Optimizing WS-Man to retrieve Lifecycle Logs from iDRAC9

This Dell EMC technical white paper provides best practices when using WS-Man APIs to retrieve the Lifecycle Logs in iDRAC9.

Dell Engineering
June 2017

Author

Abhirup Seal

A Dell EMC Technical White Paper

Revisions

Date	Description
June 2017	Initial release

Contents

Executive summary

This technical white paper is aimed at the customers of WS-Man APIs, who want to retrieve Lifecycle Controller Logs efficiently. Starting from 14th Generation of Dell EMC PowerEdge servers, iDRAC9 introduces an efficient partial Lifecycle Controller Log retrieval mechanism by using filter dialects for WS-Man.

Abbreviations

Acronym	Definition
iDRAC	Integrated Dell Remote Access Controller
DMTF	Distributed Management Task Force
CIM	Common Information Model
CIM-OM	CIM- Object Manager
WS-Man	Web Services for Management
SOAP	Simple Object Access Protocol
LC	Lifecycle Controller
LCLogs	Lifecycle Controller Logs
CQL	CIM Query Language
WQL	Windows Management Instrumentation Query Language
cimv2	Alias for http://schemas.dmtf.org/wbem/wscim/1/cim-schema/2

1 Introduction

Dell EMC PowerEdge servers include a management controller known as iDRAC. It assists the system administrators to manage and monitor the servers in a data center. iDRAC provides functionality of management by using GUI, CLI (RACADM), and programmatic interfaces such as WS-Man and Redfish.

The WS-Man interface is a DMTF (Distributed Management Task Force) specification for systems management through Web Services architecture. WS-Man is a programmatic interface that can be used to manage the server through a series of Simple Object Access Protocol (SOAP) encapsulated Web-Based Enterprise Management (WBEM) commands. Enumeration is used to inventory the devices under management and Invoke is used to change the states and configurations of a managed entity.

This technical white paper assumes that the reader has basic working knowledge of WS-Man. For more information about WS-Man specifications, see www.dmtf.org. The Lifecycle logs of the iDRAC records every action performed on the server.

Note: The Lifecycle logs or LCLogs of the iDRAC records every action performed on the server. For more information on LC Logs, please refer to this post on the Dell Techcenter.

Retrieving Lifecycle Logs optimally may be tricky and possibly time-consuming, if the correct approach is not used. Owing to the large memory storage size that the Lifecycle Logs may grow to, appropriate retrieval mechanism must be used that best suits your requirements. For example, if a system is having 5000 Lifecycle Logs, and you are interested only in a selected few, it does not make sense to retrieve all the Lifecycle Logs. Starting from 14th generation of PowerEdge servers, we have introduced efficient partial Lifecycle Logs retrieval mechanisms by using the filter dialects.

This technical white paper also takes you through the various techniques you can use to retrieve the vast data available in the Lifecycle Logs by using the WS-Man command line interface (CLI). The WS-Man technique suited for appropriate circumstances is also discussed.

The command examples provided in this technical white paper are in PowerShell and we will be using the following PowerShell variables as standard variables all through this technical white paper. Their definitions are as follows:

```
$IPAddress = "https://<IPAddress>/wsman"
$SessionOptions = New-WSManSessionOption -SkipCACheck -SkipCNCheck -
SkipRevocationCheck -OperationTimeout 120000
$InputParams = @{Key1="value1";Key2="value2"} # Hashmap of input parameters
$WQLDialect = "http://schemas.microsoft.com/wbem/wsman/1/WQL"
$CQLDialect = "http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf"
```

1.1 Active and archived logs

The number of Lifecycle Logs grows with time. Therefore, we have the concept of Active- and Archived Lifecycle Logs. Any log that is newly created becomes a part of the Active Lifecycle Logs. Whenever the size of the Active Lifecycle Logs goes beyond certain threshold, a part of it makes its way into the Archived Lifecycle Logs. In 14th generation PowerEdge servers, the archiving logic is triggered either by the total number of logs exceeding 6,100, or the internal storage size for the logs exceeding 1 MB, whichever is

earlier. 90 percent of the Active Lifecycle Logs gets added to the archives. The archived logs are stored in the form of compressed XML files, and the size of these files can go up to 10 MB.

