Workflow Optimization

Zhan Liu

January 2013| Rev 1.0

Contents

Executive Summary

This document is for systems administrators or console application developers who are interested in remotely, automatically server deployment and management by using the Remote Service API offered in the LifeCycle Controller 2 (LC2). It is especially for those who already know how to use our best practice guide (BP) to do server deployment and management, but want to optimize the workflow and significantly reduce the server deployment time.

Introduction

Using the remote API exposed by the LifeCycle Controller 2 (LC2) capability of Dell's 12th generation servers, one can easily deploy and manage server remotely and automatically. The best practice (BP) provides the guide for the workflow of each individual task, such as FCoE boot setup, iSCSI boot setup, and raid setup etc. However, it is quite time consuming to do each individual task separately. This paper provides the method to classify jobs, divide jobs into subjobs and stack jobs and subjobs, therefore significantly reduce the server deployment time.

The jobs classification

All jobs can be classified into the following three classes:

1. Class 1: Update jobs
2. Class 2: Configuration and boot jobs
3. Class 3: Boot source jobs

The jobs must be executed in the order from low classification number to high classification numbers. i.e. 1, 2, 3 if the workflow has all the jobs or 1,2 if there is no class 3 job or 1, 3 if there is no class 2 job or 2, 3 if there is no class 1 job.

1) Class 1: Update jobs
   The following are the update jobs (see BP):

   - iDRAC firmware DUP update from CIFS or TFTP share
   - BIOS firmware DUP update from CIFS or TFTP share
   - USC firmware DUP update from CIFS or TFTP share
   - CPLD firmware DUP update from CIFS or TFTP share

2) Class 2: Configuration and boot jobs
   The following are the configuration and boot jobs:

   - RAID stacking: ResetConfig, CreateVD, assign HotSpares
   - RAID stacking with BIOS attributes using Setupjobqueue
   - Boot to network ISO
   - Boot to ISO from vFlash
   - PXE Boot using embedded NICs (11G only)
   - PXE Boot using embedded NICs (12G only)
   - Set NIC attributes and iSCSI boot using setupjobqueue (11G only)
   - iSCSI boot using NDC/Broadcom (12G only)
   - iSCSI boot using QLogic (12G only)
   - iSCSI boot using Intel (12G only)
   - IO Identity

- Export LC log
- FCoE boot using QLogic (12G only)
- FCoE boot using Intel (12G only)

3) Class 3: Boot source jobs
The following are the boot source jobs:

- Changing boot order by instance
- Enable or disable boot source

How to divide job into subjobs

Basically, try to find the reboot job, which is the divider of subjobs. The job between each reboot job is a subjob.

Take NIC job as an example:

The following is the NIC job workflow.

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. GetRemoteServicesAPIStatus():
   1. System should be power off
   2. Clear all unfinished jobs
   3. Clear all pending data

B) Check NIC is enabled
   1. GetBIOSEnumerations(): ENUMERATE the *DCIM_BIOSEnumeration* class to collect information about the system.
   2. Ensure AttributeName of Slot2 is enabled
      If it is not enabled, enable it as shown below

      SetBIOSAttributes()
      - AttributeName= Slot2 AttributeValue=Enabled
      - AttributeName=BootMode AttributeValue=Bios
      - CreateBIOSConfigJob()
      - ScheduledStartTime=TIME_NOW RebootJobType=1
      - Poll jobstatus for Completed: GET the *InstanceID* of from 2).

C) Set legacy boot protocol to FCoE and enable Connect First FCoE Target
   - SetNICAttributes()
     AttributeName=LegacyBootProto AttributeValue=FCoE
     AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled
   - Disable all sources
   - Create BIOS job
   - SetNICAttributes()
     AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled

D) Configure FCoE

   1. Disable all sources
   2. Create BIOS job
   3. Set Attributes (VLAN etc) as follows

- SetNICAttributes() on NIC.Mezzanine.2B-1
  AttributeName=FCoEOffloadMode AttributeValue=Enabled

  AttributeName=VirtFIPMacAddr AttributeValue=$VirtFIPMacAddr
  AttributeName=VirtWWN AttributeValue=$VirtWWN AttributeName=VirtWWPN
  AttributeValue=$VirtWWPN AttributeName=MinBandwidth
  AttributeValue=$MinBandwidth AttributeName=MaxBandwidth
  AttributeValue=$MaxBandwidth

4. CreateNICConfigJob()

5. Set Attributes (target)as follows

   - SetNICAttributes() on NIC.Mezzanine.2B-1

     AttributeName=FirstFCoEWWPNTarget AttributeValue=$FirstFCoEWWPNTarget
     AttributeName=FirstFCoEBootTargetLUN AttributeValue=$FirstFCoEBootTargetLUN

6. CreateNICConfigJob() with RebootJobType=1

E) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
Loop through all boot sources, if boot source is IPL entry, set EnabledState=0 unless HD.

F) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from D)
G) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
H) Enable the HD boot source

I) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from F)
GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

Change NIC boot source

J) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
Check NIC boot order

K) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from I)
SetNICAttributes(): Set the attribute LegacyBootProto to the value “FCoE” and the other desired NIC attributes and values
L) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)
ScheduledStartTime=TIME_NOW RebootJobType=1

M) Poll jobstatus for Completed: GET the *InstanceID* of from F).

Then analyze the workflow, we found that after step B), D) and L), we need a reboot to actually set the values. Therefore, we can just divide this job into 3 subjobs separated by reboot job.

Subjob1:

A) The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands. GetRemoteServicesAPIStatus():

1. System should be power off

2. Clear all unfinished jobs

3. Clear all pending data

B) Check NIC is enabled

1. GetBIOSEnumerations(): ENUMERATE the DCIM_BIOSEnumeration class to collect information about the system.

2. Ensure AttributeName of Slot2 is enabled

If it is not enabled, enable it as shown below

SetBIOSAttributes()

- AttributeName= Slot2 AttributeValue=Enabled
- AttributeName=BootMode AttributeValue=Bios
- CreateBIOSConfigJob()

Reboot job:

CreateRebootJob()

Poll jobstatus for Completed: GET the InstanceID from 2).

