Managing Dell EMC Hardware with the OpenStack Ironic iDRAC Driver

May 2020
H17999.1

Technical Guide

Abstract
This technical guide describes how to manage Dell EMC server hardware in an OpenStack environment, using OpenStack Ironic with the iDRAC driver.

Dell EMC Service Provider Solutions
Copyright

The information in this publication is provided as is. Dell Inc. makes no representations or warranties of any kind with respect to the information in this publication, and specifically disclaims implied warranties of merchantability or fitness for a particular purpose.

Use, copying, and distribution of any software described in this publication requires an applicable software license.

Copyright © 2019-2020 Dell Inc. or its subsidiaries. All Rights Reserved. Dell Technologies, Dell, EMC, Dell EMC and other trademarks are trademarks of Dell Inc. or its subsidiaries. Intel, the Intel logo, the Intel Inside logo and Xeon are trademarks of Intel Corporation in the U.S. and/or other countries. Other trademarks may be trademarks of their respective owners.

Published in the USA 05/20 Technical Guide H17999.1.

Dell Inc. believes the information in this document is accurate as of its publication date. The information is subject to change without notice.
Contents

Chapter 1  Introduction ................................................. 4
Executive summary................................................................. 5
Revisions ........................................................................... 6
We value your feedback ....................................................... 6

Chapter 2  Getting started ............................................... 7
Overview ........................................................................... 8
Supported hardware ............................................................... 8
Hardware types ................................................................... 9
Supported firmware ............................................................... 10

Chapter 3  Provisioning Dell EMC hardware with the iDRAC driver ........................................ 11
Overview ........................................................................... 12
Prerequisites ........................................................................ 12
Enroll nodes .......................................................................... 12

Chapter 4  OpenStack Ussuri release deliverables ................................................................. 21
Ussuri deliverables ................................................................. 22

Chapter 5  Summary ......................................................... 23
Conclusion ........................................................................... 24

Chapter 6  References ....................................................... 25
Ironic documentation............................................................. 26
iDRAC driver documentation ................................................ 26

Appendix A  Technical support ........................................... 27
Dell EMC hardware technical support ................................... 28
Chapter 1 Introduction

This chapter presents the following topics:

- Executive summary ................................................................. 5
- Revisions .............................................................................. 6
- We value your feedback .......................................................... 6
Executive summary

Cloud operators must manage bare metal server hardware as it is brought into their environments. Ironic is an open-source project within the OpenStack Foundation that provides all bare metal management needs for all platforms and workloads. This document describes how to manage Dell EMC Hardware with the 21st release of OpenStack, known as Ussuri.

- See https://docs.openstack.org/ironic/ussuri/ for more information about the Ironic Ussuri release.

Ironic provides three drivers that can be used to manage Dell EMC hardware:

- iDRAC driver
- Redfish driver
- IPMI driver

These drivers have differing capabilities. See Hardware types on page 9 for more information.

The Ironic iDRAC driver is a component of the open-source project, Ironic. It provides extended capabilities for managing Dell EMC hardware beyond those capabilities that the generic IPMI or Redfish drivers provide.

- See https://docs.openstack.org/ironic/ussuri/admin/drivers/idrac.html for more information about the iDRAC driver.

OpenStack Marketplace

The OpenStack Marketplace offers a searchable, online directory where you can find drivers that manage Dell EMC hardware. You can filter your search by OpenStack project, vendor, or OpenStack release version.

- See https://www.openstack.org/marketplace/drivers to search for drivers.

Third-party CI

Dell EMC provides two continuous integration (CI) methods:

- Ironic
- Sushy

Ironic

Dell EMC provides third-party CI to ensure that all changes to Ironic continue to work with PowerEdge server hardware.

To view the CI results:

1. Browse to https://review.opendev.org/#/q/project:openstack/ironic.n,z.
2. Select a patch.
3. Look in the Dell Ironic CI section for the CI testing results for that patch.

