



## **Cisco C880 M5 Configuration Guide**

**December 2017**

### **Americas Headquarters**

Cisco Systems, Inc.  
170 West Tasman Drive  
San Jose, CA 95134-1706 USA  
<http://www.cisco.com>  
Tel: 408 526-4000  
800 553-NETS (6387)  
Fax: 408 527-0883

THE SPECIFICATIONS AND INFORMATION REGARDING THE PRODUCTS IN THIS MANUAL ARE SUBJECT TO CHANGE WITHOUT NOTICE. ALL STATEMENTS, INFORMATION, AND RECOMMENDATIONS IN THIS MANUAL ARE BELIEVED TO BE ACCURATE BUT ARE PRESENTED WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED. USERS MUST TAKE FULL RESPONSIBILITY FOR THEIR APPLICATION OF ANY PRODUCTS.

THE SOFTWARE LICENSE AND LIMITED WARRANTY FOR THE ACCOMPANYING PRODUCT ARE SET FORTH IN THE INFORMATION PACKET THAT SHIPPED WITH THE PRODUCT AND ARE INCORPORATED HEREIN BY THIS REFERENCE. IF YOU ARE UNABLE TO LOCATE THE SOFTWARE LICENSE OR LIMITED WARRANTY, CONTACT YOUR CISCO REPRESENTATIVE FOR A COPY.

The Cisco implementation of TCP header compression is an adaptation of a program developed by the University of California, Berkeley (UCB) as part of UCB's public domain version of the UNIX operating system. All rights reserved. Copyright © 1981, Regents of the University of California.

NOTWITHSTANDING ANY OTHER WARRANTY HEREIN, ALL DOCUMENT FILES AND SOFTWARE OF THESE SUPPLIERS ARE PROVIDED "AS IS" WITH ALL FAULTS. CISCO AND THE ABOVE-NAMED SUPPLIERS DISCLAIM ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING, WITHOUT LIMITATION, THOSE OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT OR ARISING FROM A COURSE OF DEALING, USAGE, OR TRADE PRACTICE.

IN NO EVENT SHALL CISCO OR ITS SUPPLIERS BE LIABLE FOR ANY INDIRECT, SPECIAL, CONSEQUENTIAL, OR INCIDENTAL DAMAGES, INCLUDING, WITHOUT LIMITATION, LOST PROFITS OR LOSS OR DAMAGE TO DATA ARISING OUT OF THE USE OR INABILITY TO USE THIS MANUAL, EVEN IF CISCO OR ITS SUPPLIERS HAVE BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

Cisco and the Cisco logo are trademarks or registered trademarks of Cisco and/or its affiliates in the U.S. and other countries. To view a list of Cisco trademarks, go to this URL: [www.cisco.com/go/trademarks](http://www.cisco.com/go/trademarks). Third-party trademarks mentioned are the property of their respective owners. The use of the word partner does not imply a partnership relationship between Cisco and any other company. (1110R)

Any Internet Protocol (IP) addresses used in this document are not intended to be actual addresses. Any examples, command display output, and figures included in the document are shown for illustrative purposes only. Any use of actual IP addresses in illustrative content is unintentional and coincidental.

© 2017 Cisco Systems, Inc. All rights reserved.

---

# Contents

1	Preface.....	5
1.1	Concept and target groups for this manual.....	5
1.2	Documentation overview .....	6
1.3	Notational conventions .....	7
2	IPMI .....	8
2.1	Technical Background .....	8
2.2	IPMI over LAN .....	13
2.3	Supported IPMI OEM Commands.....	14
2.3.1	SCCI-compliant Power On/Off commands .....	15
2.3.2	SCCI-compliant signaling command .....	19
2.3.3	BIOS-specific commands .....	20
2.3.4	iRMC S5-specific commands.....	22
2.4	Data Center Management Interface.....	30
2.5	Serial over LAN (SOL) .....	30
3	iRMC Configuration.....	32
3.1	Configuration via the iRMC web interface .....	32
3.2	Configuration via Remote Manager .....	33
3.3	Configuration via text console redirection .....	34
3.4	Configuration via scripting.....	35
3.5	Configuration via UEFI setup utility .....	36
3.5.1	Configuring the LAN interface.....	37
3.5.2	Configuring text console redirection.....	38
3.5.3	Configuring the serial over LAN.....	40
4	Configuration via Telnet/SSH (Remote Manager).....	42
4.1	Requirements on the managed server .....	42
4.2	Required user permissions .....	43
4.3	Logging in.....	46
4.4	Main menu .....	48
4.4.1	System Information menu.....	49
4.4.2	Power Management menu.....	50
4.4.3	Enclosure Information menu .....	51
4.4.3.1	System Eventlog.....	52
4.4.3.2	Internal Eventlog.....	53
4.4.4	Service processor menu.....	54
4.4.5	Console Redirection (EMS/SAC).....	55
4.4.6	Shell menu.....	55
4.4.7	Console Logging .....	56
4.4.8	Console Logging Run State Menu .....	57

---

---

5	Configuration via scripting .....	58
5.1	REST .....	58
5.2	Profile Management.....	61
5.2.1	Profiles .....	61
5.2.2	Automatic BIOS parameter backup.....	62
5.3	Redfish .....	63
5.4	SCCI .....	66
5.4.1	iRMC configuration data.....	66
5.4.2	SCCI file format .....	67
5.4.2.1	Parameters of SCCI provider-specific commands .....	68
5.4.2.2	Restrictions.....	70
5.4.2.3	Export / import of configuration data .....	71
5.4.3	Supported SCCI commands.....	71
5.4.4	Script Examples .....	72
5.4.4.1	cURL examples .....	72
5.4.4.2	Visual Basic (VB) script.....	73
5.4.4.3	Python script.....	73
6	Monitoring the iRMC.....	75
6.1	Monitoring with SNMP .....	75
6.2	iRMC system report.....	77
6.2.1	cURL script for download.....	77
6.2.2	Visual Basic script.....	78
6.2.3	Information sections .....	79
6.2.3.1	Summary section.....	80
6.2.3.2	BIOS section.....	81
6.2.3.3	Processor section.....	81
6.2.3.4	Memory section .....	82
6.2.3.5	Fans section .....	83
6.2.3.6	Temperature section .....	83
6.2.3.7	Power supplies section.....	84
6.2.3.8	Voltages section .....	84
6.2.3.9	IDPROMS section .....	84
6.2.3.10	SensorDataRecords section.....	84
6.2.3.11	PCIDevices section.....	85
6.2.3.12	SystemEventLog section .....	85
6.2.3.13	InternalEventLog section.....	86
6.2.3.14	BootStatus section .....	86
6.2.3.15	ManagementControllers section.....	86
6.2.3.16	Settings section .....	87
6.2.3.17	Adapters section .....	87
6.2.3.18	Interfaces section .....	87
6.2.3.19	Ports section.....	87
6.2.3.20	SNMPAgents section.....	87

---

# 1 Preface

The scalable CISCO C880 M5 is an Intel-based rack server for critical company scenarios, e.g. as database management system for medium or large-sized databases or as a consolidation basis to run an immensely large number of different applications using virtualization technologies.

Thanks to its highly developed hardware and software components, the server offers a high level of data security and availability. These include hot-plug HDD/SSD modules, hot-plug system fans, and also hot-plug power supply units, Prefailure Detection and Analysis (PDA) and Automatic Server Reconfiguration and Restart (ASR&R).

Security functions in the BIOS Setup and on the System Board protect the data on the server against manipulation. Additional security is provided by the lockable rack door.

The server occupies 5 height units (HU) in the rack.

## 1.1 Concept and target groups for this manual

This manual is aimed at system administrators, network administrators, and service staff who have a sound knowledge of hardware and software. It provides basic information on the concepts of the iRMC and deals with the following aspects in detail:

- ▮ The IPMI chapter comprises the basic principles of the IPMI protocol and the supported OEM commands, that can be used within a script.
- ▮ iRMC Configuration gives an overview of the various possibilities to configure the iRMC.
- ▮ The next chapters describe the configuration of the iRMC in detail using different interfaces:
  - Remote Manager
  - Several APIs
- ▮ Monitoring the iRMC comprises the methods of how to monitor the iRMC.

## 1.2 Documentation overview

More information on your CISCO C880 M5 can be found in the following documents:



- Cisco C880 M5 Installation Manual
- Cisco C880 M5 Configuration Guide
- Cisco C880 M5 Administration Guide
- Cisco C880 M5 User Interface Guide
- Cisco C880 M5 BIOS Setup Guide

**Further sources of information:**

- Manual for the monitor
- Documentation for the boards and drives
- Operating system documentation
- Information files in your operating system

## 1.3 Notational conventions

The following notational conventions are used in this manual:

Notational conventions	Indicates
	Indicates various types of risks, namely health risks, risk of data loss and risk of damage to devices.
	Indicates additional relevant information and tips.
<b>Bold</b>	Indicates references to names of interface elements.
monospace	Indicates system output and system elements, for example file names and paths.
<b>monospace semibold</b>	Indicates statements that are to be entered using the keyboard.
<a href="#">blue continuous text</a>	Indicates a link to a related topic.
<a href="#">purple continuous text</a>	Indicates a link to a location you have already visited.
<abc>	Indicates variables which must be replaced with real values.
[abc]	Indicates options that can be specified (syntax).
[Key]	Indicates a key on your keyboard. If you need to explicitly enter text in uppercase, the Shift key is specified, for example [Shift] + [A] for A. If you need to press two keys at the same time, this is indicated by a plus sign between the two key symbols.

### Screenshots

The screenshots are to some degree system-dependent and consequently will not necessarily match the output on your system in all the details. The menus and their commands can also contain system-dependent differences.

## 2 IPMI

### 2.1 Technical Background

The iRMC makes the BMC (Baseboard Management Controller) functions available over the IPMI interface.

#### **Intelligent Platform Management**

The “Intelligent Platform Management” initiative is a response to the increasing complexity of modern server systems. A number of manufacturers have joined this initiative in order to come up with a new solution for monitoring these server systems.

The term “Intelligent Platform Management” expresses the core aspect of this approach to the solution: Functions for monitoring and recovery of systems are implemented directly in the hardware and firmware for platform management.

#### **Objective**

The objective was to define a standardized, abstract and message-based interface between the central system controller (Baseboard Management Controller - BMC) and intelligent platform management hardware.

The standardization committees combined the central characteristics of various platform management modules into standardized descriptions.

#### **Definition**

The IPMI specification defines:

“IPMI is a hardware level interface specification that is ‘management software neutral’ providing monitoring and control functions that can be exposed through standard management software interfaces such as DMI, WMI, CIM, SNMP, etc. As a hardware level interface, it sits at the bottom of a typical management software stack”.

#### **Advantage**

The IPMI specifications ensure the independence of functions for inventory, logging, recovery and monitoring of a system by the system processor, BIOS or operating system.

This means that a system can still be involved in platform management when it is shut down and turned off.

#### **IPMI and other management standards**

IPMI is best used in conjunction with system management software running under the relevant operating system. Integration of the IPMI functionality into the management functionality offered by a management application and the operating system results in a powerful platform management environment.



An overview of the relationship between IPMI and the management software stack is shown in the following figure:

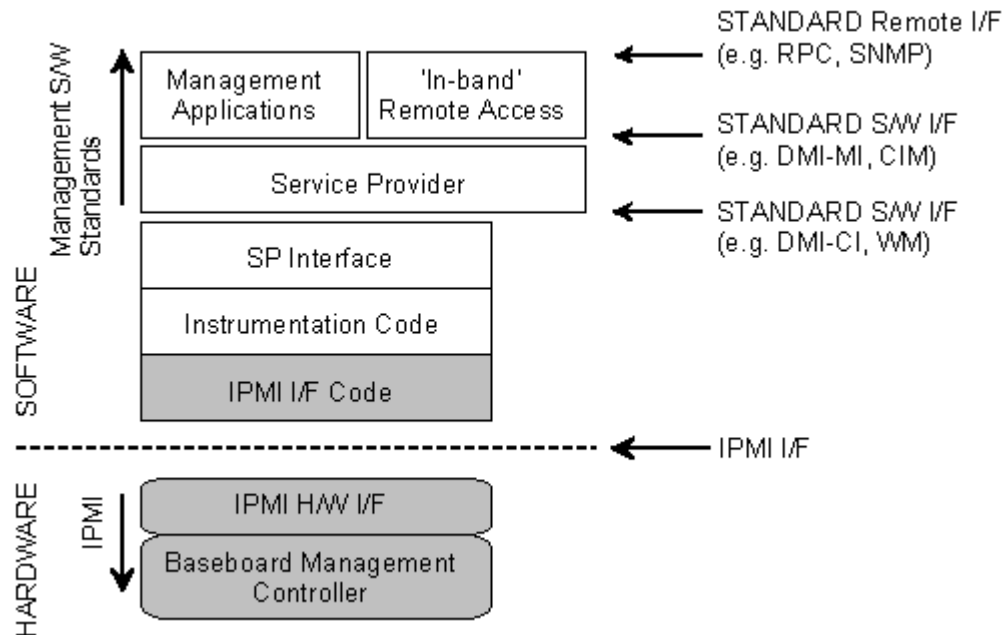


Figure 1: IPMI in the management software stack (source: IPMI specification)

### IPMI, IPMB and ICMB

The IPMI initiative resulted in three central standards:

- ▮ **IPMI: Intelligent Platform Management Interface Specification**  
describes the higher-level architecture, the current commands, event formats, data packets and properties that are used in IPMI-based systems.
- ▮ **IPMB: Intelligent Platform Management Bus**  
is an I<sup>2</sup>C based (write only) bus, which provides a standardized connection between various modules in a common housing.  
  
IPMB can also be used as a standardized interface for remote management modules.
- ▮ **ICMB: Intelligent Chassis Management Bus**  
(Not currently implemented in the remote management environment.)  
  
provides a standardized interface for exchange of platform management information and for control across systems. ICMB is designed in such a way that it can be implemented with a device that is connected to the IPMB.

### IPMI implementation

The core element of an IPMI implementation is the Baseboard Management Controller (BMC).

The BMC performs the following tasks:

- The BMC organizes the interface between the system management software and the platform management hardware.
- It provides autonomous functions for monitoring, event logging and recovery control.
- The BMC acts as a gateway between the system management software and IPMB.

IPMI allows platform management to be extended: Additional management controllers can be connected via the IPMB. The IPMB is an I<sup>2</sup>C based serial bus, which runs between the main modules of the system. It is used for communication with and between the management controllers.

With the support of multiple management controllers, IPMI provides a scalable architecture: A complex server system can use multiple controllers for monitoring different subsystems, e.g. power supplies, hot swap RAID drive modules etc.

In addition, IPMI provides 'low level' I<sup>2</sup>C commands, which can be accessed via a management controller connected to the IPMB on 'unintelligent' I<sup>2</sup>C modules that cannot process IPMI commands.

