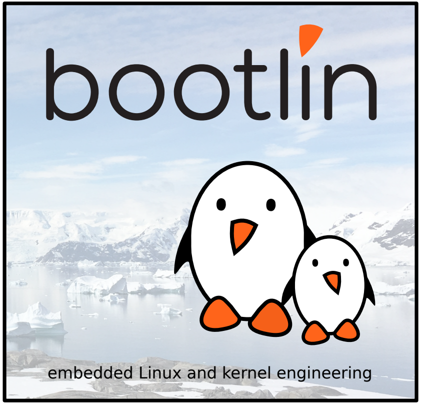
Embedded Linux Conference

Buildroot: what's new?

Thomas Petazzoni
thomas.petazzoni@bootlin.com



Thomas Petazzoni

- Co-owner and CEO at **Bootlin**
  - Embedded Linux experts
  - Engineering services: Linux BSP development, kernel porting and drivers, Yocto/Buildroot integration, real-time, boot-time, security, multimedia
  - Training services: Embedded Linux, Linux kernel drivers, Yocto, Buildroot, graphics stack, boot-time, real-time
- Co-maintainer of **Buildroot**, contributor since 2008, 5200+ patches contributed.
- Former contributor to the Linux kernel, 900+ patches contributed.
- Program committee member and regular speaker at the Embedded Linux Conference
- Living in **Toulouse**, south west of France
- thomas@bootlin.com

https://bootlin.com/company/staff/thomas-petazzoni/

Agenda

- What is Buildroot?
- Comparison with Yocto
- What's new between 2020.05 and 2022.05
  - Some numbers
  - LTS and security maintenance
  - Security vulnerability tracking
  - Security default settings
  - SELinux integration improvements
  - Vendoring support for Go/Rust
  - Python changes
  - Significant new packages
  - Testing improvements
  - Toolchain improvements
  - Architecture support

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist
- Relies on **well-known** technologies
  - *GNU Make* for the build, *Kconfig* for configuration

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist
- Relies on **well-known** technologies
  - *GNU Make* for the build, *Kconfig* for configuration
- Simple to use and learn
  - `make menuconfig` → `make` → profit

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist
- Relies on **well-known** technologies
  - *GNU Make* for the build, *Kconfig* for configuration
- Simple to use and learn
  - `make menuconfig` → `make` → profit
- **2800+** built-in packages for most popular open-source software stacks

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist
- Relies on **well-known** technologies
  - *GNU Make* for the build, *Kconfig* for configuration
- Simple to use and learn
  - `make menuconfig` → `make` → profit
- **2800+** built-in packages for most popular open-source software stacks
- Very **active community** of developers/users, used by many companies: silicon vendors, embedded system manufacturers, hobbyists

What is Buildroot?

- Is an **Embedded Linux build system**
  - Tool that **automates** the cross-compilation of a complete embedded Linux system from source
  - Builds: toolchain/compiler, bootloader and Linux kernel, complete root filesystem with user-space applications and libraries
- Similar aim as Yocto/OpenEmbedded, OpenWrt, PTXdist
- Relies on **well-known** technologies
  - *GNU Make* for the build, *Kconfig* for configuration
- Simple to use and learn
  - `make menuconfig` → `make` → profit
- **2800+** built-in packages for most popular open-source software stacks
- Very **active community** of developers/users, used by many companies: silicon vendors, embedded system manufacturers, hobbyists
- Oldest still maintained build system: started in 2001

Buildroot vs. Yocto

The one question that everybody asks!

Buildroot vs. Yocto

- What it builds
  - **Yocto**: builds a distribution, with binary packages and a package management system
  - **Buildroot**: builds a fixed functionality root filesystem, no binary packages
  - Note: binary packages are not necessarily a good thing for embedded!

