Ipmitool

Christian Külker

2022-06-30

Contents

The command line tool `ipmitool` is used to interact with a **Board Management Controller** (`BMC`) via the **Intelligent Platform Management Interface** (`IPMI`). `IPMI` is an open standard for monitoring, logging and control of hardware that is implemented independent of the main `CPU` , `BIOS` , and `OS` via a side band access. On some platforms `IPMI` can also be used to manage hardware inventory information.

- `SDR` - Sensor Data Repository
- `SEL` - System Event Log
- `FRU` - Field Replaceable Unit

1 History

Version	Date	Notes
0.1.3	2022-06-30	Formatting, changes->history, shell->bash
0.1.2	2020-05-03	Debug `SOL`
0.1.1	2020-05-02	Set MAC address raw
0.1.0	2016-02-12	Initial release

2 Install

```
aptitude install ipmitool
```

When using, make sure some kernel modules are loaded.

```
ipmi_devintf
ipmi_msghandler
ipmi_poweroff
ipmi_si
ipmi_watchdog
```

3 Set DHCP

```
ipmitool lan set 1 ipsrc dhcp
```

4 Reset the BMC

```
ipmitool mc reset cold
```

Or

```
ipmitool mc reset warm
```

5 BMC Sensor List

If the `BMC` supports it (that depends on the `OEM`) sensors can be listed. Some sensors might be on the `BMC` SOC, but most likely this sensors are made available to the `BMC` by various methods. Usually the `OEM` defines if the value is in or out of specification. Some values are depending on the `CPU` `OEM`.

```
ipmitool -I open sdr list
ACPI State       | 0x01              | ok
LPC Reset        | 0x01              | ok
Chassis          | 0x01              | ok
BMC RST          | 0x00              | ok
CPU0PECI         | 42 degrees C      | ok
CPU1PECI         | 38 degrees C      | ok
+1.1V            | 1.11 Volts        | ok
+1.4V            | 1.41 Volts        | ok
+0.9V            | 0.91 Volts        | ok
+3.0V            | 2.98 Volts        | ok
+1.5V            | 1.51 Volts        | ok
+2.5V            | 2.42 Volts        | ok
CPU0VCCP         | 0.97 Volts        | ok
CPU1VCCP         | 0.96 Volts        | ok
DDRCPU0          | 0.75 Volts        | ok
DDRCPU1          | 0.74 Volts        | ok
+1.8VICH         | 1.79 Volts        | ok
+1.5VICH         | 1.55 Volts        | ok
```

6 Set The NAC Address

Setting the MAC address if the `BMC` can be done via the `OS`.

```
ipmitool lan set <channel> macaddr <macaddr>
```

Sometimes the MAC address can only be set by Vendor specific raw commands. Sometimes the `BMC` has 2 `LAN` interfaces. The number depends on the hardware and `BMC` firmware. The channel number needs usually to be specified.

```
ipmitool raw 0x0c 0x01 0x01 0xc2 0x00
ipmitool lan set 1 macaddr aa:bb:cc:dd:ee:ff
ipmitool lan print 1

ipmitool raw 0x0c 0x01 0x08 0xc2 0x01
ipmitool lan set 8 macaddr a0:42:3f:2c:61:a2
```

```
ipmitool lan print 8
```

7 Activate SOL

To activate the **Serial Console Over LAN** (`SOL`). If the `OEM` configured the `BMC` in a secure way, a **user** and **password** is needed. However, since this is usually not changed, it can be a security problem.

```
ipmitool -I lanplus -H $REMOTEHOST -U $USER -P $PASSWORD sol activate
ipmitool -I lanplus -H $REMOTEHOST -U $USER -P $PASSWORD -v sol info
```

8 Debug SOL

To debug the setup of **Serial Over Lan** (`SOL`) add the `-v` option to enable verbose output and in some case set the privilege level explicit with the `-L` option.

```
ipmitool -v -L USER -I lanplus -H 172.20.50.102 -U admin -P admin sol
 ↪  activate
```

If his does not help use `namp` . See ipmitool Fails with Authentication Issues

9 Raw Commands

Sometimes the `OEM` implements undocumented features into the `BMC` firmware that can be reach via the **raw** option. As this commands are not standard it depends on the `BMC` what the out come will be. This is just an example.

```
ipmitool raw 0x36 0xc
```

10 Testing

Some `OEMs` implement meaningful self tests. However dummy self tests that always passes have been seen in the wild.

```
ipmitool mc selftest
Selftest: passed
```

11 Links

- ipmitool source
- IPMI v2.0 Rev 1.1