The Power of IPMI

Werner Fischer
Technology Specialist
Thomas-Krenn.AG

2<sup>nd</sup> Open Source Data Center Conference – OSDC 2010
Nuremberg, 23<sup>rd</sup> and 24<sup>th</sup> June 2010
1) About Thomas Krenn

2) IPMI basics

3) IPMI details

4) Example: Nagios/Icinga IPMI Plugin

5) Conclusions
1) About Thomas Krenn

- **Server systems, virtualization and accessories** "Made in Germany"
- **Unique service and support and 24h express delivery**
- **History**
  - 2002: founded by Max Wittenzellner and Thomas Krenn
  - 2005: turned into stock corporation
  - today:
    - 70 employees – 20 of them being technicians ;-)
    - over 9,000 customers
Agenda

2) IPMI basics
   - IPMI main features
   - IPMI Illustration
   - IPMI Messaging Interfaces
   - Channel Privilege Levels
   - IPMI Software overview

3) IPMI details

4) Example: Nagios/Icinga IPMI Plugin

5) Conclusions
2) IPMI basics

- **IPMI = Intelligent Platform Management Interface, developed by Intel, HP, NEC, Dell**
  - 1998: IPMI v1.0
  - 2001: IPMI v1.5
  - 2004: IPMI v2.0

- **IPMI main features:**
  - Monitoring (temperatures, fans, voltages, etc.)
  - Recovery Control (power on/off/reset a server)
  - Logging (System Event Log)
  - Inventory (FRU information)
2) IPMI basics

Baseboard Management Controller (BMC)

- Remote Mgmt. Card (KVM over IP, ...)
- Auxillary IPMB Connector
- ICMB bridge
- ICMB
- Chassis board
  - FRU
  - Temp. sensor
- Redundant Power board

LAN Connector
- LAN
- LAN interface

Network (LAN) Controller
- Serial Port Sharing
- M/B Serial Controller

Serial Connector
- Serial/Modem interface

PCI mgmt. bus
- System bus

M/B Serial Controller
- System interface
- System

Memory board
- Processor board
- FRU

FRU
- Temp. s.
- Memory board

Sensors & Controls
- Fan sensor
- Temp. sensor
- Power control
- Reset control
- 

SDR
- SEL
- FRU

NVS Storage

Remote Mgmt. Card (KVM over IP, ...)

BMC Serial Controller

Serial/Modem interface

LAN interface
2) IPMI basics

- **IPMI Messaging Interfaces (request/response protocol)**
  - **System Interfaces**
    - local access
    - requires root privileges
  - **Serial/Modem Interface**
    - access via serial interface or modem
    - requires IPMI user name/password (deactivate auth. NONE)
  - **LAN Interface**
    - access via network
    - requires IPMI user name/password (deactivate auth. NONE)
  - (ICMB and PCI Management Bus)
2) IPMI basics

- Channel Privilege Levels (for LAN/Serial access)

<table>
<thead>
<tr>
<th>Privilege Level</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Callback</td>
<td>Lowest Privilege Level. Allows only initiating a callback.</td>
</tr>
<tr>
<td>User</td>
<td>Allows only IPMI 'begin' commands (query sensors). Changing the BMC configuration, writing data to the BMC, executing power on/off or reset commands is prohibited.</td>
</tr>
<tr>
<td>Operator</td>
<td>Allows nearly all IPMI commands. Only changes of out-of-band interfaces are prohibited.</td>
</tr>
<tr>
<td>Administrator</td>
<td>Allows all IPMI commands.</td>
</tr>
</tbody>
</table>