Buildroot vs. Yocto

- What it builds
- Configuration
  - **Yocto**: flexible, powerful but complex configuration description
  - **Buildroot**: very simple configuration system, but sometimes limited

Buildroot vs. Yocto

- What it builds
- Configuration
- Build strategy
  - **Yocto**: complex and heavy logic, but with efficient caching of artifacts and “rebuild only what’s needed” features
  - **Buildroot**: simple but somewhat dumb logic, no caching of built artifacts, full rebuilds needed for some config changes

Buildroot vs. Yocto

- What it builds
- Configuration
- Build strategy
- Ecosystem
  - **Yocto**: (relatively) small common base in OpenEmbedded, lots of features supported in third party layers → lots of things, but varying quality
  - **Buildroot**: everything in one tree → perhaps less, but more consistent quality

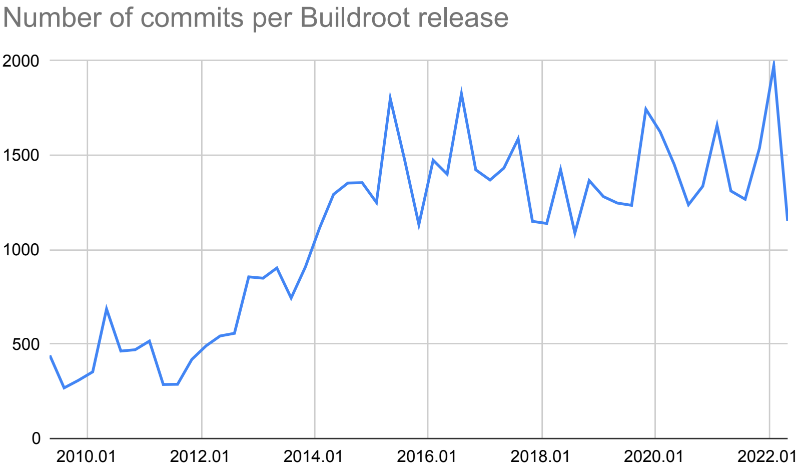
Buildroot vs. Yocto

- What it builds
- Configuration
- Build strategy
- Ecosystem
- Complexity/learning curve
  - **Yocto**: admittedly steep learning curve, *bitbake* remains a magic black box for most people
  - **Buildroot**: much smoother and shorter learning curve, tool is simple to approach, and reasonably simple to understand

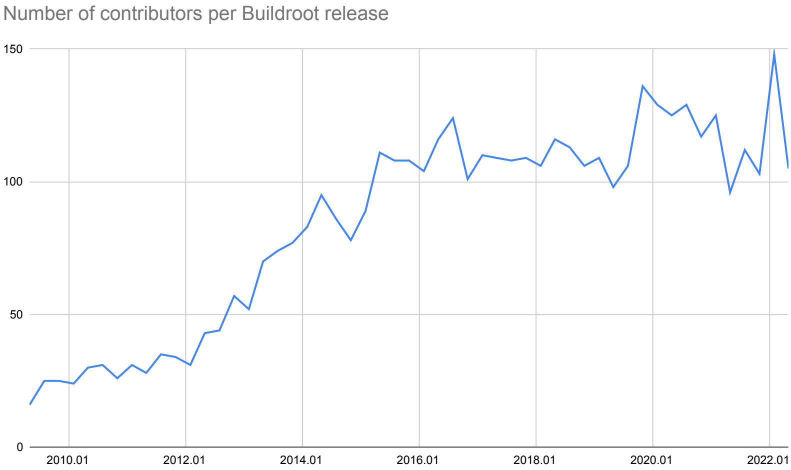
Buildroot vs. Yocto

- What it builds
- Configuration
- Build strategy
- Ecosystem
- Complexity/learning curve
- And also a matter of personal taste/preference, as often when choosing tools