2 Getting the count of entries in the active Lifecycle Logs

The `DCIM_LCRecordLog` class, along with other generic information, gives you the number of Active Lifecycle Logs entries currently present in the system. Also, newly introduced in iDRAC9, the `SequenceNumber` of the latest Lifecycle Logs entry. An instance of this class is given here:

```
DCIM_LCRecordLog
  CurrentHighestSequenceNumber = 32370
  CurrentNumberOfRecords = 3027
  ElementName = LifeCycle Log
  EnabledState = 2
  HealthState = 5
  InstanceID = DCIM:LifeCycleLog
  LogState = 2
  MaxNumberOfRecords = 0
  OperationalStatus = 2
  OverwritePolicy = 8
  RequestedState = 12
```

In the example instance here, `CurrentHighestSequenceNumber` is the sequence number of the latest Lifecycle Logs entry and `CurrentNumberOfRecords` is the total number of Active Lifecycle Logs currently present. Sequence Number is unique for each Lifecycle Logs and increases by one (1) for each log entry. What you see in the example instance here, the `CurrentHighestSequenceNumber` is 32370 and `CurrentNumberOfRecords` is 3027, which indicate that all Lifecycle Logs having SequenceNumber as 29343 (32370–3027) or less are already stored into the archives.

3 Exporting Lifecycle Logs

One of the ways to completely retrieve the Lifecycle Logs is by exporting them to a network share or to a local share (inside the iDRAC), and then stream it over WS-Man by using the new Streaming APIs introduced in 14G.

3.1 Exporting the Active Lifecycle Logs

The Active Lifecycle Logs can be exported by using the WS-Man invoke API `DCIM_LCService.ExportLCLog()`. The exported file will be in the XML format. For information about this method and its parameters, see the *Dell LCManagement Profile v4.0.0, section 8.14* available on the Dell TechCenter.

Invoke the following PowerShell command:

```
Invoke-WSManAction -Action ExportLCLog -ResourceURI cimv2/DCIM_LCService -Authentication
Basic -ConnectionURI $IPAddress -Credential root -SelectorSet
@{__cimnamespace="root/dcim"; SystemCreationClassName = "DCIM_ComputerSystem"; SystemName
= "DCIM:ComputerSystem"; CreationClassName = "DCIM_LCService"; Name = "DCIM:LCService"} -
SessionOption $SessionOptions -ValueSet $InputParams
```

3.2 Exporting the complete Lifecycle Logs

The complete Lifecycle Logs—that is, Active and Archived Lifecycle Logs—can be exported by using the WS-Man invoke API `DCIM_LCService.ExportCompleteLCLog()`. The exported file will be in the `tar.gz` format. For information about this method and its parameters, see the *Dell LCManagement Profile v4.0.0, section 8.15* available on the Dell TechCenter.

Invoke the following PowerShell command:

```
Invoke-WSManAction -Action ExportCompleteLCLog -ResourceURI cimv2/DCIM_LCService -
Authentication Basic -ConnectionURI $IPAddress -Credential root -SelectorSet
@{__cimnamespace="root/dcim"; SystemCreationClassName = "DCIM_ComputerSystem"; SystemName
= "DCIM:ComputerSystem"; CreationClassName = "DCIM_LCService"; Name = "DCIM:LCService"} -
SessionOption $SessionOptions -ValueSet $InputParams
```

4 Enumerating active Lifecycle Logs

For most practical use cases, Active Lifecycle Logs will serve your purpose. If you are interested only in the Active Logs and want to get the data directly as part of command response, you can enumerate the `DCIM_LCLogEntry` class in the root/dcim namespace. Enumeration will return all the Active Lifecycle Logs, anything between 0 and 6100 instances, and the time taken for the retrieval will be directly proportional to the number of instances.

For example, if you want to create a console in which one can browse through pages containing all the Lifecycle Logs of a particular iDRAC, and in addition to paged browsing, the browser also has to support filters on logs. In such a case, enumeration is the best choice and in the next two subsections we will describe how we can make our lives easy with:

- CQL/WQL filters to set selection criteria
- Pagination to allow partial Lifecycle Logs retrieval.