- use privilege level 'User' for monitoring purposes
### 2) IPMI basics

#### IPMI Software overview

<table>
<thead>
<tr>
<th></th>
<th>ipmitool</th>
<th>ipmiutil</th>
<th>freeipmi</th>
<th>OpenIPMI (openipmish)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>OS Support</strong></td>
<td>Linux, BSD, Solaris, (Windows)</td>
<td>Linux, BSD, Solaris, Windows, EFI</td>
<td>Linux, BSD, Solaris, (Windows)</td>
<td>Linux</td>
</tr>
<tr>
<td><strong>Target Market</strong></td>
<td>admins, developers, oem's</td>
<td>admins, developers, oem's</td>
<td>HPC, universities, admins, ...</td>
<td>(Kernel-driver)</td>
</tr>
<tr>
<td><strong>Included in Linux Distros</strong></td>
<td>Debian, Gentoo, RedHat, SLES, Ubuntu (univ.)</td>
<td>Gentoo</td>
<td>Debian (Squeeze), Gentoo, RedHat (exp.) Ubuntu (univ.),</td>
<td>Debian, Gentoo, RedHat, SLES, Ubuntu</td>
</tr>
</tbody>
</table>
Agenda

3) IPMI details

- IPMI example configuration of a LAN interface
- Remote Control
- Sensors
  - Sensor Classes
  - Sensor Types
  - SDR (Sensor Data Record) Types
  - example query with ipmitool/freeipmi
- System Event Log (SEL)
- Platform Event Filtering (PEF)
- Serial over LAN (SOL)
- Field Replaceable Unit (FRU) data

4) Example: Nagios/Icinga IPMI Plugin

5) Conclusions
3) IPMI details

- IPMI example configuration of a LAN interface

```
[root@testserver ~]# ipmitool lan print 1
Set in Progress : Set Complete
Auth Type Support : NONE MD5 PASSWORD
Auth Type Enable : Callback :
                      : User      : MD5
                      : Operator : 
                      : Admin    : MD5
                      : OEM      :
IP Address Source : Static Address
IP Address : 192.168.1.211
Subnet Mask : 255.255.255.0
MAC Address : 00:0e:0c:ea:92:a2
[...]```
3) IPMI details

- **Remote Control**

```
[user@adminpc ~]$ ipmitool -I lan -H 192.168.1.211 -U admin -P relation power status
Chassis Power is off
[user@adminpc ~]$ 

[user@adminpc ~]$ ipmitool -I lan -H 192.168.1.211 -U admin -P relation power on
Chassis Power Control: Up/On
[user@adminpc ~]$ 

