

FreeIPMI Frequently Asked Questions

Free Intelligent Platform Management System
Version 0.8.4 updated on 6 March 2010

by Albert Chu chu11@llnl.gov

Copyright © 2003-2010 FreeIPMI Core Team

This manual is for FreeIPMI (version 0.8.4, 9 February 2010). Copyright © 2006-2010 FreeIPMI Core Team

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts.

Table of Contents

0.1	IPMI - Platform Management Standard	1
0.2	What is FreeIPMI?	1
0.3	How did FreeIPMI start?	1
0.4	What operating systems does FreeIPMI run on?	1
0.5	What is the relationship between FreeIPMI, OpenIPMI, Ipmitool, and Ipmiutil?	2
0.6	What is special about FreeIPMI compared to other open source IPMI projects?	2
0.7	SSIF Driver Configuration	4
0.8	x86-64 Compilation	4
0.9	SUNBMC Driver Compilation	5
0.10	Installing FreeIPMI on FreeBSD	5

0.1 IPMI - Platform Management Standard

The IPMI specifications define standardized, abstracted interfaces to the platform management subsystem. IPMI includes the definition of interfaces for extending platform management between the board within the main chassis and between multiple chassis.

The term platform management is used to refer to the monitoring and control functions that are built in to the platform hardware and primarily used for the purpose of monitoring the health of the system hardware. This typically includes monitoring elements such as system temperatures, voltages, fans, power supplies, bus errors, system physical security, etc. It includes automatic and manually driven recovery capabilities such as local or remote system resets and power on/off operations. It includes the logging of abnormal or out-of-range conditions for later examination and alerting where the platform issues the alert without aid of run-time software. Lastly it includes inventory information that can help identify a failed hardware unit.

0.2 What is FreeIPMI?

FreeIPMI is a collection of Intelligent Platform Management IPMI system software. It provides in-band and out-of-band software and a development library conforming to the Intelligent Platform Management Interface (IPMI v1.5 and v2.0) standards.

0.3 How did FreeIPMI start?

In October 2003, California Digital Corp. (CDC) was contracted by Lawrence Livermore National Laboratory (LLNL) for the assembly of Thunder, a 1024 node Itanium2 cluster. This led to software developers from CDC and LLNL merging the IPMI software developed by both organizations into FreeIPMI.

Anand Babu, Balamurugan and Ian Zimmerman at CDC contributed the in-band KCS driver, ipmi-sensors, bmc-config, ipmi-sel, bmc-info, and portions of libfreeipmi. Albert Chu and Jim Garlick at LLNL contributed ipmipower, bmc-watchdog, ipmiping, rmcpping, portions of libfreeipmi, and ipmi support in Powerman. In October 2004, FreeIPMI 0.1.0 was officially released.

Since the 0.1.0 release, Z Research developers have contributed ipmi-chassis, ipmi-raw, ipmi-locate, and portions of ipmi-pef-config. LLNL has contributed IPMI 2.0 support, host-trange support, ipmiconsole, ipmimonitoring, ipmidetect, ipmi-sensors-config, ipmi-chassis-config, bmc-device, ipmi-oem, ipmi-demi, and portions of ipmi-pef-config.

(Note: The original FreeIPMI developers from California Digital Corp. are now at Zresearch Inc.)

0.4 What operating systems does FreeIPMI run on?

FreeIPMI was originally developed on GNU/Linux. It has been confirmed to be built on most major GNU/Linux distributions such as Redhat, Fedora, Suse, and Debian. FreeIPMI has been ported and confirmed to work on at least FreeBSD, OpenBSD, Solaris, OpenSolaris, and Windows via Cygwin. We imagine it would build cleanly on other GNU/Linux distributions and Unix operating systems. If it doesn't, it should be easily portable to them.

0.5 What is the relationship between FreeIPMI, OpenIPMI, Ipmitool, and Ipmiutil?

There are multiple implementations, APIs, interfaces, end user requirements, etc. that one can choose when developing IPMI drivers, libraries, and tools. FreeIPMI has taken some different approaches than other open-source projects.

0.6 What is special about FreeIPMI compared to other open source IPMI projects?

In our eyes, there are several reasons why FreeIPMI is particularly special.

1) Support for HPC

A number of HPC aspects have been added to the tools and/or are supported by other HPC software due to the need for FreeIPMI to work on LLNL's GNU/Linux clusters.

Scalable parallel execution of many FreeIPMI tools (ipmi-sensors, ipmi-sel, bmc-info, ipmimonitoring, etc.) across a cluster is supported through hostranged input and output. For example:

```
# > bmc-info -h "pwopr[0-5]" -u XXX -p XXX --get-device-id -C
```

```
-----
pwopr[0-1,5]
-----
Device ID           : 34
Device Revision     : 1
Device SDRs         : unsupported
Firmware Revision   : 1.0c
Device Available    : yes (normal operation)
IPMI Version        : 2.0
Sensor Device       : supported
SDR Repository Device : supported
SEL Device          : supported
FRU Inventory Device : supported
IPMB Event Receiver  : unsupported
IPMB Event Generator : unsupported
Bridge              : unsupported
Chassis Device       : supported
Manufacturer ID     : Peppercon AG (10437)
Product ID          : 4
Auxiliary Firmware Revision Information : 38420000h
-----
pwopr[2-4]
-----
Device ID           : 34
Device Revision     : 1
Device SDRs         : unsupported
Firmware Revision   : 1.17
Device Available    : yes (normal operation)
IPMI Version        : 2.0
Sensor Device       : supported
```

```

SDR Repository Device : supported
SEL Device            : supported
FRU Inventory Device  : supported
IPMB Event Receiver   : unsupported
IPMB Event Generator  : unsupported
Bridge                : unsupported
Chassis Device        : supported
Manufacturer ID       : Peppercon AG (10437)
Product ID            : 4
Auxiliary Firmware Revision Information : 38420000h

```

In the above example, its clear to see that `pwopr[2-4]` have different firmware than `pwopr[0-1,5]`. More information about hostrange support can be found in the document `freeipmi-hostrange.txt`.