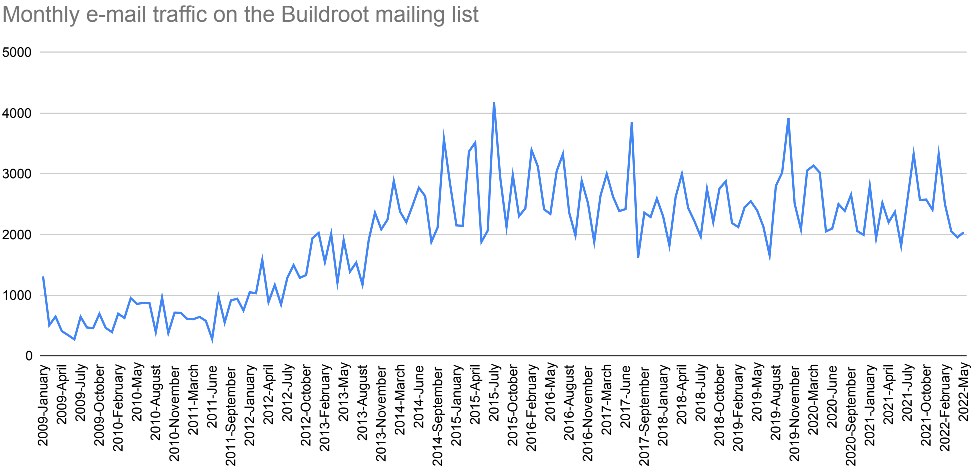
Community vitality: some numbers



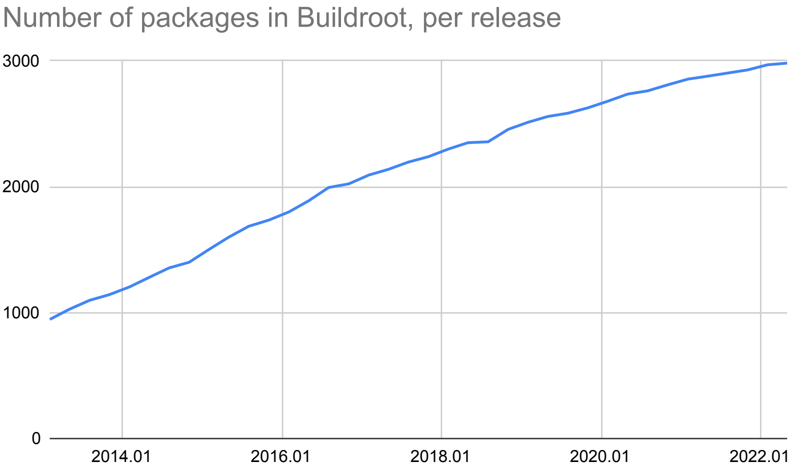
Community vitality: some numbers



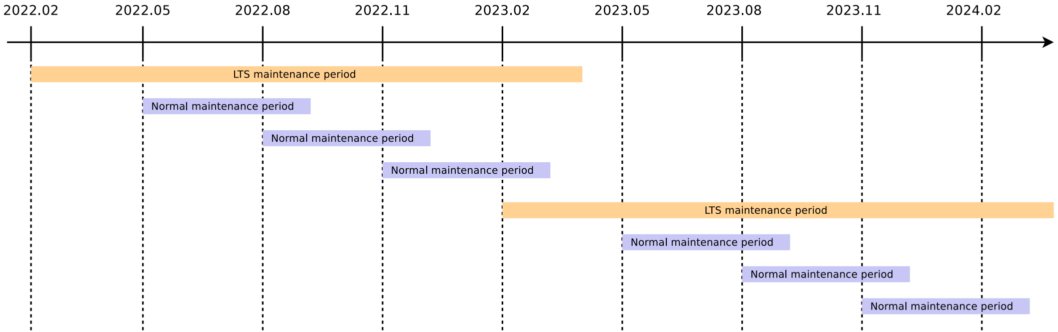
Community vitality: some numbers



Community vitality: some numbers



LTS and security maintenance


2022.02
2022.05
2022.08
2022.11
2023.02
2023.05
2023.08
2023.11
2024.02
LTS maintenance period
Normal maintenance period
Normal maintenance period
Normal maintenance period
LTS maintenance period
Normal maintenance period
Normal maintenance period
Normal maintenance period

- Maintenance: security fixes, bug fixes
- *YYYY.02* releases maintained for slightly over 12 months
- *YYYY.{05,08,11}* releases maintained for slightly over 3 months

LTS and security maintenance

- Process started with 2019.02.x
- Process now works well: review of all commits in *master* and decision if applicable to the current LTS branch
- **2020.02.x** LTS branch
  - 1219 commits
  - 228 directly security related (probably more in reality)
  - 12 point releases: 2020.02.1 → 2020.02.12
  - End of life
- **2021.02.x** LTS branch
  - 751 commits
  - 110 directly security related (same, probably more in reality)
  - 12 point releases: 2021.02.1 → 2021.02.12
  - End of life on April 6, 2022
- **2022.02.x** LTS branch, current
  - Started in February 2022
  - End of life planned on April 2023

Security vulnerability tracking

- `make pkg-stats` matches your package set with the *NIST* security vulnerability database
  - CVE database: *Common Vulnerability and Exposure*
  - CPE database: *Common Platform Enumeration*
  - HTML and JSON output

