



**OPEN**  
Compute Project

**Open Hardware Management  
Proposed Specification  
For  
Server Remote Management  
Version 0.8**

## Revision History

Date	Name	Description
1/16/2012	Grant Richard	Version 0.5 draft for general discussion.
2/14/2012	Grant Richard	Revised document based on early feedback.
3/17/2012	Grant Richard	Added comments from conference call 2/28
3/25/2012	Grant Richard	Incorporated comments from conference call and updated section Remote Specifications.
4/1/2012	Grant Richard	Incorporated Matthew Liste feedback.
4/1/2012	Matthew Liste	General scrub and some reorganization.

DRAFT

# Contents

---

1. Summary	5
2. License	7
3. Approach	8
4. Scale out – Stateless Machine System Administration Tasks	9
4.1 Initial Machine Provisioning	10
Issues	10
4.2 Default Remote Console Name	11
4.3 Rights and Credentials	11
4.4 Reset machine BMC to defaults	12
4.5 Unknown/wrong boot order or make sure the PXE is first in the boot order	12
4.6 Something odd is happening and there is a need to look at the machine's console	13
Tasks	13
4.7 Hung Machine	13
Issues	13
Remotely power off/recycle a machine	13
Signal OS to bring machine down	14
4.8 How much power is a server drawing?	14
Issues	14
4.9 Is the server being cooled effectively?	15
Issues	15
4.10 Need to provide security around actions	15
Issue	15
4.11 Need to understand what is in the machine?	16
Issue	16
Base Inventory	16
Complete Inventory	16
4.12 What happened on the machine before it crashed?	16
Issue	16
4.13 Discussed but not included	17
5. Remote Machine Management Specifications	18

5.1 Standards implementation	18
5.1.1 IPMI	18
5.1.2 DCMI	18
5.2 Specification	19
5.2.1 General Requirements - DCMI 1.5	19
5.2.2 Security	19
5.2.3 Interface Name and Discovery	19
5.2.4 Event Logging	19
5.2.5 Power	19
5.2.6 Serial Console over LAN	19
5.2.7 Inventory	20
5.2.8 Temperature	20
5.2.9 Boot Control	20
5.3 Commands Supported	20
5.4 BMCs	21
5.5 Network Interface	21

DRAFT

# 1. Summary

---

Scale computing requires a small and stable set of tools to remotely management machines. In this specification, the strategy is to define these tools by looking at the needs of a scale-out/stateless machine system administrator to trouble shoot and provision machines. These use cases tease out the actual requirements and prevents features non-essential feature out of the specification.

The specification is divided into three sections:

1. Approach
2. Scale-out / Stateless Machine System Administrator Tasks and recommended best practices
3. Remote Machine Management Specification

Upon adoption of the specification, a reference implementation will be selected and a validation suite will be produce. Examples of the scale-out SA tasks are:

- Initial machine provisioning
- Reset machine BMC to defaults
- Wrong boot order
- Odd machine behavior
- Hung Machine
- Understanding power and cooling
- Security and authorizing systems admins
- Inventory
- Machine crash analysis

For each scale-out task, there are recommended tool/functional best practices that will be used with existing standards. As well, there are some infrastructure requirements implicit in these best practices. Please note that when matching the recommendations, some of the aspects are minor features that may be dropped, or some functions may have more features than required.

For the specification part, the approach is to look at existing standards and leverage them. There were two relevant standards: IPMI 2.0/DCMI 1.5 and DMTF/SMASH -- although both standards are over featured for scale-out computing. After discussion, a subset of the DCMI 1.5 was used as it has the broadest adoption and existing commercial silicon. Discussions will continue on how to incorporate SMASH into the specification for an updated specification.

Special thanks are extended to the many people who helped contribute to this specification though email, conference call and individual calls. As well, representatives

for IPMI 2.0/DCMI 1.5 and DMTF were very generous and gracious with their time. This could have been done without all of you.

DRAFT

## 2. License

---

As of April 7, 2011, the following persons or entities have made this Specification available under the Open Web Foundation Final Specification Agreement (OWFa 1.0), which is available at <http://www.openwebfoundation.org/legal/the-owf-1-0-agreements/owfa-1-0>:

Facebook, Inc.