Invoke the following PowerShell command:

```
Get-WSManInstance -Enumerate -ResourceURI cimv2/DCIM_LCLogEntry?__cimnamespace=root/dcim
-ConnectionURI $IPAddress -Credential root -Port 443 -Authentication Basic -SessionOption
$SessionOptions
```

4.1 Filtering the Active Lifecycle Logs using CQL/WQL

In case you are interested in only some Lifecycle Logs that qualify a specific criterion or a set of criteria, you may filter the results by running enumerations with the CQL/WQL filters. To select between CQL and WQL, you must specify one of the two following filter dialects:

- **CQL**: http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf
- **WQL**: http://schemas.microsoft.com/wbem/wsman/1/WQL

The query processing logic for Lifecycle Logs has been optimized and you will get faster responses. For an example SOAP packet for CQL/WQL query-based enumeration, see Appendix A. A basic PowerShell command syntax with CQL query is given here:

```
Get-WSManInstance -Enumerate -ResourceURI cimv2/DCIM_LCLogEntry?__cimnamespace=root/dcim
-Authentication Basic -ConnectionURI $IPAddress -Credential root -Dialect $CQLDialect -
Filter "<Your CQL Query Here>" -Port 443 -SessionOption $SessionOptions
```

The following table describes with examples some use cases and the corresponding CQL/WQL queries.

Use Case	Query
All Active Lifecycle Logs of Category “Audit”	`Select * from DCIM_LCLogEntry where Category = ‘Audit’`
All Active Lifecycle Logs of PerceivedSeverity is 2 or 3	`Select * from DCIM_LCLogEntry where PerceivedSeverity = 2 or PerceivedSeverity = 3`
All Active Lifecycle Logs created before 00 hrs. 25th May, 2017	`Select * from DCIM_LCLogEntry where CreationTimeStamp < '20170525000000.000000-300'`
All Active Lifecycle Logs created on or after 00 hrs. 25th May, 2017 and are of PerceivedSeverity 3	`Select * from DCIM_LCLogEntry where CreationTimeStamp >= '20170525000000.000000-300' and PerceivedSeverity = 3`
All Active Lifecycle Logs having the keyword “wsman” in the Message property field	`Select * from DCIM_LCLogEntry where Message like '%wsman%'`

4.2 Paginating the Active Lifecycle Logs using CQL/WQL

The Lifecycle Logs Pagination, a new feature introduced in the 14G iDRAC, enables you to perform partial retrieval of the Lifecycle Logs in a much faster and efficient manner. To serve the purpose, two new hidden properties have been introduced—`PageNumber` and `PageSize`—with default values of 1 and 8192 (8K) respectively in the `DCIM_LCLogEntry` class.

In Pagination, you can imagine the Lifecycle Logs as a book, with pages and page numbers—the latest log being the first entry of the first page and the oldest one being the last entry of the last page. The `PageNumber`, `PageSize` and the contents of any page are dynamic and are decided on a per-query-basis. The Pagination-specific selectors in the CQL/WQL query work along with other Selections and Projections, the pagination logic is applied last, after all the other selection filters are processed.

The following example queries will help you understand the feature in depth.

Use Case	Query
Get the latest 100 logs	`Select * from DCIM_LCLogEntry where PageSize = 100 and PageNumber = 1`
Get latest 101st to 200th logs	`Select * from DCIM_LCLogEntry where PageSize = 100 and PageNumber = 2`
Get latest 151st to 500th logs	`Select * from DCIM_LCLogEntry where PageSize = 50 and PageNumber >= 4 and PageNumber <= 10`

Get latest 100 logs of Category “Audit”	`Select * from DCIM_LCLogEntry where Category = ‘Audit’ and PageSize = 100 and PageNumber = 1`
Get latest 100 logs of all Categories other than Audit	`Select * from DCIM_LCLogEntry where Category != 'Audit' and PageSize = 100 and PageNumber = 1`
Get latest 100 logs starting from SequenceNumber 1234	`Select * from DCIM_LCLogEntry where SequenceNumber <= 1234 and PageSize = 100 and PageNumber = 1`
Get latest 100 logs generated on or after 00 hrs. 25th May, 2017	`Select * from DCIM_LCLogEntry where CreationTimeStamp >= '20170525000000.000000-300' and PageSize = 100 and PageNumber = 1`
Get latest 200 logs having the keyword “wsman” in the Message property field and generated before 00 hrs. 25th May, 2017	`Select * from DCIM_LCLogEntry where CreationTimeStamp < '20170525000000.000000-300' and Message like '%wsman%' and PageSize = 200 and PageNumber = 1`

5 Best practices and conclusion

As noted above, there are many options to retrieve Lifecycle Controller Logs using WS-Man API's. The next step is to understand when to use a particular approach.