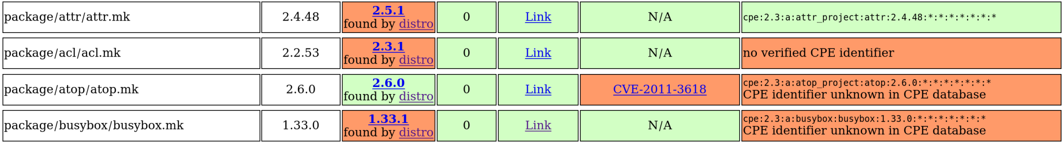
Security vulnerability tracking

- `make pkg-stats` matches your package set with the *NIST* security vulnerability database
  - CVE database: *Common Vulnerability and Exposure*
  - CPE database: *Common Platform Enumeration*
  - HTML and JSON output
- Checks if packages...
  - are affected by known CVEs
  - have a CPE identifier known in the CPE database

Security vulnerability tracking

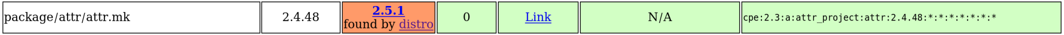
- `make pkg-stats` matches your package set with the *NIST* security vulnerability database
  - CVE database: *Common Vulnerability and Exposure*
  - CPE database: *Common Platform Enumeration*
  - HTML and JSON output
- Checks if packages...
  - are affected by known CVEs
  - have a CPE identifier known in the CPE database
- Helped with metadata from the package `.mk` file
  - `<pkg>_IGNORE_CVES` to ignore matching CVEs if they are fixed locally by a security fix backport
  - `<pkg>_CPE_ID_...` to override the default CPE identifier for the package `cpe:2.3:a:<pkg>_project:<pkg>:<pkg-version>:*:*:*:*:*:*`

pkg-stats output details



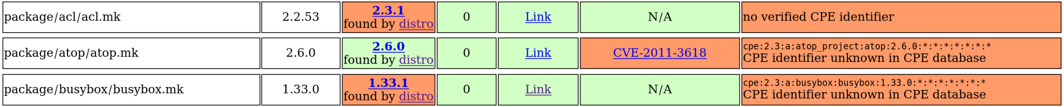
package/attr/attr.mk	2.4.48	**2.5.1** found by distro	0	Link	N/A	`cpe:2.3:a:attr_project:attr:2.4.48:*:*:*:*:*:*:*`
package/acl/acl.mk	2.2.53	**2.3.1** found by distro	0	Link	N/A	no verified CPE identifier
package/atop/atop.mk	2.6.0	**2.6.0** found by distro	0	Link	CVE-2011-3618	`cpe:2.3:a:atop_project:atop:2.6.0:*:*:*:*:*:*:*` CPE identifier unknown in CPE database
package/busybox/busybox.mk	1.33.0	**1.33.1** found by distro	0	Link	N/A	`cpe:2.3:a:busybox:busybox:1.33.0:*:*:*:*:*:*:*` CPE identifier unknown in CPE database

pkg-stats output details



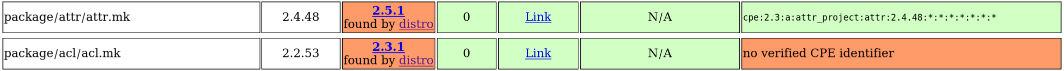
package/attr/attr.mk	2.4.48	2.5.1 found by distro	0	Link	N/A	cpe:2.3:a:attr_project:attr:2.4.48:*:*:*:*:*:*

- some <pkg>_CPE_ID_* variables defined → CPE information verified
- CPE identifier exists in the CPE dictionary
- no known CVEs



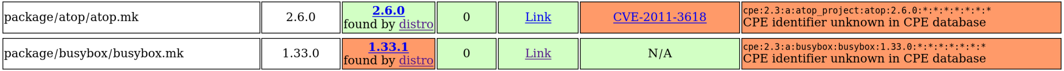
package/acl/acl.mk	2.2.53	2.3.1 found by distro	0	Link	N/A	no verified CPE identifier
package/atop/atop.mk	2.6.0	2.6.0 found by distro	0	Link	CVE-2011-3618	cpe:2.3:a:atop_project:atop:2.6.0:*:*:*:*:*:* CPE identifier unknown in CPE database
package/busybox/busybox.mk	1.33.0	1.33.1 found by distro	0	Link	N/A	cpe:2.3:a:busybox:busybox:1.33.0:*:*:*:*:*:* CPE identifier unknown in CPE database