An overview of the fundamental elements of an IPMI implementation is shown in the following figure:

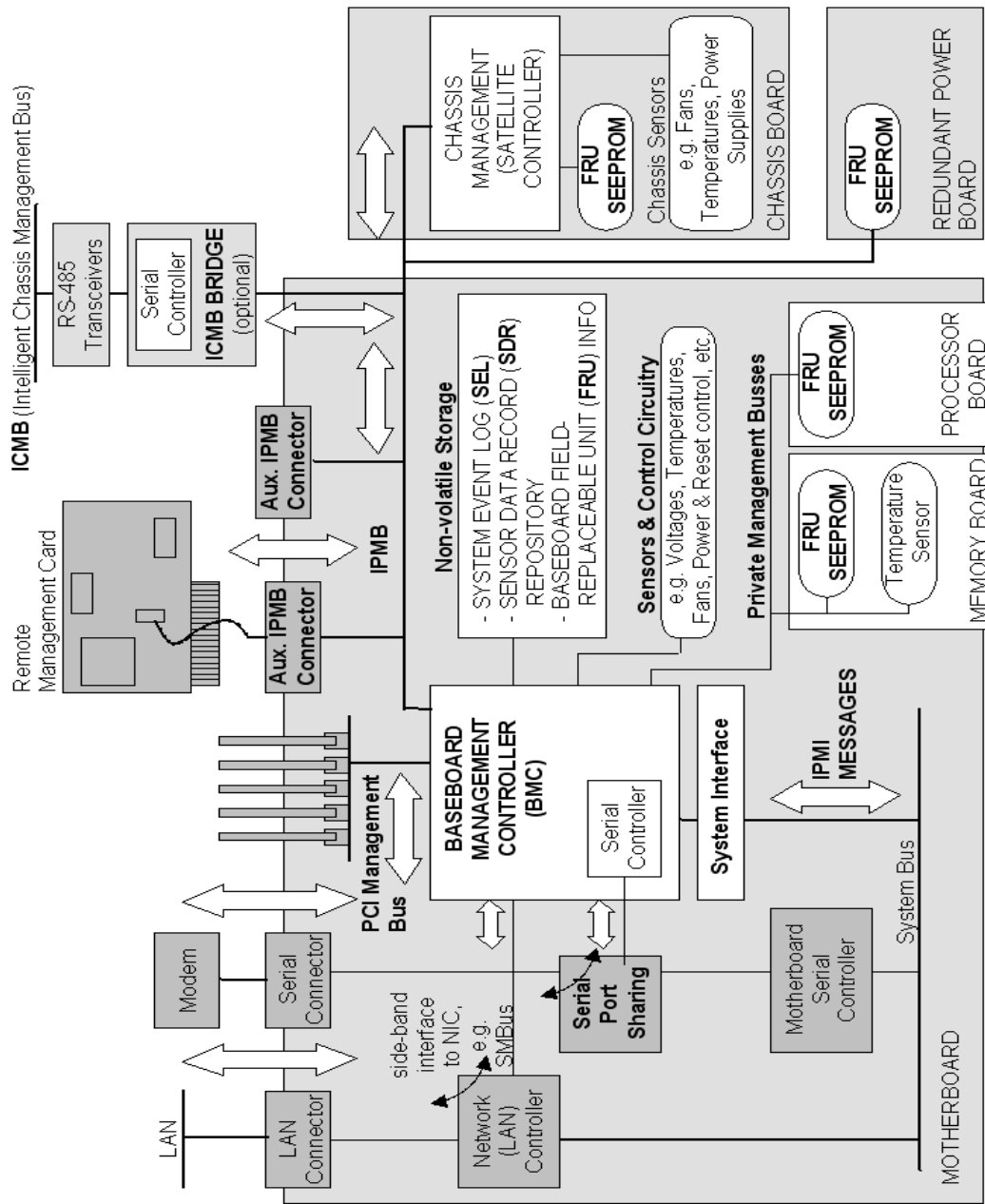


Figure 2: IPMI block diagram (source: IPMI specification)

### **IPMI and “in-band” and “out-of-band” management**

In the field of system management, a distinction is made between “in-band” and “out-of-band” management:

- The term “in-band” management is used when the operating system is running on the managed server.
- The term “out-of-band” management is used when the operating system is not running on the managed server, for instance if the hardware is faulty.

As different interfaces are available in an environment with IPMI compatible systems, you can manage IPMI compatible systems either “in-band” or “out-of-band”.

### **Channel concept under IPMI**

‘Channels’ provide the mechanisms with which IPMI messages are routed to the BMC via various connection carriers. Up to nine channels can be supported. The system interface and the primary IPMB are fixed. The other seven channels are available for the implementation.

Channels can be either ‘session based’ or ‘sessionless’. The ‘session’ concept has two meanings: It is either a concept for user authentication or a concept for routing multiple IPMI message streams via a single channel.

Examples of ‘session based’ channels are LAN channels or serial/modem channels. Examples of ‘sessionless’ channels are the system interface and the IPMB.

### **User identifications**

For ‘session based’ channels, a user login is necessary. By contrast, the ‘sessionless’ channels have no user authentication.

Under IPMI, the user configuration is channel specific. Thus, users can have different privileges depending on whether they are accessing the BMC via the LAN channel or the serial channel.

### **Fencing**

In high available clusters problems can arise if the node considered failed is not actually failed. If, for any reason, it is not really down and it is still connected to the storage solution, node still tries to communicate and write data.

With two nodes connected to the same storage running, the storage can be easily corrupted, leaving the nodes in an inconsistent state. This is where fencing comes in. With fencing, even when the cluster doesn’t know what is happening on some node, you can make sure that node doesn’t run any or certain important resources.

So fencing refers to the idea of isolating nodes considered down and preventing them from writing anything to the shared storage. The fencing feature on the iRMC can request a shutdown of a remote device, if IPMI fencing is enabled. See the "C880 M5 User Interface" manual for how to enable IPMI fencing.

### **References**

Information about the IPMI standards can be found on the Internet.

## 2.2 IPMI over LAN

IPMI-over-LAN is the current name for the specification of the LAN interface in the IPMI standard. This specification stipulates how IPMI messages can be sent to or from the BMC of a managed system - encapsulated in RMCP (Remote Management Control Protocol) data packets. These RMCP data packets are transferred via an Ethernet LAN connection using the UDP (User Datagram Protocol) under IPv4 (Internet Protocol Version 4).

The RMCP protocol has been specified to support the management of system statuses in which the operating system is not running. The RMCP is a simple inquiry/response protocol.

The interface for such a connection is provided on an onboard LAN controller assigned to the BMC.

- i** The interface can only be provided by an on-board LAN controller, not by an inserted LAN card.

Of the two ports that RMCP uses under UDP, the BMC communicates with the LAN controller via port 623 (primary RMCP Port).

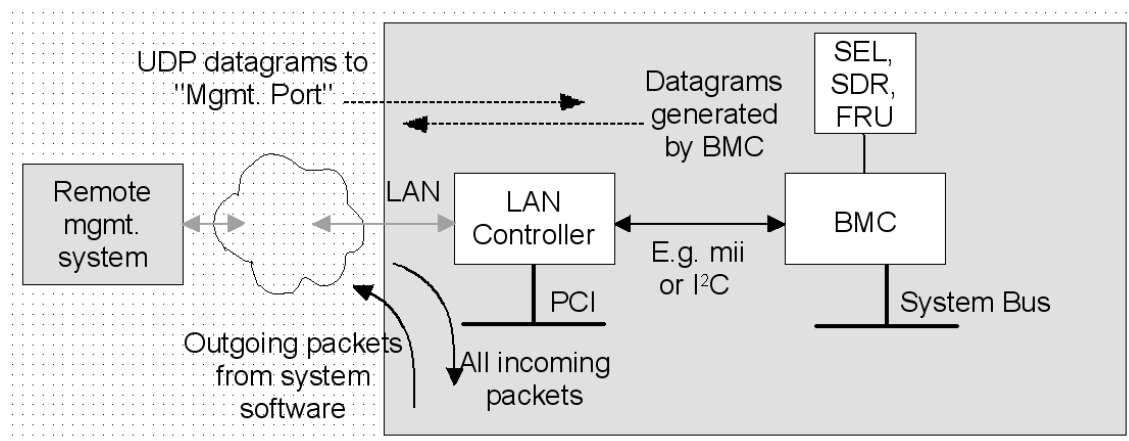


Figure 3: BMC and LAN controller

## 2.3 Supported IPMI OEM Commands

The following OEM-specific IPMI commands are supported by the iRMC S5:

### **SCCI-compliant Power On/Off commands (SCCI: Server Common Command Interface)**

0115 Get Power On Source

0116 Get Power Off Source

011C Set Power Off Inhibit

011D Get Power Off Inhibit

0120 Set Next Power On Time

### **SCCI-compliant signaling commands**

1002 Write to System Display

### **BIOS-specific command**

F109 Get BIOS POST State

F115 Get CPU Info

### **iRMC S5-specific commands**

F510 Get System Status

F512 Get EEPROM Version Info

F542 Get HDD lightpath status (Component Status Signal Read)

F543 Get SEL entry long text

F545 Get SEL entry text

F5B0 Set Identify LED

F5B1 Get Identify LED

F5B3 Get Error LED

F5E0 Set Configuration Space to Default Values

F5F8 Delete User ID

The following sections describe the individual OEM-specific IPMI commands.

### Description format

The OEM-specific IPMI commands contained in this chapter are described in the format used by the IPMI standard for describing IPMI commands.

The IPMI standard describes the IPMI commands using command tables which list the input and output parameters for each command.

You can find information on the IPMI standards on the Internet.

## 2.3.1 SCCI-compliant Power On/Off commands

### 01 15 - Get Power On Source

This command returns the reason for the most recent Power On. The possible reasons are listed below.

Request Data	-	<b>B8</b>	NetFn/LUN: OEM/Group
	-	<b>01</b>	Cmd : Command Group Communication
	1:3	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS byte first
	4	<b>15</b>	Command specifier
Response Data	-	<b>BC</b>	
	-	<b>01</b>	
	1		Completion code
	2:4	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS byte first
	3	<b>01</b>	Data Length
	4		Power on Source: Cause of last power on

Power on Source	Description
0x00	Software or command
0x01	Power switch (on the front panel or keyboard)
0x02	Automatic restart after an AC power failure
0x03	Clock or timer (hardware RTC or software timer)
0x04	Automatic restart after fan failure shutdown
0x05	Automatic restart after critical temperature shutdown

Power on Source	Description
0x08	Reboot after watchdog timeout
0x09	Remote on (modem RI line, SCSI termination power, LAN, chip card)
0x0C	Reboot after a CPU error
0x15	Reboot by hardware reset
0x16	Reboot after warm start
0x1A	Powered on by a PCI Bus Power Management Event
0x1D	Powered on by remote control via remote manager
0x1E	Reboot/reset by remote control via remote manager
0x24	Automatic Power on/off for VIOM Inventory Boot
0x25	Automatic Power on/off for VIOM Init Boot

#### 01 16 - Get Power Off Source

This command returns the reason for the most recent Power Off. The possible reasons are listed below.

Request Data	-	<b>B8</b> NetFn/LUN: OEM/Group
	-	<b>01</b> Cmd : Command Group Communication
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>16</b> Command specifier
Response Data	-	<b>BC</b>
	-	<b>01</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	3	<b>01</b> Data length
	4	Power off Source: Cause of last power off

Power off Source	Description
0x00	Software (SWOFF, SE: power off by command)
0x01	Power switch (on the front panel or keyboard)
0x02	AC power fail
0x03	Clock or timer (hardware RTC or software timer)



Power off Source	Description
0x04	Fan failure
0x05	Critical temperature
0x08	Final power-off after repeated watchdog timeouts
0x0C	Final power-off after repeated CPU errors
0x1D	Powered off by remote control via remote manager
0x24	Automatic Power on/off for VIOM Inventory Boot
0x25	Automatic Power on/off for VIOM Init Boot

### 01 1C - Set Power Off Inhibit

This command sets the Power Off Inhibit flag, which temporarily suppresses any unfounded attempt to power down the server.

If the Power Off Inhibit flag is set, the firmware saves the cause of any attempt to perform a Power Off, Power Cycle or restart of the server, but does not perform the action. The cause of the most recent attempt to perform a Power Off, Power Cycle or restart of the server is always saved at any given time. The stored action is only performed when the Power Off Inhibit flag is reset.

The Power Off Inhibit flag is automatically reset after a power failure or when the reset button is pressed.

The effect of the Power Off Inhibit flag is the same as that of the Dump flag used when creating a main memory dump. In this case, the initiator must set the flag before making the dump and reset it when the dump is complete.

Request Data	-	<b>B8</b> NetFn LUN: OEM/Group
	-	<b>01</b> Cmd : Command Group Communication
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>1C</b> Command specifier
	5	<b>00</b> Object ID
	6:7	<b>00 00</b> Value ID
	8	<b>01</b> Data length
	9	Power Off Inhibit Flag: 0 no Inhibit, 1 Inhibit
Response Data	-	<b>BC</b>
	-	<b>01</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first

**01 1D - Get Power Off Inhibit**

This command gets the value of the *Power Off Inhibit* flag.

For more information on the *Power Off Inhibit* flag, refer to the description of "01 1C - Set Power Off Inhibit".

Request Data	-	<b>B8</b>	NetFn LUN: OEM/Group
	-	<b>01</b>	Cmd : Command Group Communication
	1:3	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS Byte first
	4	<b>1D</b>	Command specifier
Response Data	-	<b>BC</b>	
	-	<b>01</b>	
	1		Completion code
	2:4	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS Byte first
	5	<b>01</b>	Response data length
	6		Power Off Inhibit Flag: 0 no Inhibit, 1 Inhibit

**01 20 - Set Next Power On Time**

This command switches on a system at the given time independent of the stored On/Off times in the configuration space.

-  The command takes effect only once.

You cancel a Power On time previously set with a 01 20 command by specifying the Power On time "0" in a subsequent 01 20 command.

Request Data	-	<b>B8</b>	NetFn LUN: OEM/Group
	-	<b>01</b>	Cmd : Command Group Communication
	1:3	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS byte first
	4	<b>20</b>	Command specifier
	5	<b>00</b>	Object ID
	6:7	<b>00 00</b>	Value ID
	8	<b>04</b>	Data length
	9:12		Time (LSB first)

Response Data	-	<b>BC</b>
	-	<b>01</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first

Time (LSB first)

Time (UNIX-specific format) when the system switches on again. Time is NOT stored in non-volatile memory. Resolution is 1 minute. After the system has switched on, Time is set to 0 internally.

If Time == 0, the system is not switched on.

## 2.3.2 SCCI-compliant signaling command

### 10 02 - Write to System Display

This command is used to write characters to the LocalView display (if connected).

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>10</b> Cmd : Command Group fan test
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>02</b> Command specifier
	5	<b>Object Index</b> : Line on display to write on.
	6:7	<b>Value ID</b> (not used)
	8	<b>Length</b> Number of characters to write, incremented by one. (The string needs not to be null-terminated; characters exceeding the length of a display line are truncated.)
	9	<b>Attribute:</b> 0 = Write string left aligned. 1 = Write string centered.
	10:10+n	<b>Characters</b> to write to the display; string needs not to be null-terminated.
Response Data	-	<b>BC</b>
	-	<b>10</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first

### 2.3.3 BIOS-specific commands

#### F1 09 - Get BIOS POST State

This command provides information whether BIOS is in POST.