Sushy

In the Ussuri timeframe, Dell EMC added a new CI for the sushy project. Sushy is the Redfish library that the Ironic Redfish and iDRAC drivers use to communicate with the
iDRAC and other BMCs, using the Redfish protocol. This CI ensures that changes to sushy continue to work with Dell EMC hardware.

To view the sushy CI results:

2. Select a patch.
3. Look in the Dell Ironic CI section for the CI testing results for that patch.

Revisions

Table 1. Revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>October 2019</td>
<td>Initial release</td>
</tr>
<tr>
<td>May 2020</td>
<td>Second release</td>
</tr>
</tbody>
</table>

We value your feedback

Dell EMC and the authors of this document welcome your feedback on the solution and the solution documentation. Contact the Dell EMC Solutions team by email or provide your comments by completing our documentation survey.

Author: Dale McDonald, Christopher Dearborn

Contributors: Richard Pioso, Arkady Kanevsky, Kurt Hey

This chapter presents the following topics:

- **Overview** .......................................................... 8
- **Supported hardware** ........................................... 8
- **Hardware types** ................................................... 9
- **Supported firmware** ............................................. 10
Overview

Operators require some basic information before provisioning Dell EMC server hardware with the Ironic iDRAC driver, including supported hardware, hardware types, and firmware. The terms iDRAC driver, and iDRAC hardware type, are used interchangeably throughout this document.

Note: This document assumes that Ironic has been installed. For information about installing Ironic, see https://docs.openstack.org/ironic/ussuri/install/index.html.

Supported hardware

This solution supports the use of Dell EMC PowerEdge rack server solutions.

The iDRAC driver should support Dell EMC PowerEdge R720xd, R620, and DSS 9000 rack servers, though they are no longer validated. Dell EMC cannot guarantee performance and compatibility with the R720xd, R620, or DSS 9000.

Note: Servers with a Chassis Management Controller (CMC) are incompatible with the iDRAC driver, since they lack an iDRAC.

Supported Dell EMC server hardware includes:

- PowerEdge R6515
- PowerEdge R7515
- PowerEdge R640
- PowerEdge R740
- PowerEdge R740xd
- PowerEdge R630
- PowerEdge R730
- PowerEdge R730xd
- PowerEdge XE2420 (tech preview)

While a variety of storage controllers should work, the following have been validated:

- BOSS
- HBA330
- PERC H730(P)
- PERC H740P

While a variety of NICs should work, the following NICs have been validated:

- Intel X710
- Intel XXV710
- Mellanox ConnectX-5
Hardware types

Ironic supports the following hardware types that are shown in Table 2 for managing Dell EMC server hardware.

Table 2. Supported hardware types

<table>
<thead>
<tr>
<th>Hardware type</th>
<th>Comment</th>
</tr>
</thead>
<tbody>
<tr>
<td>ipmi</td>
<td>Fewest features.</td>
</tr>
<tr>
<td>redfish</td>
<td>More features. In active development, new features being added.</td>
</tr>
<tr>
<td>idrac</td>
<td>Most features. Can be configured to use WS-Man, Redfish, or a mix of the two per server.</td>
</tr>
</tbody>
</table>

Table 3 compares the benefits of using either the WS-Man or Redfish protocols with the iDRAC driver.

Table 3. iDRAC driver protocol comparison

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Benefits</th>
</tr>
</thead>
</table>
| WS-Man   | Extended support for deploying nodes using WS-Man, including:  
  • Power control  
  • Boot device management  
  • In-band inspection  
  • Out-of-band inspection  
  • RAID management  
  • BIOS management |
| Redfish  | Basic support for deploying nodes using Redfish, including:  
  • Power control  
  • Boot device management  
  • In-band inspection  
  • Out-of-band inspection  
  Advanced support for deploying nodes using Redfish, including:  
  • Virtual media boot  
  • Boot mode management (UEFI/Legacy) |

Note: The iDRAC driver team is working on upgrading Redfish support to parity with WS-Man support. New features only support Redfish.
Chapter 2: Getting started

Supported firmware

The firmware that is used during validation of this solution includes those versions that are shown in Table 4, Table 5, and Table 6.