pkg-stats output details



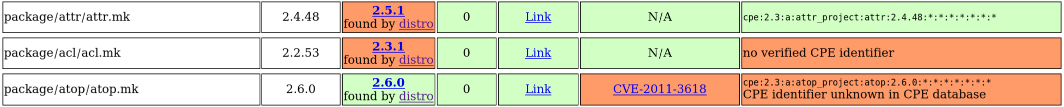
package/attr/attr.mk	2.4.48	**2.5.1** found by distro	0	Link	N/A	cpe:2.3:a:attr_project:attr:2.4.48:*:*:*:*:*:*:*
package/acl/acl.mk	2.2.53	**2.3.1** found by distro	0	Link	N/A	no verified CPE identifier

- no `<pkg>_CPE_ID_*` variable → don't know if the default CPE identifier is correct
- based on this default CPE identifier → no known CVE



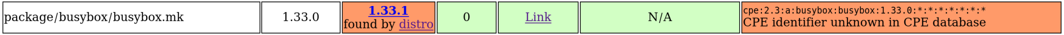
package/atop/atop.mk	2.6.0	**2.6.0** found by distro	0	Link	CVE-2011-3618	cpe:2.3:a:atop_project:atop:2.6.0:*:*:*:*:*:*:* CPE identifier unknown in CPE database
package/busybox/busybox.mk	1.33.0	**1.33.1** found by distro	0	Link	N/A	cpe:2.3:a:busybox:busybox:1.33.0:*:*:*:*:*:*:* CPE identifier unknown in CPE database

pkg-stats output details



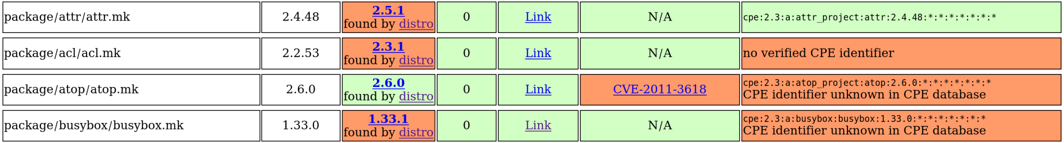
package/attr/attr.mk	2.4.48	2.5.1 found by distro	0	Link	N/A	cpe:2.3:a:attr_project:attr:2.4.48:*:*:*:*:*:*:*
package/acl/acl.mk	2.2.53	2.3.1 found by distro	0	Link	N/A	no verified CPE identifier
package/atop/atop.mk	2.6.0	2.6.0 found by distro	0	Link	CVE-2011-3618	cpe:2.3:a:atop_project:atop:2.6.0:*:*:*:*:*:*:* CPE identifier unknown in CPE database

- some `<pkg>_CPE_ID_*` variables defined → CPE information verified
- no entry in CPE dictionary → version 2.6.0 not known by NVD
- CVE-2011-3618 applicable: NVD database indicates it applies to all versions.



package/busybox/busybox.mk	1.33.0	1.33.1 found by distro	0	Link	N/A	cpe:2.3:a:busybox:busybox:1.33.0:*:*:*:*:*:*:* CPE identifier unknown in CPE database

pkg-stats output details



package/attr/attr.mk	2.4.48	2.5.1 found by distro	0	Link	N/A	cpe:2.3:a:attr_project:attr:2.4.48:*:*:*:*:*:*
package/acl/acl.mk	2.2.53	2.3.1 found by distro	0	Link	N/A	no verified CPE identifier
package/atop/atop.mk	2.6.0	2.6.0 found by distro	0	Link	CVE-2011-3618	cpe:2.3:a:atop_project:atop:2.6.0:*:*:*:*:*:* CPE identifier unknown in CPE database
package/busybox/busybox.mk	1.33.0	1.33.1 found by distro	0	Link	N/A	cpe:2.3:a:busybox:busybox:1.33.0:*:*:*:*:*:* CPE identifier unknown in CPE database

- some <pkg>_CPE_ID_* variables defined → CPE information verified
- no entry in CPE dictionary → version 1.33.0 not known by NVD
- no known CVEs