Subjob2:

C) Set legacy boot protocol to FCoE and enable Connect First FCoE Target

- SetNICAttributes()

AttributeName=LegacyBootProto AttributeValue=FCoE AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled

- Disable all sources
- Create BIOS job
- SetNICAttributes()

AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled

D) Configure FCoE

1. Disable all sources
2. Create BIOS job
3. Set Attributes (VLAN etc) as follows

- SetNICAttributes() on NIC.Mezzanine.2B-1

AttributeName=FCoEOffloadMode AttributeValue=Enabled

AttributeName=VirtFIPMacAddr AttributeValue=$VirtFIPMacAddr AttributeName=VirtWWN AttributeValue=$VirtWWN AttributeName=VirtWWPN AttributeValue=$VirtWWPN AttributeName=MinBandwidth AttributeValue=$MinBandwidth AttributeName=MaxBandwidth AttributeValue=$MaxBandwidth

4. CreateNICConfigJob()

5. Set Attributes (target)as follows

- SetNICAttributes() on NIC.Mezzanine.2B-1

AttributeName=FirstFCoEWWPNTarget AttributeValue=$FirstFCoEWWPNTarget AttributeName=FirstFCoEBootTargetLUN AttributeValue=$FirstFCoEBootTargetLUN

7. CreateNICConfigJob()

The reboot job

CreateRebootJob() with RebootJobType=1

Poll jobstatus for Completed

Subjob3:

GetBootSourceSettings(): ENUMERATE the DCIM_BootSourceSetting class to collect information about the boot sources.

Loop through all boot sources, if boot source is IPL entry, set EnabledState=0 unless HD.

F) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from D)

G) GetBootSourceSettings(): ENUMERATE the DCIM_BootSourceSetting class to collect information about the boot sources.

H) Enable the HD boot source

I) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from F)

GetBootSourceSettings(): ENUMERATE the DCIM_BootSourceSetting class to collect information about the boot sources.

Change NIC boot source

J) GetBootSourceSettings(): ENUMERATE the DCIM_BootSourceSetting class to collect information about the boot sources.

Check NIC boot order

K) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from I)

SetNICAttributes(): Set the attribute LegacyBootProto to the value "FCoE" and the other desired NIC attributes and values

L) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

ScheduledStartTime=TIME_NOW RebootJobType=1

N) Poll jobstatus for Completed: GET the InstanceID from F).

How to stack jobs

Let us use the following example to explain how to stack jobs.

Suppose we have the above NIC job and the following RAID job.

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

B) [LC1.5.1 only] Disable CSIOR (Collect System Inventory on Restart).
NOTE: On 11G systems, CSIOR must be disabled to circumvent a sync behavior that prohibits successful RAID stacking.

a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]
C) ENUMERATE the *DCIM_ControllerView* class to find RAID controller's instanceID & FQDD (they are often identical.)

a. Integrated RAID card example is ”RAID.Integrated.1-1”

b. External RAID card example is ”RAID.Slot.1-1”

D) ResetConfig(): Delete all virtual disks and unassign all HotSpare physical disks. [ReturnValue=0].

E) CreateVirtualDisk(): RAID 1 on physical disk 0 & 1, for example. [ReturnValue=0].

F) AssignSpare(): Create dedicated hotspare using Create VD instanceID [ReturnValue=0].

G) CreateRAIDConfigJob(): Apply steps D) – F) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096].

H) SetAttribute(): Set BIOS attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

I) CreateBIOSConfigJob(): Apply step H) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096]

J) CreateRebootJob(): Pass RebootJobType of 3 parameter

1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown
K) SetupJobQueue(): Use RAID JID(G), BIOS JID(J), and reboot RID(K) [ReturnValue=0]

L) Poll jobstatus for Completed: GET the *InstanceID* of from G) or J).

M) The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for ‘reloading’ then poll for ‘ready’, while the GetRemoteServicesAPIStatus() can just poll for ‘ready.’
N) ENUMERATE the *DCIM_VirtualDiskView* class to ensure successful virtual disk creation.

a. RAIDTypes parameter will be 4, for a RAID 1 configuration

b. PhysicalDiskIDS parameter will list physical disks used

O) ENUMERATE the *DCIM_PhysicalDiskView* class to ensure successful hotspare assignments.

The raid job can be divided into the following subjobs:

Subjob1:

A)The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

B) [LC1.5.1 only] Disable CSIOR (Collect System Inventory on Restart).

NOTE: On 11G systems, CSIOR must be disabled to circumvent a sync behavior that prohibits successful RAID stacking.

a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]
C) ENUMERATE the *DCIM_ControllerView* class to find RAID controller's instanceID & FQDD (they are often identical.).

a. Integrated RAID card example is "RAID.Integrated.1-1"

b. External RAID card example is "RAID.Slot.1-1"

D) ResetConfig(): Delete all virtual disks and unassign all HotSpare physical disks. [ReturnValue=0].

E) CreateVirtualDisk(): RAID 1 on physical disk 0 & 1, for example. [ReturnValue=0].

F) AssignSpare(): Create dedicated hotspare using Create VD instanceID [ReturnValue=0].

G) CreateRAIDConfigJob(): Apply steps D) – F) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096].

H) SetAttribute(): Set BIOS attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

I) CreateBIOSConfigJob(): Apply step H) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096]

Reboot job:

J) CreateRebootJob(): Pass RebootJobType of 3 parameter

1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown
K) SetupJobQueue(): Use RAID JID(G), BIOS JID(J), and reboot RID(K) [ReturnValue=0]

L) Poll jobstatus for Completed: GET the *InstanceID* of from G) or J).

M) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

Subjob2:

N) ENUMERATE the *DCIM_VirtualDiskView* class to ensure successful virtual disk creation.

a. RAIDTypes parameter will be 4, for a RAID 1 configuration

b. PhysicalDiskIDS parameter will list physical disks used

O) ENUMERATE the *DCIM_PhysicalDiskView* class to ensure successful hotspare assignments.

