Prototype of Configurable Redfish Query Proxy Module

Chanyoung Park, Yoonsue Joe, Myoungwan Yoo, Dongeun Lee, Kyungtae Kang*
Hanyang University, XSLAB, Texas A&M University-Commerce
Background: What are BMC and Redfish?

Provide server baseboard management things
- Power control
- Fan control
- OS status
- etc.

Web Service
ex)
HP iLO,
DELL iDRAC,
OpenBMC
bmcweb,
...

Clients
ipmi
Redfish
Browser
...

[Image of a circuit board with components labeled]
Background: What are BMC and Redfish?

Provide server baseboard management things:
- Power control
- Fan control
- OS status
- etc.

Web Service
ex) HP iLO, DELL iDRAC, OpenBMC bmcweb,

Clients
  - ipmi
  - Redfish
  - Browser
  - ...

---

4
Background: What are BMC and Redfish?

Provide server baseboard management things
- Power control
- Fan control
- OS status
- etc.

Web Service
ex) HP iLO, DELL iDRAC, OpenBMC bmcweb, ...

Clients
ipmi
Redfish
Browser
...
Redfish is build on top of Open Data Protocol (OData).
- OData provides standardized client side query
  ex) http://url?$top=3
  http://url?$expand&select=Name,SystemType
- Redfish follows the format of OData, but to implement OData query features are optional
- Redfish suggests to support some query features from OData and a couple of new query features
  ex) $top, $skip, $expand, $filter, $select, only, excerpt
Background: What is Redfish Query?

Redfish is build on top of Open Data Protocol (OData).
- OData provides standardized client side query
  ex) http://url?$top=3
      http://url?$expand&select=Name,SystemType
- Redfish follows the format of OData, but to implement OData query features are optional
- Redfish suggests to support some query features from OData and a couple of new query features
  ex) $top, $skip, $expand, $filter, $select, only, excerpt
Background: What is Redfish Query?

Redfish is build on top of Open Data Protocol (OData).
- OData provides standardized client side query
  ex) http://url?$top=3
      http://url?$expand&select=Name,SystemType
- Redfish follows the format of OData, but to implement OData query features are optional
- Redfish suggests to support some query features from OData and a couple of new query features
  ex) $top, $skip, $expand, $filter, $select, only, excerpt
What’s Hard? And Our Approach

Why? Because implementing query features are not easy + time consuming.
- Developers must implement all query features in data layer by themselves, if they are not using RDB.
- OData ecosystem is mature in some platform and languages
- Redfish query features are optional

Vendor A: Let’s support $filter query in resource Chassis
Vendor B: Let’s support $top, $skip queries in resource Events
Vendor C: Let’s support $select query in all resources
Vendor D: We have no plan to support any query features

Confusing…
What’s Hard? And Our Approach

Why? Because implementing query features are not easy + time consuming.
- Developers must implement all query features in data layer by themselves, if they are not using RDB.
- OData ecosystem is mature in some platform and languages
- Redfish query features are optional

Vendor A: Let’s support $filter query in resource Chassis
Vendor B: Let’s support $top, $skip queries in resource Events
Vendor C: Let’s support $select query in all resources
Vendor D: We have no plan to support any query features
What’s Hard? And Our Approach

Why? Because implementing query features are not easy + time consuming.
- Developers must implement all query features in data layer by themselves, if they are not using RDB.
- OData ecosystem is mature in some platform and languages
- Redfish query features are optional

Vendor A: Let’s support $filter query in resource Chassis
Vendor B: Let’s support $top, $skip queries in resource Events
Vendor C: Let’s support $select query in all resources
Vendor D: We have no plan to support any query features
What’s Hard? And Our Approach

Vendor A: Let’s support $filter query in resource Chassis

Vendor B: Let’s support $top, $skip queries in resource Events

Vendor C: Let’s support $select query in all resources

Vendor D: We have no plan to support any query features

Delegate $filter processing when accessing resource Chassis (fast)

Processing unsupported queries instead of BMC Servers, through make multiple additional communications with backends. (Slow)
What’s Hard? And Our Approach

Vendor A: Let’s support $filter query in resource Chassis

Vendor B: Let’s support $top, $skip queries in resource Events

Vendor C: Let’s support $select query in all resources

Vendor D: We have no plan to support any query features

RedfishTool

Do similar things to some target resources, with some query features

Difference with RedfishTool
1. We target all resources and query features.
2. We can leverage backend server’s query processing by configuration.
Approach Details

Client Request:
https://bmc-host/redfish/v1/resource_A?$expand=.($levels=1)&$select=Name,Id&$top=10

(a) Redfish service’s target resource handler does not support any query.

(b) Redfish service’s target resource handler supports $expand query.

Configure resource_A handler can process $expand query.
Approach Details

Client Request:
https://bmc-host/redfish/v1/resource_A?$expand=.($levels=1)&$select=Name,Id&$top=10

(a) Redfish service's target resource handler does not support any query.
(b) Redfish service’s target resource handler supports $expand query.
Approach Details

Client Request:
https://bmc-host/redfish/v1/resource_A?$expand=.($levels=1)&$select=Name,Id&$top=10

(a) Redfish service's target resource handler does not support any query.