Security default settings

- Default configuration settings changed to enable more security-hardening features
- *PIC/PIE* (position independent) → needed for some other security features
- *SSP* (Stack Smashing Protection) enabled by default: `-fstack-protector`
- *RELRO* (RELocation Read Only) enabled by default, making additional ELF sections read-only
- *FORTITY_SOURCE* enabled by default, adds additional checks in the C library for buffer overflows

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled
- Reduce the size of the default policy from 2.4 MB to 250 KB, keeping only *base* SELinux policy modules

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled
- Reduce the size of the default policy from 2.4 MB to 250 KB, keeping only *base* SELinux policy modules
- Allow the *refpolicy* package to
  - Enable additional modules
  - Provide additional custom modules

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled
- Reduce the size of the default policy from 2.4 MB to 250 KB, keeping only *base* SELinux policy modules
- Allow the *refpolicy* package to
  - Enable additional modules
  - Provide additional custom modules
- Allow packages to enable additional SELinux modules
  - From the standard refpolicy using `<pkg>_SELINUX_MODULES`
  - Custom per-package ones in `package/<pkg>/selinux/`

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled
- Reduce the size of the default policy from 2.4 MB to 250 KB, keeping only *base* SELinux policy modules
- Allow the *refpolicy* package to
  - Enable additional modules
  - Provide additional custom modules
- Allow packages to enable additional SELinux modules
  - From the standard refpolicy using `<pkg>_SELINUX_MODULES`
  - Custom per-package ones in `package/<pkg>/selinux/`
- Many packages annotated with `<pkg>_SELINUX_MODULES`

```
SYSTEMD_SELINUX_MODULES = systemd udev xdg
```

SELinux integration improvements

- Set the SELinux file security context at build time and not run-time → allows read-only root filesystems with SELinux enabled
- Reduce the size of the default policy from 2.4 MB to 250 KB, keeping only *base* SELinux policy modules
- Allow the *refpolicy* package to
  - Enable additional modules
  - Provide additional custom modules
- Allow packages to enable additional SELinux modules
  - From the standard refpolicy using `<pkg>_SELINUX_MODULES`
  - Custom per-package ones in `package/<pkg>/selinux/`
- Many packages annotated with `<pkg>_SELINUX_MODULES`

```
SYSTEMD_SELINUX_MODULES = systemd udev xdg
```

- Contributions to upstream SELinux refpolicy to make it work with Buildroot

Vendoring support for Go/Rust (1)

- Go and Rust have **language-specific package managers**
- These package managers automatically download the dependencies
  - Described by `go.mod` in Go
  - Described by `Cargo.toml` in Rust

Vendoring support for Go/Rust (1)

- Go and Rust have **language-specific package managers**
- These package managers automatically download the dependencies
  - Described by `go.mod` in Go
  - Described by `Cargo.toml` in Rust
- They break fundamental features of build systems
  - Integration into a download infrastructure: caching, local backup site
  - Legal/license information collection: source code, license files
  - Reproducibility

Vendoring support for Go/Rust (2)

- Buildroot download infrastructure has been extended with *post-download helpers*

Vendoring support for Go/Rust (2)

- Buildroot download infrastructure has been extended with *post-download helpers*
- After downloading the main package source code, ability to run custom logic to finalize the download
  - `support/download/go-post-process`
  - `support/download/cargo-post-process`

Vendoring support for Go/Rust (2)

- Buildroot download infrastructure has been extended with *post-download helpers*
- After downloading the main package source code, ability to run custom logic to finalize the download
  - `support/download/go-post-process`
  - `support/download/cargo-post-process`
- Runs the Go or Rust specific tools to retrieve the dependencies
- Makes sure that
  - The tarball contains the full source code, dependencies included
  - The hash used by Buildroot to validate the tarball covers also the dependencies
  - All source code and license files are available in the tarball

Vendoring support for Go

- Existing `golang-package` infrastructure
- Extended to use the *post-download* helper

package/tinifier/tinifier.mk

```
TINIFIER_VERSION = 3.4.0
TINIFIER_SITE = $(call github,tarampampam,tinifier,v$(TINIFIER_VERSION))
TINIFIER_LICENSE = MIT
TINIFIER_LICENSE_FILES = LICENSE
TINIFIER_GOMOD = ./cmd/tinifier

$(eval $(golang-package))
```