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F1</b> Cmd : Command Group BIOS
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>09</b> Command Specifier
Response Data	-	<b>BC</b>
	-	<b>F1</b>
	1	Completion Code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	5	<p>[7:3] - reserved</p> <p>[0] - BIOS POST State : 0 = BIOS is not in POST 1 = BIOS is in POST</p> <p>[1] - BIOS Video State:</p> <p>0 = Video initialization is not yet done 1 = Video initialization is done</p> <p>[2] - Uncorrectable Error Indicators</p> <p>State: 0 = Indicators are not yet cleared</p>

#### F1 15 - Get CPU Info

This command returns CPU-internal information. The iRMC gets this information from the BIOS during the POST phase.

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F1</b> Cmd : Command Group BIOS
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>15</b> Command Specifier
	5	Socket Number (0-based) of the CPU


Response Data	-	<b>BC</b>
	-	<b>F1</b>
	1	<b>Completion Code:</b> 01 = Unpopulated CPU Socket
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	5:6	CPU ID, LS Byte first
	7	Platform ID
	8	Brand ID
	9:10	Maximal Core Speed of the CPU [MHz], LS Byte first
	11:12	Intel Ultra path Interconnect in Mega Transactions per second, LS Byte first
	13	T-Control Offset
	14	T-Diode Offset
	15	CPU T <sub>J</sub> max
	16:17	Record ID CPU Info SDR, LS Byte first
	18:19	Record ID Fan Control SDR, LS Byte first
	20:21	CPU ID High Word, LS Byte first (optional) (N/A = 0000h)
	22:23	Thermal Design Power TDP value with 1/8W granularity, LS Byte first (optional) (N/A = FFFFh)
	24:25	Cache Size (L1) (sum of all L1 caches of socket) Unit [KB] if MSBit = 0, [MB] if MSBit = 1, LS Byte first (N/A = FFFFh)
	26:27	Cache Size (L2) (sum of all L2 caches of socket) Unit [KB] if MSBit = 0, [MB] if MSBit = 1, LS Byte first (N/A = FFFFh)
	28:29	Cache Size (L3) (sum of all L3 caches of socket) Unit [KB] if MSBit = 0, [MB] if MSBit = 1, LS Byte first (N/A = FFFFh)
	30	Maximal Corecount (N/A = FFh)
	31	Current Corecount (N/A = FFh)
	32	Maximal Threads per Core (N/A = FFh)
	33	Current Threads per Core (N/A = FFh)
34:35	CPU Family from SMBIOS (structure 4, offset 6)	
36	CPU Manufacturer: 0 = unknown, 1 = INTEL, 2 = AMD	

### 2.3.4 iRMC S5-specific commands

#### F5 10 - Get System Status

This command returns a variety of internal information on the system such as the power state, error status, etc.

Request Data	-	<b>B8</b>	NetFnILUN: OEM/Group
	-	<b>F5</b>	Cmd : Command Group Memory
	1:3	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS byte first
	4	<b>10</b>	Command Specifier
	5:8	<b>Timestamp</b>	
Response Data	-	<b>BC</b>	
	-	<b>F5</b>	
	1		Completion Code
	2:4	<b>80 28 00</b>	IANA-Enterprise-Number FTS, LS byte first
	5		<b>System Status</b>
	6		<b>Signaling</b>
	7		<b>Notifications</b>
	8		<b>POST Code</b>

 The Timestamp is only relevant for evaluating the *Notifications* Byte.

	<b>System Status</b>	<b>Signaling</b>	<b>Notifications</b>
Bit 7 -	System ON	Localize LED	SEL Modified (new SEL entry)
Bit 6 -			SEL Modified (SEL Cleared)
Bit 5 -			SDR Modified
Bit 4 -	SEL entries available		Nonvolatile IPMI variable modified
Bit 3 -		CSS LED	ConfigSpace modified
Bit 2 -	Watchdog active	CSS LED	
Bit 1 -			
Bit 0 -	Post State	Global Error LED	New Output on LocalView display

#### F5 12 - Get EEPROM Version Info

This command returns information on the current versions (bootloader, firmware and SDR) stored in the EEPROM(s).

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group Memory
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>12</b> Command Specifier
	5	EEPROM# 00=EEPROM 1; 01=EEPROM 2
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	5	<b>Status</b> 00=Checksum error runtime FW, 01=OK
	6	<b>Major FW Revision</b> Binary coded
	7	<b>Minor FW Revision</b> BCD coded
	8:10	<b>Aux. FW Revision</b> Binary coded (major/minor/res.)
	11	<b>Major FW Revision</b> ASCII coded letter
	12	<b>Major SDRR Revision</b> BCD coded
	13	<b>Minor SDRR Revision</b> BCD coded
	14	<b>SDRR Revision Char.</b> ASCII coded letter
	15	<b>SDRR-ID</b> LSB binary coded
	16	<b>SDRR-ID</b> MSB binary coded
	17	<b>Major Booter Revision</b> Binary coded
18	<b>Major Booter Revision</b> BCD coded	
19:20	<i>Aux. Booter Revision</i> Binary coded (major/minor)	

#### **F5 42 - Get HDD lightpath status (Component Status Signal Read)**

Light path diagnostics is a system of LEDs on the control panel and on the system board. When an error occurs, LEDs are lit. If the control panel indicates an error, use the descriptions of the LEDs to diagnose the problem and take corrective action.

This command returns information on the state of a hard disk drive (HDD) slot.

Request Data	-	<b>B8</b> NetFn LUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group iRMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>42</b> Command specifier
	5	<b>Entity ID</b> ( <i>Table 37-12</i> of IPMI 1.5 Spec.) of Component whose status signal is to be read.
	6	<b>Entity Instance</b> (0-based) of Component whose status signal is to be read.
	7	<b>Sensor Type</b> ( <i>Table 36-3</i> of IPMI Spec.) of the sensor which reports the status of the component to which the status signal is associated.
	(8)	Option (optional) Bit 7:2 - Reserved Bit 1 : Completion code 0x02 suppressed Bit 0 - 1 : Return ID string of component status Sensor
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	<b>Completion Code:</b> 01 = Status signal not available 02 = Component not present
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	5	<b>Signal Status:</b> 00 = ok 01 = Identify 02 = Prefailure warning 03 = Failure
	6	<b>CSS and Physical LED available:</b> Bit 6:0 - 0= No physical LED available Bit 6:0 > 00 = Physical LED available, single or multiple color, code Bit 7 = 0: No CSS component Bit 7 = 1: CSS component
	(7)	Length of ID string of component status sensor (only present if Bit 0 in requested Byte 8 is set)
	(8:m)	Length of ID string of component status sensor in ASCII characters (only present if Bit 0 in requested Byte 8 is set)



**F5 43 - Get SEL entry long text**

This command translates a given SEL entry into long text.

Request Data	-	<b>B8</b> NetFn LUN: OEM/Group
	-	<b>F5</b> Cmd : Command group iRMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>43</b> Command specifier
	5:6	<b>Record ID</b> of SEL record, LS Byte first 0x0000: get first record 0xFFFF: get last record
	7	<b>Offset</b> in response SEL text
	8	<b>MaxResponseDataSize</b> size of converted SEL data (16:n) in response (on Pilot-1 designs the max data size is 56 bytes)
	Response Data	-
-		<b>F5</b>
1		Completion Code:
2:4		<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
5:6		Next Record ID
7:8		<b>Actual Record ID</b>
9		<b>Record type</b>
10:13		<b>Timestamp</b>
14		<b>Severity:</b> Bit 7: 0 = No CSS component 1 = CSS component Bit 6-4: 000 = INFORMATIONAL 001 = MINOR 010 = MAJOR 011 = CRITICAL 1xx = Unknown' Bit 3-0: reserved, read as 0000
15		<b>Data length</b> of the whole text
16:n		<b>Converted SEL data</b> requested part (n = 16 + MaxResponseDataSize - 1)
n + 1		<b>String Terminator</b> trailing '\0' character

**F5 45 - Get SEL Entry Short Text**

This command translates a given System Event Log SEL entry into ASCII text.

Request Data	-	<b>B8</b> NetFnLUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group iRMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	4	<b>45</b> Command specifier
	5:6	<b>Record ID</b> of SDR, LS Byte first
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS Byte first
	5:6	<b>Next Record ID</b>
	7:8	<b>Actual Record ID</b>
	9	<b>Record type</b>
	10:13	<b>Timestamp</b>
	14	<b>Severity:</b> Bit 7: 0 = No CSS component 1 = CSS component Bit 6-4: 000 = INFORMATIONAL 001 = MINOR 010 = MAJOR 011 = CRITICAL 1xx = Unknown' Bit 3-0: reserved, read as 0000
	15	<b>Data length without trailing null terminator</b>
	16:35	<b>Converted SEL data as ASCII</b> , 20 bytes maximum
n+1	<b>Trailing Null</b>	

**F5 B0 - Set Identify LED**

This command allows you to switch the Identify LED (blue) of the server on and off. In addition, you can set and read the GPIOs that are directly connected to the Identify LED.

- i** You can also switch the Identify LED on and off using the Identify switch on the server.

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group BMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>B0</b> Command specifier
	5	<b>Identify LED:</b> 0: Identify LED off 1: Identify LED on 2: Identify LED blinking
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first

**F5 B1 - Get Identify LED**

This command returns information on the status of the Identify LED (blue) of the server.

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group BMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>B1</b> Command specifier
Response Data	-	<b>BC</b>
	-	F5
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	5	<b>State of Identify LED:</b> 0: Identify LED off 1: Identify LED on 2: Identify LED blinking

**F5 B3 - Get Error LED**

This command returns information on the status of the server's global error LED (red) and CSS LED (yellow). The global error LED indicates the most serious error status of the components. The CSS LED indicates, whether the customer himself can repair the fault.

Request Data	-	<b>B8</b> NetFnILUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group BMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>B3</b> Command specifier
Response Data	-	BC
	-	F5
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	5	<b>State of Error LED:</b> 0: CSS off / GEL off 1: CSS off / GEL on 2: CSS off / GEL blink 3: CSS on / GEL off 4: CSS on / GEL on 5: CSS on / GEL blink 6: CSS blink / GEL off 7: CSS blink / GEL on 8: CSS blink / GEL blink

**F5 E0 - Reset ConfigSpace variables to default**

This command forces all configuration space variables to be set to default values.

Request Data	-	<b>B8</b> NetFn/LUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group BMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>E0</b> Command Specifier
	5:7	<b>43 4C 52</b> = 'CLR' Security Code
	8	<b>0xAA</b> : Initiate set Config Space Variable to default <b>0xBB</b> : Initiate set Config Space Variable to default except network parameter as specified in note 1 <b>0x00</b> : Get reset to default status
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	Completion Code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	5	<b>0x00</b> : completed set to default <b>0x01</b> : set to default in progress

**F5 F8 - Delete User ID**

The system supports up to 16 users. This command allows individual iRMC S5 users to be deleted.

 The system can no longer be managed if all iRMC users are deleted.

Request Data	-	<b>B8</b> NetFn/LUN: OEM/Group
	-	<b>F5</b> Cmd : Command Group BMC
	1:3	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first
	4	<b>F8</b> Command specifier
	5:8	User ID (1-16)
Response Data	-	<b>BC</b>
	-	<b>F5</b>
	1	Completion code
	2:4	<b>80 28 00</b> IANA-Enterprise-Number FTS, LS byte first

## 2.4 Data Center Management Interface

The iRMC supports the DCMI (Data Center Management Interface) protocol, which is compliant with the IPMI V2.0 standard. DCMI has been designed to improve manageability and energy efficiency of server systems that are deployed in large data centers.

To meet the hardware management requirements of servers within data centers, DCMI supports, among others, the following key features:

- Inventory functions (server identification)
- Power Management and power monitoring
- Power consumption monitoring and control
- Event logging
- Temperature monitoring

Detailed information about DCMI can be found on the Internet.

## 2.5 Serial over LAN (SOL)

“Serial Over LAN” is an interface compliant with the IPMI V2.0 standard, which controls transfer of serial data over a LAN connection. In particular, SOL specifies the packet formats and protocols for transferring serial data streams over a LAN between the serial controller on the managed computer and a remote workstation. SOL is based on the IPMI- over-LAN specification.

In order to establish a SOL connection, a remote management application first initiates an IPMI-over-LAN session with the BMC. After this has been done, the SOL services can be activated from the remote workstation. The data traffic between the serial controller and the remote workstation is handled over the same IPMI session as the IPMI commands.

As soon as an SOL connection has been established, data transfer between the serial controller and the remote workstation is carried out as follows:

- Transfer from the serial controller to the remote workstation:  
The data stream issued by the serial controller is partitioned by the BMC, packaged and then sent to the remote workstation over the LAN.
- Transfer from the remote workstation to the serial controller:  
BMC unpacks the characters contained in the packages sent by the remote workstation and forwards them to the serial controller as a character stream.

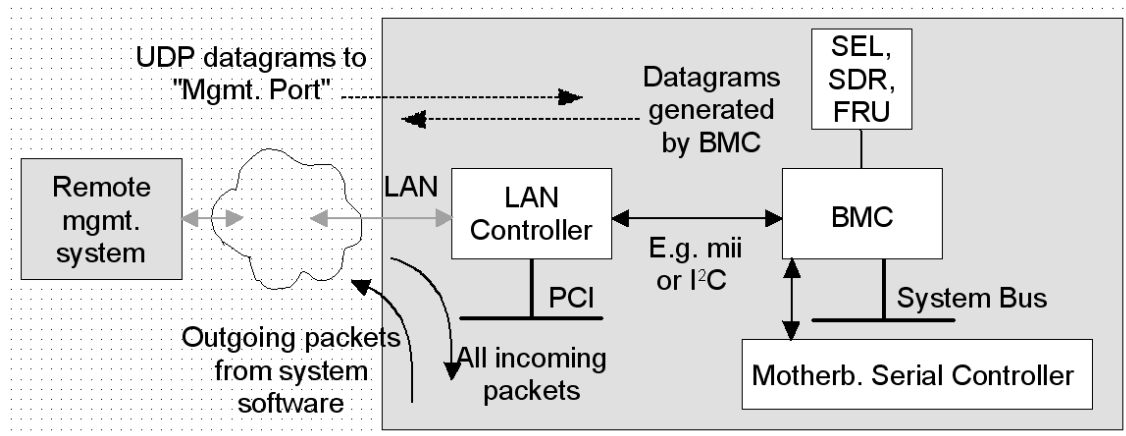


Figure 4: BMC and SOL

The SOL character data is then exchanged between the BMC of the managed system and the remote workstation as SOL messages. The SOL messages are encapsulated in RMCP+ data packets and transferred in UDP datagrams over an Ethernet LAN connection using IPv4 (Internet Protocol Version 4). The RMCP+ protocol is based on the RMCP protocol, but includes extensions for encryption, authentication, etc.

Serial over LAN permits “headless” management by console redirection by both the BIOS and the operating system of the managed server. High-cost concentrator solutions are not required.