You can review the signed copies of the Open Web Foundation Agreement Version 1.0 for this Specification at <http://opencompute.org/licensing/>, which may also include additional parties to those listed above.

Your use of this Specification may be subject to other third party rights. THIS SPECIFICATION IS PROVIDED "AS IS." The contributors expressly disclaim any warranties (express, implied, or otherwise), including implied warranties of merchantability, noninfringement, fitness for a particular purpose, or title, related to the Specification. The entire risk as to implementing or otherwise using the Specification is assumed by the Specification implementer and user. IN NO EVENT WILL ANY PARTY BE LIABLE TO ANY OTHER PARTY FOR LOST PROFITS OR ANY FORM OF INDIRECT, SPECIAL, INCIDENTAL, OR CONSEQUENTIAL DAMAGES OF ANY CHARACTER FROM ANY CAUSES OF ACTION OF ANY KIND WITH RESPECT TO THIS SPECIFICATION OR ITS GOVERNING AGREEMENT, WHETHER BASED ON BREACH OF CONTRACT, TORT (INCLUDING NEGLIGENCE), OR OTHERWISE, AND WHETHER OR NOT THE OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

## 3. Approach

---

Our approach to delivering and maintaining remote management project are:

- Categorize the remote management capabilities into two sets: 1) *Base* that must be present in all machines and 2) *Extended* that may vary from platform to platform but will always have a uniform interface and performance.
- The Base functions will be modeled against what is absolutely necessary for system administration and management for scale-out and stateless node requirements. All other features will be targeted to an OSs agent or orchestration layer to reduce complexity and cost in the BMC.
- Any solution must be uniformly implemented across platforms and architectures include but not limited to Intel, AMD and ARM.
- Survey and use existing remote management technologies and implementations to determine the best technology to leverage. Identify gaps between Remote Hardware Management and existing standards. This will provide command line and API/interfaces.
  - DMTF – SMBIOS, CIM, WBEM, SMASH/CLP, ASF, WS-Man, SMI-S
  - Intel (with Partnerships) – DCMI, IPMI, AMT
- Coordinate with other Open Compute tracks – especially motherboard, rack and data center designs.
- As part of the every deliverable, a methodology to validate the functionality and maintain validity thereof must be included.
- As it is common to have some non-scale platforms in a scale environment, encourage closed source and specialty hardware manufacturers to comply with Open Hardware Management and to submit those platforms for validation.
- For every tool or function, make sure that security is considered to insure that systems are not compromised by using these tools.

## 4. Scale out – Stateless Machine System Administration Tasks

Remote management’s base requirements will be established by using the most flexible and minimal administration for scale-out compute machines – stateless nodes.

Stateless compute nodes are chosen as:

- Stateless nodes are used in groups where node failure recovery is part of the architecture.
- Administrative requirements are uniform across stateless nodes.
- Pools of stateless nodes typically have very similar (or identical) OS/software distributed across similar pools of hardware so software faults are either epidemic or “random”.
- The most common strategy for fixing faulted stateless nodes is either rebuilding the OS/software or replacing the node.

Because of this, scale-out stateless node administration requirements are light and any scale-out platform will need the same capabilities.

In the section below, the requirements needed for scale-out stateless nodes will be discussed. These are capabilities such as machine provisioning, faults, inventory, power and temperature. Out of the discussion below there are recommended best practices at this stage. The goal is to over time evolve these into firm standards. The recommendations are summarized in the table below, and discussed in more detail in the remainder of this chapter

Item	Recommendation
Initial Machine Provisioning	Predictable defaults settings for all Remote Management functions shall be established. It is acceptable that users can ask their server delivery vendors to alter these standards to their company why defaults.
Default Remote Console Name	The default console name will be an OEM identifier and OEM unique identifier. Example of OEM unique identifiers would be serial number or asset tag. In any event, the combination of the two must result in a unique string.
Rights and Credentials	The default remote console password will support authentication, integrity and confidentiality. As part of the initial configuration, the default password must be reset to insure continued secure operations. It is acceptable that users may work with their server suppliers to change the default remote console password to their own standard. This is acceptable and tooling around remote management interface configuration should provide for this.
Reset BMC to defaults	Resetting the machine’s BMC to defaults is not part of the Base standard. It will be available through a BMC firmware configuration push or should be as part of the