Vendoring support for Rust

- Newly added `cargo-package` infrastructure
- Uses a *post-download* helper

package/bat/bat.mk

```
BAT_VERSION = 0.19.0
BAT_SITE = $(call github,sharkdp,bat,v$(BAT_VERSION))
BAT_LICENSE = Apache-2.0 or MIT
BAT_LICENSE_FILES = LICENSE-APACHE LICENSE-MIT

$(eval $(cargo-package))
```

Python changes: Python 2.x removed

- Python 2.x EOL upstream in January 2020
- Kept for some time in Buildroot, marked deprecated, to help migration
- Finally removed in *2022.02*
- Allowed to remove a lot of complexity that was needed to support Python 2.x and Python 3.x in parallel

Python changes: PEP517 build system support

- Standard replacement for `setup.py`
- Uses a `pyproject.toml` file
- For now Buildroot supports `flit` based PEP517 build systems
- Needs `<pkg>_SETUP_TYPE = flit`

package/python-cssselect2/python-cssselect2.mk

```
PYTHON_CSSSELECT2_VERSION = 0.6.0
PYTHON_CSSSELECT2_SOURCE = cssselect2-$(PYTHON_CSSSELECT2_VERSION).tar.gz
PYTHON_CSSSELECT2_SITE = https://files.pythonhosted.org/packages/68/62/[...]
PYTHON_CSSSELECT2_SETUP_TYPE = flit
PYTHON_CSSSELECT2_LICENSE = BSD-3-Clause
PYTHON_CSSSELECT2_LICENSE_FILES = LICENSE

$(eval $(python-package))
```

Significant new packages

- ≈ 290 new packages added between 2020.05 and 2022.05
- GNU Octave
- Tracing: bpftool, uftrace, ply, babeltrace2
- ARM Mali GPU drivers
- Zabbix
- liburing
- WirePlumber
- OpenCV 4
- libvirt
- OpenZFS
- PostGIS
- Additional Qt5 modules: Qt5Knx, Qt5Coap, Qt5Mqtt, Qt5Lottie
- 63 additional Python packages

CI testing improvements

- Already existing
  - **Build-time** testing of semi-random configurations, `autobuild.buildroot.org`
  - Suite of **run-time tests**, `support/testing`, tested in Gitlab CI
  - Defconfigs build tested, and if possible boot tested, in Gitlab CI

CI testing improvements

- Already existing
  - **Build-time** testing of semi-random configurations, `autobuild.buildroot.org`
  - Suite of **run-time tests**, `support/testing`, tested in Gitlab CI
  - Defconfigs build tested, and if possible boot tested, in Gitlab CI
- Improvement
  - Switch to **fully random configurations** for build-time testing
  - Architecture/toolchain config used to be taken from a set of pre-defined config
  - Only package set was randomized
  - Now the full configuration is randomized
  - Allowed to **detect many corner cases**, and fix them
  - Still on-going

Architecture support

- Buildroot already **supports more CPU architectures** than any other build system
  - Synopsys ARC (LE/BE), AArch64 (LE/BE), ARM (LE/BE, including no-MMU Cortex-M), C-SKY, x86 (32-bit/64-bit), m68k, Microblaze (LE/BE), MIPS (32-bit/64-bit, LE/BE), NIOSII, OpenRISC, PowerPC, Power64 (LE/BE), RISC-V (32-bit/64-bit), SuperH, SPARC (32-bit/64-bit), Cadence Xtensa

Architecture support

- Buildroot already **supports more CPU architectures** than any other build system
  - Synopsys ARC (LE/BE), AArch64 (LE/BE), ARM (LE/BE, including no-MMU Cortex-M), C-SKY, x86 (32-bit/64-bit), m68k, Microblaze (LE/BE), MIPS (32-bit/64-bit, LE/BE), NIOSII, OpenRISC, PowerPC, Power64 (LE/BE), RISC-V (32-bit/64-bit), SuperH, SPARC (32-bit/64-bit), Cadence Xtensa
- Addition of support for the **S390x** CPU architecture
  - Contributed and maintained directly by IBM