We notice that after NIC subjob1 and RAID subjob1, a reboot job is necessary. Since this two subjobs have no dependency, we can combined them together and use just one reboot to complete the two subjobs. The same happens to NIC subjob2 and RAID Subjob2. Now the stacked workflow for this example is the below:

- NIC subjob1
- RAID subjob1
- The reboot job
- NIC subjob2
- RAID sub job2
- The reboot job
- NIC subjob3
- The reboot job

Here, the key is that even though both NIC and RAID jobs need reboot to complete, they don't need to be done by two separate reboot jobs. They can be done by one reboot job. Since reboot job usually consume most of the server deployment and configuration time, the deployment and configuration time will be significantly reduced. Next section, we will see which subjobs can be stacked together in one reboot job and how to optimize the workflow.

How to optimize the workflow

We already know how to classify jobs, how to divide jobs into subjobs, and how to stack the subjobs. In this section, we will see how we can optimize the workflow and reduce the server deployment time.

Most of the workflow may have one or all of the following jobs. We can follow this example as a basic guide for optimization.

Suppose we have a workflow with the following jobs need to be done.

1. - update NIC firmware
2. - BIOS jobs
3. – Raid jobs
4. - NIC jobs
5. - Boot Sources

First, classify the jobs:

Class 1:

1. - update NIC firmware

Class 2:

2. - BIOS jobs
3. – Raid jobs
4. - NIC jobs

Class 3:

5. - Boot Sources

Second, divide the jobs into the following subjobs. Since we only have one job in Class 1 and 3, therefore we only need to analyze jobs in class 2 and try to stack them. Please refer to the Appendix and the above section to figure out how to get the subjobs. Here we just suppose that the subjobs have already been got.

2 - BIOS jobs

3. Raid

Subjob1: 3a - Enable RAID

Subjob2: 3b - Configure RAID

4. NIC jobs

Subjob1: 4a- NIC enable (if required)

Subjob2: 4b- NIC partition enablement (if required)

Subjob3: 4c- NIC attributes (possible REBOOTs inside this as well)

Then we come up with this table, here 1, 2, 3a, etc represent the job (subjob) need to be done, to show the parallel / serial relationship of jobs.

<table>
<tr><td colspan="3">1 - update NIC firmware</td><td>reboot</td></tr>
<tr><td rowspan="3">2 - BIOS jobs</td><td>3a - Enable RAID</td><td>4a - NIC enable</td><td>reboot</td></tr>
<tr><td>3b- Configure RAID</td><td>4b - NIC partition enablement</td><td>reboot</td></tr>
<tr><td></td><td>4c - NIC attributes</td><td>reboot</td></tr>
<tr><td colspan="3">5 - Boot Sources</td><td>reboot</td></tr>
</table>

First, the jobs must be executed in this order:

Class 1, Class 2 and Class 3.

Second, in the same class, subjobs must be done in this order a, b, c, … if they are within the same job. The subjobs have dependency if they are in the same job. The subjobs have no dependency if they are in different jobs within the same class. Therefore, any subjob can be stack with other subjobs which are not in the same job but in the same class.

Now, the workflow can be stacked as follows.

Job 3 must be done in this order: 3a, reboot, and 3b, reboot

Job 4 must be done in this order: 4a, reboot, 4b, reboot, and 4c, reboot

Since Job 2, 3, 4 are in the same class (Class 2 in this case), they can be done in parallel to reduce the number of reboots, which consume most of the deployment time.

The following are those possible useful workflow if any of 3a, 3b, 4a, 4b, and 4c job has already be done by default.

1) If none of the subjob has been done by default
   1, reboot
   2, 3a, 4a, reboot

   3b, 4b, reboot

   4c, reboot

   5, reboot

2) If 4a is done.
   1, reboot

   2, 3a, 4b, reboot

   3b, 4c, reboot

   5, reboot

3) if 4a, 4b is done
   1, reboot

   2, 3a, 4c, reboot

   3b, reboot

   5, reboot

4) If 4a, 3a is done
   1, reboot

   2, 3b, 4b, reboot

   4c, reboot

   5, reboot

5) If 4a, 4b, and 3a is done
   1, reboot
   3b, 4c, reboot
   5, reboot

General Optimization Rules to Follow

First, follow the best practice (BP) to figure out the individual workflows (jobs) needed for your server deployment and management task (your workflow).

Second, classify the jobs into the three classes by following the section “The jobs classification” in this paper

Third, divide each job into subjobs by following section “How to divide job into subjobs” or refer to “appendix: The Subjobs of Each Individual Workflow “.

Fourth, stack the jobs by following section “How to stack jobs” to reduce the number of reboot.

Fifth, form the optimized workflow to further reduce the number of reboot by following section “How to optimize the workflow”

“Appendix: The Subjobs of Each individual Workflow” lists all subjobs of each known individual workflow. You can simply use this appendix to get those subjobs and follow the steps in this paper to do the workflow optimization.

Appendix: The Subjobs of Each Individual Workflow

RAID stacking: ResetConfig, CreateVD, assign HotSpares

Subjob1:

A) The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) [LC1.5.1 only] Disable CSIOR (Collect System Inventory on Restart).

NOTE: On 11G systems, CSIOR must be disabled to circumvent a sync behavior that prohibits successful RAID stacking.
a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

Reboot job:

CreateRebootJob()
Poll jobstatus for Completed

Subjob2:

C) ENUMERATE the *DCIM_ControllerView* class to find RAID controller’s instanceID & FQDD (They are often identical.)
a. Integrated RAID card example is ”RAID.Integrated.1-1”
b. External RAID card example is ”RAID.Slot.1-1”
D) ResetConfig(): Delete all virtual disks and unassign all HotSpare physical disks. [ReturnValue=0] and create a reboot job

Reboot job:

CreateRebootJob()
Poll jobstatus for Completed

Subjob3:

E) CreateVirtualDisk(): RAID 1 on physical disk 0 & 1, for example. [ReturnValue=0].
F) AssignSpare(): Create dedicated hotspare using Create VD instanceID [ReturnValue=0].
G) AssignSpare(): Create global hotspare [ReturnValue=0].

