Securing Yocto Built Systems

ELCE 2025

Michael Opdenacker

Root Commit

Aug. 26, 2025

Michael Opdenacker

root commit

Embedded Linux expert and trainer

- https://rootcommit.com/about/michael-opdenacker/
- Former founder of Bootlin
- New founder of Root Commit
- Contributor to the Yocto Project and other Free Software projects
- Never missed a single ELC Europe since 2007!
- Free Software enthusiast and advocate (member of April.org)

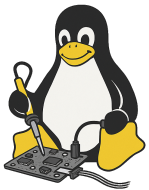
April
promouvoir et défendre le logiciel libre

What I Do For a Living — What's Funding This Work

root commit

Consulting and engineering work

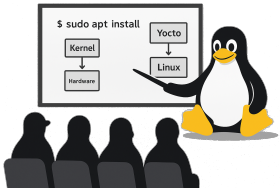
- Yocto Project
  Project reviews, system implementation, new features
- Linux Kernel
  Driver development, board bring-up, debugging
- Embedded Linux
  Boot time, bug fixing, security and other optimizations



Training — https://rootcommit.com/training

- Yocto Project and OpenEmbedded — Free Materials!
- Linux kernel, board support, driver development
  Free Materials after next course!
- Embedded Linux
- Linux Boot Time Reduction

What's special: focus on practical activities, interactivity and learning techniques.

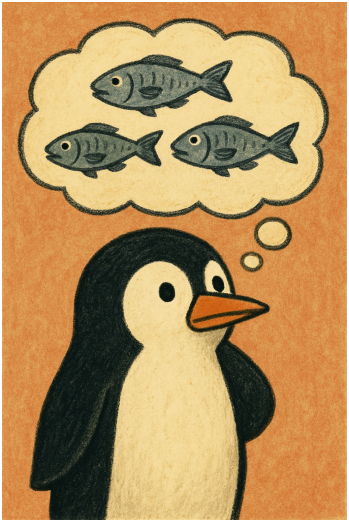


Introduction

Goals of This Presentation?

root commit

- Yocto's Manual "Making Images More Secure" is too short and highlevel: https://docs.yoctoproject.org/dev-manual/securing-images.html Find opportunities to improve it.
- One of my customer projects has a "Security Audit" task
- Want to share what I'm learning!
- Trying to focus on Yocto specific techniques



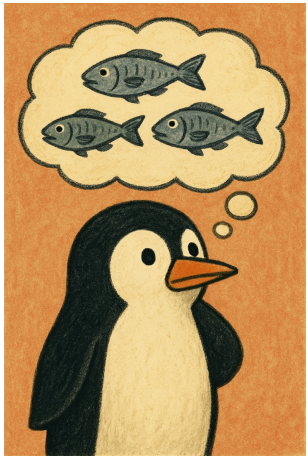
Not Covered in This Presentation

root commit

Linux security is a very wide topic.
Some aspects require dedicated presentations and can be SoC specific

- Firewalls
- Block device / filesystem encryption
- Extended file attributes
- Linux Security Modules (SELinux, AppArmor, SMACK, Tomoyo)

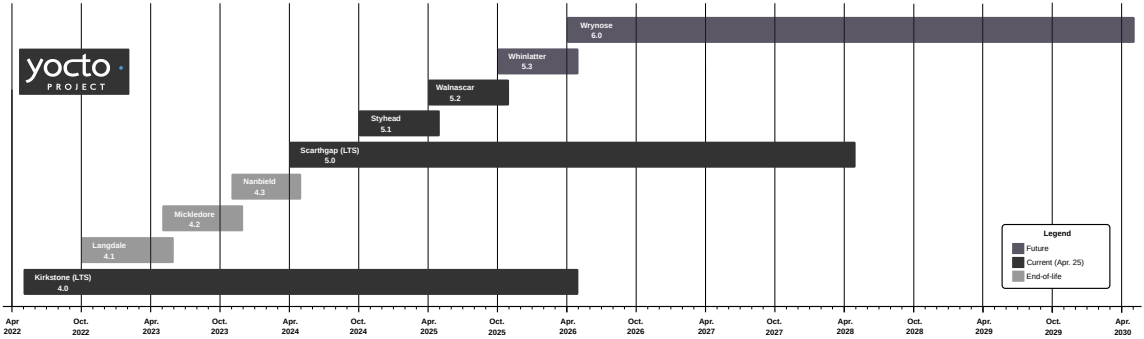
- Software updates
- Secure boot
- Verified boot
- Using TPM devices
- Integrity measurement



