

Linux firmware for iRMC controller on Fujitsu Primergy servers

Vladimir Shakhov

R&D Senior Development Engineer

Fujitsu Technology Solutions, R&D department



Table of content

① Fujitsu Primergy servers family

What is iRMC

Software - ServerView Suite

Open standards: IPMI protocol and others

iRMC internals

Demo: WebIF and IPMI

② Road to Linux: from iRMC S1/ThreadX to iRMC S4/Linux

Early days - ThreadX : S1 - S2/S3

Migration to Linux: S4

Demo: RemoteManager - bug-to-bug compatible

③ Linux based firmware

Components

Development environment

FOSS legal questions

Demo: inside the Linux on iRMC



Fujitsu Primergy Servers

Lineage of x86-based servers:
Blade (BX), Rack (RX), Tower (TX) and Cloud (CX).



FUJITSU

iRMC S4 in the wild



FUJITSU

iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.



iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.



Control x86 hardware

Work independent if x86 host on or off.

iRMC - integrated Remote Management Controller

ARM-based SoC

Emulex Pilot3 iBMC ASIC

Integrated BMC

Super I/O

Graphics controller

KVMS: Remote Keyboard, Video, Mouse and Storage

CPU: 32-bit 400MHz ARM9 processor with MMU.



Control x86 hardware

Work independent if x86 host on or off.

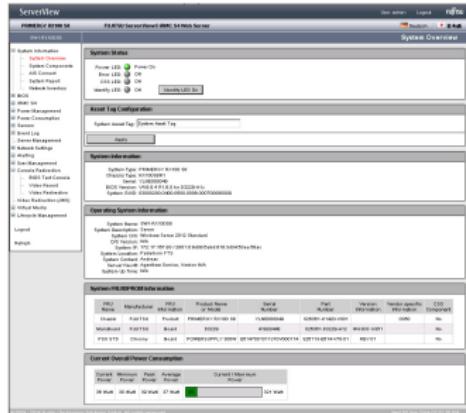
Own Operation System

Very typical Embedded Linux.



iRMC basic features

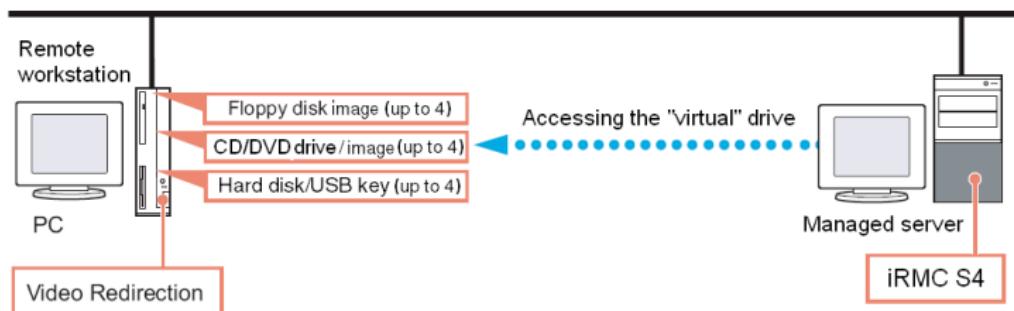
- Web access (own web-server)
- Security (SSL and SSH included)
- ServerView suite Integration
- Power management
- SNMPv1/v2c/v3 support
- Text console redirection
- “Headless” system operation
- CLP - command line interface



FUJITSU

iRMC advanced features

- Advanced Video Redirection (AVR)
- Virtual Media
- Embedded Lifecycle Management (eLCM)



FUJITSU

Open standards

<HTML>

http://



OpenSSL
Cryptography and SSL/TLS Toolkit

Net-SNMP

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

FUJITSU

Open standards

<HTML>

http://



OpenSSL
Cryptography and SSL/TLS Toolkit

Net-SNMP

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

FUJITSU

Open standards

<HTML>

http://



OpenSSL
Cryptography and SSL/TLS Toolkit

Net-SNMP

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

SNMP ver 1/2x/3

Popular protocol for network management.