H) CreateRAIDConfigJob(): Apply steps D) – F) [ReturnValue=4096].

Reboot job:

CreateRebootJob()
I) Poll jobstatus for Completed: GET the *InstanceID* from H).

Subjob4 :

J) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'
K) ENUMERATE the *DCIM_VirtualDiskView* class to ensure successful virtual disk creation.
a. RAIDTypes parameter will be 4, for a RAID 1 configuration
b. PhysicalDiskIDS parameter will list physical disks used
L) ENUMERATE the *DCIM_PhysicalDiskView* class to ensure successful hotspare assignments.
a. HotSpareStatus parameter of 2, indicates global hotspare
b. HotSpareStatus parameter of 1, indicates dedicated hotspare

NOTE: H200 controller is unique in that it always returns 2 for both dedicated and global hotspares

RAID stacking with BIOS attributes using Setupjobqueue

Subjob1 :

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) [LC1.5.1 only] Disable CSIOR (Collect System Inventory on Restart).

NOTE: On 11G systems, CSIOR must be disabled to circumvent a sync behavior that prohibits successful RAID stacking.
a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

Reboot job:

CreateRebootJob()
Poll jobstatus for Completed

subjob2:

C) ENUMERATE the *DCIM_ControllerView* class to find RAID controller's instanceID & FQDD (they are often identical.)
a. Integrated RAID card example is "RAID.Integrated.1-1"
b. External RAID card example is "RAID.Slot.1-1"
D) ResetConfig(): Delete all virtual disks and unassign all HotSpare physical disks. [ReturnValue=0].
E) CreateVirtualDisk(): RAID 1 on physical disk 0 & 1, for example. [ReturnValue=0].
F) AssignSpare(): Create dedicated hotspare using Create VD instanceID [ReturnValue=0].

G) CreateRAIDConfigJob(): Apply steps D) – F) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096].
A) SetAttribute(): Set BIOS attribute EmbNic1Nic2 to Enabled [ReturnValue=0]
B) CreateBIOSConfigJob(): Apply step H) without reboot type, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096]

Reboot job:

C) CreateRebootJob(): Pass RebootJobType of 3 parameter
1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown
D) SetupJobQueue(): Use RAID JID(G), BIOS JID(J), and reboot RID(K) [ReturnValue=0]
E) Poll jobstatus for Completed: GET the *InstanceID* from G) or J).
F) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

Subjob4:
G) ENUMERATE the *DCIM_VirtualDiskView* class to ensure successful virtual disk creation.
a. RAIDTypes parameter will be 4, for a RAID 1 configuration
b. PhysicalDiskIDS parameter will list physical disks used
H) ENUMERATE the *DCIM_PhysicalDiskView* class to ensure successful hotspare assignments.
a. HotSpareStatus parameter of 2, indicates global hotspare
b. HotSpareStatus parameter of 1, indicates dedicated hotspare

NOTE: H200 controller is unique in that it always returns 2 for both dedicated and global hotspares
I) ENUMERATE the *DCIM_BIOSEnumeration* class to ensure BIOS settings were correctly set.

Boot to network ISO

Subjob1 :

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

NOTE: GetRemoteServicesAPIStatus() will return "not ready" if drivers or an ISO is already attached.
B) DetachDrivers(): Ensures any drivers are detached.
C) DetachISOImage(): Ensures all images are detached.
D) GetDriverPackInfo(): Displays available OS drivers. This is only required for end to end OS deployment.
E) UnpackAndAttach(): Unpacks and attaches desired driver pack. The resulting concrete job is invoked immediately. This is only required for end to end OS deployment.
F) Poll concrete job until 'Success'.
G) BootToNetworkISO(): The resulting concrete job is invoked immediately.
H) Poll concrete job until 'Success'.

NOTE: OS is still booting at this point, so sleep to allow completion. Steps I) through J) are providing when the BootToNetwork image is no longer desired.

I) DetachDrivers(): [ReturnValue=0].
J) DetachISOImage(): [ReturnValue=0].
K) RequestMonoSystemStateChange(): [ReturnValue=0].
NOTE: Modular systems (i.e. M610, M710, etc.) use RequestModSystemStateChange().

Boot to ISO from vFlash

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

NOTE: GetRemoteServicesAPIStatus() will return "not ready" if drivers or an ISO is already attached.
B) DetachDrivers(): Ensures any previous drivers are detached.
C) DetachISOImage(): Ensures all previous images are detached.
D) DetachISOFromVFlash(): Ensures all previous images are detached.
E) DeleteISOFromVFlash(): Ensures all previous images are deleted.
F) DownloadISOToVFlash(): Download desired image from network to vFlash.
G) Poll concrete job until 'Success'.
H) GetDriverPackInfo(): Displays available OS drivers. This is only required for end to end OS deployment.
I) UnpackAndAttach(): Unpacks and attaches desired driver pack. The resulting concrete job is invoked immediately. This is only required for end to end OS deployment.
J) Poll concrete job until 'Success'.
K) BootToISOFromVFlash(): The resulting concrete job is invoked immediately.
L) Poll concrete job until 'Success'.

NOTE: OS boot is complete at this point, sleep 600 seconds to allow for completion. Steps M) through P) are providing when the BootToNetwork image is no longer desired.
M) DetachDrivers(): [ReturnValue=0].
N) DetachISOFromVFlash(): [ReturnValue=0].
O) DeleteISOFromVFlash(): [ReturnValue=0].
P) RequestMonoSystemStateChange(): Reboot to finish removal of OS [ReturnValue=0].

NOTE: Modular systems (i.e. M610, M710, etc.) use RequestModSystemStateChange().