Exporting the logs is required only when you really want "all" the logs. However, that is rarely the case. Alternately, you can enumerate the complete Active logs also if that suits the purpose. More often, you will be interested in the more recent set of logs only. If you compare between Extrinsic Methods (Invoke APIs to export) vs Intrinsic Methods (Enumerate with or without filter), Intrinsic methods are much easier to script with. WS-Man being a programmatic interface, is heavily used across the industry to write monitoring scripts or consoles.

Note: The timing–related data values given in this section are obtained in Dell EMC lab test environments. Results in actual customer environments may vary. These values must not be seen as absolute entities, rather they must be used for comparison between various approaches in the same setup. The values will be used in this technical white paper to show relative advantages of using one method over the other. The following test was performed on a system having 3000+ Lifecycle Logs.

Typically in the Dell EMC test environment, enumerating 3000+ Lifecycle Logs may take anywhere between 30–40 seconds. However, if you are interested in a part of the Lifecycle Logs, you may use the CQL/WQL filtering and will definitely save time. How much you save depends on the number of filtered results and the type of query.

For example, if you want "All Active LCLogs of PerceivedSeverity 2 or 3"; in the Dell EMC test setup we got around 350 instances in less than seven seconds. Thus, filtering may greatly enhance the user experience. Prior to the 14G with iDRAC9, there was not a mechanism to retrieve partial results page-wise. With this addition of Pagination, you can extract a part of a filtered result. In pure Pagination, that is, without other filtering logic, retrieving 100 logs takes around 1–4 seconds depending on whether they are the most recent (<1 second) or the oldest(3-4 seconds). Similarly, retrieving 500 logs takes around 6-9 seconds, and 1000 logs around 12-15 seconds.

In summary, if you need only partial logs, it is extremely beneficial to write intelligent CQL/WQL queries, add in the Pagination logic wherever applicable, and then make "Filtered Enumerations" to get better performance.

6 Appendix A

Example SOAP packet syntax for an Enumeration with CQL filter query:

```
<?xml version='1.0' encoding='UTF-8'?>
<s:Envelope xmlns:s="http://www.w3.org/2003/05/soap-envelope"
xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing"
xmlns:wsman="http://schemas.dmtf.org/wbem/wsman/1/wsman.xsd"
xmlns:wsen="http://schemas.xmlsoap.org/ws/2004/09/enumeration">
<s:Header>
<wsa:To s:mustUnderstand="true">https://<IPAddress>:<PORT>/wsman</wsa:To>
<wsman:ResourceURI s:mustUnderstand="true">http://schemas.dmtf.org/wbem/wscim/1/cim-
schema/2/DCIM_LCLogEntry</wsman:ResourceURI>
<wsa:ReplyTo>
<wsa:Address
s:mustUnderstand="true">http://schemas.xmlsoap.org/ws/2004/08/addressing/role/anonymous</
wsa:Address></wsa:ReplyTo>
<wsa:Action
s:mustUnderstand="true">http://schemas.xmlsoap.org/ws/2004/09/enumeration/Enumerate</wsa:
Action>
<wsman:MaxEnvelopeSize s:mustUnderstand="true">512000</wsman:MaxEnvelopeSize>
<wsa:MessageID s:mustUnderstand="true">urn:uuid:e469a6ee-5004-11e7-bb45-
415645000030</wsa:MessageID>
<wsman:OperationTimeout>PT120.0S</wsman:OperationTimeout>
<wsman:SelectorSet>
<wsman:Selector Name="__cimnamespace">root/dcim</wsman:Selector>
</wsman:SelectorSet>
</s:Header>
<s:Body>
<wsen:Enumerate>
<wsman:OptimizeEnumeration />
<wsman:MaxElements>256</wsman:MaxElements>
<wsman:Filter Dialect="http://schemas.dmtf.org/wbem/cql/1/dsp0202.pdf">Select * from
DCIM_LCLogEntry where Message like '%wsman%'</wsman:Filter>
</wsen:Enumerate>
</s:Body>
</s:Envelope>
```