FreeIPMI Frequently Asked Questions

Free Intelligent Platform Management System
Version 0.4.3 updated on 5 September 2007

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.

0.3 How did it 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 driver, libfreeipmi, ipmi-sensors, bmc-config, ipmi-sel, and bmc-info. Albert Chu and Jim Garlick at LLNL contributed ipmipower, bmc-watchdog, ipmiping, rmcpping, 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 pef-config. LLNL has contributed IPMI 2.0 support, hostrange support, ipmiconsole, ipmimonitoring, ipmidetect, and pef-config.

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

0.4 Who maintains FreeIPMI?

Primary maintenance of FreeIPMI is currently managed by Albert Chu at Lawrence Livermore National Laboratory (LLNL). Overall maintenance of FreeIPMI has been handled by staff at CDC, Zresearch, and LLNL during the lifetime of FreeIPMI.

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 towards development than other open-source projects.

Some examples of the differences in approaches:

1) In-Band Driver

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.

(Note: As of FreeIPMI 0.3.0, the OpenIPMI linux kernel driver is also supported in FreeIPMI as an alternate in-band IPMI driver.)

2) Libraries

The OpenIPMI library concentrates on higher level APIs. The FreeIPMI library primarily concentrates on APIs for IPMI packet construction. Some higher level management APIs are provided although they are not considered the primary focus of libfreeipmi.

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 Linux clusters.

Ipmipower is capable of scaling to large nodes for cluster support. It 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. Native support of IPMI via libipmiconsole is slated for future support in Conman.

Scalable parallel execution of ipmi-sensors, ipmi-sel, ipmi-raw, ipmi-chassis, ipmi-fru, bmc-info, and ipmimonitoring across a cluster is supported through hostranged input and output.

```
# > bmc-info -h "pwopr[0-5]" -u XXX -p XXX -C
----------------
pwopr[0-1,5]
----------------
Device ID:         22
Device Revision:   1
Firmware Revision: 1.12
                   [Device Available (normal operation)]
IPMI Version:      2.0
Additional Device Support:
                   [Sensor Device]
                   [SDR Repository Device]
                   [SEL Device]
                   [FRU Inventory Device]
                   [Chassis Device]
```

```
Manufacturer ID:   28C5h
Product ID:        4h
Aux Firmware Revision Info: 38420000h
Channel Information:
       Channel No: 1
      Medium Type: 802.3 LAN
    Protocol Type: IPMB-1.0
       Channel No: 5
      Medium Type: Asynch. Serial/Modem (RS-232)
    Protocol Type: IPMB-1.0
----------------
pwopr[2-4]
----------------
Device ID:         22
Device Revision:   1
Firmware Revision: 1.23
                   [Device Available (normal operation)]
IPMI Version:      2.0
Additional Device Support:
                   [Sensor Device]
                   [SDR Repository Device]
                   [SEL Device]
                   [FRU Inventory Device]
                   [Chassis Device]
Manufacturer ID:   28C5h
Product ID:        4h
Aux Firmware Revision Info: 38420000h
Channel Information:
       Channel No: 1
      Medium Type: 802.3 LAN
    Protocol Type: IPMB-1.0
       Channel No: 5
      Medium Type: Asynch. Serial/Modem (RS-232)
    Protocol Type: IPMB-1.0
```

In the above example, its clear to see that pwopr[2-4] has different firmware than pwopr[0-1,5].

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.

Ipmimonitoring is used for host monitoring IPMI based sensors on a cluster. It is currently supported by Skummee (https://sourceforge.net/projects/skummee) on LLNL clusters up to 1152 nodes in size.

The bmc-config 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.

2) Easy setup

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

3) Portability

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

4) Additional features

By “splitting” various IPMI features into multiple tools, each tool is capable of providing the user with options and features, which may not be easy or capable in a “all in one” tool.

0.7 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
```

0.8 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 errors. Pass libdir appropriately to configure script to workaround this problem. (i.e. –libdir=/usr/lib64)

Example:

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