Set hard drive to first in boot order

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

B) Change BootMode to BIOS, if current value is UEFI.
a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

Reboot job:

CreateRebootJob()
Poll jobstatus for Completed

Subjob2:

C) GetBootConfigSettings(): ENUMERATE the *DCIM_BootConfigSetting* class to identify the *ElementName* field containing *BootSeq* and corresponding *InstanceID* (IPL or UEFI).
ElementName = Hard drive C: BootSeq
D) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class.
a. The *CurrentAssignedSequence* attribute of each instance defines the instance's place in the zero based indexed boot sequence
b. The CurrentEnabledStatus attribute defines whether the boot source, such as the hard drive, is enabled
c. If the current sequence is 0 and the status is enable, skip to the end
E) ChangeBootOrderByInstanceID(): using instanceID = IPL [ReturnValue=0]
F) ChangeBootSourceState(): using instanceID = IPL and EnabledState=1 [ReturnValue=0]
G) Poll jobstatus for Completed: GET the *InstanceID* of from E).
H) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'
I) ENUMERATE the *DCIM_BootSourceSetting* class.
a. The CurrentAssignedSequence of the "Hard drive C" should be 0
b. The CurrentEnabledStatus of the "Hard drive C" should be 1

Export (backup) image to vFlash

Subjob1 :

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) BackupImage(): Performs backup operation [ReturnValue=4096].
C) Poll jobstatus for Completed: GET the *InstanceID* of from B).
NOTE: The available space on the SD card will be reduced by 384MB upon completion of successful backup.

Export (backup) image to CIFS or NFS share

Subjob1 :

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) BackupImage(): Performs backup operation [ReturnValue=4096].
C) Poll jobstatus for Completed: GET the *InstanceID* of from B).

Import (restore) image from vFlash

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) RestoreImage(): Performs restore operation [ReturnValue=4096].
C) Poll jobstatus for Completed: GET the *InstanceID* of from B).

Import (restore) image from CIFS or NFS share

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

B) RestoreImage(): Performs restore operation [ReturnValue=4096].
C) Poll jobstatus for Completed: GET the *InstanceID* of from B).

iDRAC firmware DUP update from CIFS or TFTP share

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) GetSoftwareIdentities(): ENUMERATE the *DCIM_SoftwareIdentity* class to list the firmwares on the system.
C) Search the results from B) for:

[LC1.5.0/LC1.5.1] "ElementName = iDRAC6" and note the accompanying instanceID to be used in D).
[LC2 1.0] "ElementName = Integrated Dell Remote Access Controller" and note the accompanying instanceID to be used in D).
Use the Software Inventory registered profile version to determine the applicable string to search for.

B) InstallFromURI(): Invokes firmware update operation [ReturnValue=4096].

Reboot job:

E) CreateRebootJob(): Pass parameter RebootJobType of value 3.

1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown LC Integration Best Practices Specification
C) SetupJobQueue(): Use JID(D) and reboot RID(E) [ReturnValue=0]; The StartTimeInterval parameter is set to TIME_NOW, meaning the operations will be invoked immediately.
G) Poll RID jobstatus for Reboot Completed: GET the *InstanceID* of from E).
H) Poll JID jobstatus for Completed: GET the *InstanceID* of from D).
I) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

BIOS firmware DUP update from CIFS or TFTP share

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) GetSoftwareIdentities(): ENUMERATE the *DCIM_SoftwareIdentity* class to list the firmwares on the system.
C) Search the results from B) for "ElementName = BIOS" and for "Status = Installed", then note the accompanying instanceID to be used in D)
D) InstallFromURI(): Invokes firmware update operation [ReturnValue=4096]

Reboot job:

E) CreateRebootJob(): Pass parameter RebootJobType of value 3

1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown
D) SetupJobQueue(): Use JID(D) and reboot RID(E) [ReturnValue=0]; The StartTimeInterval parameter is set to TIME_NOW, meaning the operations will be invoked immediately
G) Poll RID jobstatus for Reboot Completed: GET the *InstanceID* from E
H) Poll JID jobstatus for Completed: GET the *InstanceID* from D).
I) [LC1.5.0/LC1.5.1] Sleep for 5 minutes to allow reboot, POST, and CSIOR to complete

See Appendix 33.4.3 and 33.4.7 for more information about POST and CSIOR
J) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

USC firmware DUP update from CIFS or TFTP share

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) GetSoftwareIdentities(): ENUMERATE the *DCIM_SoftwareIdentity* class to list the firmwares of the system.
C) Search the results from B) for "ElementName = Dell Lifecycle Controller" and note the accompanying instanceID to be used in D). There may be additional characters and numbers after the substring "Controller".
D) InstallFromURI(): Invokes firmware update operation [ReturnValue=4096]

NOTE: The USC update is applied immediately, and cannot be scheduled for a later time.
E) Poll jobstatus for Completed: GET the *InstanceID* of from D).
F) RequestiDRACStateChange(): Must reset idrac for changes to take effect [ReturnValue=0]
G) [LC1.5.0/LC1.5.1]Sleep for 10 minutes to allow reboot, POST, and CSIOR to complete

See Appendix 33.4.3 and 33.4.7 for more information about POST and CSIOR
H) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

PXE Boot using embedded NICs (11G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) Enable CSIOR (Collect System Inventory on Restart)
a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

Reboot job:

CreateRebootJob()

Poll jobstatus for Completed:

Subjob2:

C) Call subroutine sub_setEmbNICs_NIC1_NIC2.win to perform the following:
a. GetBIOSEnumerations(): Enumerate the DCIM_BIOSEnumeration to obtain the current values of EmbNic attributes
b. DeletePendingBIOSConfiguration(): Ensures there is no other pending BIOS configuration
c. SetAttribute(): Set parent attribute EmbNic1Nic2 to DisabledOS [ReturnValue=0]
d. SetAttribute(): Set child attributes EmbNic1 and EmbNic2 to Disabled [ReturnValue=0]
e. CreateBIOSConfigJob(): Creates jobID and applies configuration immediately with reboot job type of 3 [ReturnValue=4096]

Reboot job:

CreateRebootJob()
Poll jobstatus for Completed

NOTE: The following RS Status polling for SSIB task
f. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'
NOTE: The following RS Status polling is for PXE to be set in the boot list during CSIOR
g. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