	orchestration layer.
<b>Change Boot Order to provide PXE boot.</b>	Include in the Base specification is the ability to change the boot order. All NICs on the system must boot PXE.
<b>Remote Console</b>	The Base standard will include a Remote Character Console over LAN that will interaction with a machines boot process and access to the installed OS.
<b>Remote power on/off</b>	The Base standard will include the ability to remotely power on and off a machine. This action will be analogous to removing or plugging-in the power cable.
<b>OS signal for soft boot</b>	Having a function that would first try to bring the machines down from an OS perspective was discussed and it was recommended that this not be part of the Base specification. Although a “Must Reboot” or “Must Power down” function is important, this function belonged in a machine maintenance orchestration layer and not as part of the BMC.
<b>Power draw information</b>	For the base specification, the current power draw will be immediately available and the daily minimum/maximum power draws will be able for the past 28 days. Other features, like understanding total kw consumed over a period of time for internal chargeback or more extensive logging can be accomplished via an agent or orchestration layer.
<b>Temperature probes</b>	<ul style="list-style-type: none"> <li>- Probes should be standardized on in-take, CPU and ambient motherboard temperatures probes.</li> <li>- Through a remote interface, these temperatures must be immediately accessed.</li> <li>- The lowest, average and highest temperatures will be recorded in a log to facilitate failure analysis with rolling 28 day retention.</li> </ul>
<b>Security Roles</b>	As part of the security specification, there will be at least three levels of authentication/authorization: 1) BMC administrator, 2) Machine administrator and 3) Machine Viewer. The levels will be hierarchical so that Machine administrator has the rights of a Machine Viewer. As part of OCP common practice, none of these passwords should be the default.
<b>Basic Inventory</b>	As part of the Base, these will be Asset Tag, Device ID, GUID, Manufacturer, Firmware/Software information and Management Controller ID. The minimum list must be able to get requested and received in one command.
<b>Extended Inventory</b>	For a complete/extended inventory, OCP Remotely Managed machines will have an OS based agent like SMBIOS. This agent’s specification is out of scope.
<b>Logging server faults/data</b>	As part of the Base features, a small log will exists outside the OS that can contain information about hardware faults like NMI, multi-parity error, temperature, power and other machine diagnostic/state information.

## 4.1 Initial Machine Provisioning

### *Issues*

Initial provisioning of a new node requires knowing the machine / remote management agent’s name to begin configuration or that the node has its configuration pre-set by the

vendor or integrator. Frequently, scale-out infrastructures find the machine's remote management interfaces a variety of ways but they are often fragile as it requires either:

- *MAC address and/or other inventory information feeds/bar codes from the machine's manufacturer*
- *Having someone attach a keyboard-video-mouse (KVM) to inventory and/or setup the machine*
- *Developing heuristics and searching through IP lease information to find "new" machines that have gotten a DHCP leases for their management interfaces.*

As well, the default authentication for the interface for each manufacturer is well known. This makes gaining login credentials uniform but, many times, this information is not changed or inconsistently changed.

Another credential strategy is to have integrators set this to a scale-out consumer's standard. This is not ideal as this can be error prone and makes on-boarding a new vendor more difficult.

**Recommendation:** Predictable defaults settings for all Remote Management functions shall be established. It is acceptable that users can ask their server delivery vendors to alter these standards to their respective defaults

## 4.2 Default Remote Console Name

There should be a process that has a standard and unique default name for a machine's BMC and, as part of the configuration, this name is changed to provide a workable and straight forward implementation. Suggestions for default names have been:

- *Default name should always be manufacture's OID '-Serial Number*
- *OID + serial number*
- *OID + MAC*
- *GUID (but raises concerns raised about Tier 2 setting this consistently)*

**Recommendation:** The default console names will be an OEM identifier and OEM unique identifier. Example of OEM unique identifiers would be serial number or asset tag. In any event, the combination of the two must result in a unique string.

## 4.3 Rights and Credentials

There should be consistent initial authentication for the BMC across scale-out machines with two strategies:

1. A standardized password across all OCP machines. Examples could be:

- OID + Serial Number
  - Hash of different read only system identifiers like OID and MAC.
2. A password that is provided by the scale purchaser and is pre-configured.
    - a. For example, scale user XYZ, may as that all machine coming into their environment have all of their administrator's account to their remote management interface set to *passwordXYZ*

For the scale out and stateless node, requiring LDAP, AD or another directory authentication is too heavy weight for a minimum standard.