## 3 iRMC Configuration

The following tools are available for configuring the iRMC:

- ▮ iRMC web interface ("[Configuration via the iRMC web interface](#)" on page 32)
- ▮ Remote Manager ("[Configuration via Remote Manager](#)" on page 33)
- ▮ Text console redirection ("[Configuration via text console redirection](#)" on page 34)
- ▮ Scripts ("[Configuration via scripting](#)" on page 35)
- ▮ UEFI setup utility ("[Configuration via UEFI setup utility](#)" on page 36)

### 3.1 Configuration via the iRMC web interface

You can use the iRMC web interface to configure the iRMC. The web interface is described in the "Cisco C880 M5 User Interface Guide".

#### LAN parameters

You configure the LAN parameters on the **Baseboard Management Controller** page. The following groups are provided for configuration:

- ▮ **Network Interface** group for the LAN settings
- ▮ **DNS** and **DNS Name Registration** groups to configure the DHCP and DNS settings

In the **Web Access** group of the services page you configure the ports and network services.

#### Alerting

You configure the parameters for alerting on the **Baseboard Management Controller** page of the **Settings** menu. The following groups are provided for configuration:

- ▮ **SNMP** group to configure SNMP trap forwarding.
- ▮ **Email Alerting** group to configure email alerting.

#### Text Console Redirection

You configure text console redirection in the **BIOS Console Redirection** group of the **Services** page.



### 3.2 Configuration via Remote Manager

The Remote Manager is a Telnet-based interface of the iRMC. You can call the Remote Manager over any Telnet/SSH client.

The iRMC supports secure connections over SSH. The Remote Manager interface is identical for Telnet and SSH connections.

In principle, any Telnet/SSH client that interprets VT100 sequences can be used to access the iRMC. It is nevertheless recommended that the iRMC web interface or the Remote Management Front end is used.

See "[Configuration via Telnet/SSH \(Remote Manager\)](#)" on page 42 for more details.

### 3.3 Configuration via text console redirection

Text console redirection will be available depending on the configuration of text console redirection and on the operating system of the server:

- Either for the duration of the BIOS POST phase only or
- Beyond the BIOS POST phase while the operating system is running

You configure the text console redirection via LAN with the UEFI setup utility.

#### **Linux**

The settings depend on the used Linux operating system. Refer to the documentation of your operating system.

### 3.4 Configuration via scripting

The iRMC supports remote configuration and limited scripting via the **/config** URL in the iRMC S5.

On the **Baseboard Management Controller** page of the iRMC web interface you can save (export) the current iRMC configuration data in a configuration file (.pre).

As well, you can import (restore) iRMC configuration data from an existing configuration file (.pre), i.e. load configuration data onto the iRMC.

To import an iRMC configuration, you can alternatively send the corresponding SCCI command file to the **/config** URL of the iRMC via the HTTP POST operation.

You can import configuration files via the SCCI API using commands of the following languages:

- cURL
- VB script
- Python

For more information, refer to "[Configuration via scripting](#)" on page 58.

### 3.5 Configuration via UEFI setup utility

UEFI (Unified Extensible Firmware Interface) is a standard firmware interface for PCs and Servers, which is designed to replace BIOS.

There are several ways to enter Setup Utility:

- Press the F2 function key or Delete key in POST window.
- Select “BIOS Setup” on Boot Device Selector (Settings/System/Boot Options in iRMC Web interface) before powered on the C880 M5 server. Then, powered on the server.

### 3.5.1 Configuring the LAN interface

You can configure the iRMC's LAN interface using the UEFI setup utility:

1. Call the UEFI setup utility of the managed server.
2. Open the iRMC LAN parameter configuration menu:  
Management – iRMC LAN Parameters Configuration

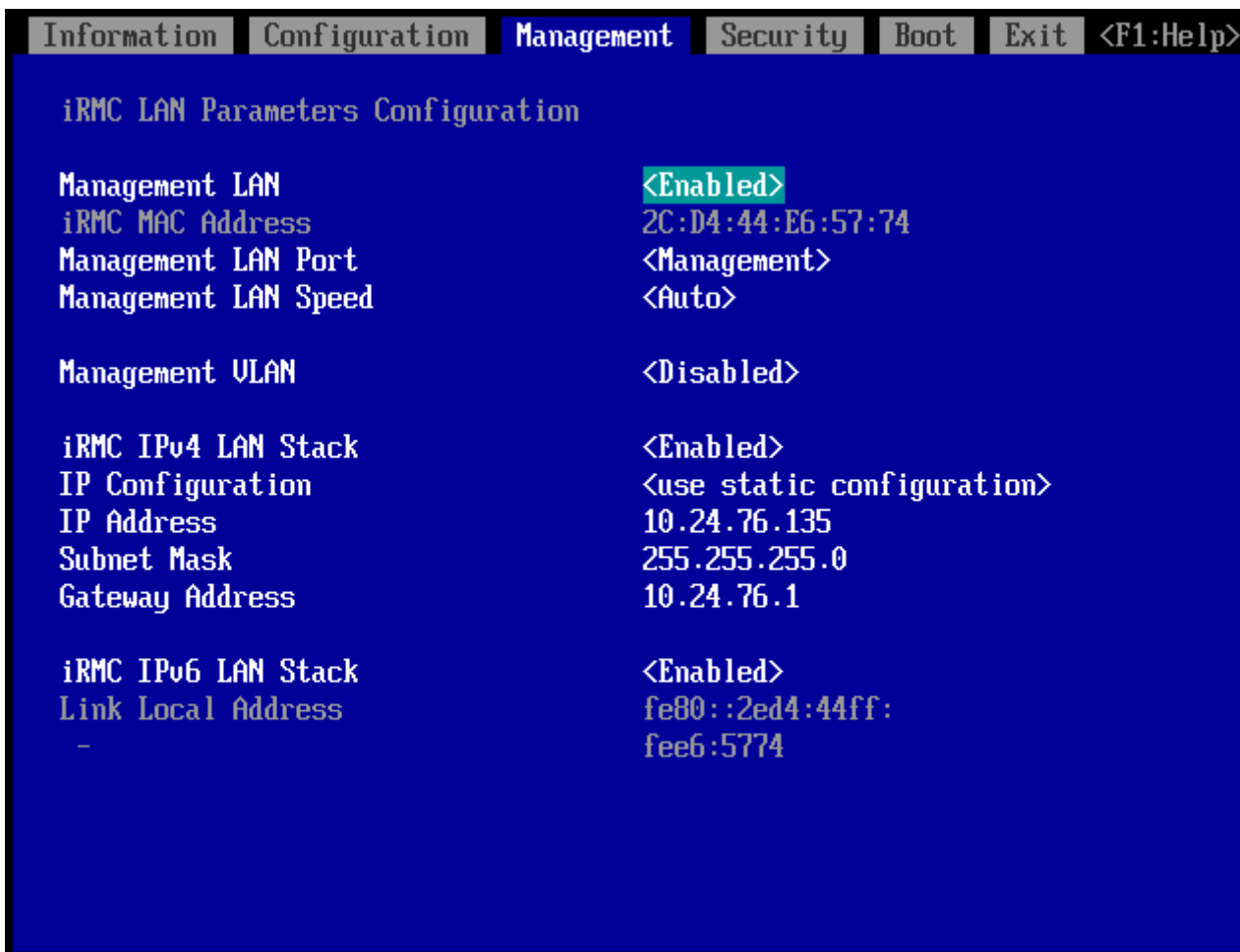


Figure 5: **Management** menu

3. In the **Management LAN** field, enter **Enabled**.
4. In the **Management LAN Port** field, enter **Management**.
  - i** For more information, refer to the "Cisco C880 M5 User Interface Guide" and/or refer to the "Cisco C880 M5 BIOS Setup Guide".
5. Save the settings.
6. If you want to use console redirection on the iRMC, continue with configuring text console redirection. For more information, refer to "[Configuration via text console redirection](#)" on page 34.

7. If you do not want to use text console redirection on the iRMC, exit the UEFI setup and continue with testing the LAN interface. (For more information, refer to the "Cisco C880 M5 Administration Guide")

### 3.5.2 Configuring text console redirection

1. Call the UEFI setup utility of the managed server.
2. Open the **Management** menu:



Figure 6: Management menu

3. In the **Serial Multiplexer** field, enter **iRMC**.

4. Open the **Console Redirection** menu:



Figure 7: **Console Redirection** menu

5. In the **Console Redirection** field, enter **Serial1**. In this case, the terminal uses the first serial interface.
6. In the **Baud Rate** field, specify the baud rate.
7. Do not change the setting in the **Protocol** field. (The setting depends on the terminal type used.)
8. The setting in the **Flow Control** field also depends on the terminal type used. The settings must be the same on both terminal and managed server.
9. Save your settings and exit the UEFI setup utility.
10. Continue with testing the LAN interface. (For more information, refer to the "Cisco C880 M5 Administration Guide")

### 3.5.3 Configuring the serial over LAN

1. Call the UEFI setup utility of the managed server.
2. Open the **Management** menu:



Figure 8: **Management** menu

3. In the **Serial Multiplexer** field, enter **iRMC**.
4. Open the **Serial Port 1 Configuration** tab with **Configuration/Super IO Configuration/Serial Port 1 Configuration** to configure the serial port.



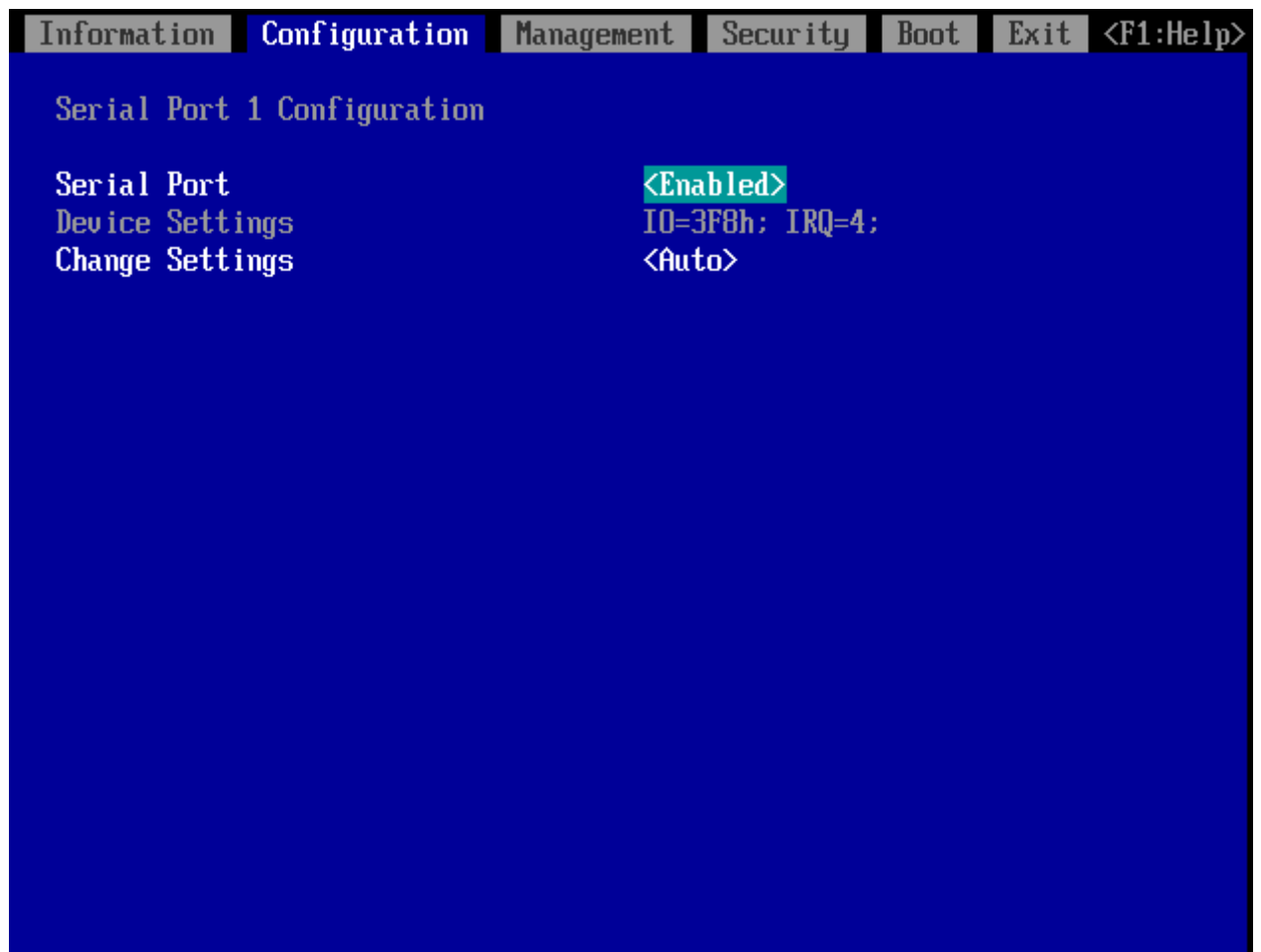


Figure 9: **Serial Port 1 Configuration** tab

5. In the **Serial Port** field, enter **Enabled**.
6. Save your settings and exit the UEFI setup utility.
7. Continue with testing the LAN interface. (For more information, refer to the "Cisco C880 M5 Administration Guide").

# 4 Configuration via Telnet/SSH (Remote Manager)


A Telnet-based interface is available for the iRMC. This is known as the Remote Manager. You can call the Remote Manager over any Telnet/SSH client.

The iRMC supports secure connections over SSH (Secure Shell). The Remote Manager interface is identical for Telnet and SSH connections. In principle, any Telnet/SSH client that interprets VT100 sequences can be used to access the iRMC. It is nevertheless recommended that the iRMC web interface or the Remote Management Front end (referred to below simply as the Remote Management Front end) be used.

This chapter describes operation of the iRMC from the Remote Manager and the various functions in detail.

## 4.1 Requirements on the managed server

Access via Telnet must be activated for the iRMC (for more information, refer to the section "Ports and Network Services - Configuring ports and network services" in the "Cisco C880 M5 User Interface Guide").

-  Access via the Telnet protocol is deactivated by default for security reasons, as passwords are transmitted in plain text.

## 4.2 Required user permissions

The following table provides an overview of the permissions required to use the individual functions available on the iRMC web interface.