Getting Started

Use a Long Term Support Yocto Release

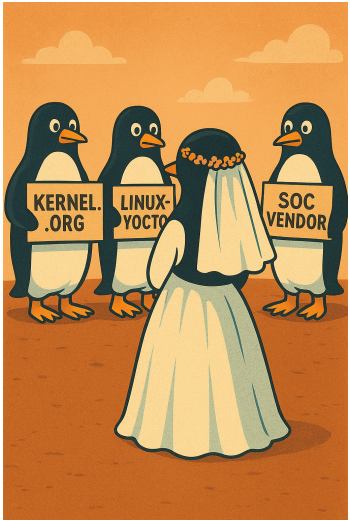
- Critical to get bug fixes and security updates from the project and community
- Also guarantees the availability of third party layers
- Exception: in development phase when the next LTS release is close, could use the latest stable release in the meantime.



Choose your kernel carefully

root commit

- An **LTS kernel** from kernel.org is the best solution to get kernel updates quickly
- **linux-yocto** is pretty well supported, you can wait a few weeks for updates.
- **Vendor kernels** support your hardware well, but they are not meant to include vulnerability fixes in a timely fashion. They can also be pretty outdated.



Create Your Own Distribution

root commit

Build your own distribution, don't start from Poky

- Poky is made for testing purposes, it has way too many unnecessary features
- Unnecessary features increase the attack surface
- No problem to start from scratch!

conf/distro/distro-prod.conf

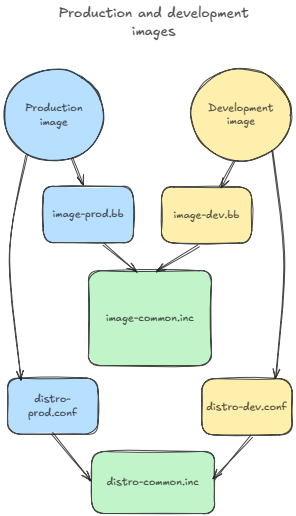
```
DISTRO = "distro-prod"
DISTRO_NAME = "Production Distro"
DISTRO_VERSION = "1.0"

INIT_MANAGER = "systemd"
```

Production and Development Images

Good practise early enough during development

- Maintain a development image alongside the production image
- Development settings isolated from the start from the production one
- Less risk of forgetting debug features in the production image

Note: diagram created with https://excalidraw.io ❤️



A look at the meta-security layer

root commit

Maintained by Scott Murray and Marta Rybczynska (thanks!)

- meta-integrity: support for Linux Integrity Measurement Architecture (IMA)
- meta-parsec: support for PARSEC, common API to hardware security and cryptographic services
- meta-tpm: TPM device support
- recipes-compliance: support for openSCAP (security audits)
- recipes-core: mostly images and package groups
- recipes-ids: support for Intrusion Detection Systems (Samhain, Suricata)
- recipes-mac: support for AppArmor and SMACK
- recipes-scanners: support for multiple scanners (but some are broken)
- recipes-security: support for multiple utilities

Note: the security DISTRO_FEATURE has a very limited scope: it just adds a few configuration fragments to the kernel (SMACK, AppArmor, dm-verity, eCryptfs, LKRG)

My conclusion: mostly useful for the features it supports. Other packages won't be compiled with tighter security features.

System Hardening

System Hardening

Reduce the Attack Surface

Disable Debugging Image Features (1)

root commit

- Walnascar (6.2) removed the `debug-tweaks` image feature.
  There was too much risk to forget it in a configuration file.
- It was corresponding to these image features:
  - `allow-empty-password`
  - `allow-root-login`
  - `empty-root-password`
  - `post-install-logging`

Disable Debugging Image Features (2)

Enable these debugging image features **only** in the development image:

- `allow-empty-password`
- `empty-root-password`
- `tools-debug`: installs debugging tools such as `strace` and `gdb`
- `perf`: installs profiling tools such as `perf`, `systemtap`, and `lttng`
- `tools-sdk`: installs a full SDK that runs on the device.