(b) Redfish service's target resource handler supports $expand query.
Approach Details

Client Request:
https://bmc-host/redfish/v1/resource_A?$expand=.(levels=1)&$select=Name,Id&$top=10

(a) Redfish service’s target resource handler does not support any query.

(b) Redfish service’s target resource handler supports $expand query.

Configure resource_A handler can process $expand query.
Preliminary Experimental Result

<table>
<thead>
<tr>
<th>Through Proxy</th>
<th>Bypass Proxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>No query</td>
<td>No query</td>
</tr>
<tr>
<td>Generates two subrequests</td>
<td>Generates two subrequests</td>
</tr>
<tr>
<td>0 subrequests (delegates to</td>
<td>four subrequests</td>
</tr>
<tr>
<td>Redfish service)</td>
<td></td>
</tr>
<tr>
<td>Generates one subrequest</td>
<td>Generates one subrequest</td>
</tr>
</tbody>
</table>

- **Through Proxy**
  - No query: Generates two subrequests
  - Generates two subrequests
  - 0 subrequests (delegates to Redfish service): Generates four subrequests
  - Generates one subrequest

- **Bypass Proxy**
  - No query
  - Generates two subrequests
  - Four subrequests
  - Generates one subrequest

**Graph:**
- Shows proxy overhead
- Shows query processing performance: client program vs proxy
- Shows query processing performance: client program vs proxy + backend server
- Shows query processing performance: client program vs proxy

**Legend:**
- LAN-P
- LAN-NP
- VPN-P
- VPN-NP
- MOCK-P
- MOCK-NP

Response Time (ms)
## Preliminary Experimental Result

<table>
<thead>
<tr>
<th>Through Proxy</th>
<th>Bypass Proxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>No query</td>
<td>No query</td>
</tr>
<tr>
<td>Generates two subrequests</td>
<td>Generates two subrequests</td>
</tr>
<tr>
<td>0 subrequests (delegates to Redfish service)</td>
<td>four subrequests</td>
</tr>
<tr>
<td>Generates one subrequest</td>
<td>Generates one subrequest</td>
</tr>
</tbody>
</table>

### Through Proxy
- **LAN-P**
- **LAN-NP**
- **VPN-P**
- **VPN-NP**
- **MOCK-P**
- **MOCK-NP**

**Exp. Scenario**
- Shows proxy overhead
- Shows query processing performance client program vs proxy
- Shows query processing performance client program vs proxy + backend server

**Response Time (ms)**
- A: 46.2, 43.2
- B: 20.7, 24.3
- C: 47.0, 100.6, 234.4
- D: 16.7, 16.8

---

*19*
### Preliminary Experimental Result

<table>
<thead>
<tr>
<th>Through Proxy</th>
<th>Bypass Proxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>No query</td>
<td>No query</td>
</tr>
<tr>
<td>Generates two subrequests</td>
<td>Generates two subrequests</td>
</tr>
<tr>
<td>0 subrequests</td>
<td>four subrequests</td>
</tr>
<tr>
<td>Generates one subrequest</td>
<td>Generates one subrequest</td>
</tr>
</tbody>
</table>

**Figure:**

- **A:** Shows proxy overhead
- **B:** Shows query processing performance client program vs proxy
- **C:** Shows query processing performance client program vs proxy + backend server
- **D:** Shows query processing performance client program vs proxy

### Table:

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Through Proxy</th>
<th>Bypass Proxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>No query</td>
<td>No query</td>
</tr>
<tr>
<td>B</td>
<td>Generates two subrequests</td>
<td>Generates two subrequests</td>
</tr>
<tr>
<td>C</td>
<td>0 subrequests (delegates to Redfish service)</td>
<td>four subrequests</td>
</tr>
<tr>
<td>D</td>
<td>Generates one subrequest</td>
<td>Generates one subrequest</td>
</tr>
</tbody>
</table>

### Graph:

- X-axis: Response Time (ms)
- Y-axis:
  - LAN-P
  - LAN-NP
  - VPN-P
  - VPN-NP
  - MOCK-P
  - MOCK-NP
Preliminary Experimental Result

<table>
<thead>
<tr>
<th>Exp. Scenario</th>
<th>Through Proxy</th>
<th>Bypass Proxy</th>
</tr>
</thead>
<tbody>
<tr>
<td>No query</td>
<td>No query</td>
<td></td>
</tr>
<tr>
<td>Generates two subrequests</td>
<td>Generates two subrequests</td>
<td></td>
</tr>
<tr>
<td>0 subrequests (delegates to Redfish service)</td>
<td>four subrequests</td>
<td></td>
</tr>
<tr>
<td>Generates one subrequest</td>
<td>Generates one subrequest</td>
<td></td>
</tr>
</tbody>
</table>

- Shows proxy overhead
- Shows query processing performance client program vs proxy
- Shows query processing performance client program vs proxy + backend server

Response Time (ms)
Conclusion

• Preliminary evaluation demonstrated system performance is improved in terms of query response time

• Redfish services can easily support the query functions recommended in a specification document

• We plan to use caching for performance improvements

• We plan to continue developing to support all query features specified by Redfish, and tests in various environments