Table 4. Supported PowerEdge server firmware versions

<table>
<thead>
<tr>
<th>PowerEdge Servers</th>
<th>Component</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>R6515, R7515</td>
<td>Lifecycle controller</td>
<td>4.10.10.10</td>
</tr>
<tr>
<td></td>
<td>BIOS</td>
<td>1.1.6</td>
</tr>
<tr>
<td>R640, R740, R740xd</td>
<td>Lifecycle controller</td>
<td>4.10.10.10</td>
</tr>
<tr>
<td></td>
<td>BIOS</td>
<td>2.5.4</td>
</tr>
<tr>
<td>R630, R730, R730xd</td>
<td>Lifecycle controller</td>
<td>2.70.70.70</td>
</tr>
<tr>
<td></td>
<td>BIOS</td>
<td>2.11.0</td>
</tr>
</tbody>
</table>

Table 5. Supported storage controller firmware versions

<table>
<thead>
<tr>
<th>Storage controller</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>BOSS-S1 RAID controller</td>
<td>2.5.13.3022</td>
</tr>
<tr>
<td>HBA330 disk controller</td>
<td>16.17.00.05</td>
</tr>
<tr>
<td>PERC H730(P) RAID controller</td>
<td>25.5.6.0009</td>
</tr>
<tr>
<td>PERC H740P RAID controller</td>
<td>50.9.4-3025</td>
</tr>
</tbody>
</table>

Table 6. Supported network interface card firmware versions

<table>
<thead>
<tr>
<th>Network interface card</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel X710</td>
<td>19.0.12</td>
</tr>
<tr>
<td>Intel XXV710</td>
<td>19.0.12</td>
</tr>
<tr>
<td>Mellanox ConnectX-5</td>
<td>16.25.40.62</td>
</tr>
</tbody>
</table>
Chapter 3 Provisioning Dell EMC hardware with the iDRAC driver

This chapter presents the following topics:

- Overview ................................................................. 12
- Prerequisites ......................................................... 12
- Enroll nodes .......................................................... 12
Overview

This chapter provides basic information required to provision Dell EMC server hardware with the Ironic iDRAC driver.

Prerequisites

Some configuration of Ironic is required before provisioning bare metal nodes using the iDRAC driver. This configuration includes:

1. Installing required libraries
2. Enabling the idrac hardware type
3. Enabling selected interfaces

For detailed instructions see the Enabling section of the iDRAC driver documentation, at https://docs.openstack.org/ironic/ussuri/admin/drivers/idrac.html.

4. Avoiding potential timeouts on power off, by increasing `post_deploy_get_power_state_retries` in the `[agent]` section of `/etc/ironic/ironic.conf` to 18. It should look like this example:

```
[agent]
post_deploy_get_power_state_retries = 18
```

5. Preventing potential timeouts on RPC responses, by increasing `rpc_response_timeout` in the `[agent]` section of `/etc/ironic/ironic.conf` to 600. It should look like this example:

```
[agent]
rpc_response_timeout = 600
```

6. Preventing nodes from intermittently going into maintenance mode by setting `sync_power_state_interval` in the `[conductor]` section of `/etc/ironic/ironic.conf` to 70. It should look like this example:

```
[conductor]
sync_power_state_interval = 70
```

Note: Be sure to stop the Ironic conductor service before editing the `ironic.conf` file. Restart the service after you have finished editing.

Enroll nodes

After the initial configuration of Ironic has been completed, nodes can be enrolled into Ironic.

1. Follow the general enrollment process specified at https://docs.openstack.org/ironic/ussuri/install/enrollment.html.
2. Deviations from that process include:
   a. Create nodes on page 13
Chapter 3: Provisioning Dell EMC hardware with the iDRAC driver

b. Select interfaces on page 13

c. Set iDRAC credentials on page 15

Create nodes
When creating a node in Ironic, you must specify the driver that Ironic should use to manage the node.