Review all `IMAGE_FEATURES` on https://docs.yoctoproject.org/ref-manual/features.html#ref-features-image

recipes-core/images/image-dev.bb

```
IMAGE_FEATURES += "allow-empty-password \
                   allow-root-login \
                   empty-root-password \
                   tools-debug \
                   tools-sdk \
                   post-install-logging"
```

Disable Distro Features

root commit

In the production image, it's also good to remove unnecessary DISTRO_FEATURES

```
$ bitbake-getvar DISTRO_FEATURES
DISTRO_FEATURES="acl alsa bluetooth debuginfod ext2 ipv4 ipv6 pcmcia usbgadget usbhost wifi
xattr nfs zeroconf pci 3g nfc x11 vfat seccomp systemd usrmerge pulseaudio gobject-
introspection-data ldconfig"
```

Remove the unnecessary ones:

conf/distro/distro-prod.conf

```
DISTRO_FEATURES:remove = "alsa bluetooth debuginfod pcmcia usbgadget usbhost \
wifi xattr nfs zeroconf pci 3g nfc x11 vfat pulseaudio
```

Result:

```
$ bitbake-getvar DISTRO_FEATURES
DISTRO_FEATURES="acl ext2 ipv4 ipv6 seccomp systemd usrmerge gobject-introspection-
data ldconfig"
```

Remove Unnecessary Packages (1)

root commit

To figure out installed packages

- `bitbake-getvar` doesn't help here, because images are defined by **packagegroups**:

```
$ bitbake-getvar -r core-image-minimal IMAGE_INSTALL
IMAGE_INSTALL="packagegroup-core-boot  u-boot"
```

- The best way is to look at the contents of the image manifest in `tmp/deploy/images/<machine>/<image>-<machine>.rootfs.manifest`

```
base-files genericarm64 3.0.14
base-passwd armv8a 3.6.4
busybox armv8a 1.36.1
busybox-hwclock armv8a 1.36.1
busybox-syslog armv8a 1.36.1
busybox-udhcpc armv8a 1.36.1
eudev armv8a 3.2.14
grub-bootconf genericarm64 1.00
grub-common armv8a 2.12
grub-editenv armv8a 2.12
grub-efi armv8a 2.12
lnlt-ifupdown genericarm64 1.0
init-system-helpers-service armv8a 1.66
initscripts armv8a 1.0
initscripts-functions armv8a 1.0
kbd armv8a 2.6.4
kernel genericarm64 6.13.11
kernel-6.13.11 genericarm64 6.13.11
kernel-image-6.13.11 genericarm64 6.13.11
kernel-image-image-6.13.11 genericarm64 6.13.11
keymaps genericarm64 1.0
kmod armv8a 33
ldconfig armv8a 2.40+git0+626c048f32
libblkid1 armv8a 2.40.2
libc6 armv8a 2.40+git0+626c048f32
libcrypto3 armv8a 3.3.1
libkmod2 armv8a 33
liblzma5 armv8a 5.6.2
libz1 armv8a 1.3.1
modutils-initscripts armv8a 1.0
netbase noarch 6.4
openssl-conf armv8a 3.3.1
openssl-ossl-module-legacy armv8a 3.3.1
packagegroup-core-boot genericarm64 1.0
sysvinit armv8a 3.04
sysvinit-inittab genericarm64 2.88dsf
sysvinit-pidof armv8a 3.04
ttyrun armv8a 2.34.0
u-boot genericarm64 2024.07
u-boot-env genericarm64 2024.07
update-alternatives-opkg armv8a 0.7.0
update-rc.d noarch 0.8+git0+b8f9501050
```

Manifest for `core-image-minimal` + `u-boot` on release Styhead

Remove Unnecessary Packages (2)

root commit

To remove the unwanted packages from the production image:

- For the same reasons, `IMAGE_INSTALL:remove` doesn't always work.
- Instead, use `PACKAGE_EXCLUDE`:

conf/distro/distro-prod.conf

```
# Packages to skip installing
PACKAGE_EXCLUDE += "busybox-udhcpc kbd-keymaps-pine"
```

System Hardening

Harden Password Security

Use Stronger Passwords 😉

https://www.linkedin.com/posts/clayr_genius-takes-many-forms-activity-7211897352754798592-Ir0T/

Device Specific Credentials

Yocto lets you create specific users, groups and passwords using the `extrausers` class.

