Two Years Experience of Industrial-grade Open Source Base Layer Development and its Future

Yoshitake Kobayashi, Toshiba Corp., CIP TSC Chair
Urs Gleim, Siemens AG, CIP Board Chair
Open Source Summit Europe, Edinburgh, October 22, 2018
What is CIP?
Our Civilization runs on Linux
This morning around the corner…
<table>
<thead>
<tr>
<th>Transport</th>
<th>Energy</th>
<th>Industry</th>
<th>Others</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rail automation</td>
<td>Power Generation</td>
<td>Industry automation</td>
<td>Healthcare</td>
</tr>
<tr>
<td>Vehicle control</td>
<td></td>
<td>CNC control</td>
<td>Building automation</td>
</tr>
<tr>
<td>Automatic ticket gates</td>
<td>Turbine Control</td>
<td>Industrial communication</td>
<td>Broadcasting</td>
</tr>
</tbody>
</table>
There are issues to be solved…
A Power Plant System:

25-60 years products life-cycle

Very reluctant to perform product updates and upgrades of hardware and base software platform
Industrial IoT: Edge and Fog Computing

Functionality is moving from the cloud to the “Edge”

- Increasing number of networked industrial-grade devices
- Security management requires harmonized software landscape

Application examples on IIoT infrastructure

- Plant analytics
- SCADA functionality
- Plant (device) mgmt.
- Local / real-time analytics
- IoT Gateways

Data collection
- Pre-processing
- Sensor / actor connectivity

IoT: Internet of Things   IIoT: Industrial IoT   SCADA: Supervisory Control And Data Acquisition
The key challenges

Apply IoT concepts to industrial systems.

Ensure quality and longevity of products.

Keep millions of connected systems secure.

Industrial gradeness
- Reliability
- Functional Safety
- Real-time capabilities

Sustainability
- Product life-cycles of decades
- Backwards compatibility
- Standards

Security
- Security & vulnerability management
- Firmware updates
- Minimize risk of regressions
We maintain different flavors and version of Linux

...in each division...

...for several products...

...for many years.

...without having business advantages from doing this.

And other companies do the same.
Establishing an Open Source Base Layer of industrial-grade software to enable the use and implementation of software building blocks for Civil Infrastructure Systems

https://www.cip-project.org/

since April 2016
What is “Open Source Base Layer (OSBL)”?

- OSBL is a set of industrial grade core open source software components, tools and methods
  - Reference implementation
  - Start for a minimal set for controllers in industrial-grade systems

3 months upstream kernel

2-6 years Long Term Support (LTS) for desktop/server

2-6 years LTSI, support of embedded hardware

10-15 years super long support incl. core packages by [https://cip-project.org](https://cip-project.org) (Siemens, Hitachi, Toshiba, Renesas et al.)

Non-CIP packages
Linux distribution (e.g. Debian) may extend/include CIP packages.

CIP Reference file system image with SDK (CIP Core packages)

CIP SLTS Kernel

CIP Reference Hardware

User space

Kernel

Hardware
The backbone of CIP are the member companies

- Member companies
- Open source projects (Upstream work)
- Budget
- Developers, maintainers
- Contribution & usage / integration
- Optional: funding of selected projects
- CIP Super Long Term Support Project
- CIP source code repositories

Member Companies:
- HITACHI
- Renesas
- Siemens
- Toshiba
- Codethink
- Cybertrust
- Moxa
- Plat’Home

Open Source Summit Europe 2018
Mapping CIP into the company

Layered Linux distribution for industrial products, utilizing and influencing the relevant Open Source projects:

- **CIP Core Packages** *(tens)*
  - Corporate team / central project
- **CIP Kernel** *(10+ years maintenance)*
- **additional packages** *(hundreds)*

"distribution"

- Business Units / Products
- Companies / Divisions
- Domain-specific extensions
- Domain-specific extensions
- ... (vertical arrows)
- Firmware Update
- Security Hardening
- Container Runtime
- ... (horizontal arrows)
- Kernel and Base Packages, SDK, Build chain, QA

Up to 70% effort reduction achievable for OSS license clearing and vulnerability monitoring, kernel and package maintenance, application adaptation and testing for an individual product.