FUJITSU

Open standards

<HTML>

http://



OpenSSL
Cryptography and SSL/TLS Toolkit

Net-SNMP

Intelligent Platform Management Interface

IPMI - standardized, abstract, message-based interface between BMC and intelligent hardware for platform management. Key component of system.

Web: HTTP, HTML, JavaScript

Web-based control interface.

SNMP ver 1/2x/3

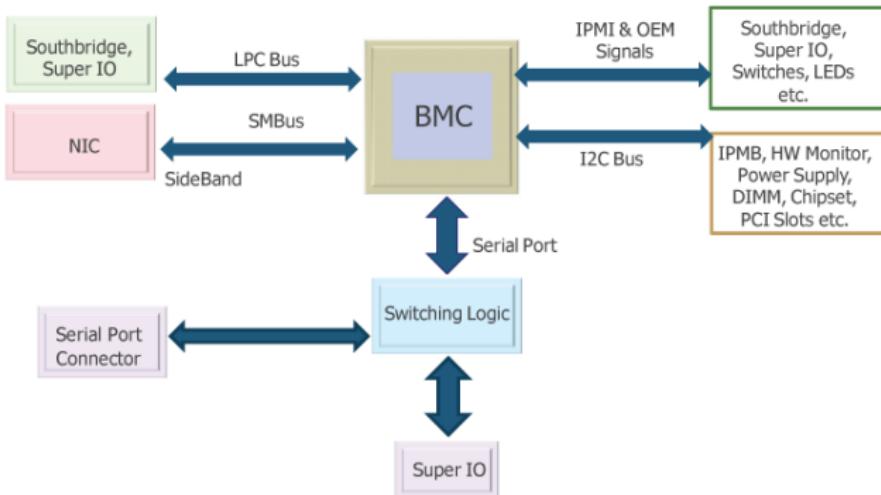
Popular protocol for network management.

Security: SSH and SSL

FUJITSU

IPMI - key interface of a system

IPMI Block Diagram



FUJITSU

Demo 1

Web interface: AVR, VirtualMedia, remote boot

Scenario 1: AVR, show boot settings

AVR: show Windows, Start LCM Custom Image, AVR: Show Linux



Web interface: AVR, VirtualMedia, remote boot

Scenario 1: AVR, show boot settings

AVR: show Windows, Start LCM Custom Image, AVR: Show Linux

Scenario 2: IPMI - via ipmitool

```
$ ipmitool -U admin -P admin -H 192.168.1.1 -I lanplus [command line]
```

command line variants:

- chassis status
- lan print
- user list
- sensor



Pro

- Advanced Real-Time Operation System
- Small footprint
- Fast performance



Pro

- Advanced Real-Time Operation System
- Small footprint
- Fast performance

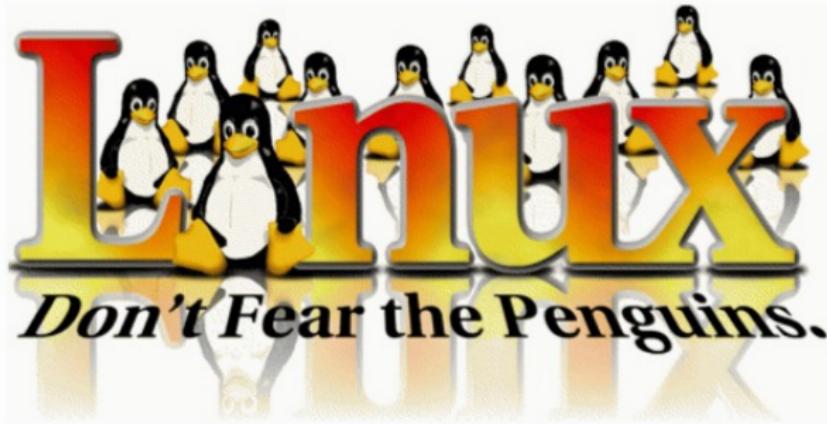
Contra

- Lack of available developers
- Lack of 3rd party ready components
- High cost of support
- Long features time-to-market
- Environment compatible only with themselves