1. Pass `--driver idrac` to the `openstack baremetal node create` command to manage the node using the iDRAC hardware type.

2. See the Enrolling section of the iDRAC driver documentation located in the Ironic documentation, at https://docs.openstack.org/ironic/ussuri/admin/drivers/idrac.html#enrolling.

Select interfaces
Ironic defines a set of interfaces that every hardware type can implement. Some of these interfaces are required, while others are optional. The iDRAC driver supports the following interfaces:

- **Inspect** - Provides out-of-band inspection capabilities.
- **Management** - Provides boot device management.
- **Power** - Provides power control.
- **RAID** - Provides RAID volume configuration.
- **Vendor** - Provides BIOS settings configuration.
- **Boot** – Provides options for booting ramdisks and user instances.
- **Deploy** – Provides deployment management.
- **Network** – Provides supported network types.
- **Rescue** – Provides supported rescue mode types.
- **Storage** - Provides supported storage interfaces.

Table 6 describes for each interface:

- If WS-Man can be used
- If Redfish can be used
- If there are protocol-independent choices

<table>
<thead>
<tr>
<th>Interface</th>
<th>WS-Man</th>
<th>Redfish</th>
<th>Protocol independent</th>
</tr>
</thead>
<tbody>
<tr>
<td>Inspect</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Management</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Power</td>
<td>X</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>RAID</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Vendor</td>
<td>X</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Boot</td>
<td></td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>
Some interfaces support multiple protocol-independent choices. For the protocol-independent choices that are available for each interface, see the [OpenStack Ironic iDRAC driver documentation](#).

By default, the iDRAC hardware type uses the WS-Man version of these interfaces to manage nodes when it is supported. The Boot interface defaults to iPXE, which is protocol-independent, but virtual media boot using Redfish can be used as well.

Virtual media boot works by:

1. Building an image.
2. Downloading the image to the server.
3. Mounting it in the virtual CD/DVD drive, from which the server then boots.

The transfer of the image uses HTTPS, which avoids using the potentially unreliable TFTP protocol.

**Note:** The WS-Man management and power interfaces must be used as a pair. If you use WS-Man for one of these interfaces, then you must use WS-Man for the other.

Alternatively, Redfish can be used for the interfaces that are indicated above. To use Redfish for an interface:

1. Specify the following parameters to the `openstack baremetal node create` command:
   b. Inspect: `--inspect-interface idrac-redfish`
   c. Management: `--management-interface idrac-redfish`
   d. Power: `--power-interface idrac-redfish`

2. Alternatively, the interfaces to use on a node can be set after node creation.
   a. Get the UUID of the node in Ironic by running the following command:

   ```
   openstack baremetal node list
   ```
Chapter 3: Provisioning Dell EMC hardware with the iDRAC driver

b. Set the interfaces by running the following command:

   openstack baremetal node set <uuid> --inspect-interface \ idrac-redfish

   c. Or, you can set multiple interfaces at once by specifying --xxx-interface and a supported interface multiple times. For example:

   openstack baremetal node set <uuid> \ --power-interface idrac-redfish \ --management-interface idrac-redfish

   **Note:** The WS-Man interface name can be explicitly specified as idrac-wsman.

Set iDRAC credentials

For Ironic to be able to access the iDRAC, you must set the credentials for the iDRAC on the node in Ironic:

- If using the WS-Man interfaces, the WS-Man credentials must be specified.
- If using the Redfish interfaces, the Redfish credentials must be specified.
- If using both the WS-Man interfaces and the Redfish interfaces, both sets of credentials must be specified.

The following parameters are used to specify the WS-Man credentials on the node in Ironic:

- **drac_username** - The WS-Man username to use when communicating with the iDRAC. Usually `root`.
- **drac_password** - The password for the WS-Man user to use when communicating with the iDRAC.
- **drac_address** - The IP address of the iDRAC.

