeBackplane",
  "Version": "1.20"
}
{
  "Name": "Power Supply Firmware",
  "Description": "PowerSupplies",
  "Version": "1.00"
}
{
  "Name": "Innovation Engine (IE) Firmware",
  "Description": "InnovationEngineFirmware",
  "Version": "0.2.1.2"
}
{
  "Name": "Server Platform Services (SPS) Firmware",
  "Description": "SPSFirmwareVersionData",
  "Version": "4.1.4.339"
}
{
  "Name": "Smart Storage Energy Pack",
  "Description": "SmartStorageEnergyPack",
  "Version": "2.1"
}
{
  "Name": "Redundant System ROM",
  "Description": "SystemRomBackup",
  "Version": "U32 v2.22 (11/13/2019)"
}
{
  "Name": "Intelligent Provisioning",
  "Description": "Intelligent Provisioning",
  "Version": "3.30.213"
}
{
  "Name": "Power Management Controller FW Bootloader",
  "Description": "PowerManagementControllerBootloader",
```

Expand collection only if it contains one element

Without the `only` query parameter

```
In [6]: echo "Chassis collection"
```

```
curl --insecure --no-proxy "localhost, 127.0.0.1" --silent \
--header "$HeaderContentType" --header "$HeaderODataVersion" \
--header "X-Auth-Token: $Token" \
--request GET "${iLO5_URI}"/redfish/v1/Chassis' | jq
```

Chassis collection

```
{
  "@odata.context": "/redfish/v1/$metadata#ChassisCollection.ChassisCollection",
  "@odata.etag": "W/\"AA6D42B0\"",
  "@odata.id": "/redfish/v1/Chassis",
  "@odata.type": "#ChassisCollection.ChassisCollection",
  "Description": "Computer System Chassis View",
  "Name": "Computer System Chassis",
  "Members": [
    {
      "@odata.id": "/redfish/v1/Chassis/1"
    }
  ],
  "Members@odata.count": 1
}
```

With the `only` query parameter

In [7]: `echo "Chassis collection"`

```
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \  
--header "$HeaderContentType" --header "$HeaderODataVersion" \  
--header "X-Auth-Token: $Token" \  
--request GET "${iLO5_URI}"/redfish/v1/Chassis?only' | jq
```

Chassis collection

```
{
  "@odata.context": "/redfish/v1/$metadata#Chassis.Chassis",
  "@odata.etag": "W/\"0942D38B\"",
  "@odata.id": "/redfish/v1/Chassis/1",
  "@odata.type": "#Chassis.v1_6_0.Chassis",
  "Id": "1",
  "AssetTag": "",
  "ChassisType": "RackMount",
  "IndicatorLED": "Off",
  "Links": {
    "ManagedBy": [
      {
        "@odata.id": "/redfish/v1/Managers/1"
      }
    ],
    "ComputerSystems": [
      {
        "@odata.id": "/redfish/v1/Systems/1"
      }
    ]
  },
  "Manufacturer": "HPE",
  "Model": "ProLiant DL360 Gen10",
  "Name": "Computer System Chassis",
  "NetworkAdapters": {
    "@odata.id": "/redfish/v1/Chassis/1/NetworkAdapters"
  },
  "Oem": {
    "Hpe": {
      "@odata.context": "/redfish/v1/$metadata#HpeServerChassis.HpeServerChassis",
      "@odata.type": "#HpeServerChassis.v2_3_1.HpeServerChassis",
      "Actions": {
        "#HpeServerChassis.DisableMCTPOnServer": {
          "target": "/redfish/v1/Chassis/1/Actions/Oem/Hpe/HpeServerChassis.DisableMCTPOnServer"
        },
        "#HpeServerChassis.FactoryResetMCTP": {
          "target": "/redfish/v1/Chassis/1/Actions/Oem/Hpe/HpeServerChassis.FactoryResetMCTP"
        }
      },
      "Firmware": {
        "PlatformDefinitionTable": {
          "Current": {
            "VersionString": "9.8.0 Build 15"
          }
        },
        "PowerManagementController": {
          "Current": {
            "VersionString": "1.0.7"
          }
        },
        "PowerManagementControllerBootloader": {
          "Current": {
            "Family": "25",
            "VersionString": "1.1"
          }
        },
        "SPSFirmwareVersionData": {
          "Current": {
            "VersionString": "4.1.4.339"
          }
        }
      }
    }
  }
}
```


Delete sessions

It is extremely important to delete Redfish sessions to avoid reaching the maximum number of opened sessions in a BMC, preventing any access to it. Read this [article \(https://developer.hpe.com/blog/managing-ilo-sessions-with-redfish\)](https://developer.hpe.com/blog/managing-ilo-sessions-with-redfish) for more detail.

```
In [8]: echo "Body response of a session deletion:"
```

```
curl --insecure --noproxy "localhost, 127.0.0.1" --silent \  
--header "$HeaderContentType" --header "$HeaderODataVersion" \  
--header "X-Auth-Token: $Token" \  
--request DELETE $SessionLocation | jq
```

Body response of a session deletion:

```
{  
  "error": {  
    "code": "iLO.0.10.ExtendedInfo",  
    "message": "See @Message.ExtendedInfo for more information.",  
    "@Message.ExtendedInfo": [  
      {  
        "MessageId": "Base.1.4.Success"  
      }  
    ]  
  }  
}
```

Wrap up

In this notebook you performed the following actions:

- Create and delete a token based Redfish session
- Retrieved iLO 5 supported OData protocol features
- Expand a collection with OData `$expand` query operator
- Expand a collection with OData `only` query operator

HPE DEVELOPER COMMUNITY PROGRAM

BUILD, COMMUNICATE, COLLABORATE.



hpedev.io



hpedev.slack.com



[Monthly Newsletter](#)



<mailto:hpedev@hpe.com>



[@HPE_DevCom](https://twitter.com/HPE_DevCom)



[Yammer Group](#) (Internal)



PLEASE GIVE US YOUR FEEDBACK



iOS



Android



THANK YOU

Francois.donze@hpe.com