Why Linux



FUJITSU



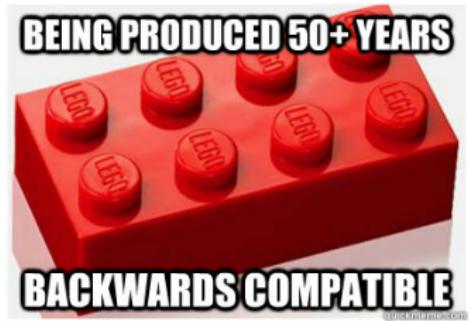
Cost of development and support

- More developers available
- Huge amount of 3rd party ready components
- Faster development
- HW platform fast enough to run it

Main challenges

Backward compatibility

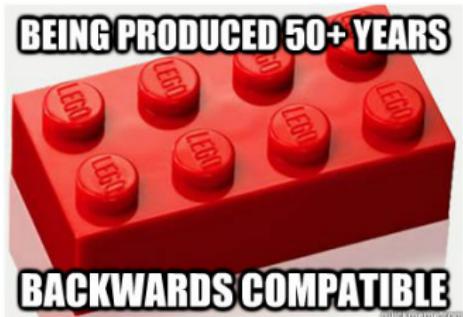
- Same interfaces (UI, protocols)
- Binary firmware upgrades



Main challenges

Backward compatibility

- Same interfaces (UI, protocols)
- Binary firmware upgrades



Code re-use



- OS API are different
- OS layout completely different
- HW-related stuff to rewrite from scratch

Demo 2: OpenSSH + RemoteManager

Interface ~~bug to bug~~ byte to byte identical to ThreadX.

```
*****
*   Welcome to PRIMERGY Remote Manager      *
*   Firmware Revision 98.10a (1.00)          *
*   SDR 3.16  ID 0401 TX1320M1              *
*   Firmware built Nov  5 2015 16:35:12 CET  *
*****
System Type  : PRIMERGY TX1320 M1
System ID    : YLXLXXXX36
System Name   : SUT-PW
System OS     : Windows Server 2016 Technical Preview 3 Standard
System Status: OK (Identify LED is OFF)
Power Status : Off
Asset Tag    : System Asset Tag

Main Menu

(1) System Information...
(2) Power Management...
(3) Enclosure Information...
(4) Service Processor...

(c) Change password
(*) Console Redirection (EMS/SAC)
(s) Start a Command Line shell...
(l) Console Logging

Enter selection or (0) to quit: □
```