The following parameters are used to specify the Redfish credentials on the node in Ironic:

- **redfish_username** - The Redfish username to use when communicating with the iDRAC. Usually `root`.
Chapter 3: Provisioning Dell EMC hardware with the iDRAC driver

- **redfish_password** - The password for the Redfish user to use when communicating with the iDRAC.
- **redfish_address** - The URL of the iDRAC (https://<ip>).
- **redfish_system_id** - The Redfish ID of the server to be managed. This ID should always be: /redfish/v1/Systems/System.Embedded.1.

**Note:** For other Redfish protocol parameters, see https://docs.openstack.org/ironic/ussuri/admin/drivers/redfish.html.

Credentials can be associated with the node in Ironic either when creating the node or by modifying the node after it has been created:

1. To specify the WS-Man credentials when creating a node in Ironic pass the following parameters to the openstack baremetal node create command:
   
   ```
   --driver-info drac_username=<username> \ 
   --driver-info drac_password=<password> \ 
   --driver-info drac_address=<ip>
   ```

2. To specify the Redfish credentials when creating a node in Ironic, pass the following parameters to the openstack baremetal node create command:

   ```
   --driver-info redfish_username=<username> \ 
   --driver-info redfish_password=<password> \ 
   --driver-info redfish_address=https://<ip> \ 
   --driver-info redfish_system_id=/redfish/v1/Systems/System.Embedded.1
   ```

Alternatively, the credentials for Ironic to use can be set or modified after the node has been created.

1. Get the UUID of the node in Ironic by following Step 2.a on page 14.
2. Set the wanted parameters on the node by running the following command:

   ```
   openstack baremetal node set <uuid> \ 
   --driver-info <parameter_name>=<parameter_value>
   ```

**Note:** You can set multiple parameters at once by specifying --driver-info, and a parameter name and value, multiple times.

### Node creation examples

To put it all together, create a node in Ironic as per the following examples:

- **WS-Man example** on page 16
- **Redfish example** on page 17
- **Redfish and WS-Man example** on page 17

#### WS-Man example

The following example creates a node in Ironic that is managed by the iDRAC hardware type and uses WS-Man for all interfaces:
openstack baremetal node create \
--driver idrac \
--driver-info drac_username=root \
--driver-info drac_password=1234 \
--driver-info drac_address=192.168.110.100

**Redfish example**
The following example creates a node in Ironic that is managed by the iDRAC hardware type and uses Redfish for all interfaces:

openstack baremetal node create \
--driver idrac \
--driver-info redfish_username=root \
--driver-info redfish_password=1234 \
--driver-info redfish_address=https://192.168.110.100 \
--driver-info redfish_system_id=/redfish/v1/Systems/System.Embedded.1 \
--inspect-interface idrac-redfish \
--management-interface idrac-redfish \
--power-interface idrac-redfish \
--vendor-interface no-vendor \
--raid-interface no-raid

**Note:** Redfish support in the iDRAC driver does not currently include BIOS or RAID management.

**Redfish and WS-Man example**
The following example creates a node in Ironic that is managed by the iDRAC hardware type. It uses Redfish for the power, management, and inspect interfaces, and WS-Man for the remaining interfaces:

openstack baremetal node create \
--driver idrac \
--driver-info redfish_username=root \
--driver-info redfish_password=1234 \
--driver-info redfish_address=https://192.168.110.100 \
--driver-info redfish_system_id=/redfish/v1/Systems/System.Embedded.1 \
--driver-info drac_username=root \
--driver-info drac_password=1234 \
--driver-info drac_address=192.168.110.100 \
--inspect-interface idrac-redfish \
--management-interface idrac-redfish \
--power-interface idrac-redfish \
--vendor-interface idrac-redfish \
--raid-interface idrac-redfish

