

GNU FreeIPMI Frequently Asked Questions

Free Intelligent Platform Management System
Version 0.2.0 updated on 3 April 2006

by Albert Chu chu11@l1n1.gov

Copyright © 2002-2005 FreeIPMI Core Team

Permission is granted to make and distribute verbatim copies of this manual provided the copyright notice and this permission notice are preserved on all copies.

Permission is granted to copy and distribute modified versions of this manual under the conditions for verbatim copying, provided that the entire resulting derived work is distributed under the terms of a permission notice identical to this one.

Permission is granted to copy and distribute translations of this manual into another language, under the above conditions for modified versions, except that this permission notice may be stated in a translation approved by the Foundation.

Table of Contents

0.1	IPMI - Platform Management Standard	1
0.2	What is GNU FreeIPMI?	1
0.3	How did it start?	1
0.4	Who should refer to this FAQ?	2
0.5	What is the relationship between GNU FreeIPMI, OpenIPMI, Ipmitool, and Ipmiutil?	2
0.6	What is special about GNU FreeIPMI compared to other open source IPMI projects?	2
0.7	SSIF Driver Configuration	3
0.8	x86-64 Compilation	3
0.9	libguile-srfi-srfi dynamic-link error	4

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 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 GNU FreeIPMI?

GNU FreeIPMI is a Free Intelligent Platform Management System Software. It provides “Remote-Console” (out-of-band), “System Management Software” (in-band) and a development library conforming to Intelligent Platform Management Interface (IPMI v1.5) standards.

GNU FreeIPMI User’s Guide concentrates installation, usage, troubleshooting and bug reporting. It corresponds to 0.2.0 release.

0.3 How did it start?

In 2002 Anand Babu at California Digital Corp. began working on a set of IPMI tools for the GNU system. This work formed the basis of FreeIPMI project.

In September 2003, Albert Chu at Lawrence Livermore National Laboratory (LLNL) began work on `ipmipower`, a tool for scalable remote power control. The project also included an IPMI library for packet construction. The tool was to be used for remote power control on Thunder, a 1024 node Itanium2 cluster for LLNL.

In October 2003, California Digital Corp. was contracted by LLNL for the assembly of Thunder and the development of several tools for IPMI management on Thunder. It was at this time that FreeIPMI’s major development efforts began. The Thunder project allowed developers from both California Digital and LLNL to collaborate on various aspects of FreeIPMI, including various IPMI tools and the `libfreeipmi` library.

Anand Babu, Balamuruan and Ian Zimmerman at California Digital Corp. worked on the an in-band driver, `libfreeipmi`, `fish`, `ipmi-sensors`, `bmc-config`, `ipmi-sel`, `bmc-info`. Jim Garlick and Albert Chu at LLNL worked on portions of the `libfreeipmi` library, worked to port LLNL tools to use `libfreeipmi`, and support `ipmipower` in `Powerman`. In May 2004, the LLNL developed tools `ipmipower`, `bmc-watchdog`, `ipmiping`, and `rmcpping` were officially merged into the GNU FreeIPMI project. In October 2004, FreeIPMI 0.1.0 was officially released. Around same time, FreeIPMI project was accepted by the Free Software Foundation as a GNU project.

In September 2005, Supercomputing team of California Digital moved out to form Z RESEARCH Inc. Maintenance of the GNU FreeIPMI also moved along from California Digital Corp. to Z RESEARCH Inc.

0.4 Who should refer to this FAQ?

If you want to use the *Intelligent Platform Management Interface* functionalities available on modern motherboards running GNU or any POSIX compliant operating systems, this guide is right for you.

0.5 What is the relationship between GNU 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. GNU FreeIPMI has taken one approach towards developing them while other open-source projects have taken other approaches.

Some examples of the differences in approaches:

1) In-Band Driver

GNU FreeIPMI implements their in-band IPMI driver in userspace using `iopl()` calls (and `/dev/io` interface under FreeBSD) while OpenIPMI implements a kernel module for its in-band IPMI driver.

2) Libraries

OpenIPMI (to our knowledge) concentrates on an APIs for IPMI driver. GNU FreeIPMI concentrates on an APIs for driver, IPMI commands and high level management interface.

3) Command-Line Interface

GNU FreeIPMI provides a bunch of independent tools and extensible Scheme interface for scripting.

Ipmitool offers a single tool for all IPMI features it supports.

The GNU FreeIPMI developers have contributed patches to the other projects on multiple occasions.

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

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

1) Support for HPC

Ipmpower is capable of scaling to very large numbers of nodes. At LLNL, in conjunction with Powerman, ipmpower currently power controls a 1024 node cluster in about 0.3 seconds. Without Powerman it can power control 1024 nodes in about 5 seconds.

At LLNL, `bmc-config`'s configuration file and command-line interface are used to configure BMCs of 1024 nodes in under 1 minute.

2) Easy setup

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

3) Portability

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

4) Additional features

By "splitting" various IPMI features into multiple tools, each tool is capable of providing the user with more options and features.

0.7 SSIF Driver Configuration

GNU 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 GNU 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 such as SUSE GNU/Linux, 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 error. Fix is simple, pass libdir appropriately to configure script.

Example:

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

0.9 libguile-srfi-srfi dynamic-link error

If you see error messages like these under “RedHat” x86-64 platform,

```
/usr/share/guile/1.6/srfi/srfi-13.scm:159:1: In procedure dynamic-link
in
expression (load-extension "libguile-srfi-srfi-13-14-v-1"
"scm_init_srfi_13"):
/usr/share/guile/1.6/srfi/srfi-13.scm:159:1: file:
"libguile-srfi-srfi-13-14-v-1", message:
"libguile-srfi-srfi-13-14-v-1.so:
cannot open shared object file: No such file or directory"
```

It may be because

- Duplicate Installation 1) you have installed both the 32bit (guile-1.6.4-14.i386.rpm) and 64bit (guile-1.6.4-14.x86-64.rpm) packages. To verify # rpm -qf /usr/bin/guile guile-1.6.4-14 guile-1.6.4-14
Will list guile twice. Fix is, just remove the i386 version. # rpm -e guile-1.6.4-14.i386
- LTDL_LIBRARY_PATH 2) Guile installed in a non standard path. “ltdl” guile dynamic module loader doesn’t use ld.so.conf. Instead set LTDL_LIBRARY_PATH path properly.