Remote Manager menu items	Required IPMI privilege level				Required permission			
	OEM	Administrator	Operator	User	Configure User Accounts	Configure iRMC Settings	Video Redirection Enabled	Remote Storage Enable
<b>View System Information</b>	X	X	X	X				
View Chassis / Mainboard / OS Information.						X		
Set Asset Tag.						X		
Set System Name.						X		
Set System Operating System Information.						X		
Set System Description.						X		
Set System Location Information (SNMP).						X		
Set System Contact Information (SNMP).						X		
<b>Power Management</b>	X	X	X					
<b>View Enclosure Information</b>	X	X	X	X				

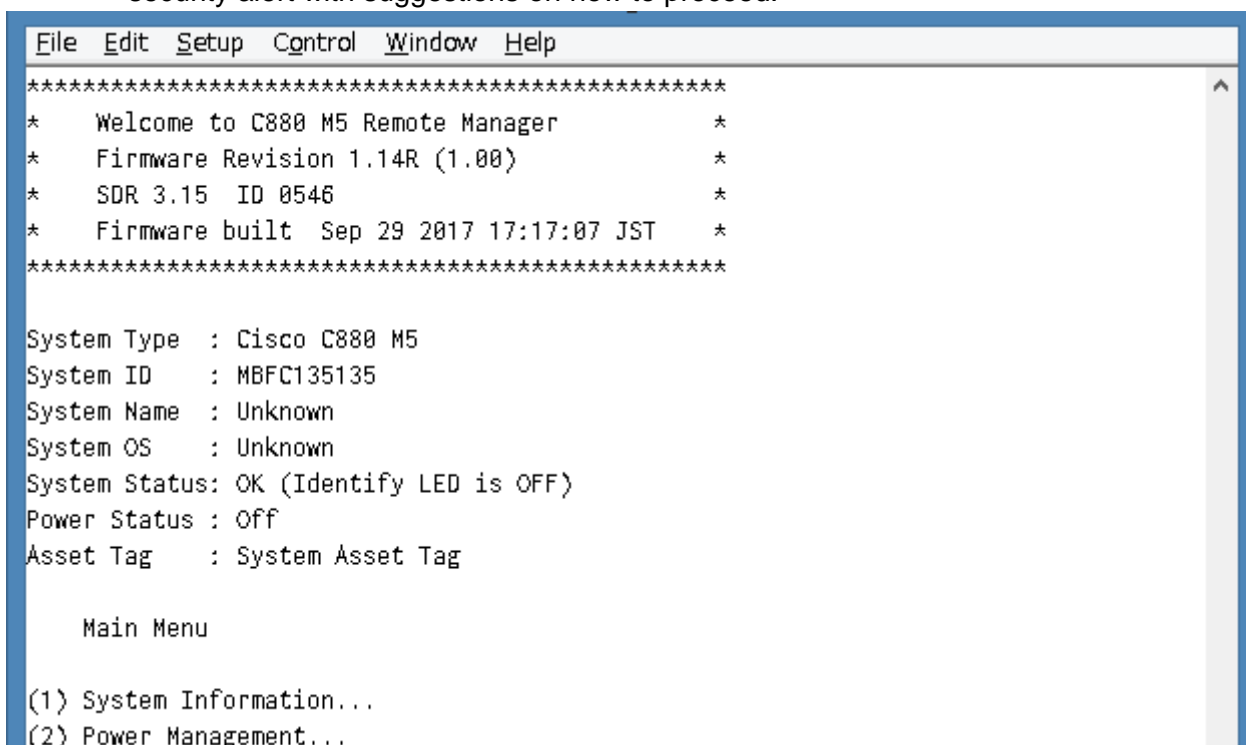
Remote Manager menu items	Required IPMI privilege level				Required permission			
	OEM	Administrator	Operator	User	Configure User Accounts	Configure iRMC Settings	Video Redirection Enabled	Remote Storage Enable
System Eventlog - View/Dump System Eventlog.	X	X	X	X				
System Eventlog - Clear System Eventlog.	X	X	X					
Internal Eventlog - View/Dump Internal Eventlog.	X	X						
Internal Eventlog - Clear Internal Eventlog.	X	X						
Sensor overviews (Temperature, Fans ...)	X	X	X	X				
<b>View Service Processor</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>				
Service Processor - List IP Parameters.	X	X	X			X		
Service Processor - Configure IP Parameters.	X	X	X			X		
Service Processor - Toggle Identify LED.	X	X	X	X				
Service Proc. - Reset iRMC S5 (warm/cold reset).	X	X						
<b>Change password</b>					<b>X</b>			
<b>Console Redirection (EMS/SAC)</b>	<b>X</b>	<b>X</b>	<b>X</b>					
<b>Start a command Line shell</b>	<b>X</b>	<b>X</b>	<b>X</b>	<b>X</b>				

Remote Manager menu items	Required IPMI privilege level				Required permission			
	OEM	Administrator	Operator	User	Configure User Accounts	Configure iRMC Settings	Video Redirection Enabled	Remote Storage Enable
Console Logging	X	X	X					

## 4.3 Logging in

When connecting to the iRMC, you are required to enter your login credentials (user name and password). As soon as a connection to the iRMC S5 has been established, the main menu window of the Remote Manager (Telnet/SSH window) is displayed at the terminal client at the remote workstation.

- i** When logging in over an SSH connection: If the host key of the managed server is not yet registered at the remote workstation, the SSH client issues a security alert with suggestions on how to proceed.

A screenshot of a terminal window showing the main menu of the Remote Manager for a Cisco C880 M5 device. The window has a menu bar with 'File', 'Edit', 'Setup', 'Control', 'Window', and 'Help'. The main content area displays a welcome message and system information, followed by a 'Main Menu' section with two options: '(1) System Information...' and '(2) Power Management...'.

```
File Edit Setup Control Window Help
*****
* Welcome to C880 M5 Remote Manager *
* Firmware Revision 1.14R (1.00) *
* SDR 3.15 ID 0546 *
* Firmware built Sep 29 2017 17:17:07 JST *
*****

System Type : Cisco C880 M5
System ID : MBFC135135
System Name : Unknown
System OS : Unknown
System Status: OK (Identify LED is OFF)
Power Status : Off
Asset Tag : System Asset Tag

Main Menu

(1) System Information...
(2) Power Management...
```

Figure 10: Remote Manager: Main menu

The Remote Manager window contains information on the affected system. This information identifies the server and indicates its operating status (Power Status). Some details (e.g. the System Name) are only shown for servers and only if the server is configured appropriately.

In order to be able to use the Remote Manager, you must log in with a user name and a password.

Then an appropriate event will be written to the Event log and the relevant main menu of the Remote Manager displayed (for more information, refer to section "Main menu of the Remote Manager").

You can terminate the login process at any time using [Ctrl] + [D].

## 4.4 Main menu

The main menu of the Remote Manager provides the following functions:

```
Main Menu

(1) System Information...
(2) Power Management...
(3) Enclosure Information...
(4) Service Processor...

(c) Change password
(r) Console Redirection (EMS/SAC)
(s) Start a Command Line shell...
(l) Console Logging

Enter selection or (0) to quit: █
```

Figure 11: Remote Manager: **Main Menu**

### (1) System Information

View information on the managed server and set the Asset Tag, for more information, refer to ["System Information menu" on page 49](#).

### (2) Power Management

Power the server up or down, for more information, refer to ["Power Management menu" on page 50](#).

### (3) Enclosure information

Request information on the current system status, e.g. check error and event messages from the error log and event log (temperature, fan, etc.), for more information, refer to ["Enclosure Information menu" on page 51](#).

### (4) Service Processor

Configure the iRMC (e.g. update firmware or change IP address), for more information, refer to ["Service processor menu" on page 54](#).

### (c) Change password

Change the password.

### (r) Console Redirection (EMS/SAC)

Text console redirection, for more information, refer to ["Console Redirection \(EMS/SAC\)" on page 55](#).

### (s) Start a Command Line shell

Start a Command Line shell, for more information, refer to ["Shell menu" on page 55](#).

### (l) Console logging

Redirect output of messages to the text console, for more information, refer to ["Console Logging" on page 56](#).



## 4.4.1 System Information menu

The **System Information** menu opens if you choose **(1) System Information** from the main menu.

```
System Information Menu

(1) View Chassis Information
(2) View Mainboard Information
(3) View OS and SNMP Information

(4) Set ASSET Tag
(*) Set System Name
(*) Set System Operating System Information
(*) Set System Description
(*) Set System Location Information (SNMP)
(*) Set System Contact Information (SNMP)

Enter selection or (0) to quit: █
```

Figure 12: Remote Manager: **System Information** menu

### **(1) View Chassis Information**

Information on the chassis of the managed server and its product data.

### **(2) View Mainboard Information**

Information on the main board of the managed server and its product data.

### **(3) View OS and SNMP Information**

Information on the operating system and the version of the managed server and on the SNMP settings.

### **(4) Set ASSET Tag**

Sets a customer-specific asset tag for the managed server.

## 4.4.2 Power Management menu

The **Power Management** menu opens if you choose **(2) Power Management** from the main menu.

```
Power Management Menu

(1) Immediate Power Off
(2) Immediate Reset
(3) Power Cycle
(*) Power On

(5) Graceful Power Off (Shutdown)
(6) Graceful Reset (Reboot)

Enter selection or (0) to quit: █
```

Figure 13: Remote Manager: **Power Management** menu

### **(1) Immediate Power Off**

Powers the server down, regardless of the status of the operating system.

### **(2) Immediate Reset**

Completely restarts the server (cold start), regardless of the status of the operating system.

### **(3) Power Cycle**

Powers the server down completely and then powers it up again after a configured period.

### **(\*) Power On**

Switches the server on.

### **(5) Graceful Power Off (Shutdown)**

Graceful shutdown and power off.

This menu item is not available on C880 M5 server.

### **(6) Graceful Reset (Reboot)**

Graceful shutdown and reboot.

This menu item is not available on C880 M5 server.

### 4.4.3 Enclosure Information menu

The **Enclosure Information** menu opens if you choose **(3) Enclosure Information** from the main menu.

```

Enclosure Information Menu

(e) System Eventlog
(i) Internal Eventlog
(t) Temperature
(v) Voltages/Current
(f) Fans
(p) Power Supplies
(d) Door Lock
(m) Memory Sensors
(c) CPU Sensors
(s) Component Status
(l) List All Sensors

Enter selection or (0) to quit: █

```

Figure 14: Remote Manager: **Enclosure Information** menu

#### **(e) System Eventlog**

Call the System Eventlog menu (for more information, refer to the section "[System Eventlog](#)" on page 52).

#### **(i) Internal Eventlog**

Call the internal Eventlog menu (for more information, refer to the section "[Internal Eventlog](#)" on page 53).

#### **(t) Temperature**

Display information on the temperature sensors and their status.

#### **(v) Voltages/Current**

Display information on the voltage and current sensors and their status.

#### **(f) Fans**

Display information on the fans and their status.

#### **(p) Power Supplies**

Display information on the power supplies and their redundancy status.

#### **(d) Door Lock**

Display information on whether the front panel or housing are open.

#### **(m) Memory Sensors**

Display information on the memory statuses.

#### **(c) CPU Sensors**

Localize the processors of the server.

#### **(s) Component Status**

Display detailed information on all sensors that have a C880 M5 diagnostic LED.

#### **(l) List All Sensors**

Display detailed information on all sensors.

### 4.4.3.1 System Eventlog

The **System Eventlog** menu opens if you select **(e) System Eventlog** from the **Enclosure Information** sub menu.

```
System Eventlog Menu

(1) View System Eventlog (text, newest first)
(2) View System Eventlog (text, oldest first)
(3) Dump System Eventlog (raw, newest first)
(4) Dump System Eventlog (raw, oldest first)

(5) View System Eventlog Information
(6) Clear System Eventlog

Enter selection or (0) to quit: █
```

Figure 15: Remote Manager: **System Eventlog** menu

**(1) View System Eventlog (text, newest first)**

The contents of the System Event log are output to screen in a readable form and in chronological order (the most recent entry first).

**(2) View System Eventlog (text, oldest first)**

The contents of the System Event log are output to screen in a readable form and in reverse chronological order (the oldest entry first).

**(3) Dump System Eventlog (raw, newest first)**

The contents of the System Event log are dumped in chronological order (the most recent entry first).

**(4) Dump System Eventlog (raw, oldest first)**

The contents of the System Event log are dumped in reverse chronological order (the oldest entry first).

**(5) View System Eventlog Information**

Display information on the System Event log.

**(6) Clear System Eventlog**

Clear the contents of the System Event log.

### 4.4.3.2 Internal Eventlog

The **Internal Eventlog** menu opens if you select **(i) Internal Eventlog** from the **Enclosure Information** sub menu.

```
Internal Eventlog Menu

(1) View Internal Eventlog (text, newest last)
(2) Dump Internal Eventlog (raw, newest last)
(3) View Internal Eventlog Information
(4) Clear Internal Eventlog
(5) Change Internal Eventlog mode

Enter selection or (0) to quit: █
```

Figure 16: Remote Manager: **Internal Eventlog** menu

**(1) View Internal Eventlog (text, newest last)**

The contents of the internal event log are output to screen in a readable form and in reverse chronological order (the most recent entry last).

**(2) Dump Internal Eventlog (raw, newest last )**

The contents of the internal event log are dumped in reverse chronological order (the most recent entry last).

**(3) View Internal Eventlog Information**

Display information on the internal event log.

**(4) Clear Internal Eventlog**

Clear the contents of the internal event log.

**(5) Change Internal Eventlog mode**

Changes the buffer mode of the internal event log from Ring Buffer mode to Linear Buffer mode and vice versa.

#### 4.4.4 Service processor menu

The **Service processor** menu opens if you choose **(4) Service Processor** from the main menu.

```
Service Processor Menu

(1) Configure IP Parameters
(2) List IP Parameters

(3) Toggle Identify LED

(4) Reset iRMC S5 (Warm reset)
(5) Reset iRMC S5 (Cold reset)

Enter selection or (0) to quit: █
```

Figure 17: Remote Manager: **Service Processor** menu

##### (1) Configure IP Parameters

Configure the IPv4/IPv6 address settings of the iRMC in a guided dialog. Refer to section "Network Interface Setting" in the "Cisco C880 M5 User Interface Guide" for details on the individual settings.

##### (2) List IP Parameters

Display the IP settings.

##### (3) Toggle Identify LED

Switch the C880 M5 identification LED on/off.

##### (4) Reset iRMC S5 (Warm reset)

Reset the iRMC. The connection is closed. Only the interfaces are re-initialized.

##### (5) Reset iRMC S5 (Cold reset)

Reset the iRMC. The connection is closed. The entire iRMC is re-initialized.

- i** It is recommended to reboot the server after a **Reset iRMC S5 (Cold Reset)** or **Reset iRMC S5 (Warm Reset)**.

### 4.4.5 Console Redirection (EMS/SAC)

You can start console redirection with the **(r) Console Redirection (EMS/SAC)** command of the main menu.

**i** Text-based console redirection only works over the LAN with Serial 1.

If serial over LAN is to be used while the operating system is running, the **Serial 1 Multiplexer** must be set to **iRMC**.

Use the keyboard shortcut [ESC] + [(] or [~] + [.] (tilde dot) to exit the text console.

### 4.4.6 Shell menu

**(s) Start a Command Line shell** in the main menu opens the **Shell Menu** in which you can start a SMASH CLP shell.

This menu item is not available on C880 M5 server.

```
Shell Menu

(1) Start SMASH CLP shell...

Enter selection or (0) to quit: █
```

Figure 18: Remote Manager: **Shell Menu**

## 4.4.7 Console Logging

The **(I) Console Logging** command in the main menu allows you to redirect message output (logging) to the text console (serial interface).

When you select **(1) Console Logging** from the main menu, the **Console Logging** menu opens.

```
Console Logging Menu

(1) Change Logging Run state
(2) Clear Console Logging buffer
(3) Replay Console (Fast mode)
(4) Replay Console (Continuous mode)

Enter selection or (0) to quit: █
```

Figure 19: Remote Manager: **Console Logging** menu

### **(1) Change Logging Run state**

Show and change the logging run state. For more information, refer to ["Console Logging Run State Menu" on page 57](#).

### **(2) Clear Console Logging buffer**

Clear the console logging buffer.