[user@adminpc ~]$ ipmitool -I lan -H 192.168.1.211 -U admin -P relation power status
Chassis Power is on
[user@adminpc ~]$ 
```
### 3) IPMI details

- **Sensors: Sensor Classes (1/2)**

<table>
<thead>
<tr>
<th>Discrete</th>
<th>Threshold</th>
</tr>
</thead>
</table>
| multiple states possible:  
  - up to 15 states  
  - each state is reflected by a bit  
  - multiple state bits can active at a time | changes event status on analog reading comparison to threshold values |
| can provide:  
  - generic states (Table 42-2) or  
  - sensor-specific states (Table 42-3) | provides:  
  - analog reading of the sensor and  
  - discr. threshold comparison status bit (generic state, Table 42-2) |
| other classes similar to discrete:  
  - Digital: term often used for discrete sensors with two possible states  
  - OEM: discrete sensor where the meaning of the states (offsets) are OEM defined |
### 3) IPMI details

**Sensors: Sensor Classes (2/2)**

<table>
<thead>
<tr>
<th>Discrete</th>
<th>Threshold</th>
</tr>
</thead>
</table>
| [root@test ~]# ipmitool sdr get "PS2 Status"
Sensor ID : PS2 Status (0x71)
Entity ID : 10.2 (Power Supply)
Sensor Type (Discrete) : Power Supply
States Asserted : Power Supply
  - [Presence detected]
  - [Power Supply AC lost]
Assertion Events : Power Supply
  - [Presence detected]
  - [Power Supply AC lost]
Assertions Enabled : Power Supply
  - [Presence detected]
  - [Failure detected]
  - [Predictive failure]
  - [Power Supply AC lost]
  - [Presence detected]
Deassertions Enabled : Power Supply
  - [Presence detected]
  - [Power Supply AC lost] |
| [root@test ~]# ipmitool sdr get "Fan 1"
Sensor ID : Fan 1 (0x50)
Entity ID : 29.1 (Fan Device)
Sensor Type (Analog) : Fan
Sensor Reading : 5719 (+/- 0) RPM
Status : ok
Nominal Reading : 6708.000
Normal Minimum : 2451.000
Normal Maximum : 10965.000
Lower critical : 1720.000
Lower non-critical : 1978.000
Positive Hysteresis : 86.000
Negative Hysteresis : 86.000
Minimum sensor range : Unspecified
Maximum sensor range : Unspecified
Event Message Control : Per-threshold
Readable Thresholds : lcr lnc
Settable Thresholds : lcr lnc
Threshold Read Mask : lcr lnc
Assertion Events : Assertions Enabled : lnc lcr
Deassertions Enabled : lnc lcr
3) IPMI details

- **Sensors: Sensor Types**

<table>
<thead>
<tr>
<th>Sensor Types:</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Temperature</td>
<td>Voltage</td>
</tr>
<tr>
<td>Current</td>
<td>Fan</td>
</tr>
<tr>
<td>Physical Security</td>
<td>Platform Security</td>
</tr>
<tr>
<td>Processor</td>
<td>Power Supply</td>
</tr>
<tr>
<td>Power Unit</td>
<td>Cooling Device</td>
</tr>
<tr>
<td>Other</td>
<td>Memory</td>
</tr>
<tr>
<td>Drive Slot / Bay</td>
<td>POST Memory Resize</td>
</tr>
<tr>
<td>System Firmwares</td>
<td>Event Logging Disabled</td>
</tr>
<tr>
<td>Watchdog</td>
<td>System Event</td>
</tr>
<tr>
<td>Critical Interrupt</td>
<td>Button</td>
</tr>
<tr>
<td>Module / Board</td>
<td>Microcontroller</td>
</tr>
<tr>
<td>Add-in Card</td>
<td>Chassis</td>
</tr>
<tr>
<td>Chip Set</td>
<td>Other FRU</td>
</tr>
<tr>
<td>Cable / Interconnect</td>
<td>Terminator</td>
</tr>
<tr>
<td>System Boot Initiated</td>
<td>Boot Error</td>
</tr>
<tr>
<td>OS Boot</td>
<td>OS Critical Stop</td>
</tr>
<tr>
<td>Slot / Connector</td>
<td>System ACPI Power State</td>
</tr>
<tr>
<td>Watchdog</td>
<td>Platform Alert</td>
</tr>
<tr>
<td>Entity Presence</td>
<td>Monitor ASIC</td>
</tr>
<tr>
<td>LAN</td>
<td>Management Subsystem Health</td>
</tr>
<tr>
<td>Battery</td>
<td>Session Audit</td>
</tr>
<tr>
<td>Version Change</td>
<td>FRU State</td>
</tr>
</tbody>
</table>
### 3) IPMI details

- **Sensors: SDR (Sensor Data Record) Types**

<table>
<thead>
<tr>
<th>Name (Chapter 43)</th>
<th>SDR Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Full Sensor Record</td>
<td>01h</td>
</tr>
<tr>
<td>Compact Sensor Record</td>
<td>02h</td>
</tr>
<tr>
<td>Event-Only Record</td>
<td>03h</td>
</tr>
<tr>
<td>Entity Association Record</td>
<td>08h</td>
</tr>
<tr>
<td>Device-relative Entity Association Record</td>
<td>09h</td>
</tr>
<tr>
<td>Reserved Records</td>
<td>0Ah:0Fh</td>
</tr>
<tr>
<td>Generic Device Locator Record</td>
<td>10h</td>
</tr>
<tr>
<td>FRU Device Locator Record</td>
<td>11h</td>
</tr>
<tr>
<td>Management Controller Device Locator Record</td>
<td>12h</td>
</tr>
<tr>
<td>Management Controller Confirmation Record</td>
<td>13h</td>
</tr>
<tr>
<td>BMC Message Channel Info Record</td>
<td>14h</td>
</tr>
<tr>
<td>OEM Record</td>
<td>C0h</td>
</tr>
</tbody>
</table>
### 3) IPMI details

- **Sensors: example query with ipmitool**
  - use 'ipmitool sdr type [...]’ for sensor type
  - use 'ipmitool sdr elist [...]’ for SDR type