It would also be ideal to force a password change after the initial configuration but this is better left to the orchestration layer rather than including into the BMC.

**Recommendation:** The default remote console password will support authentication, integrity and confidentiality. As part of the initial configuration, the default password must be reset to insure continued secure operations. It is acceptable that users may work with their server suppliers to change the default remote console password to their own standard. This is acceptable and tooling around remote management interface configuration should provide for this.

#### 4.4 Reset machine BMC to defaults

When redeploying a node, there is a requirement to reset the machine's Remote Management interface to its default.

**Recommendation:** Resetting the machine's BMC to defaults is not part of the Base standard. It will be available through a BMC firmware configuration push or should be as part of the orchestration layer.

#### 4.5 Unknown/wrong boot order or make sure the PXE is first in the boot order

Most scale environments leverage PXE for image load and maintenance by loading a maintenance/disposable OS that controls OS load/system maintenance. Without PXE being first, the hard drive/cd/ect will boot with an OS, and maintenance events cannot take place.

There are two options to the boot issue that are widely deployed. This first is to require PXE to always be in the first option in the boot sequence. With that strategy, there is no requirement to have any control of boot options – as less is often best. But, the issues with this are longer boot times and opening to attack by compromising PXE responders. The alternate is to have a remote access method to change boot options. This is possible but will introduce more features into the BMC.

Ideally, all NICs will be PXE enabled. The advantage to this is that no matter which NIC is connected, or however multiple NIC are enumerated, there will always be a successful PXE boot. With the PXE boot and an OS environment, more details about the machine can be discovered and any errors corrected.

**Recommendation:** Include in the Base specification the ability to change the boot order. All NICs on the system must boot PXE.

## 4.6 Something odd is happening and there is a need to look at the machine's console

In general, there is a need to understand what is going on with the machine even if the OS is not accessible or available. For example, this is useful for emerging problems or with boot issues after firmware updates. All of this can be accomplished by re-directing the serial console.

### *Tasks*

- Remote character console is essential for resolving or researching faults or other inconsistent behavior. Examples would be tailing logs, running sar/top and watching boot POST.
- Understanding what that machine resource are being used vs what is in the machine can be important. For example, a machine may be operating on 48GB of memory yet there is 64GB of memory physically installed. Inventory is more widely discussed later in the document.

**Recommendation:** The Base standard will include a Remote Character Console over LAN that will interaction with a machines boot process and access to the installed OS.

## 4.7 Hung Machine

### *Issues*

Need to hard power off/on without someone visiting the machine or having to purchase smart/ip addressable power strips.

### *Remotely power off/recycle a machine*

Call program to hard power off or hard power recycle a machine that the OS can't be logged into

**Recommendation:** The Base standard will include the ability to remotely power on and off a machine. This action will be analogous to removing or plugging-in the power cable.

## ***Signal OS to bring machine down***

Having a single comment to bring down a machine by first trying to shut down the OS and, if that doesn't work, by removing the power, is of value. The complexity comes in at the OS level where the BMC must signal the OS to start shutdown. This soft power signal needs to be correctly implemented and application/services need to be brought down gracefully. As well, in many stateless and scale environments, the hard power down is acceptable without first attempting soft power down.

**Recommendation:** Having a function that would first try to bring the machines down from an OS perspective was discussed and it was recommended that this not be part of the Base specification. Although a "Must Reboot" or "Must Power down" function is important, this function belonged in a machine maintenance orchestration layer and not as part of the BMC.

## **4.8 How much power is a server drawing?**

### ***Issues***

There are a number of reasons to check the power draw for a machine. A few are:

- Understanding minimum and peak workload power draw across a cabinet(s) or group of machines.
- Need to understand glitches for changing workload and/or changes in aggregate workload behaviors.
- Power strip (MOA) circuit breaker tripping

For scale out nodes, the requirements were for current voltage and, if possible, minimum and maximum power draws over a period of time (for example in the past 24 hours). For diagnostics, having the minimum/maximum data saved in the log would be ideal.

**Recommendation:** For the base specification, the current power draw will be immediately available and the daily minimum/maximum power draws will be available for the past 28 days. Other features, like understanding total kw consumed over a period of time for internal chargeback or more extensive logging can be accomplished via an agent or orchestration layer.