### **(3) Replay Console (Fast mode)**

Show the console log (in fast mode).

### **(4) Replay Console (Continuous mode)**

Show the console log (in continuous mode).



## 4.4.8 Console Logging Run State Menu

The **Console Run State** menu opens when you select **(1) Change Logging Run State** in the **Console Logging** menu.

```

Console Logging Run State Menu
State: STOPPED (Normal Mode)

(r) Start Console Logging
(*) Stop Console Logging

(t) Toggle to Text Mode
(*) Toggle to Normal Mode

Enter selection or (0) to quit: █

```

Figure 20: Remote Manager: **Console Logging Run State** menu

### **(r) Start Console Logging**

Start output of messages to the text console.

### **(\*) Stop Console Logging**

Stop output of messages to the text console.

### **(t) Toggle to Text Mode**

Switch to text mode.

All escape sequences are filtered out before messages are output to the console.

### **(\*) Toggle to Normal Mode**

Switch to normal mode.

In normal mode, only the following escape sequences are filtered out before messages are output to the console:

```

<ESC>(
<ESC>stop
<ESC>Q
<ESC>R<ESC>r<ESC>R
<ESC>^

```

This means that color, pseudo-graphics, etc. can also be represented to a limited extent.

## 5 Configuration via scripting

The iRMC S5 supports APIs (Application Programming Interface) for scripted configuration. With scripting only one iRMC has to be configured according to the requirements of the environment. This configuration is then uploaded to all other C880 M5 servers without the need to access them all one by one.

- **RESTful**

Representational state transfer is a way to provide interoperability between computer systems on the internet. REST-compliant web services allow requesting systems to access and manipulate textual representations of web resources using a uniform and predefined set of stateless operations.

- **Redfish**

Redfish is a DMTF standard specification and schema that specifies a RESTful interface. It utilizes a range of IT technologies that have been selected because of their widespread use. These technologies create a new foundation from which servers can be managed using common programming and scripting languages, such as Python, Java and C.

- **SCCI**

The Server Control Command Interface is a generic API defined by Cisco for different server management controller hardware as well as software. It can be easily extended to new commands or to new configuration items.

The system report of the iRMC provides information about the current settings and hardware fittings.

### 5.1 REST

REST (Representational State Transfer) is a resource-oriented architecture (ROA) style that describes how web standards can be used in a web-compatible manner, thus increasing the reusability, extensibility, clarity, and simple integration of external systems into an existing network structure of web services. In order to use the web browser to navigate from one resource to another, REST requires simple, clear, uniform interfaces. It is therefore necessary to comply with certain specifications and be constrained by a particular set of access methods (e.g. HTTP methods).

REST represents an alternative to web services that are based on a service-oriented architecture (SOA). If web services use the REST architecture, they are known as RESTful APIs (Application Programming Interfaces) or REST APIs. Unlike SOAP-based web services, there are no "official" standards for RESTful APIs. The reason for this is that REST is an

architectural style, while SOAP is a protocol. Even if REST is not a standard in itself, RESTful implementations use standards such as HTTP, URI, JSON, and XML.

Each REST resource has a generic interface via the HTTP methods GET, POST, PUT, and DELETE. Log conventions are not required in order to enable a client and server to communicate with one another. Most application scenarios can be covered by these access methods.

In the case of resource-oriented web services (which also includes RESTful web services), messages are not sent to a fixed service endpoint. Instead, the recommended access methods (e.g. GET, PUT, POST, and DELETE) operate them directly on addressable resources.

The client's state changes each time information or resources are requested and modified. The term **Representational State Transfer** (REST) serves to visualize the transfer from an application's current state to its next state. State transfers occur by transferring data that will represent the application's next state.

Resources are the basic components of a RESTful API. A resource must be addressable and have at least one display form (representation). The request's header contains information about the type of client used and the representation required by this client. Each request sent to the server must contain all information needed to interpret the request. The URI (uniform resource identifier/name and address of a resource) identifies the resource, while the HTTP header contains information such as the access type (e.g. GET), the return format or authentication. REST requires that the URI represents exactly one page of content and that a REST server delivers the same web page content when it responds to multiple requests with the same URI.

Links or URIs provide the client with a series of state transfers. The client is therefore managed intuitively by the application. In this context, it is important that each operation delivers a status code and additional links. The provision of additional links in the respective representations enables other services to be integrated.

Depending on the application's request, various representations of a resource can be delivered, for example, in different languages or formats (HTML, JSON or XML). HTTP status codes embed the access control at interface level. The user data does not have to be taken into consideration – this supports service performance and complies with data protection.

The representation of a resource can also be chosen directly. The **Content-Type** attribute must be explicitly set in the HTTP header for this purpose. In the representation, it is important to specify which operations are permitted on a resource, and which are not permitted.

### Main features

The main features of the REST architectural style are:

- Uniquely identifiable resources
- URIs or links for unique addressability

- Different representations of the resources, for example, in different languages or formats (HTML, JSON or XML)

**i** The **representation** of a resource can refer to other resources. If a client follows a link in a representation, it changes from one state to another.

- Addressability
- Statelessness

A REST message contains all of the information required by the server or client in order to understand the message. Neither the server nor the application saves state information between two messages. This is known as a stateless protocol – each operation stands alone and is fully isolated from earlier operations.

- Uniform interfaces via standard HTML functions (e.g. the HTTP methods GET, PUT, POST, and DELETE).
- Communication occurs on demand. The client is active and requests a representation from a passive server or modifies a resource.
- Status codes

Each time a request is sent to the server, the client must be informed about all potential application scenarios. Specific status codes are used for this purpose. If, for example, a request is confirmed with the HTML status code **201 - Created**, this means that a new resource has been created. The table shows some HTML status codes:

HTML status code	Meaning	HTML status code	Meaning
200	OK	400	Bad Request
201	Created	401	Unauthorized
202	Accepted	403	Forbidden
204	No Content	404	Not Found

### HTML methods

The following table shows the HTTP methods generally used to implement RESTful APIs. These methods can be used to request or change resources:

HTTP methods	Description
GET <sup>1)</sup>	Requests the specified resource from the server. GET does not cause any other side effects. The server's state is not changed. For this reason, GET is considered to be secure.
POST	Inserts a new (sub)resource below the specified resource. POST is not free from side effects. For example, database fields can be changed or processes can be started on the server.

HTTP methods	Description
PUT <sup>1)</sup>	The specified resource is created. If the resource exists already, it is changed.
PATCH	One part of the specified resource is changed. Side effects are permitted here.
DELETE <sup>1)</sup>	Deletes the specified resource.

<sup>1)</sup> The GET, PUT, and DELETE methods are idempotent, which means that, if the same request is sent multiple times, there are no additional side effects compared to sending the request once only.

## 5.2 Profile Management

Using the profile management you can send and retrieve a full server configuration via the RESTful API.

Profile management supports the following functions:

- Create a profile (or sub profile) inside the iRMC's profile store.
- Retrieve a profile (or sub profile) from the iRMC's profile store.
- Apply a profile (or sub profile) to the iRMC to be executed.
- Get session information that provides status and log information for creating and applying a profile.
- Control the version of a profile.

### 5.2.1 Profiles

A profile is a set of configuration parameters for a server and is available in XML or JSON format.

The server profile contains the following sections, also called sub profiles.

SystemConfig

The SystemConfig section is divided into two sub profiles:

Sub profile	Meaning
IrmcConfig	Comprises the parameters of the iRMC configuration.

Sub profile	Meaning
BiosConfig	Comprises the parameters of the BIOS configuration.

#### AdapterIrmcConfig

Comprises the parameters of the virtualization configuration (according to Virtual IO Management).

#### HWConfigurationIrmc

Comprises the parameters of the out-of-band RAID configuration.

When you use the eLCM Service Platform support the following additional sub profiles are available to allow an unattended deployment process:

#### HWConfiguration

Comprises the parameters of the RAID and LAN adapter configuration.

#### OSInstallation

Comprises the parameters of OS installation and deployment configuration.

The iRMC advertises the currently supported version of the profile to enable profile administration by the clients. Furthermore, each sub profile has its own version with major and minor information.

The iRMC accepts all profiles with the same major number and the same or lower minor number. In these cases backward compatibility is ensured.

## 5.2.2 Automatic BIOS parameter backup

To create a BiosConfig sub profile in the profile store of the iRMC, a BIOS parameter backup is necessary. When you request a BIOS parameter backup the managed node initiates a boot to read the parameters of the current BIOS configuration.

The **Automatic BiosParameter Backup** function on the **BIOS** page automatizes creating the BiosConfig sub profile.

With **Automatic BiosParameter Backup** enabled the parameters of the BIOS configuration will be sent automatically during each boot process. Thus, the function speeds up the request for **BiosConfig** sub profile creation but slows down each host system boot.

The configuration can be changed with the Config Space variable 1CC0 ConfPermanentBiosConfigStorageEnabled.

## 5.3 Redfish

Redfish is a standard for out-of-band management firstly released in 2015 by the Distributed Management Task Force (DMTF). It uses a data model representation inside a hypermedia RESTful interface. The data model is defined in terms of a standard, machine-readable schema, with the payload of the messages expressed in JSON and the protocol using the open data protocol (OData) version 4. Because it is a hypermedia API, Redfish is capable of representing a variety of implementations by using a consistent interface. It has mechanisms for discovering and managing data center resources, handling events, and managing long-lived tasks.

Redfish has been designed to support the full range of server architectures from monolithic servers to converged infrastructure and hyper-scale architecture. Redfish explicitly addresses converged infrastructure and rack-level management with a modeling that can scale for the management of multiple nodes, nested chassis, and server blades within a larger, actively managed enclosure.

The Redfish data model, which defines the structure and format of data representing server status, inventory and available operational functions, is vendor-neutral. Administrators can then create management automation scripts that can manage any Redfish compliant server thus increasing the efficient operation of a heterogeneous server farm.

### Main features

The main features of the Redfish architectural style are:

- Increased simplicity and usability
- Encrypted connections and generally heightened security
- A programmatic interface that can easily be controlled through scripts
- Based on widely-used standards for web APIs and data formats

HTTPS is a secure version of HTTP that enables secure communications by operating HTTP within a network connection encrypted by TLS or SSL. Redfish utilizes HTTPS encryption for secure and reliable communication. All Redfish network traffic, including event notifications, can be sent encrypted across the network.

The web interface employed by Redfish is supported by many programming languages, and its tree-like structure makes information easier to locate. Data returned from a Redfish query can be turned into a searchable dictionary consisting of key-value-pairs. By looking at the values in the dictionary, it is easy to locate settings and current status of a Redfish managed node. These settings can then be updated and actions issued to one or multiple nodes.

### JSON data

Redfish represents data using JSON. JSON is a lightweight data-interchange format that is easy for people to read and write and also for machines to parse. JSON is based on a subset of the JavaScript programming language, using a text format that is completely

language independent. But JSON uses conventions familiar to programmers of the C-family of languages such as C, C++, C#, Java, JavaScript, Perl, and Python.

### OData

OData is an open protocol standard for the definition and exchange of information using RESTful APIs. OData was originally created in 2007 by Microsoft and subsequently standardized by the OASIS standards body. OData provides Redfish the required framework to ensure that the data structures remain interchangeable between server vendors.

### Redfish architecture

Because the RESTful API employed by Redfish is web-based, access is provided using URIs, which can be typed into a web browser. The Redfish API uses a simple folder structure that starts with the Redfish root at /redfish/.

From the top-level root, the RESTful interface branches out to cover a number of "Collections" which each in turn include multiple sub-items, creating a tree-like structure. For more information on the RESTful API, refer to ["REST" on page 58](#).

The URI is the primary unique identifier of resources. Redfish URIs consist of three parts as described in RFC3986: Part one defines the scheme and authority of the URI, part two specifies the root service and version and part three defines a unique resource identifier.

### Redfish operations

In Redfish, HTTP methods implement the operations of a RESTful API. This allows the user to specify the type of request being made.

Redfish operation	HTTP methods	Description
Read	GET <sup>1)</sup>	Requests the specified resource from the server. GET does not cause any other side effects. The server's state is not changed. For this reason, GET is considered to be secure.
Create	POST	Inserts a new (sub)resource below the specified resource. POST is not free from side effects. For example, database fields can be changed or processes can be started on the server.
Update	PUT <sup>1)</sup>	The specified resource is created. If the resource exists already, it is changed.
	PATCH	One part of the specified resource is changed. Side effects are permitted here.
Delete	DELETE <sup>1)</sup>	Deletes the specified resource.



### Authentication

Depending upon the sensitivity of a given resource, Redfish clients will be required to authenticate their access. The required credentials and supported forms of authentication are determined by the platform being managed. In the case of iRMC, authentication is supported using local iRMC user credentials or any of the other supported authentication methods, such as LDAP and Active Directory.

Access to iRMC data is allowed by authenticated and authorized users only, except as noted below. Authentication is achieved using a subset of the common HTTP headers supported by a Redfish service – in particular, the X-Auth-Token header. More details on authentication are provided in the “Session Management” section of the Redfish specification.

The Redfish API provides access to Redfish URIs by using two methods:

- ▮ **Basic authentication:** In this method, user name and password are provided for each Redfish API request.
- ▮ **Session based authentication:** This method is used when issuing multiple Redfish operation requests.

## 5.4 SCCI

The SCCI is a generic application programming interface (API) defined by Cisco for different Server Management Controller hardware. It can be easily extended to cover new commands or new configuration items.

On the **Baseboard Management Controller** page of the iRMC web interface you can save (export) the current iRMC configuration data in a configuration file (.pre). You can also restore (import) the iRMC configuration data from an existing configuration file, i.e. load configuration data onto the iRMC.

To import iRMC configuration data, you can alternatively send the corresponding SCCI command file to the /config URI of the iRMC via the HTTP POST operation.

You can import configuration files via the SCCI API using commands of the following languages:

- cURL
- VB script
- Python

This section provides information on the following topics:

- How to use an SCCI (Server Common Command Interface) compliant interface for configuring the iRMC.
- SCCI commands supported by the iRMC.
- Scripted configuration of the iRMC using various script languages.

**i** Please note that the interface described is mainly for remote configuration and is not an SCCI implementation. It only uses the SCCI command and configuration definitions and the SCCI file format.

### 5.4.1 iRMC configuration data

The iRMC stores internal configuration data in separate sections of its NVRAM (non-volatile RAM):

- ConfigSpacedata, which is addressed by the firmware via an internal description or mapping table.
- Original, manufacturer-specific OMD NVCFG data, which is accessed by offset definitions.

Some configuration data from the original OMD NVCFG data is internally mapped by the firmware to be accessible via ConfigSpace access methods. For instance, DNS servers and DNS configuration of the iRMC can be accessed both via IPMI OEM LAN configuration parameters and via ConfigSpace. Both methods access the same low level data structures in the original NVCFG area.

Non-iRMC-specific software components in some cases also map standard IPMI related commands and configuration items, such as standard IPMI user configuration or IPv4 network configuration. This implements an abstraction level between the IPMI BMC layer and higher software levels.