Subjob3:

D) Call subroutine sub_setEmbNICs_NIC1_NIC2.win to perform the following:
a. GetBIOSEnumerations(): Enumerate the DCIM_BIOSEnumeration to obtain the current values of EmbNic attributes

b. DeletePendingBIOSConfiguration(): Ensures there is no other pending BIOS configuration

c. SetAttribute(): Set parent attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

d. SetAttribute(): Set child attributes EmbNic1 and EmbNic2 to EnabledPxe [ReturnValue=0]

e. CreateBIOSConfigJob(): Creates jobID and applies configuration immediately with reboot job type of 3 [ReturnValue=4096]

Reboot job:

CreateRebootJob()
NOTE: The following RS Status polling for SSIB task
f. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'
NOTE: The following RS Status polling is for PXE to be set in the boot list during CSIOR
g. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

Subjob4:

E) Sleep 500 seconds to allow PXE boot to occur. Users would then select applicable PXE boot options before continuing.

Proceed to step F) to disable PXE boot.

F) Call subroutine sub_setEmbNICs_NIC1_NIC2.win to perform the following:

a. GetBIOSEnumerations(): Enumerate the DCIM_BIOSEnumeration to obtain the current values of EmbNic attributes

b. DeletePendingBIOSConfiguration(): Ensures there is no other pending BIOS configuration

c. SetAttribute(): Set parent attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

d. SetAttribute(): Set child attributes EmbNic1 and EmbNic2 to Enabled [ReturnValue=0]

e. CreateBIOSConfigJob(): Creates jobID and applies configuration immediately with reboot job type of 3 [ReturnValue=4096]

Reboot job:

CreateRebootJob()

NOTE: The following RS Status polling for SSIB task

f. The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

NOTE: The following RS Status polling is for PXE to be set in the boot list during CSIOR

g. The GetRSStatus() method must first poll for 'reloading' then poll for 'ready', while the GetRemoteServicesAPIStatus() can just poll for 'ready.'

PXE Boot using embedded NICs (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands.

GetRemoteServicesAPIStatus()

B) Enable CSIOR (Collect System Inventory on Restart)

a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

C) Call subroutine sub_setEmbNICs_NIC1_NIC2_12G.win to perform the following:

a. GetBIOSEnumerations(): Enumerate the DCIM_BIOSEnumeration to obtain the current values of EmbNic attributes

b. DeletePendingBIOSConfiguration(): Ensures there is no other pending BIOS configuration

c. SetAttribute(): Set parent attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

d. SetAttribute(): Set child attributes EmbNicPort1BootProto to Pxe and EmbNicPort2BootProto to None [ReturnValue=0]

e. CreateBIOSConfigJob(): Creates jobID and applies configuration immediately with reboot job type of 3 [ReturnValue=4096]

Reboot job:

CreateRebootJob()

NOTE: The following polling is for SSIB task

f. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands.

GetRemoteServicesAPIStatus()
D) Sleep 500 seconds to allow PXE boot to occur. Users would then select applicable PXE boot options before continuing.

Subjob2:

Proceed to step E) to disable PXE boot.
E) Call subroutine sub_setEmbNICs_NIC1_NIC2_12G.win to perform the following:
a. GetBIOSEnumerations(): Enumerate the DCIM_BIOSEnumeration to obtain the current values of EmbNic attributes

b. DeletePendingBIOSConfiguration(): Ensures there is no other pending BIOS configuration

c. SetAttribute(): Set parent attribute EmbNic1Nic2 to Enabled [ReturnValue=0]

d. SetAttribute(): Set child attributes EmbNicPort1BootProto to None and EmbNicPort2BootProto to None [ReturnValue=0]

e. CreateBIOSConfigJob(): Creates jobID and applies configuration immediately with reboot job type of 3 [ReturnValue=4096]

NOTE: The following polling is for SSIB task
f. The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands.

GetRemoteServicesAPIStatus()

Set NIC attributes and iSCSI boot using setupjobqueue (11G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.
B) ENUMERATE the *DCIM_NICView* and *DCIM_SoftwareIdentity* classes to collect information about the system.
C) Enable CSIOR (Collect System Inventory on Restart), if not enabled
a. SetAttribute(): Sets attribute to be configured [ReturnValue=0]

b. CreateConfigJob(): Creates jobID and applies configuration [ReturnValue=4096]

Reboot job:

CreateRebootJob()
c. Poll jobstatus for Completed: GET the *InstanceID* from B).

Subjob2:

D) ENUMERATE the *DCIM_NICEnumeration*, *DCIM_NICString*, *DCIM_NICInteger*, and *DCIM_BIOSEnumeration* classes to collect information about the system.

E) SetBIOSAttributes(): Set all the following attributes, if at least one is not set to desired value
a. EmbNic1Nic2=Enabled
b. BootMode=BIOS
c. ProcVirtualization= Enabled
d. ErrPrompt=Disabled
e. EmbNic1=Enabled
F) CreateBIOSConfigJob(): Apply step E) with reboot type 3 and ScheduledStartTime parameter of TIME_NOW, which invokes the operation immediately [ReturnValue=4096]

Reboot job:

CreateRebootJob()
G) Poll jobstatus for Completed: GET the *InstanceID* from F

NOTE: The following status polling for SSIB task
F) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

NOTE: The following status polling is for subsequent CSIOR
G) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

Subjob3:

H) ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the system.
I) ChangeBootSourceState(): Loop through boot sources and set their enabled state to zero, except for NIC [ReturnValue=4096]
J) ChangeBootOrderByInstanceID(): Set the boot order of the NIC to first (CurrentAssignedSequence = 0) [ReturnValue=0]
K) SetAttribute(): Set BIOS attributes EmbNic1 to EnablediScsi [ReturnValue=0]
L) CreateBIOSConfigJob(): Apply steps J) – L) without reboot, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096]
M) SetAttribute(): Set various NIC attributes
N) CreateNICConfigJob(): Apply steps N) without reboot, without UntilTime, and without ScheduledStartTime parameter TIME_NOW. [ReturnValue=4096]

Reboot job:
O) CreateRebootJob(): Pass RebootJobType of 3 parameter

1 = PowerCycle
2 = Graceful reboot without forced shutdown
3 = Graceful reboot with forced shutdown
P) SetupJobQueue(): Use BIOS JID(L), NIC JID(N), and reboot RID(O) [ReturnValue=0]
Q) Poll jobstatus for Completed: GET the *InstanceID* of from M).
R) Poll jobstatus for Completed: GET the *InstanceID* of from O).
S) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

iSCSI boot using NDC/Broadcom (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands.