Architecture support

- Buildroot already **supports more CPU architectures** than any other build system
  - Synopsys ARC (LE/BE), AArch64 (LE/BE), ARM (LE/BE, including no-MMU Cortex-M), C-SKY, x86 (32-bit/64-bit), m68k, Microblaze (LE/BE), MIPS (32-bit/64-bit, LE/BE), NIOSII, OpenRISC, PowerPC, Power64 (LE/BE), RISC-V (32-bit/64-bit), SuperH, SPARC (32-bit/64-bit), Cadence Xtensa
- Addition of support for the **S390x** CPU architecture
  - Contributed and maintained directly by IBM
- Addition of support for the **RISC-V 64-bit no-MMU** architecture
  - Contributed mainly by Western Digital

Architecture support

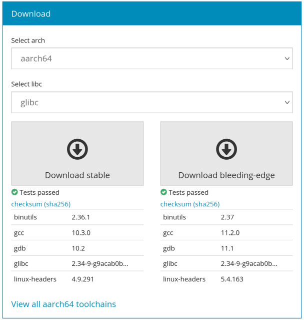
- Buildroot already **supports more CPU architectures** than any other build system
  - Synopsys ARC (LE/BE), AArch64 (LE/BE), ARM (LE/BE, including no-MMU Cortex-M), C-SKY, x86 (32-bit/64-bit), m68k, Microblaze (LE/BE), MIPS (32-bit/64-bit, LE/BE), NIOSII, OpenRISC, PowerPC, Power64 (LE/BE), RISC-V (32-bit/64-bit), SuperH, SPARC (32-bit/64-bit), Cadence Xtensa
- Addition of support for the **S390x** CPU architecture
  - Contributed and maintained directly by IBM
- Addition of support for the **RISC-V 64-bit no-MMU** architecture
  - Contributed mainly by Western Digital
- Removal of NDS32 happening soon
  - Follows removal of NDS32 from upstream Linux

Toolchain support

- Two choices in Buildroot for the toolchain/compiler:
  - **Internal toolchain**: Buildroot builds the full toolchain from source, i.e binutils, C library, kernel headers, gcc, gdb
  - **External toolchain**: Buildroot uses an existing pre-compiled cross-compilation toolchain

Toolchain support

- Two choices in Buildroot for the toolchain/compiler:
  - **Internal toolchain**: Buildroot builds the full toolchain from source, i.e binutils, C library, kernel headers, gcc, gdb
  - **External toolchain**: Buildroot uses an existing pre-compiled cross-compilation toolchain
- Improvements
  - **Internal toolchain**: mainly updates to follow the latest upstream release of all components
  - **External toolchain**: main change is the direct support for 198 pre-built toolchains from toolchains.bootlin.com



Download

Select arch: aarch64

Select libc: glibc

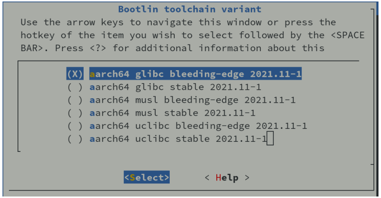
Download stable — ✓ Tests passed — checksum (sha256)

binutils	2.36.1
gcc	10.3.0
gdb	10.2
glibc	2.34-9-g9acab0b...
linux-headers	4.9.291

Download bleeding-edge — ✓ Tests passed — checksum (sha256)

binutils	2.37
gcc	11.2.0
gdb	11.1
glibc	2.34-9-g9acab0b...
linux-headers	5.4.163

View all aarch64 toolchains



```
                    Bootlin toolchain variant
 Use the arrow keys to navigate this window or press the
 hotkey of the item you wish to select followed by the <SPACE
 BAR>. Press <?> for additional information about this

            (X) aarch64 glibc bleeding-edge 2021.11-1
            ( ) aarch64 glibc stable 2021.11-1
            ( ) aarch64 musl bleeding-edge 2021.11-1
            ( ) aarch64 musl stable 2021.11-1
            ( ) aarch64 uclibc bleeding-edge 2021.11-1
            ( ) aarch64 uclibc stable 2021.11-1

                   <Select>      < Help >
```

Buildroot training course

- Bootlin has a full training course on Buildroot
- Taught by your speaker
- Training materials are freely available
    - Like for all Bootlin training courses
- Next public on-line course September 5-9, 2022

Online training course
Embedded System Development with Buildroot

bootlin
Embedded Linux and kernel engineering

https://bootlin.com/training/buildroot/

Questions? Suggestions? Comments?

Thomas Petazzoni

thomas.petazzoni@bootlin.com

https://bootlin.com/pub/conferences/2022/elc/petazzoni-buildroot-whats-new/