### Benefits of remote iRMC configuration

Remote configuration of the iRMC via web-based access provides the following benefits:

- ▮ Uses HTTP POST (Power on self test) operation for uploading files onto the iRMC. No special tool is required. Any generic tool or scripting environments supporting authenticated HTTP POST operations can be used.
- ▮ Uses built-in authentication and authorization methods of the iRMC Web server.
- ▮ Supports HTTP 1.1 Basic and Digest authentication based on RFC 2617 with local iRMC user accounts.
- ▮ Features optional built-in strong encryption with standard HTTPS-based access.
- ▮ Can be used with global user accounts (managed by an LDAP directory service) and HTTP 1.1 Basic authentication.
  - ❗ If HTTP 1.1 Basic authentication is used, it is recommended that, for encryption and confidentiality reasons, you use the HTTPS protocol to protect the username/password combination.
- ▮ Uses a configuration file format that is based on XML. You can edit the file manually, or export it from a reference installation or from the Server Configuration Manager.
- ▮ The configuration file can be re-used with other SCCI based installation methods (e.g. Server Configuration Manager).
- ▮ Can be easily extended to new configuration items and new supported SCCI commands.

## 5.4.2 SCCI file format

- ❗ The format of the XML configuration file (.pre) with iRMC S5-specific notes is shown below.

The configuration file is based on XML syntax:

- Each configuration setting consists of a simple XML fragment starting with a <CMD> tag.
- The complete sequence of configuration settings is enclosed in a pair of tags <CMDSEQ> and </CMDSEQ>.

The following is an example of a typical command sequence comprising two configuration settings:

```
<CMDSEQ>
<CMD Context="SCCI" OC="ConfigSpace" OE="3800" OI="0" Type="SET">
<DATA Type="xsd::hexBinary" Len="1">04</DATA>
<CMD Context="SCCI" OC="ConfigSpace" OE="3801" OI="0" Type="SET">
<DATA Type="xsd::hexBinary" Len="1">00</DATA> </CMD>
</CMDSEQ>
```


The Context parameter is used internally to select the provider of the operation. Currently, SCCI is the only supported provider.

### 5.4.2.1 Parameters of SCCI provider-specific commands

The following SCCI-provider-specific commands are available:

#### Operation Code (OC)

Hex value or string specifying the command/operation code.

-  The iRMC only supports a limited set of SCCI commands. For a list of supported commands refer to the ["Supported SCCI commands" on page 71](#) table.

#### Operation Code Extension (OE)

Hex value for extended operation code. Default: OE=0


For ConfigSpace Read-/Write operations, this value defines the ConfigSpace ID.

#### Object Index (OI)

Hex value selecting an instance of an object. Default: OI=0"


#### Operation Code Type (Type)

For configuration settings, the values GET (read operation) and SET (write operation) are supported. Default: Type=GET

-  SET operations require data. For specifying the appropriate data type, use the Data (DATA) parameter described below.

#### Cabinet Identifier (CA)

Allows you to select an extension cabinet and use its cabinet ID number.

-  Do not use this parameter to request for the system cabinet!

**Data (DATA)**

If a SET parameter (write operation) is specified: Data type (Type parameter), and, in some cases, data length (LEN parameter) are required.

Currently, the following data types are supported:

- **xsd::integer**

Integer value

Example

```
<DATA Type="xsd::integer">1234</DATA>
```

- **xsd::hexBinary**

Stream of bytes. Each byte is coded in two ASCII characters. Use the Len parameter as shown in the example below to specify the length of the stream (i.e. the number of bytes).

The data type `xsd::hexBinary` can be used without any restriction. The number of bytes used is determined by the Len parameter.

Example

A stream of four bytes 0x00 0x01 0x02 0x04 will be coded as the following ASCII stream:

```
<DATA Type="xsd::hexBinary" Len="4">0001020304</DATA>
```

- **xsd::string**

Normally used for the transfer of strings. Additionally, the string type can be used for IPv4 addresses and MD5-based user passwords. In this case, the string data is internally converted to the accepted target format.

Transferring encrypted data

A proprietary data encryption is supported for some sensitive data such as user or service (LDAP/SMTP) access passwords.

Encrypted="1" must be set in the <DATA> tag to indicate that the data to be written is encrypted.

Examples

Transferring the string "Hello World":

```
<DATA Type="xsd::string">Hello World</DATA>
```

Transferring a password as clear (readable) text:

```
<DATA Type="xsd::string">My Readable Password</DATA>
```

Transferring an encrypted password:

```
<DATA Type="xsd::string"
Encrypted="1">TpVITJwCyHEIsC8tk24ci83JuR9I</DATA>
```

Transferring the IPv4 address "192.23.2.4"

```
<DATA Type="xsd::string">192.23.2.4</DATA>
```

- ❗ The `xsd::string` data type is restricted to readable strings, IP addresses and MD5-based user passwords.

For all other data, the `xsd::hexbinary` data type must be used!

- ℹ Do not directly specify language-specific characters, e.g. ä, à, å, æ, in strings unless they are actually needed by the using application!

Both SCCI and the ConfigSpace interface do not store any character encoding information. Thus, any non-US-ASCII-characters will be interpreted internally by the using application and therefore should be avoided.

If you do actually need to specify special characters, make sure that you edit and save your file in UTF-8 format including the correct BOM.

### Command Status (Status)

After the configuration settings are transferred, the Status contains the result of the operation. If the operation has completed successfully, the value 0 is returned.

- ℹ For a specification of all public configuration settings (ConfigSpace) see the `SCCI_CS.pdf` file, which is distributed with the C880 M5 Scripting Toolkit.

### 5.4.2.2 Restrictions

All commands specified in the `.pre` file are normally executed sequentially. The following are exemptions from this rule:

- ▮ To prevent broken network connectivity, commands for IPv4 and VLAN network configuration are executed at the end of a command sequence.
- ▮ Currently, IPv6 configuration is limited to the configuration of the non-volatile IPv6 configuration parameters.  
As a workaround, you can proceed as follows:
  1. Arrange your script as follows:
    - a. At the beginning of the script: Disable IPv6.
    - b. Configure IPv6 parameters.
    - c. At the end of the script: Enable IPv6
  2. Submit the script from an IPv4 address.
- ▮ The SSL certificate and the related matching private key are executed at the end of a command sequence. Both components must be present in the same `.pre` file and are checked for matching each other.
- ▮ If a power management operation for the managed server or a reboot of the iRMC is required or desired:  
It is recommended (but not required) to run these commands in separate command files. You can achieve this e.g. by splitting the configuration and power management operations into separate tasks.

- Optional time delays between the execution of consecutive commands must be implemented outside the script.

For example, you can achieve this as follows:

1. Divide the script appropriately into separate scripts.
2. Use the functional range of the client to insert time delays between sending the individual files.

### 5.4.2.3 Export / import of configuration data

The **Baseboard Management Controller** page of the iRMC web interface allows you to save (export) the current iRMC configuration data in a configuration file (.pre). As well, you can import iRMC configuration data from an existing configuration file (.pre), i.e. load configuration data onto the iRMC (for more information, refer to the "Cisco C880 M5 User Interface Guide").

To import an iRMC configuration, you can alternatively send the corresponding SCCI command file to the /config URI of the iRMC via the HTTP POST operation.

### 5.4.3 Supported SCCI commands

The iRMC S5 supports the following SCCI commands:

SCCI OpCode	SCCI Command String	Meaning
0xE002	ConfigSpace	ConfigSpace write
0x0111	PowerOnCabinet	Power on the server.
0x0112	PowerOffCabinet	Power off the server.
0x0113	PowerOffOnCabinet	Power cycle the server.
0x0204	ResetServer	Hard reset the server.
0x020C	RaiseNMI	Pulse the NMI (Non Maskable Interrupt).
0x0203	ResetFirmware	Perform a BMC reset.
0x0250	ConnectRemoteFdiImage	Connect or disconnect a floppy disk image on a remote image mount (NFS or CIFS share).


SCCI OpCode	SCCI Command String	Meaning
0x0251	ConnectRemoteCdImage	Connect or disconnect a CD/DVD .iso image on a remote image mount (NFS or CIFS)
0x0252	ConnectRemoteHdImage	Connect or disconnect a hard disk image on a remote image mount (NFS or CIFS share).

## 5.4.4 Script Examples

### 5.4.4.1 cURL examples

The open source command-line tool cURL allows you to transfer data specified with URL syntax. You can download the latest version of the source code as well as precompiled versions for different operating systems from <http://curl.haxx.se/>.

The following are some examples of how to use cURL to send a configuration file to the iRMC.

 For details on the cURL command line options please refer to the cURL documentation.

- HTTP Access with Basic Authentication (default) and the default iRMC S5 admin account:  

```
curl --basic -u admin:admin --data @Config.pre
http://<iRMC IP address>/config
```
- HTTP Access with Digest Authentication and the default iRMC admin account:  

```
curl --digest -u admin:admin --data @Config.pre
http://<iRMCIPaddress>/config
```
- HTTPS Access with no certificate check (-k) and Digest authentication and the default iRMC admin account:  

```
curl --digest -k -u admin:admin --data @Config.pre
https://<iRMC IP address>/config
```
- HTTPS Access with a LDAP user account.  
Please note that for LDAP users you have to specify Basic authentication:  

```
curl --basic -k -u LDAPuser:LDAPpassword --data @Config.pre
https://<iRMC IP address>/config
```



### 5.4.4.2 Visual Basic (VB) script

The following VB script sends a configuration file to the iRMC:

```
IP_ADDRESS = "<iRMC IP address>"
USER_NAME = "admin"
PASSWORD = "admin"
FILE_NAME = ".\\ConfigFile.pre"
Const ForReading = 1
Set objFSO = CreateObject("Scripting.FileSystemObject")
Set objFile = objFSO.OpenTextFile(FILE_NAME, ForReading)
' -----
On Error Resume Next
Set xmlHttp = CreateObject("Microsoft.XMLHTTP")
xmlHttp.Open "POST", "http://" & IP_ADDRESS & "/config", False, USER_NAME, PASSWORD
xmlHttp.setRequestHeader "Content-Type", "application/x-www-form-urlencoded"
xmlHttp.Send objFile.ReadAll
Wscript.Echo xmlHttp.responsexml.xml
```

### 5.4.4.3 Python script

```
#!/usr/bin/python3
import sys
import httplib2
from urllib.parse import urlencode
#=====
# iRMC
USER = 'admin'
PWD = 'admin'
IP_ADDR = '192.168.1.100'
#=====
h = httplib2.Http()
# Basic/Digest authentication
h.add_credentials(USER, PWD)
def doit(data,ausgabe=sys.stdout):
    try:
        resp, content = h.request("http://%s/config" % IP_ADDR, "POST", data)
        if resp['status'] == '200':
            data = content.decode('utf-8')
            print(data,file=ausgabe)
        else:
            print('STATUS:',resp['status'],file=ausgabe)
            print(str(resp),file=ausgabe)
    except Exception as err:
```

---

```
print('ERROR:',str(err),file=ausgabe)
print()

# Example 1 - send a configuration file to the iRMC
try:
    data = open('ConfigFile.pre').read()
    doit(data)
except Exception as err:
    print('ERROR:',str(err),file=ausgabe)
# Example 2 - Set Config Space Values
# 0x200 (ConfCabinetLocation) and
# 0x204 (ConfSystemContact) direct from the script
#
LocationContact = "<?xml version='1.0' encoding='UTF-8'?>
<CMDSEQ>
<!-- ConfCabinetLocation -->
<CMD Context='SCCI' OC='ConfigSpace' OE='200' OI='0' >
<DATA Type='xsd:string'>%s</DATA>
</CMD>
<!-- ConfSystemContact -->
<CMD Context='SCCI' OC='ConfigSpace' OE='204' OI='0' >
<DATA Type='xsd:string'>%s</DATA>
</CMD>
</CMDSEQ>
'"
doit(LocationContact % ("Ostsee","Kiel"))
```

## 6 Monitoring the iRMC

### 6.1 Monitoring with SNMP

SNMP (Simple Network Management Protocol) is an application-layer protocol that provides a message format for communication between SNMP managers and agents. SNMP provides a standardized framework and a common language used for the monitoring and management of devices in a network.

The SNMP framework has three parts:

- SNMP manager
- SNMP agent
- MIB

The SNMP manager is the system used to control and monitor the activities of network hosts using SNMP. The most common managing system is called a Network Management System (NMS). The term NMS can be applied to either a dedicated device used for network management, or the applications used on such a device. A variety of network management applications are available for use with SNMP. These features range from simple command-line applications to feature-rich graphical user interfaces.

The SNMP agent is the software component within the managed device that maintains the data for the device and reports these data, as needed, to managing systems. Agent and MIB reside on the routing device (router, access server, or switch). To enable the SNMP agent, you must define a relationship between the manager and the agent.

The Management Information Base (MIB) is a virtual information storage area for network management information, which consists of collections of managed objects. Within the MIB there are collections of related objects, defined in MIB modules. MIB modules are written in the SNMP MIB module language, as defined in STD 58, RFC 2578, RFC 2579, and RFC 2580.

The SNMP agent contains MIB variables whose values the SNMP manager can request or change with Get or Set operations. A manager can get a value from an agent or store a value into that agent. The agent gathers data from the MIB, the repository for information about device parameters and network data. The agent can also respond to manager requests to Get or Set data.

A key feature of SNMP is the ability to generate notifications from an SNMP agent. These notifications do not require that requests be sent from the SNMP manager. Asynchronous notifications can be generated as traps or inform requests. Traps are messages alerting the SNMP manager to a condition on the network. Inform requests (informs) are traps that

include a request for confirmation of receipt from the SNMP manager. Notifications can indicate improper user authentication, restarts, the closing of a connection, loss of connection to a neighbor router, or other significant events.

Traps are less reliable than informs because the receiver does not send any acknowledgment when it receives a trap. The sender cannot determine if the trap was received. An SNMP manager that receives an inform request acknowledges the message with an SNMP response protocol data unit (PDU). If the manager does not receive an inform request, it does not send a response. If the sender never receives a response, the inform request can be sent again. Thus, informs are more likely to reach their intended destination.

However, traps are often preferred because informs consume more resources in the router and in the network. Unlike a trap, which is discarded as soon as it is sent, an inform request must be held in memory until a response is received or the request times out. Also, traps are sent only once, while an inform may be retried several times. The retries increase traffic and contribute to a higher overhead on the network. Thus, traps and inform requests provide a trade-off between reliability and resources. If it is important that the SNMP manager receives every notification, use inform requests. However, if you are concerned about traffic on your network or memory in the router and you need not receive every notification, use traps.

The iRMC supports the following versions of SNMP:

- ▮ SNMPv1 is a full Internet standard, defined in RFC 1157. (RFC 1157 replaces the earlier versions that were published as RFC 1067 and RFC 1098.) Security is based on community strings.
- ▮ SNMPv2c is an experimental Internet protocol defined in RFC 1901, RFC 1905, and RFC 1906. SNMPv2c is an update of the protocol operations and data types of SNMPv2p (SNMPv2 Classic), and uses the community-based security model of SNMPv1.
- ▮ SNMPv3 is an interoperable standards-based protocol defined in RFCs 2273 to 2275. SNMPv3 provides secure access to devices by a combination of authenticating and encrypting packets over the network.  
The security features provided in SNMPv3 are as follows:
  - ▮ Message integrity: ensuring that a packet has not been tampered with in transit.
  - ▮ Authentication: determining that the message is from a valid source.
  - ▮ Encryption: scrambling the contents of a packet prevent it from being learned by an unauthorized source.