GetRemoteServicesAPIStatus():

B) GetBIOSEnumerations: ENUMERATE the *DCIM_BIOSEnumeration* class to collect information about the system.

C) GetNICViews: ENUMERATE the *DCIM_NICVIEW* class to collect information about the NIC FQDDs.

D) GetBootSourceSettings: ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the NIC FQDDs.

- Check whether the FQDD and IPL fields are in the boot order
- SetNICAttributes(): Set the attribute LegacyBootProto to the value "iSCSI" and the other desired NIC attributes and values
- CreateNICConfigJob()

Reboot job:

```
CreateRebootJob()
Poll jobstatus for Completed
```

Subjob2:

E) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the NICs.

- Check the CurrentEnabledStatus to ensure it is enabled

F) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from D)

G) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from D)

H) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

Reboot job:

```
CreateRebootJob()
```

I) Poll jobstatus for Completed: GET the *InstanceID* from F

iSCSI boot using QLogic (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands.

GetRemoteServicesAPIStatus():

B) GetBIOSEnumerations(): ENUMERATE the *DCIM_BIOSEnumeration* class to collect information about the system.

- Ensure AttributeName of IntegratedNetwork1 is enabled
- If it is not enabled, enable it as shown below
- SetBIOSAttributes()
- AttributeName=IntegratedNetwork1 AttributeValue=Enabled
- AttributeName=BootMode AttributeValue=Bios
- CreateBIOSConfigJob()

Reboot job:

- CreateRebootJob() RebootJobType=1

- Poll jobstatus for Completed: GET the *InstanceID* from 2).

Subjob2:

C) GetNICViews: ENUMERATE the *DCIM_NICVIEW* class to collect information about the NIC FQDDs.
- Check if specified FQDD is present in NICViews, If not, go to NICError

D) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
- Loop through all boot sources, if boot source is IPL entry, set EnabledState=0 unless HD

E) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from D)

F) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
- Enable the HD boot source

G) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from F)

H) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
- Change NIC boot source

I) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
- Check NIC boot order

J) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from I)
- SetNICAttributes(): Set the attribute LegacyBootProto to the value “iSCSI” and the other desired NIC attributes and values

K) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

Reboot job:

- CreateRebootJob() RebootJobType=1

L) Poll jobstatus for Completed: GET the *InstanceID* from F).

Notes:
1) QLogic will not show up in the boot list until it connects to an iSCSI target. So if iSCSI is misconfigured, or the network is down, it does not show up.
2) RAID and SATA HDs cannot be disabled in the boot list. Either disable the controller, but then they are not available as secondary disks, or move them down in the HD boot list.
3) It is recommended to disable the entire HD list from the boot order until iSCSI is on the top, to prevent it from booting into another HD

iSCSI boot using Intel (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands.
- GetRemoteServicesAPIStatus()

B) GetNICViews: ENUMERATE the *DCIM_NICVIEW* class to collect information about the NIC FQDDs.
▫ Check if specified NIC FQDD is present in NICViews

C) GetNICEnumerations: ENUMERATE the *DCIM_NICEnumeration* class to collect information about the system.
▫ Check if TcpIpViaDHCP=Enabled and IscsiViaDHCP=Disabled

D) GetBootSourceSettings: ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the NIC FQDDs.
▫ Loop through all boot sources, if boot source is IPL entry, set CurrentEnabledStatus =0 unless HD [Steps D)-F)]
E) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from C)
▫ Set CurrentEnabledStatus=1 for NIC FQDD boot source

F) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the NICs.
▫ Check the CurrentEnabledStatus state

G) Configure iSCSI
▫ CreateBIOSConfigJob(): Target=(BIOS FQDD)

▫ SetNICAttributes(): Target=(NIC FQDD) Set the attribute LegacyBootProto to the value iSCSIPrimary

▫ CreateNICConfigJob(): Target=(NIC FQDD)

Reboot job:

CreateRebootJob()
▫ Poll jobstatus for Completed using instanceID from CreateNICConfigJob()

Subjob2:

H) Move iSCSI to the top of the HD Boot List by looping through boot sources

▫ GetBootSourceSettings()

▫ ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from GetBootSourceSettings())

I) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the NIC FQDDs.

▫ Loop through boot sources to confirm the NIC FQDD and “BCV” are in an instanceID

J) Set NIC to first in boot order
▫ GetBootSourceSettings()

▫ ChangeBootOrderByInstanceID(): Use InstanceID=BCV and source=(instanceID from GetBootSourceSettings())

▫ CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

Reboot job:

`CreateRebootJob()`
Poll jobstatus for Completed: `GET` the *InstanceID* from BIOS config job

FCoE boot using QLogic (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a "ready" state before executing any other WSMAN commands. The GetRSStatus() method or the GetRemoteServicesAPIStatus() method may be used depending on the version of the LC Management registered profile.

1. System should be power off
2. Clear all unfinished jobs
3. Clear all pending data

B) Check NDC is enabled

1. GetBIOSEnumerations(): ENUMERATE the *DCIM_BIOSEnumeration* class to collect information about the system.

2. Ensure AttributeName of IntegratedNetwork1 is enabled

If it is not enabled, enable it as shown below
- SetBIOSAttributes()

AttributeName=IntegratedNetwork1 AttributeValue=Enabled
AttributeName=BootMode AttributeValue=Bios
- CreateBIOSConfigJob()

Reboot job:

- CreateRebootJob() RebootJobType=1
- Poll jobstatus for Completed: GET the *InstanceID* from 2).