`Ipmipower` is capable of scaling to large nodes for cluster support and is supported by Powerman (<http://sourceforge.net/projects/powerman>) for scalable power management. At LLNL, in conjunction with Powerman, `ipmipower` is used for power control on clusters ranging from sizes of 4 to 1152. It is capable of power controlling all of the nodes in the largest clusters in under a second.

`Ipmiconsole` is currently supported by Conman (<http://home.gna.org/conman/>) for scalable console management.

`Ipmimonitoring` interprets sensor readings as well as just reporting them. It can be used for host monitoring IPMI on a cluster. By mapping sensor readings into NOMINAL, WARNING, or CRITICAL states, it makes monitoring sensors easier across large numbers of nodes. Skummee (<http://sourceforge.net/projects/skummee>) uses `libipmimonitoring` on LLNL clusters up to 1152 nodes in size. FreeIPMI monitoring plugins for Ganglia (<http://ganglia.info/>) and Nagios (<http://www.nagios.org/>) have also been developed and made available for download (<http://www.gnu.org/software/freeipmi/download.html>).

The `bmc-config`, `ipmi-chassis-config`, `ipmi-pef-config`, `ipmi-sensors-config`, and configuration file and command-line interface are used to easily copy the BMC configuration from one node to every other node in a cluster quickly. It has been used to modify the BMC configuration across large LLNL clusters in a few minutes. They also have the capability to verify (via the `diff` option) that the desired configuration has been properly stored to firmware.

`Ipmidetector` can be used to enhance the efficiency of the hostranged input by eliminating those nodes in the cluster that have been temporarily removed for servicing.

2) Additional OEM support

FreeIPMI contains support for a number of OEM extensions and OEM sensors and/or events. `Ipmi-oem` currently supports OEM command extensions from Dell, Fujitsu, Inventec, Sun Microsystems, and Supermicro. `Ipmi-sensors` and `Ipmi-sel` support OEM sensors and/or events from Dell, Inventec, Sun Microsystems, and Supermicro.

3) Additional flexibility and features

By implementing various IPMI sub-sections into multiple tools, each tool is capable of providing the user with more flexibility and ultimately more features in addition to those listed above. It may not be as easy (or architecturally possible) to do in a all-in-one tool.

4) Easy setup

By implementing drivers in userspace libraries, there is no need to build/setup/manage any kernel modules/drivers.

5) Portability

Likewise, by implementing everything in userspace libraries and tools, portability to multiple operating systems and architectures should be easier.

0.7 SSIF Driver Configuration

FreeIPMI's SSIF driver works on top of kernel's i2c device interface.

Under GNU/Linux load these kernel modules: i2c-dev, i2c-i801, i2c-core before using FreeIPMI.

To identify SSIF device address:

Example:

```
$> lspci (in the output look for this entry)
00:1f.3 SMBus: Intel Corp. 6300ESB SMBus Controller (rev 01)
    Subsystem: Intel Corp.: Unknown device 342f
    Flags: medium devsel, IRQ 17
    I/O ports at 0400 [size=32]
    ----

$> cat /proc/bus/i2c
i2c-0    smbus      SMBus I801 adapter at 0400          Non-I2C SMBus adapter
    ----
```

Make sure the "0400" above matches with the "0400" address under proc. Also make sure "i2c-0" is not different. If it appears different then grep for "i2c-0" in this code "ipmitool.c" and change. "i2c-X" is the label assigned to each slave device attached on the i2c bus.

BMC address Locator:

Refer to the SM BIOS IPMI Device Information Record Type 38, record 06h and 08h. Use the value of record 06h as the IPMBAddress and load the SMBus controller driver at the address value read from record 08h.

Usual values for record 06h -> 0x42

Usual values for record 08h -> 0x400

0.8 x86-64 Compilation

Under some x86-64 platforms, native 64 bit libraries reside under lib64 and 32 bit libs under lib. Autotools by default installs libfreeipmi.so under /usr/lib, instead of /usr/lib64 causing dynamic linking errors. Pass libdir appropriately to configure script to workaround this problem. (e.g. --libdir=/usr/lib64)

Example:

```
# ./configure --prefix=/usr --libdir=/usr/lib64
```

0.9 SUNBMC Driver Compilation

In order to compile the sunbmc driver, the `bmc_intf.h` header file must be in your include path. This can be added by to the include search path by adding the path to the `CPPFLAGS` environment variable. e.g.

```
in bash
    export CPPFLAGS='-I/my/bmc_intf_path/'
in csh/tcsh
    setenv CPPFLAGS '-I/my/bmc_intf_path/'
```

If the file is not on your system, the `bmc_intf.h` file may simply be downloaded from <http://opensolaris.org/> and installed into any location.

0.10 Installing FreeIPMI on FreeBSD

You can install a binary package of `freeipmi` or use the port, located in `ports/sysutils/freeipmi`, to build it from the source. See `ports(7)` and 'Packages and Ports' section (http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/ports.html) in The FreeBSD Handbook.

Please contact port maintainer (MAINTAINER line in the port's Makefile), if you have problems building from the port.