**OSS** Open Source Software
**CIP** Civil Infrastructure Platform project ([https://www.cip-project.org/](https://www.cip-project.org/))
**QA** quality assurance
**SDK** software development kit

Open Source Summit Europe 2018
CIP activities and status
Scope of activities

User space
- App container infrastructure (mid-term)
- App Framework (optionally, mid-term)
- Domain Specific communication (e.g. OPC UA)
- Shared config. & logging
- Multimedia
- CIP Core Packages
- Safe & Secure Update
- Monitoring
- Security
- Real-time support
- Real-time / safe virtualization

Kernel space
- Linux Kernel
- Super Long Term Supported Kernel (STLS)

Middleware/Libraries
- On-device software stack

Tools
- Build environment (e.g. bitbake, dpkg)
- Test automation
- Tracing & reporting tools
- Configuration management
- Device management (update, download)
- Application life-cycle management

Concepts
- Functional safety architecture/strategy, including compliance w/ standards (e.g., NERC CIP, IEC61508)
- Long-term support Strategy: security patch management
- Standardization collaborative effort with others
- License clearing
- Export Control Classification

Product development and maintenance
CIP focuses on upstream development

Contribute, Collaborate and use by CIP

Upstream Projects
- mainline
- LTS
- Real-Time LINUX
- debian
- LAVA
- KernelCI
- Reproducible Builds
- EdgeX Foundry
- yocto project

Contributing by CIP members as future candidates
- Jailhouse
- fossology

1. Upstream first
2. Use the upstream code
3. Integrate

CIP Open Source Base Layer (OSBL)
CIP SLTS kernel development (Upstream first development model)

- Upstream projects for CIP
  - Linux mainline and LTS

- How CIP collaborate with upstream?
  - Mainline
    - All backport patches should be up-streamed before merging
    - Many patches especially Renesas board related features has already up-streamed
  - Linux stable
    - CIP SLTS is based on LTS
    - CIP kernel team participate into LTS review process
4.4-stable review patch. If anyone has any objections, please let me know.

------------------
From: Christoph Hellwig <hch@lst.de>

commit f507b54dccfd8000c517d740bc45f20c74532d18 upstream.

The job structure is allocated as part of the request, so we should not free it in the error path of bsg_prepare_job.

Signed-off-by: Christoph Hellwig <hch@lst.de>
Reviewed-by: Ming Lei <ming lei@redhat.com>
Signed-off-by: Jens Axboe <axboe@kernel.dk>
Signed-off-by: Greg Kroah-Hartman <gregkh@linuxfoundation.org>

---
block/bsg-lib.c | 1 -
1 file changed, 1 deletion(-)

--- a/block/bsg-lib.c
+++ b/block/bsg-lib.c
@@ -147,7 +147,6 @@ static int bsg_create_job(struct device
    failjob_rls_rqst_payload:
        kfree(job->request_payload.sg_list);
    failjob_rls_job:
-       kfree(job);
        return -ENOMEM;
    }

Reviewed by Ben Hutchings for 4.4-stable

On Tue, 2017-10-03 at 14:21 +0200, Greg Kroah-Hartman wrote:
> 4.4-stable review patch. If anyone has any objections, please let me know.
> 
> ---------------
> > From: Christoph Hellwig <hch@lst.de>
> > > commit f507b54dccfd8000c517d740bc45f20c74532d18 upstream.
> > > The job structure is allocated as part of the request, so we should not free it in the error path of bsg_prepare_job.
> > 
> > That function doesn't exist here (it was introduced in 4.13). Instead, this backport has modified bsg_create_job(), creating a leak. Please revert this on the 3.18, 4.4 and 4.9 stable branches.
> > 
> > < -- snip -- >
> >
> > Ben Hutchings
> > Software Developer, Codethink Ltd.
CIP SLTS (linux-4.4.y-cip), Maintenance period 10 years

- CIP SLTS kernel tree is now available on kernel.org under the CIP group
  - https://git.kernel.org/pub/scm/linux/kernel/git/cip/linux-cip.git/
  - CIP RT kernel will be available soon
- Mentor: Ben Hutchings (Codethink)
- Maintainer: Nobuhiro Iwamatsu (Cybertrust), N.N. (will be announced soon)
CIP SLTS Kernel development

• Development status
  • The latest CIP kernel 4.4.154-cip28

• CIP Reference boards
  • QEMU x86_64
  • AM335x Beaglebone Black (Armv7)
  • RZ/G1M iWave Qseven Development Kit (Armv7)
  • RZ/G2M-96CE(tentative name) (Armv8)

• CIP Reference board candidates (under consideration)
  • Physical x86_64 board
  • Cyclone V Development Kit
Next SLTS kernel version

**Next CIP SLTS kernel**

- Mainline
- Stable (linux-stable)
- CIP SLTS (linux-4.4.y-cip)
- Stable (linux-stable-x.y)
- NEXT CIP SLTS (based on LTS)