Subjob2:

C) CheckConnectFirstFCoETarge(): ENUMERATE the NIC FADD and check if ConnectFirstFCoETarget is eabled, if not, enable ConnectFirstFCoETarget as show below

- Disable all sources
- Create BIOS job
- SetNICAttributes()

AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled
- CreateNICConfigJob

Reboot job:

CreateRebootJob() with RebootJobType=1
Poll jobstatus for Completed

Subjob3:

D) Configure FCoE

1. Disable all sources
2. Create BIOS job

3. Set Partition Attributes as follows:

SetNICAttributes() on NIC.Integrated.1-1-4

AttributeName=FCoEOffloadMode AttributeValue=Enabled
AttributeName=VirtFIPMacAddr AttributeValue=$VirtFIPMacAddr AttributeName=VirtWWN
AttributeValue=$VirtWWN AttributeName=VirtWWPN AttributeValue=$VirtWWPN
AttributeName=MinBandwidth AttributeValue=$MinBandwidth AttributeName=MaxBandwidth
AttributeValue=$MaxBandwidth
4. CreateNICConfigJob()
5. Set Port Attributes as follows:

SetNICAttributes() on NIC.Integrated.1-1-1
AttributeName=FirstFCoEWWPNTarget AttributeValue=$FirstFCoEWWPNTarget
AttributeName=FirstFCoEBootTargetLUN AttributeValue=$FirstFCoEBootTargetLUN
6. CreateNICConfigJob()

Reboot job:

CreateRebootJob() with RebootJobType=1
Poll jobstatus for Completed

Subjob4:

E) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

Loop through all boot sources, if boot source is IPL entry, set EnabledState=0 unless HD.
F) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from D)

G) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
H) Enable the HD boot source

I) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from F)

GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
Change NIC boot source
J) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

Check NIC boot order
K) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from I)

SetNICAttributes(): Set the attribute LegacyBootProto to the value “FCoE” and the other desired NIC attributes and values
L) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

Reboot job:

CreateRebootJob() RebootJobType=1
M) Poll jobstatus for Completed: GET the *InstanceID* from F).

FCoE boot using Intel (12G only)

Subjob1:

A) The Lifecycle Controller remote service must be in a “ready” state before executing any other WSMAN commands. GetRemoteServicesAPIStatus():

1. System should be power off
2. Clear all unfinished jobs
3. Clear all pending data

B) Check NIC is enabled

1. GetBIOSEnumerations(): ENUMERATE the *DCIM_BIOSEnumeration* class to collect information about the system.

2. Ensure AttributeName of Slot2 is enabled

If it is not enabled, enable it as shown below
SetBIOSAttributes()
- AttributeName= Slot2 AttributeValue=Enabled
- AttributeName=BootMode AttributeValue=Bios
- CreateBIOSConfigJob()

Reboot job:

- CreateRebootJob() RebootJobType=1
- Poll jobstatus for Completed: GET the *InstanceID* from 2).

Subjob2:

C) CheckConnectFirstFCoETarge(): ENUMERATE the NIC FADD and check if ConnectFirstFCoETarget is eabled, if not, enable ConnectFirstFCoETarget as show below

- SetNICAttributes()

AttributeName=LegacyBootProto AttributeValue=FCoE AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled
- Disable all sources

- Create BIOS job

- SetNICAttributes()

AttributeName=ConnectFirstFCoETarget AttributeValue=Enabled
- CreateNICConfigJob

Reboot job:

CreateRebootJob() with RebootJobType=1
Poll jobstatus for Completed

Subjob3:

D) Configure FCoE

1. Disable all sources
2. Create BIOS job
3. Set Attributes (VLAN etc) as follows
- SetNICAttributes() on NIC.Mezzanine.2B-1

AttributeName=FCoEOffloadMode AttributeValue=Enabled

AttributeName=VirtFIPMacAddr AttributeValue=$VirtFIPMacAddr AttributeName=VirtWWN AttributeValue=$VirtWWN AttributeName=VirtWWPN AttributeValue=$VirtWWPN AttributeName=MinBandwidth AttributeValue=$MinBandwidth AttributeName=MaxBandwidth AttributeValue=$MaxBandwidth
4. CreateNICConfigJob()

5. Set Attributes (target)as follows
▫ SetNICAttributes() on NIC.Mezzanine.2B-1

AttributeName=FirstFCoEWWPNTarget AttributeValue=$FirstFCoEWWPNTarget AttributeName=FirstFCoEBootTargetLUN AttributeValue=$FirstFCoEBootTargetLUN
6. CreateNICConfigJob()

Reboot job:

CreateRebootJob() with RebootJobType=1
Poll jobstatus for Completed

Subjob4:

E) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

Loop through all boot sources, if boot source is IPL entry, set EnabledState=0 unless HD.
F) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=0 source=(instanceID from D)

G) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

H) Enable the HD boot source

I) ChangeBootSourceState(): Use InstanceID=IPL EnabledState=1 source=(instanceID from F)

GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.
Change NIC boot source
J) GetBootSourceSettings(): ENUMERATE the *DCIM_BootSourceSetting* class to collect information about the boot sources.

Check NIC boot order
K) ChangeBootOrderByInstanceID(): Use InstanceID=IPL source=(instanceID from I)

SetNICAttributes(): Set the attribute LegacyBootProto to the value “FCoE” and the other desired NIC attributes and values
L) CreateBIOSConfigJob(): Use Target=(BIOS FQDD)

Reboot job:

CreateRebootJob() RebootJobType=1
M) Poll jobstatus for Completed: GET the *InstanceID* of from F).

References

[1] Lifecycle Controller Best Practice Specification
http://www.delltechcenter.com/page/Lifecycle+Controller

Glossary

Acronym	Description
CSIOR	Collect System Inventory on Restart
BIOS	Basic Input / Output System
NIC	Network Interface Controller
NDC	Network Daughter Card
iDRAC	Integrated DELL Remote Access Controller
LC2	LifeCycle Controller 2