```bash
[root@testserver ~]# ipmitool sdr type Other
PS1 +12V Power   | 7Ch  | ok  | 10.1 | 80 Watts
PS2 +12V Power   | 7Dh  | ok  | 10.2 | 104 Watts
[root@testserver ~]# ipmitool sdr type Other -v
Sensor ID         : PS1 +12V Power (0x7c)
Entity ID         : 10.1 (Power Supply)
Sensor Type (Analog) : Other
Sensor Reading    : 80 (+/- 6) Watts
Status            : ok
Nominal Reading   : 372.000
Normal Minimum    : 100.000
Normal Maximum    : 744.000
Upper critical    : 840.000
Upper non-critical: 792.000
[...]```
3) IPMI details

- **Sensors: example query with freeipmi**

```bash
[root@testserver ~]# ipmimonitoring
Record_ID | Sensor Name | Sensor Group     | Monitoring Status|
Sensor Units | Sensor Reading

[...]
17 | Fan 5 | Fan | Nominal | RPM | 9052.000000
18 | Fan 6 | Fan | Nominal | RPM | 8060.000000
19 | PS1 AC Current | Current | Nominal | A | 0.124000
20 | PS2 AC Current | Current | Nominal | A | 0.992000
[...]
33 | Power Redundancy | Power Unit | Critical | N/A | 'Redundancy Lost' 'Non-redundant:Sufficient Resources from Redundant'
34 | BMC Watchdog | Watchdog 2 | Nominal | N/A | 'OK'
35 | Platform Secu V | Platform Security Violation Attempt | Nominal | N/A | 'OK'
36 | Physical Scrty | Physical Security | Critical | N/A | 'General Chassis Intrusion'
```
3) IPMI details

- System Event Log (SEL)
  - stored in non-volatile storage

```
[root@testserver ~]# ipmitool sel elist
  40 | 06/21/2010 | 14:29:29 | Power Supply PS1 Status | Power Supply AC lost | Asserted
  54 | 06/21/2010 | 14:29:29 | Power Unit Power Redundancy | Fully Redundant
  68 | 06/21/2010 | 14:29:29 | Power Unit Power Redundancy | Redundancy Lost
  7c | 06/21/2010 | 14:29:29 | Power Unit Power Redundancy | Non-Redundant: Sufficient from Redundant
[...]
  2fc | 06/21/2010 | 15:20:32 | Physical Security Physical Scrty | General Chassis intrusion | Asserted
```

```
[root@testserver ~]# ipmitool sel elist
  Power Supply PS1 Status | Power Supply AC lost | Asserted
  Power Unit Power Redundancy | Fully Redundant
  Power Unit Power Redundancy | Redundancy Lost
  Power Unit Power Redundancy | Non-Redundant: Sufficient from Redundant
[...]
  Physical Security Physical Scrty | General Chassis intrusion | Asserted
```
3) IPMI details

- Platform Event Filtering (PEF)
  - BMC takes selected actions on event messages
  - actions can be:
    - system power off
    - system reset
    - generating SNMP trap
  - tools: freeipmi (pef-config), ipmiutil

