FreeIPMI Frequently Asked Questions

Free Intelligent Platform Management System
Version 1.0.4 updated on 22 April 2011

by Albert Chu chu11@llnl.gov

Table of Contents

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. FreeIPMI also supports IPMI-related specifications such as the Data Center Management Interface (DCMI) and Intel Node Manager.

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, hostrange support, ipmiconsole, libipmiconsole, ipmidetect, ipmi-sensors-config, ipmi-chassis-config, bmc-device, ipmi-oem, ipmi-dcmi, libipmimonitoring, 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 atleast FreeBSD, OpenBSD, Solaris, OpenSolaris,

and Windows via Cygwin. We imagine it would build cleanly on other operating systems. If it doesn't, it should be easily portable to them.

0.5 What are the differences 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.

The section below points out a number of the reasons why we feel FreeIPMI is particularly special compared to the other projects.

The Ipmiutil project has a good chart describing many of the differences between the projects: http://ipmiutil.sourceforge.net/docs/ipmisw-compare.htm.

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 and large data centers

A number of features have been added into the tools to support HPC and/or large data centers. Much of this original support was added to support the large cluster environments at Lawrence Livermore National Laboratory (LLNL).

Scalable parallel execution of many FreeIPMI tools (ipmi-sensors, ipmi-sel, bmc-info, 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 (http://www.gnu.org/software/freeipmi/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 2000. It has been used to determine power status or power control LLNL's largest clusters in under a second.

Ipmiconsole is currently supported by Conman (http://conman.googlecode.com/) for scalable console management.

Ipmi-sensors and libipmimonitoring are capable of interpreting sensor readings as well as just reporting them. It can be used for host monitoring IPMI sensor severity 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) currently uses libipmimonitoring to monitoring sensors on LLNL clusters of up to 2000 nodes in size. FreeIPMI sensor 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).

Ipmi-sel and libipmimonitoring are capable of interpreting system event log (SEL) entries as well as just reporting them. It can be used for host monitoring IPMI event severity on a cluster. By mapping events into NOMINAL, WARNING, or CRITICAL states, it makes monitoring system events easier across large numbers of nodes. Skummee (http://sourceforge.net/projects/skummee) currently uses libipmimonitoring to monitoring the SEL on LLNL clusters of up to 2000 nodes in size.

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.

Ipmidetect 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, IBM, Intel, Inventec, Quanta, Sun Microsystems, and Supermicro. Ipmi-sensors and Ipmi-sel support OEM sensors and/or events from Dell, Intel, Inventec, Quanta, 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 an all-in-one tool.

4) Extra IPMI support

In addition to the features listed above, FreeIPMI also supports the Data Center Management Interface, or DCMI, via the FreeIPMI tool ipmi-dcmi.

5) Easy setup

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

6) Portability

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

0.7 What setup is needed for Serial over LAN (SOL) or Ipmiconsole

The setup of Serial-over-LAN (SOL) and/or Ipmiconsole is highly dependent on your motherboard. However, most motherboards require the following:

1) Adjust the BIOS COM port for serial redirection over SOL instead of the normal serial port and set the appropriate baud rate. If you do not know which port is the SOL port, you may need to play around and guess. It is likely a non-default setting, since many manufacturers may still assume the default redirection is out of the normal serial port. If you do not have a serial port on your motherboard, this part can probably be skipped.

2) Configure IPMI on the motherboard to use SOL. Many motherboards may have this enabled by default, however you may wish to verify with FreeIPMI's bmc-config. More information can be found in the bmc-config.conf(5) manpage on the settings. However, the key settings are to enable SOL on the system, enable SOL for individual users, and select the appropriate baud. On many motherboards, the selected baud must match what is configured in the BIOS.

3) Adjust your operating systems serial console settings to use the appropriate COM port. For Linux, the following guide (http://www.vanemery.com/Linux/Serial/serial-console.html)

provides a pretty good overview of setting of a serial console on Linux. The only difference for setting up a serial console with Ipmiconsole or SOL, is the ttySX terminal may need to be changed.

0.8 Does my system support IPMI?

Unfortunately, there are no universally defined mechanisms for determining if a system supports IPMI. The following may provide hints.

1) FreeIPMI's ipmi-locate can be used to determine if IPMI can be found on your system. Users are cautioned though, the failure to discover IPMI via ipmi-locate is not sufficient to disprove that IPMI exists on your system. Your system may not publish such information or may expect clients to communicate at default locations.

2) dmidecode may be similarly used to probe for devices that support IPMI on your system. You may grep for IPMI or specifying the IPMI DMI type on the command line.

```
# > dmidecode --type 38
# dmidecode 2.10
SMBIOS 2.5 present.

Handle 0x0049, DMI type 38, 18 bytes
IPMI Device Information
        Interface Type: KCS (Keyboard Control Style)
        Specification Version: 2.0
        I2C Slave Address: 0x10
        NV Storage Device: Not Present
        Base Address: 0x0000000000000CA2 (I/O)
        Register Spacing: Successive Byte Boundaries
```

Again, the failure to find an IPMI supported device is not sufficient to show lack of IPMI support.

Ultimately, some amount of information from product documents or trial and error may be necessary to determine if IPMI is supported on your system.

0.9 SSIF Driver Configuration

FreeIPMI's SSIF driver works on top of kernel'2 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
```

0.10 x86-64 Compilation

By default, FreeIPMI's build autotools (e.g. configure) should detect if you are on a 64 bit system and should build against 64 bit libraries. However, some multi-architecture installs (e.g. you have 32 bit and 64 bit libraries installed) may lead to builds and installs of 32 bit instead of 64 bit. For those noticing this, pass libdir appropriately to the configure script to workaround this problem. (e.g. –libdir=/usr/lib64)

Example:

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

0.11 libgcrypt requirement

FreeIPMI requires the libgcrypt library to be installed for a variety of encryption requirements in IPMI 2.0. If you are building FreeIPMI and receive a 'libgcrypt required to build libfreeipmi' error, please install libgcrypt. For Linux users, this may require the install of the libgcrypt-devel package as well. For those who do not need IPMI 2.0 encryption, FreeIPMI may be built without it by specifying –without-encryption when executing configure.

0.12 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.