## 4.9 Is the server being cooled effectively?

### *Issues*

As part of the total data center environment, understanding server temperatures is necessary to tune the datacenter and make the most efficient use of cooling. As well, with rising data center temperatures, having good temperature data becomes more critical. Finally, during failure situations, understanding temperature is an important data point to know if a server needs to be powered down.

**Recommendation:** As part of the base standard, the following features are required:

- Probes should be standardized on in-take, CPU and ambient motherboard temperatures probes.
- Through a remote interface, these temperatures must be immediately accessed.
- The lowest, average and highest temperatures will be recorded in a log to facilitate failure analysis with rolling 28 day retention.

It was identified that there must be temperature alarms both for absolute temperature and temperature changes over time and these will be communicated to the Event/Alerts/Logging project.

## 4.10 Need to provide security around actions

### *Issue*

Many of the remote management actions can be disruptive. These are remote actions such as power off and boot order changes. User name and password should be used to protect against:

- Users (internal or external) that want to cause mischief
- Script that are malformed or buggy.

In general, all accesses to the machines should be governed by a password, as even the read only attribute may provide enough information to become a security liability.

Furthermore, the default account names and password are often not changed since, at scale, this is often a formidable task.

**Recommendation:** As part of the security specification, there will be at least three levels of authentication/authorization: 1) BMC administrator, 2) Machine administrator and 3) Machine Viewer. The levels will be hierarchical so that Machine administrator has the

rights of a Machine Viewer. As part of OCP common practice, none of these passwords should be the default.

## 4.11 Need to understand what is in the machine?

### *Issue*

In the event that inventory systems are not correct or incomplete, is it necessary to understand what is in the machine. This is not always possible and some devices are enumerated differently between vendors.

### *Base Inventory*

The BMC will return the very minimum about of information required to remote configure and administer the machine.

**Recommendation:** As part of the Base, these will be:

- Asset Tag
- Device ID
- GUID
- Manufacturer
- Firmware/Software information
- Management Controller ID

The minimum list must be able to get requested and received in one command.

### *Complete Inventory*

In scale computing with many similar nodes, there is not the variety of hardware that is found in traditional business compute so complete inventory is less critical. Yet, for scale computing, knowing exactly what the machine has vs. what is reported becomes a valuable diagnostic tool.

**Recommendation:** For a complete/extended inventory, OCP Remotely Managed machines will have an OS based agent similar to SMBIOS. This agent's specification is out of scope.

## 4.12 What happened on the machine before it crashed?

### *Issue*

In scale out environments, machines will appear to fault randomly. Although machine failures are architected into scale environments, often these faults are on the leading

edge of issues, or even a 1% problem over many 10s/100s of thousands of machine is significant.

**Recommendation:** As part of the Base features, a small log will exist outside the OS that can contain information about hardware faults like NMI, multi-parity error, temperature, power and other machine diagnostic/state information.

#### 4.13 Discussed but not included

The following items were discussed but not included in the Base specification.

- A watch-dog timer and/or server heart beat feature was discussed. Since the existing base standards support the primitives behind this, this feature will be left to a Remote Management agent or orchestration layer.
- Scalable updating of firmware was discussed. This is a large enough topic so will be specified in its separate project.

DRAFT

## 5. Remote Machine Management Specifications

---

The approach of OCP Remote Machine Management is to leverage existing standards. The two relevant remote server management standards are IPMI 2.0/DCMI 1.5 and DMTF's SMASH. Both standards encompassing most of the functionality required. The concern is that both have many more features than are required and, with that, maintenance on the "unused" portion has the potential have bugs and/or faults. Understanding this, the Remote Machine Management specification will accept version "subset" version as a trade-off to quicker implementation.

After looking both standards, for this specification, IPMI 2.0/DCMI 1.5 is being used with the following discussion:

- For out of band support, when surveying the market for existing chipsets that would support either standard there was IPMI 2.0/DCMI 1.5 silicon available. A sample of these are Winbond WPCM450, Aspeed 2050 and Intel 5520.
- For in-band support, there are Linux and Windows drivers.
- Almost all OEMs either supported IPMI 2.0/DCMI 1.5 or had add-in technology to support this so scale out scripts can be re-used for non-scale out maintenance and configuration activities.
- Subsequent version of this specification should review SMASH as it is gaining more acceptance with full implementations in BMC's and a major OEM's support.