[root@testserver ~]# pef-config --info
PEF version: 2.0
Alert action support: Yes
Power down action support: Yes
Power reset action support: Yes
Power cycle action support: Yes
OEM action support: Yes
Diagnostic interrupt action support: Yes
[...]
3) IPMI details

- Serial over LAN (SOL) (1/2)
  - allows text-based access to
    - BIOS
    - GRUB
    - operating systems
      - Linux serial console
      - Windows debugger
        (http://support.microsoft.com/kb/151981/)
  - SOL is implemented as a payload type in RMCP+
3) IPMI details

- Serial over LAN (SOL) (2/2)
3) IPMI details

- Field Replaceable Unit (FRU) data

```
[root@testserver ~]# ipmitool fru print
FRU Device Description : BuiltIn FRU Device (ID 0)
  Chassis Type : Rack Mount Chassis
  Chassis Serial : 9000041568
  Chassis Extra : SR2500LX
  Board Mfg Date : Sat Apr 28 11:37:00 2007
  Board Mfg : Intel
  Board Product : S5000PAL0
  Board Serial : BZAU71700054
  Board Part Number : D13607-805
  Product Manufacturer : Intel
  Product Name : S5000PAL
  Product Part Number : ...........
  Product Serial : ...........
[...]
FRU Device Description : Pwr Supply 1 FRU (ID 2)
  Product Manufacturer : DELTA
  Product Name : DPS-750EBA
  Product Part Number : D20850-006
  Product Version : 02
  Product Serial : DLD0712004978
```
Agenda

1) About Thomas Krenn
2) IPMI basics
3) IPMI details
4) Example: Nagios/Icinga IPMI Plugin
5) Conclusions
4) Example: Nagios/Icinga IPMI Plugin

- how the Nagios IPMI Sensor Monitoring Plugin works
  - it's a shell script (Bash)
  - it uses ipmitool, gawk
  - you can use the plugin with every IPMI-compatible server
  - it follows the *Nagios plug-in development guidelines*
  - clear illustration within the Nagios web interface

<table>
<thead>
<tr>
<th>Host</th>
<th>Service</th>
<th>Status</th>
<th>Last Check</th>
<th>Duration</th>
<th>Attempt</th>
<th>Status Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>examplehost</td>
<td>IPMI Fans</td>
<td>OK</td>
<td>2009-11-02 16:06:33</td>
<td>0d 0h 6m 12s</td>
<td>1/4</td>
<td>Fan Status: OK</td>
</tr>
<tr>
<td></td>
<td>IPMI Power Supply</td>
<td>OK</td>
<td>2009-11-02 16:06:24</td>
<td>0d 0h 32m 21s</td>
<td>1/4</td>
<td>Power Supply Status: OK</td>
</tr>
<tr>
<td></td>
<td>IPMI Pysical Security</td>
<td>WARNING</td>
<td>2009-11-02 16:06:14</td>
<td>0d 0h 40m 32s</td>
<td>4/4</td>
<td>Physical Security Status: Warning [Intrusion = Inc]</td>
</tr>
<tr>
<td></td>
<td>IPMI Temperature</td>
<td>OK</td>
<td>2009-11-02 16:06:02</td>
<td>0d 0h 31m 43s</td>
<td>1/4</td>
<td>Temperature Status: OK</td>
</tr>
<tr>
<td></td>
<td>IPMI Voltage</td>
<td>OK</td>
<td>2009-11-02 16:05:48</td>
<td>0d 0h 31m 57s</td>
<td>1/4</td>
<td>Voltage Status: OK</td>
</tr>
</tbody>
</table>
5) Conclusions

- IPMI has multiple very useful features
- IPMI is built into most modern Servers
- IPMI has rarely been used until now

→ so start using the Power of IPMI today ;-)!

- Further information
  - http://download.intel.com/design/servers/ipmi/IPMI2_0E4_Markup_061209.pdf
  - http://lists.thomas-krenn.com