**Backported patches**
- Maintained by Ben Hutchings

**Feature backports**
- Stop backporting. Focus to security fix only

**CIP will pick up LTS version**

Approx. 2-3 years

**CIP** will pick up LTS version
Real-time Linux development (PREEMPT_RT)

- Upstream projects for CIP
  - Real-time Linux Project

- How CIP collaborate with upstream
  - Contributing PREEMPT_RT mainlining
    - Join Real-time Linux Project as Gold member
  - Contributing RT stable maintenance
    - CIP member Daniel Wagner from Siemens is maintaining 4.4.y-stable-rt
    - CIP 4.4-rt-cip kernel based on 4.4.y-stable-rt
  - Test results are available on CI-RT
    - [https://ci-rt.linutronix.de/RT-Test/](https://ci-rt.linutronix.de/RT-Test/)

- Contributions
  - Contributing RT patch mainlining
  - Join Real-time Linux Project as Gold member
  - Contributing RT stable maintenance
  - CIP member Daniel Wagner from Siemens is maintaining 4.4.y-stable-rt
  - CIP 4.4-rt-cip kernel based on 4.4.y-stable-rt
  - Test results are available on CI-RT

- Test results are available on CI-RT
  - [https://ci-rt.linutronix.de/RT-Test/](https://ci-rt.linutronix.de/RT-Test/)
CIP Testing

- Upstream projects
  - CIP is using LAVA2 and KernelCI for testing environment
  - CIP member contributing
    • Fuego

- How CIP collaborate with upstream?
  - CIP testing B@D created with KernelCI and LAVA2
  - Sharing CIP testing results to public
    • https://lists.cip-project.org/pipermail/cip-testing-results/
  - Posting patches to creating features
    • Send test results from Fuego to KernelCI
    • LAVA support on Fuego

- What’s next?
  • Closer collaboration with KernelCI
CIP testing

CIP Testing project
(https://wiki.linuxfoundation.org/civilinfrastructureplatform/ciptesting)

• B@D designed to:
  • Test Linux kernels and base systems locally.
  • On hardware connected to your dev machine.

• B@D features
  • Based on kernelci.org
  • Linux and Windows 10 as Host OS supported.
  • Shipped as VM and Vagrant based environments.
  • Results and logs sharing capabilities.

• Check the source code involved
  • https://gitlab.com/cip-project/cip-testing/board-at-desk-single-dev/tree/master
• Upstream projects for CIP
  • Debian
    • Debian-LTS
    • Debian-Cross
  • Deby / ISAR / ELBE / Yocto Project
• CIP Core will move to have 2 profiles
  • Tiny (Bitbake + Debian source code)
  • Debian (Debian binary (dpkg) based)
• How CIP collaborate with upstream?
  • Support Debian-LTS project
  • Contributing patches to Debian-cross
  • CIP Core uses Yocto Project and Deby
    • Deby is a layer for Poky to use Debian source code
• Upstream projects for CIP
  • Debian
    • Debian-LTS
  • Debian-LTS
  • Debian
• CIP Core project
  • Tiny (Bitbake + Debian source code)
  • Debian (Debian binary (dpkg) based)
• How CIP Core collaborate with upstream
  • Support Debian-LTS project
  • Contributing patches to Debian-cross
  • CIP Core uses Yocto Project and Deby
    • Deby is a layer for Poky to use Debian source code

The latest status of this activity will present at the following talk

Debian & Yocto: State of the Art - Kazuhiro Hayashi, Toshiba Corporation & Manuel Traut, Linutronix GmbH & Baurzhan Ismagulov

➔ Tuesday, October 23 • 14:30 - 15:10
An example of minimal package set for CIP base layer

Candidates for initial component set

- Kernel
  - Linux kernel + backported patches
  - PREEMPT_RT patch
- Bootloader
  - U-boot
- Shells / Utilities
  - Busybox
- Base libraries
  - Glibc
- Tool Chain
  - Binutils
  - GCC
- Security
  - OpenSSL

NOTE: The maintenance effort varies considerably for different packages.

Keep these packages for Reproducible build

- Flex
- Bison
- autoconf
- automake
- bc
- bison
- Bzip2
- Curl
- Db
- Dbus
- Expat
- Flex
- gawk
- Gdb

- Git
- Glib
- Gmp
- gettext
- Kbd
- Libibverbs
- Libtool
- Libxml2
- Mpclib
- Mpfr4
- Ncurses
- Make
- M4

- pax-utils
- Pciutils
- Perl
- pkg-config
- Popt
- Procps
- Quilt
- Readline
- sysfsutils
- Tar
- Unifdef
- Zlib
## Gaps and Common Goals between Debian and CIP