**Monitoring Ironic and iDRAC interactions**
Once the node has been set up in Ironic, interactions between Ironic and the node can be seen by searching in `/var/log/ironic/ironic-conductor.log` for either:

- The node UUID
- The node IP address
Prepare nodes for deployment

To prepare the nodes for deployment you must perform the following procedures:

1. Placing nodes in a known good state on page 18
2. Inspect nodes on page 19

Placing nodes in a known good state

Before deploying the nodes using Ironic, it is recommended that you put the iDRACs into a known good state. You do this task by running the manual known_good_state cleaning step against the node. This cleaning step currently resets the iDRAC and clears the Lifecycle Controller job queue on the targeted node. It may be extended in the future to perform other best practice operations to put the server into the best possible known good state.

Note: These cleaning steps are only available on nodes whose management interface is set to idrac-wsman. Support for these cleaning steps in the idrac-redfish management interface will be added in a future release.

You can use this procedure to return a node to an initial clean state when returning a node to the unallocated bare metal pool.

To run the known_good_state cleaning step:

1. Get the UUID of the target node by following Step 2.a on page 14.
2. Run the following command:
   
   ```
   openstack baremetal node clean --clean-steps '[{"interface": "management", "step": "known_good_state"}]' <uuid>
   ```
3. Monitor the progress of the cleaning step by periodically running the following command, and then watching for the node exiting the cleaning state:
   
   ```
   openstack baremetal node list
   ```

Note: The cleaning step may take a few minutes to complete.

If you do not want to reset the iDRAC and clear the job queue in the same operation, you can run them separately.

To reset the iDRAC:

1. Get the UUID of the target node by following Step 2.a on page 14.
2. Run the command in Step 2 above, substituting reset_idrac for the cleaning step name.
3. Run the command in Step 3 above.

To clear the Lifecycle Controller job queue:

1. Get the UUID of the target node by following Step 2.a on page 14.
2. Run the command in Step 2 above, substituting clear_job_queue for the cleaning step name.
3. Run the command in Step 3 above.

**Inspect nodes**

You can inspect nodes using either:

- **In-band inspection** - Causes the node to PXE boot an inspection RAM disk hosted by Ironic.
- **Out-of-band inspection** - Ironic communicates directly with the iDRAC through the driver that is configured on the node.

**In-band inspection**

To inspect a node using in-band inspection:

1. Set the inspect interface to `inspector` by running the following command:

```
openstack baremetal node set <uuid> --inspect-interface inspector
```

2. Run the following command:

```
openstack baremetal node inspect <uuid>
```

**Out-of-band inspection**

Because out-of-band inspection in Ironic does not require a server reboot, it runs quicker than in-band introspection.

- **Using the idrac-wsman inspect interface**
- **Using the idrac-redfish inspect interface**

**Using the idrac-wsman inspect interface**

1. Set the inspect interface to `idrac-wsman` by running the following command:

```
openstack baremetal node set <uuid> --inspect-interface \ idrac-wsman
```

2. Run the following command:

```
openstack baremetal node inspect <uuid>
```

**Using the idrac-redfish inspect interface**

1. Set the inspect interface to `idrac-redfish` by running the following command:

```
openstack baremetal node set <uuid> --inspect-interface \ idrac-redfish
```

2. Run the following command:

```
openstack baremetal node inspect <uuid>
```

**Enable PXE booting on the node’s port in Ironic**

Each interface can PXE-enable the node’s port in Ironic:

- **Using the idrac-wsman inspect interface**
- **Using the idrac-redfish inspect interface**
Chapter 3: Provisioning Dell EMC hardware with the iDRAC driver

**Using the idrac-wsman inspect interface**

The `idrac-wsman` inspect interface automatically PXE enables the correct port.

**Using the idrac-redfish inspect interface**

If the `idrac-redfish` inspect interface is used, then you must PXE-enable the node port in Ironic prior to deployment. The MAC address of the port must be set to PXE boot on the server.