iRMC Firmware components

Free and Open Source Software

- Linux Kernel
- U-Boot bootloader
- Busybox
- GNU Glibc
- Net-SNMP
- OpenSSH



FUJITSU

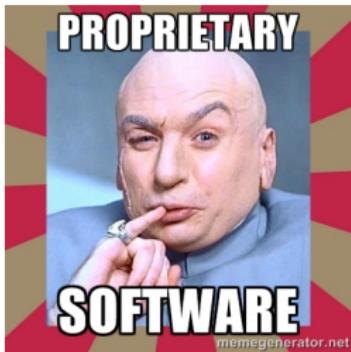
iRMC Firmware components

Free and Open Source Software

- Linux Kernel
- U-Boot bootloader
- Busybox
- GNU Glibc
- Net-SNMP
- OpenSSH



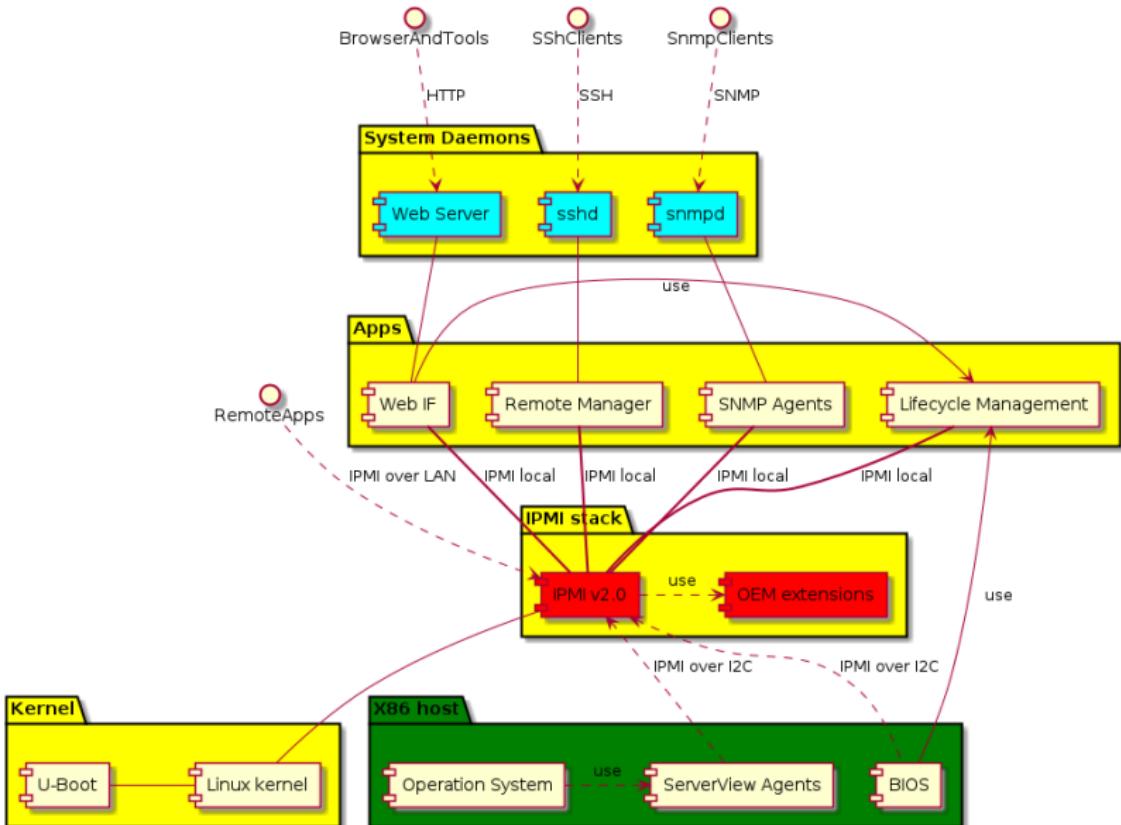
Closed source



- IPMI full stack powered by AMI
MegaRAC
- WebServer
- SNMP agents

FUJITSU

iRMC firmware internals



SU

Development environment

LXC containers + X2go for developers

The same environment for all to build and debug.

Read-only root filesystem on container.

Debian GNU/Linux based.

Custom package system by AMI MegaRAC technology

Used only in development and build process.

Not used for updates.