<table>
<thead>
<tr>
<th>Debian</th>
<th>CIP requires</th>
<th>Chance to collaborate with Debian</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Support</strong></td>
<td><strong>Support</strong></td>
<td><strong>Longer</strong> term maintenance for limited number of packages (CIP joined Debian-LTS)</td>
</tr>
<tr>
<td>- Term: 3+2 years by Debian-LTS</td>
<td>- Term: 10+ years</td>
<td></td>
</tr>
<tr>
<td>- Num of source pkgs: over 25000 (67776 binary pkgs)</td>
<td>- Num of pkgs: 10+ (minimum)</td>
<td></td>
</tr>
<tr>
<td><strong>Build</strong></td>
<td><strong>Build</strong></td>
<td><strong>Contributing to Debian-cross</strong> (RFC posted to Debian-cross)</td>
</tr>
<tr>
<td>- Should support native build</td>
<td>- Need to have both native and cross build</td>
<td></td>
</tr>
<tr>
<td>- Working on cross build packaging (Debian-cross)</td>
<td>- Binary / Source code should be managed and reproducible</td>
<td></td>
</tr>
<tr>
<td>- Reproducible build</td>
<td><strong>OSS license compliance</strong></td>
<td><strong>Exchange and share the license review results</strong></td>
</tr>
<tr>
<td><strong>OSS license compliance</strong></td>
<td></td>
<td><strong>Contributing test cases to upstream</strong></td>
</tr>
<tr>
<td>- DEP-5 adoption is ongoing</td>
<td>- Generate reports automatically</td>
<td></td>
</tr>
<tr>
<td><strong>Testing</strong></td>
<td><strong>Testing</strong></td>
<td></td>
</tr>
<tr>
<td>- Packages has to be tested</td>
<td>- Easy to redistribute</td>
<td></td>
</tr>
<tr>
<td>- autopkgtest</td>
<td>- All packages should be tested in timely manner</td>
<td></td>
</tr>
</tbody>
</table>
Security working group

- CIP launched a new working group to focus on cybersecurity

**Goal**
- Provide guidelines and reference implementations to help developers to meet cybersecurity standard requirements (IEC 62443)

**Status**
- Just started

---

* CIP deliverables*

- Certified platform
  - Board (e.g. Reference board)
- Middleware (e.g. Device driver)
- Linux Kernel (CIP)

Compliant environment for evaluation
- Equipment for evaluation (TBD.)
- Test cases
- Reference implementations

Component

<table>
<thead>
<tr>
<th>H/W</th>
<th>S/W</th>
<th>Document</th>
</tr>
</thead>
<tbody>
<tr>
<td>User equipment</td>
<td>User application</td>
<td>User manual</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Design document</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Evaluation document</td>
</tr>
</tbody>
</table>

Guideline and evidence
- Security requirement
- Implement for security
- Evaluation evidence
- Application note
- Implements guideline

*: Noted that this image is under planning and for only illustrative purposes.
Software update working group

• CIP launched a new working group to focus software update

• Goal
  • Incorporate a common solution for software updates into CIP core

• Status
  • Just started
Collaboration: EdgeX Foundry on CIP Core

• CIP has joined EdgeX Foundry as Associate Member

• EdgeX Foundry on CIP Core
  • Purpose
    • To demonstrate CIP Core provides maintained base system for IoT systems

  • Goal
    • Create a sample implementation to run EdgeX Foundry on CIP Core
    • Proof of concept project

  • Status
    • Source code is available on CIP GitLab
      https://gitlab.com/cip-playground/edgex-cip
Collaboration: Reproducible builds

• CIP became a sponsor of Reproducible builds

• Collaboration plan
  • Ensure reproducible build for CIP Core over lifetime
Summary

• CIP today focuses on
  • **Kernel maintenance**: maintaining Linux kernels for very long time including real-time support
  • **Testing**: providing a test infrastructure and evolve tests
  • **CIP Core packages**: a set of industrial-grade components that require super long-term maintenance including the required build tool chains
  • **Security**: Improving to have security features and to follow Cyber Security Standard
  • **Collaboration**: Linux stable, Debian/Debian-LTS, Real Time Linux, Reproducible Builds, EdgeX Foundry

• New activities started: Security (IEC 62443-4-2), SW update
Conclusion

• Our Civilization needs an Open Source Base Layer of industrial-grade software
  • CIP provides this, using Linux

• Sustainability is ensured by
  • The backing of big industrial and semiconductor companies
  • Close cooperation with and building with mature Open Source projects (Debian, PREEMPT_RT, KernelCI, ...)
  • Providing suitable tool chains
  • Ensuring in-depth tests

• Contribution and collaboration with upstream projects are the key CIP activities
Questions?
Contact Information and Resources

To get the latest information, please contact:
  • CIP Mailing list: cip-dev@lists.cip-project.org

Other resources
  • CIP Web site: https://www.cip-project.org
  • CIP Wiki: https://wiki.linuxfoundation.org/civilinfrastructureplatform/
  • CIP source code
    • CIP GitLab: http://www.gitlab.com/cip-project
    • CIP kernel: git://git.kernel.org/pub/scm/linux/kernel/git/cip/linux-cip.git
Thanks for your attention!