By default, the SNMP service on the iRMC S5 is disabled.

The SNMP service on the iRMC S5 supports GET requests on the following SNMP MIBs:

- ▮ SNMP STATUS.MIB
- ▮ SNMP OS.MIB
- ▮ SNMP SC2.MIB
- ▮ SNMP MIB-2.MIB

When the SNMP service is enabled, information provided by these MIBs can be used by any system running an SNMP manager.

## 6.2 iRMC system report

Typically, the collected information includes, among others, iRMC information (sensor, IDPROM/FRU, eventlog) etc.

A subset of this information, comprising mainly service incidents, can be made available directly out-of-band from the iRMC.

This section gives some examples on scripted download and automatic evaluation of the iRMC report and shows system report items provided by the iRMC.

### 6.2.1 cURL script for download

cURL is an open source command line tool for transferring data specified with URL syntax. The latest version of the source code as well as precompiled versions for different operating systems can be downloaded from <http://curl.haxx.se/>.

The following are some examples of how to retrieve the System Report file with cURL from the iRMC, for details of the cURL command line options please refer to the cURL documentation. As default cURL sends the retrieved data to standard output, you can redirect or pipe it into additional processing or save the retrieved data with the `-o` `<outputfilename>`.

- HTTP access with Digest authentication, the default iRMC admin account and saving (`-o`) to 'report.xml':  

```
curl --digest -o report.xml -u admin:admin  
http://192.168.1.100/report.xml
```
- HTTPS Access with no certificate check (`-k`) and Digest authentication and the default iRMC admin account:  

```
curl --digest -k -u admin:admin  
https://192.168.1.100/report.xml
```
- HTTPS Access with a LDAP user account:  
Please note that, for LDAP users, you have to specify basic authentication since the authentication parameters need to be passed to the LDAP server for verification:  

```
curl --basic -k -u LDAPuser:LDAPpassword  
https://192.168.1.100/report.xml
```

## 6.2.2 Visual Basic script

Scripting is also possible with Visual Basic. The following VB script retrieves the report.xml from the iRMC and saves it into a local file, also named report.xml:

```
IP_ADRESSE = "192.168.1.100"
USER_NAME = "admin"
PASSWORD = "admin"
FILE_NAME = ".\report.xml"
ADDONS = "/report.xml"
' -----
On Error Resume Next
Function SaveBinaryData(FileName, ByteArray)
    Const adTypeBinary = 1
    Const adSaveCreateOverWrite = 2
    Dim BinaryStream
    Set BinaryStream = CreateObject("ADODB.Stream")
    BinaryStream.Type = adTypeBinary
    BinaryStream.Open
    BinaryStream.Write ByteArray
    BinaryStream.SaveToFile FileName, adSaveCreateOverWrite
    WScript.Echo "Antwort:" & BinaryStream.Read
End Function
Set xmlHttp = CreateObject("Msxml2.XMLHTTP")
xmlHttp.Open "GET", "http://" & IP_ADRESSE & ADDONS, False, USER_NAME, PASSWORD
xmlHttp.Send
If InStr(xmlHttp.GetResponseHeader("Content-Type"), "xml") > 0 Then
    SaveBinaryData FILE_NAME,xmlHttp.ResponseBody
Else
    Wscript.Echo ADDONS & " not found on " & IP_ADRESSE
End If
```

### 6.2.3 Information sections

The following system report sections are supported:

Section	Sub section	Remarks/Limitations
System	BIOS	Only BIOS version string from ConfigSpace
	Processor	
	Memory	
	Fans	
	Temperatures	
	PowerSupplies	
	Voltages	
	IDPROMS	
	SensorDataRecords	
	PCIDevices	Only PCI Vendor and Device Id of cards in slots, no on-board device information
	SystemEventLog	
	InternalEventLog	
	BootStatus	
ManagementController	Only iRMC S5	
Network	Settings	
	Adapters	
	Interfaces	
	Ports	

### 6.2.3.1 Summary section

The generated XML contains as first section a summary section with some information about the date and time of the record creation, the current iRMC's IP addresses as well as summary about the number Critical/Major and Warning (Minor) entries in the SystemEventLog section and it has an inventory list of available sections.

Sample output:

```
<?xml version="1.0" encoding="UTF-8"?>
<Root Schema="2" Version="1.06P" OS="iRMC"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
  <Summary>
    <Created>
      <IsAdmin>true</IsAdmin>
      <Date>2017/07/03 11:27:15</Date>
      <BuildDuration>3</BuildDuration>
      <Company>CISCO</Company>
      <Computer>iRMCFDAF9F</Computer>
      <OS>iRMC S5 1.06P SDR: 3.07      ID 0546      C880M5</OS>
      <Domain></Domain>
      <HostIPv4Address>10.172.103.13</HostIPv4Address>
      <HostIPv6Address>fe80::219:99ff:fe80::219:99ff</HostIPv6Address>
    </Created>
    <Errors Count="1">
      <Eventlog>
        <Message>59 important error(s) in event log!</Message>
      </Eventlog>
    </Errors>
    <Warnings Count="1">
      <Eventlog>
        <Message>23 important warning(s) in event log!</Message>
      </Eventlog>
    </Warnings>
    <Content>
      <Item Name="System/Bios"></Item>
      <Item Name="System/Processor"></Item>
      <Item Name="System/Memory"></Item>
      <Item Name="System/Fans"></Item>
      <Item Name="System/Temperatures"></Item>
      <Item Name="System/PowerSupplies"></Item>
      <Item Name="System/Voltages"></Item>
```



```

<Item Name="System/IDPROMS"></Item>
<Item Name="System/SensorDataRecords"></Item>
<Item Name="System/PCIDevices"></Item>
<Item Name="System/SystemEventlog"></Item>
<Item Name="System/InternalEventlog"></Item>
<Item Name="System/BootStatus"></Item>
<Item Name="System/ManagementControllers"></Item>
<Item Name="Network/Settings"></Item>
  <Item Name="Network/Adapters"></Item>
  <Item Name="Network/Interfaces"></Item>
  <Item Name="Network/Ports"></Item>
</Content>
</Summary>
<System>

```

### 6.2.3.2 BIOS section

Since the iRMC has no access to the SMBIOS structures of the server, only a very limited subset of information is provided.

Example:

```

<Bios Schema="1">
<SMBIOS Version="Unknown">
<Type0 Name="BIOS Information" Type="0">
<BiosVersion>V1.0.0.0 R1.1.0 for D3858-A1x</BiosVersion>
</Type0>
</SMBIOS>
</Bios>

```

### 6.2.3.3 Processor section

The information generated is based on the F113 and F115 OEM IPMI cmd and is compliant with the CDiagReport.h.

```

<Processor Schema="1">
<CPU Boot="true">
<SocketDesignation>CPU</SocketDesignation>
<Manufacturer>Intel</Manufacturer>
<Model>
<Version>Intel(R) Xeon(R) Platinum 8156 CPU @ 3.60GHz</Version>

```

```

<BrandName>Intel(R) Xeon(R) Platinum 8156 CPU @ 3.60GHz</BrandName>
</Model>
<Speed>3600</Speed>
<Status Description="ok">1</Status>
<CoreNumber>4</CoreNumber>
<CoresEnabled>4</CoresEnabled>
<LogicalCpuNumber>8</LogicalCpuNumber>
<QPISpeed Unit="MT/s">10400</QPISpeed>
<Level1CacheSize Unit="KByte">256</Level1CacheSize>
<Level2CacheSize Unit="KByte">4096</Level2CacheSize>
<Level3CacheSize Unit="KByte">16896</Level3CacheSize>
</CPU>
</Processor>

```

#### 6.2.3.4 Memory section

The information generated is retrieved by decoding the memory SPD data as well as evaluating the memory status and configuration sensors and is compliant with the CDiagReport.himplementation.

```

<Memory Schema="2">
<Installed>3145728</Installed>
<Modules Count="96">
<Module Name="SB#0-DIMM#0A0" CSS="true">
<Status Description="ok">1</Status>
<Approved>true</Approved>
<Size>32768</Size>
<Type>DDR4</Type>
<BusFrequency Unit="MHz">2666</BusFrequency>
<SPD Size="512" Revision="1.1" Checksum="true">
<Checksum>
<Data>55437</Data>
<Calculated>55437</Calculated>
</Checksum>
<ModuleManufacturer>Micron Technology</ModuleManufacturer>
<ModuleManufacturingDate>2017,8</ModuleManufacturing Date>
<ModulePartNumber>36ASF4G72PZ-2G6D1 </ModulePartNumber>
<ModuleRevisionCode>49</ModuleRevisionCode>
<ModuleSerialNumber AsString="FD03E615">4244891157</ModuleSerialNumber>
<ModuleType>RDIMM</ModuleType>
<DeviceType>DDR4</DeviceType>
<DeviceTechnology>2Gx4/17x10x4</DeviceTechnology>

```

```

<BusFrequency Unit="MHz">2666</BusFrequency>
<VoltageInterface>1.2V</VoltageInterface>
<BurstLengths>8;(4);</BurstLengths>
<CASLatencies>10;11;12;13;14;15;16;17;18;19;20;</CASLatencies>
<DataWith>72</DataWith>
<NumberRanks>2</NumberRanks>
</SPD>
<ConfigStatus Description="Normal">0</ConfigStatus>
</Module>

```

### 6.2.3.5 Fans section

Fan data is retrieved/generated from FAN sensors and is compliant with the CDiagReport.himplementation.

```

<Fans Schema="1" Count="2">
<Fan Name="FANU#0-FAN#00" CSS="true">
<Status Description="ok">1</Status>
<CurrSpeed>8080</CurrSpeed>
<CurrMaxSpeed>8640</CurrMaxSpeed>
<NomMaxSpeed>8480</NomMaxSpeed>
</Fan>
<Fan Name="PSU#0-FAN#0" CSS="true">
<Status Description="ok">1</Status>
<CurrSpeed>7440</CurrSpeed>
</Fan>
</Fans>

```

### 6.2.3.6 Temperature section

The information generated is compliant with the CDiagReport.h implementation.

```

<Temperatures Schema="1" Count="7">
<Temperature Name="Ambient" CSS="false">
<Status Description="ok">6</Status>
<CurrValue>24</CurrValue>
<WarningThreshold>38</WarningThreshold>
</Temperature>
<Temperature Name="SB#0-CPU#0" CSS="false">
<Status Description="ok">6</Status>
<CurrValue>69</CurrValue>
<WarningThreshold>103</WarningThreshold>
</Temperature>

```

### 6.2.3.7 Power supplies section

The information generated is compliant with the CDiagReport.h implementation.

```
<PowerSupplies Schema="1" Count="4">
<PowerSupply Name="PSU#0" CSS="true">
<Status Description="ok">1</Status>
<Load>220</Load>
</PowerSupply>
</PowerSupplies>
```

### 6.2.3.8 Voltages section

The information generated is compliant with the CDiagReport.h implementation.

```
<Voltages Schema="1" Count="73">
<Voltage Name="BATT 3.0V" CSS="false">
<Status Description="ok">1</Status>
<CurrValue>3.09</CurrValue>
<NomValue>3.00</NomValue>
<Thresholds>
<MinValue>1.95</MinValue>
<MaxValue>3.62</MaxValue>
</Thresholds>
</Voltage>
```

### 6.2.3.9 IDPROMS section

The information generated is compliant with the CDiagReport.h implementation. In addition, the actual name retrieved from the FRU SDR record is provided as "Name" attribute in the instance tag.

Since an entry is quite long, for an example please check a generated file.

### 6.2.3.10 SensorDataRecords section

The information generated is compliant with the CDiagReport.h implementation.

Since an entry is quite long, for an example please check a generated file.

### 6.2.3.11 PCIDevices section

The iRMC does not have any direct access to PCI data and therefore can only report a limited subset on information. This information is based on what the server BIOS has sent with the F119 OEM IPMI cmd and which can be retrieved with the F11A OEM IPMI cmd.

```
<pcidevices schema="1">
<Device>
<ConfigSpace>
<Revision>1</Revision>
<VendorId>1000</VendorId>
<SubVendorId>1734</SubVendorId>
<DeviceId>005D</DeviceId>
<SubDeviceId>1212</SubDeviceId>
<BaseClass>Mass storage controller</BaseClass>
<SubClass>RAID controller</SubClass>
</ConfigSpace>
<Slot>17</Slot>
</Device>
</PCIDevices>
```

### 6.2.3.12 SystemEventLog section

The information generated is compliant with the CDiagReport.h implementation.

```
<SystemEventlog Schema="1">
<Entry>
<Date>2017/07/03 10:21:13</Date>
<Severity>Info</Severity>
<ErrorCode>37003A</ErrorCode>
<Message><![CDATA[System status - OS running]></Message>
<Data Size="14">
<HexDump Lines="1" BytesPerLine="14">
<Line Offset="0">
<Hex> 02 52 7F EE 59 20 00 04 F3 FE 73 25 FF 80 </Hex>
</Line>
</HexDump>
</Data>
</Entry>
```

### 6.2.3.13 InternalEventLog section

The information generated is compliant with the CDiagReport.h implementation.

```
<InternalEventlog Schema="1">
<Entry>
<Date>2017/07/03 11:07:59</Date>
<Severity>INFO</Severity>
<ErrorCode>2300B1</ErrorCode>
<Message>iRMC Browser http connection user 'admin' login from 10.172.103.28</Message>
</Entry>
```

### 6.2.3.14 BootStatus section

The information generated is compliant with the CDiagReport.h implementation.

```
<BootStatus Schema="1">
<PowerOnReason AsString="Power Switch">1</PowerOnReason>
<PowerOffReason AsString="Software">0</PowerOffReason>
<PowerFailBehavior AsString="remain off">1</PowerFailBehavior>
</BootStatus>
```

### 6.2.3.15 ManagementControllers section

Only information about the hosting iRMC is provided.

```
<ManagementControllers Schema="1">
<iRMC Name="iRMC">
<Firmware>1.06P</Firmware>
<IPAddress>10.172.103.13</IPAddress>
<IPSubnetMask>255.255.255.0</IPSubnetMask>
<IPGateway>10.172.103.1</IPGateway>
<MACAddress>00-19-99-FD-AF-9F</MACAddress>
<ManagementLANPort>0</ManagementLANPort>
<IPNominalSpeed>0</IPNominalSpeed>
</iRMC>
</ManagementControllers>
```

**6.2.3.16 Settings section**

No information in this section.

```
<Settings Schema="1"></Settings>
```

**6.2.3.17 Adapters section**

No information in this section.

```
<Adapters Schema="1">
</Adapter>
```

**6.2.3.18 Interfaces section**

No information in this section.

```
<Interfaces Schema="1">
</Interfaces>
```

**6.2.3.19 Ports section**

No information in this section.

```
<Ports Schema="1"></Ports>
```

**6.2.3.20 SNMPAgents section**

No information in this section.

```
<SNMPAgents Schema="1">
  <ProductVersion>N/A</ProductVersion>
</SNMPAgents>
```