First, determine the MAC address of the port on the server that is configured to PXE boot:

1. Log in to the server iDRAC GUI.
   a. Browse to **Configuration > BIOS Settings > Boot Setting**, and then examine the **Boot Mode**.
   b. If the Boot Mode is **UEFI**, then skip to step 2 below.
   c. If the Boot Mode is **BIOS**, then expand **BIOS Boot Settings** and examine the **Boot Sequence**. Note the name of the PXE port.
   d. Skip to step 4 below.
2. Examine the **Current Value** setting for **PXE Device1** through **PXE Device4**.
   a. Note the PXE device whose value is **Enabled**.
3. Expand the **PXE Device** settings for that device.
   a. The name of the PXE port is displayed in the **Interface** field.
4. Browse to **System > Inventory > Hardware Inventory** to find the port with that name.

---

**Note:** The **CurrentMACAddress** field displays the MAC address of the PXE port.

---

5. Find the port associated with the node in Ironic that has that MAC address by running the following command:

   ```bash
   openstack baremetal port list --node <node_uuid> | grep -i <MAC_address_found_above>
   ``
   a. The output displays the UUID of the port.
6. Enable PXE booting on the port by running the following command:

   ```bash
   openstack baremetal port set --pxe-enabled <port_uuid>
   ``

---

**Note:** Automatically enabling PXE booting on the port in the `idrac-redfish inspect` interface will be added in a future release.
This chapter presents the following topics:

**Ussuri deliverables** .............................................................. 22
Ussuri deliverables

Dell EMC has added the following enhancements to the OpenStack Ussuri release.

- Automatic conversion of drives between JBOD and RAID mode when creating or deleting virtual disks on a PERC H730(P) RAID controller
- Continued support of continuous integration testing on Dell EMC hardware in Ironic
- Support of continuous integration testing in the sushy project on Dell EMC hardware
- Support for virtual media boot when using the `idrac-redfish-virtual-media` boot interface
- Support for boot mode management
- Support for Dell EMC PowerEdge XE2420 servers (tech preview)
- Support for AMD-based Dell EMC servers:
  - PowerEdge R6515
  - PowerEdge R7515
- Support for Redfish with the following Dell EMC PowerEdge servers:
  - R630
  - R730
  - R730xd
Chapter 5 Summary

This chapter presents the following topics:

Conclusion ........................................................................................................ 24
Conclusion

While the generic Ironic IPMI and Redfish drivers can be used to manage Dell EMC hardware, the Ironic iDRAC driver provides more features and flexibility.

As the industry transitions to Redfish, the Ironic iDRAC driver is transitioning to Redfish as well. The iDRAC driver now enables operators to select using the Redfish protocol for some Ironic interfaces, while using the existing WS-Management protocol for other Ironic interfaces. Operators can start using Redfish while still retaining access to the extended WS-Management RAID and BIOS configuration capabilities.

Future releases of the iDRAC driver will upgrade Redfish support to parity with WS-Management support.
This chapter presents the following topics:

Ironic documentation ........................................................................................................26
iDRAC driver documentation ...........................................................................................26
Chapter 6: References

**Ironic documentation**

The following Ironic documentation provides additional and relevant information:
- [https://docs.openstack.org/ironic/ussuri/](https://docs.openstack.org/ironic/ussuri/)

**iDRAC driver documentation**

The following iDRAC driver documentation provides additional and relevant information:
- [https://docs.openstack.org/ironic/ussuri/admin/drivers/idrac.html](https://docs.openstack.org/ironic/ussuri/admin/drivers/idrac.html)
This appendix presents the following topics:

Dell EMC hardware technical support ............................................................. 28
Dell EMC hardware technical support

You can obtain technical support for Dell EMC hardware solutions directly from Dell EMC.

Dell.com/support is focused on meeting customer needs with proven services and support.

Technical documents and videos provide expertise that helps to ensure customer success on:

- Dell EMC Server platforms
- Dell EMC Storage platforms
- Dell EMC Networking platforms