### 5.1 Standards implementation

- Overall, Interface and protocol needs to be consistent **with** measurements within 1% of tolerance.
- In-band and out-of-band functions need to be supported.

#### 5.1.1 IPMI

When citing IPMI, the following document is referenced.

- IPMI v2.0 rev. 1.0 specification markup for IPMI v2.0/v1.5 errata revision 4 ([http://download.intel.com/design/servers/ipmi/IPMI2\\_OE4\\_Markup\\_061209.pdf](http://download.intel.com/design/servers/ipmi/IPMI2_OE4_Markup_061209.pdf))

#### 5.1.2 DCMI

When citing IPMI, the following document is referenced.

- DCMI Specification version 1.5  
([http://www.intel.com/Assets/PDF/prodspec/DCMI\\_Spec\\_V1\\_5\\_Rev.pdf](http://www.intel.com/Assets/PDF/prodspec/DCMI_Spec_V1_5_Rev.pdf) )

## 5.2 Specification

To avoid replicating the information in the standard, they are referenced by section below. The DCMI specification often references the IPMI specification and this is understood.

### 5.2.1 General Requirements - DCMI 1.5

- Section 5.3.1.2 LAN Interface Requirements
- Section 5.4.1 Mandatory Requirements (Protocol)
- Section 5.2 – General Manageability Access Requirements
  - **Only:** Out-of-band | LAN | SOL over RMCP+ | Single | AEP only
- Section 5.3.1.1 System Interface Requirements
- Section 5.4.1.3 Serial Over LAN Protocol for LAN Access

### 5.2.2 Security

- Security: DCMI 1.5- Section 4.1 Security Access
- User Privilege: DCMI 1.5 Section 4.2

### 5.2.3 Interface Name and Discovery

- DCMI 1.5
    - Section 5.5.1 In-Band Discovery Requirement
    - 5.5.2.1 DHCP enabled management controller
    - 5.5.2.3 RMCP Ping / Pong
- Note: 5.5.2.2 is not included as static IPs are uncommon in scale computing.

### 5.2.4 Event Logging

- 6.3/3.1.3 DCMI Logging

### 5.2.5 Power

Power on / off : DCMI 1.5 Section 3.1.2 Chassis Power  
Power draw: DCMI 1.5 Section 6.6.1 Get Power Reading

### 5.2.6 Serial Console over LAN

- DCMI 5.4.1.3
- Command/protocol DCMI/ Section7.1

### 5.2.7 Inventory

- DCMI: 3.1.1 Identification
- DCMI: 6.4 Identification and Discovery Support

### 5.2.8 Temperature

- Sensor messages DCMI 1.5 / section 3.1.5 compliant
- Command described in DCMI 1.5 / Section 6.7.3 Get Temperature Readings

### 5.2.9 Boot Control

- Standard: DCMI/IPMI Section 6.7.4

## 5.3 Commands Supported

The following command are a subset of the DCMI 1.5 commands as referenced in Table 6.1 and the respective specifications later in the standards document. Command completion codes are specified in DCMI 1.5 Section 8 – Completion Codes.

<b>DCMI Capabilities &amp; Discovery Info</b>
Get DCMI Capabilities
Set & Get DCMI Configuration Parameters
Set & Get Controller ID String
<b>Platform &amp; Asset Identification Commands</b>
Set & Get Asset Tag
Get Device ID
Get System GUID
<b>Boot Control</b>
Set & Get Boot Options
<b>Sensor &amp; SDR Commands</b>
Get DCMI Sensor Info
<b>Logging Command</b>
Get SEL info
Get SEL Entry
Clear SEL
<b>Power Management</b>
Get Power Reading
<b>Thermal Management</b>
Get Temperature Readings
<b>Remote Management</b>
Set & Get LAN Configuration Options
Set & Get Channel Access
Set & Get User Access

Set & Get User Name
Set User Password
Set & Get Payload Access

## 5.4 BMCs

The details around the BMC design and placement on the motherboard may vary between motherboards. BMC and board designers can differentiate themselves by optimizing on power.

## 5.5 Network Interface

The interface to the BMC must be through a shared network port.

DRAFT