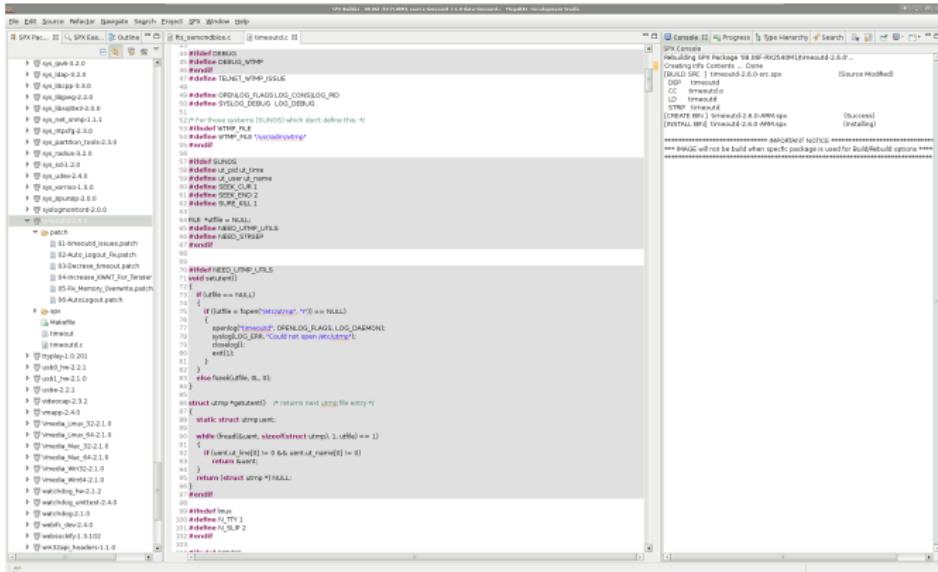
Package format similar to DEB, but not the same.

Eclipse-based IDE + AMI MegaRAC extensions

Rich IDE + version control + packaging system integration

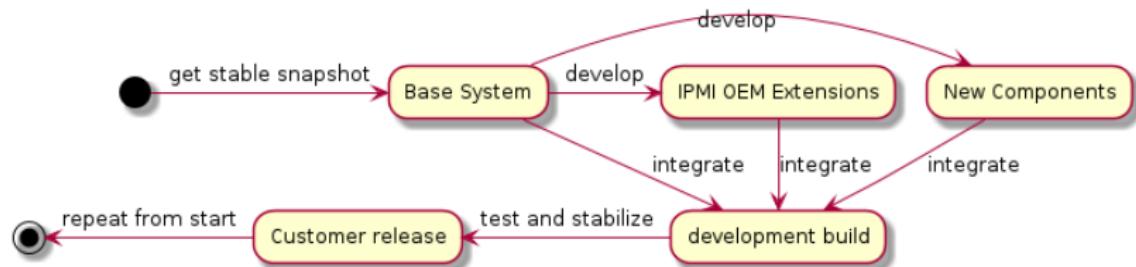


IDE: Eclipse + AMI MegaRAC extensions



FUJITSU

Development cycle



Very typical Embedded Linux development cycle (simplified view):

- ① Get base system snapshot and freeze it
- ② Develop new components and IPMI OEM extensions
- ③ Bug fix and stabilization
- ④ Test it hard
- ⑤ Release firmware to customers
- ⑥ Repeat once again from step (1).

FOSS legal questions

- Following the FOSS licenses
- Special policy for FOSS components using
- Consolidation of components legal status
- Rare upstream communication ¹
- FOSS component sources - by demand from support



¹no significant changes in upstream

Demo 3

Development login via SSH.
Show typical Embedded Linux system.



Questions? Remarks?



shaping tomorrow with you

- Fujitsu Primergy servers: <http://www.fujitsu.com/fts/products/computing/servers/primergy/>
- iRMC S4 manual: <http://manuals.ts.fujitsu.com/file/11470/irmc-s4-ug-en.pdf>
- Emulex Pilot 3 iBMC specs: <http://www.emulex.com/products/controllers/management-controllers/pilot-baseboard-management-controller/specifications>
- AMI MegaRAC technology by American Megatrends Inc: <http://ami.com/products/remote-management/>
- ThreadX RTOS: <http://rtos.com/products/threadx>
- Fujitsu Technology Solutions: <http://www.fujitsu.com/fts>

contact: Vladimir.Shakhov at ts.fujitsu.com