- However, these apply to all devices
- Shared credentials, if compromized on a device, compromise the security of the whole fleet
- One solution: set credentials at first boot
- Another solution: expire credentials to force users to modify them at the next (actually first) login.
- Drawback: as long as you haven't logged in, the original password remains.

```
$ man passwd
...
   -e, --expire
        Immediately expire an account's password.
        This in effect can force a user to change
        their password at the user's next login.
...
```

```
# passwd -e root
```

```
qemux86-64 login: root
Password:
Your password has expired.  Choose a new password.
Changing password for root
Enter the new password (minimum of 5 characters)
Please use a combination of upper and lower case letters and numbers.
New password:
Re-enter new password:
passwd: password changed.
```

Setting Root Password At First Boot (1)

root commit

If you have `systemd`, you can use `systemd-firstboot` to set the root password, without having to create a shared initial password.

Compile `systemd` with `firstboot` support, typically in your `distro.conf` file:

conf/distro/secure-distro.conf

```
PACKAGECONFIG:append:pn-systemd = " firstboot"
```

Setting Root Password At First Boot (2)

root commit

Create a `firstboot.conf` file to customize the `getty` (login prompt) service

recipes-core/systemd/files/firstboot.conf

```
[Unit]
Description=First Boot Root Password Setup

[Service]
ExecStartPre=/bin/sh -c '[ -f /etc/firstboot.done ] || /usr/bin/sed -i "/^root:/d" /etc/shadow'
ExecStartPre=/bin/sh -c '[ -f /etc/firstboot.done ] || /usr/bin/systemd-firstboot --prompt-root-password'
ExecStartPre=/bin/sh -c '[ -f /etc/firstboot.done ] || /bin/touch /etc/firstboot.done'
TTYReset=yes
TTYVHangup=yes
```

The `-f /etc/firstboot.done` test is repeated, because if you set it as a precondition, the unit fails and `getty` is not even started.

Setting Root Password At First Boot (3)

root commit

Create a recipe to deploy this file

recipes-core/systemd/systemd-firstboot-getty.bb

```
DESCRIPTION = "Systemd service add-on to prompt for a root password at first boot"
LICENSE = "MIT"
LIC_FILES_CHKSUM = "file://${COMMON_LICENSE_DIR}/CC-BY-4.0;md5=9b33bbd06fb58995fb0e299cd38d1838"

SRC_URI += "file://firstboot.conf"

S = "${UNPACKDIR}"

do_install() {
    install -d ${D}${systemd_system_unitdir}/serial-getty@.service.d
    install -m 0644 ${S}/firstboot.conf ${D}${systemd_system_unitdir}/serial-getty@.service.d
}

FILES:${PN} += "${systemd_system_unitdir}/serial-getty@.service.d/firstboot.conf"
```

System Hardening

User-space Hardening

Use Compiler Hardening Features

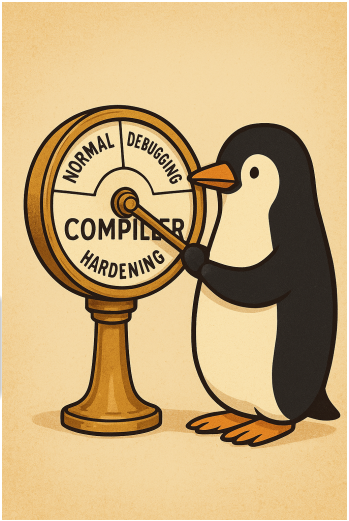
root commit

OpenEmbedded Core ships with compiler hardening features:

- meta/conf/distro/include/security_flags.inc
- meta/conf/distro/include/rust_security_flags.inc

conf/distro/distro-prod.conf

```
require conf/distro/include/security_flags.inc
require conf/distro/include/rust_security_flags.inc
```

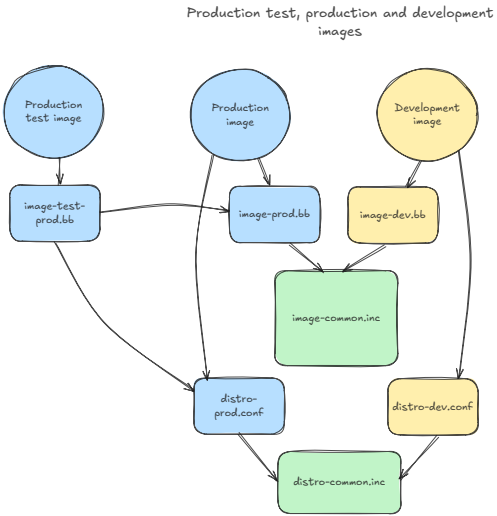


Need for Production Test Image

root commit

At this stage, need some tools to test the production image

- Don't want to put them in the development image, will get false positives with development tools and features.
- Don't want to leave them in the production image, could be used to reveal weak spots. They may also pull unwanted dependencies.
- Solution: create a third "Production test" image.



Looking for User-space Hardening Checkers

root commit

Looked at tools in `meta-security`

- `buck-security`: security scanner for Debian and Ubuntu Linux.
  Builds but currently broken, and the project has been idle for 10 years.
  Not interested in reviving a Perl based project 😉.
- `checksec`: scan your system for several simple security holes.
  Also broken, the commands given as examples don't work.
  Maintained by Debian but not actively maintained anyway.
- `nikto`: web server scanner
  Still maintained, but broken too (also in Perl)

Does anybody know a good (and modern!) security scanner we could use for userspace hardening?

Answers from the audience:

- Lynis (https://cisofy.com/lynis/)
- OpenSCAP (https://www.open-scap.org/)

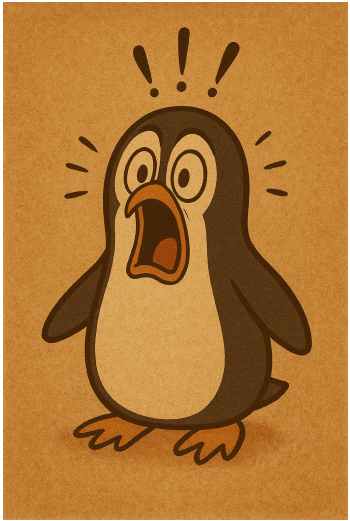
Both are supported by `meta-security`)

System Hardening

Run Public Services With Limited Resources

The `lighttpd` case

- Surprised to see that `lighttpd` in OE-core is run as `root`
- If the web server is compromised, the whole system is!
- Important to run public services with restricted privileges and resources.



Easy To Fix With Systemd!

root commit

Let's modify `lighttpd.service` provided by the `lighttpd` sources:

- Make the service run with user `lighttpd` and group `daemon`:

```
User=lighttpd
Group=daemon
```

- We can also restrict privileges to the bare minimum:

```
# If not starting lighttpd as root,
# minimal capability to bind to ports < 1024:
CapabilityBoundingSet=CAP_NET_BIND_SERVICE
```

- Another issue: `lighttpd` won't be able to write its log to `/var/log` (owned by root by default). As `/var/log` is volatile (`tmpfs`), a solution is to change its permissions before starting the service:

```
# "+" means running the command as root
ExecStartPre=+/bin/chgrp daemon /var/log
ExecStartPre=+/bin/chmod 775 /var/log
```

- Then the server runs as intended:

```
root@qemux86-64:~# ps | grep light
  249 lighttpd  9312 S    /usr/sbin/lighttpd -D -f ...
  258 root      4056 S    grep light
```

Practical Implementation

root commit

Changes to my custom layer:

```
recipes-extended/lighttpd
|-- files
|   `-- lighttpd.service
`-- lighttpd_%.bbappend

2 directories, 2 files
```

recipes-extended/lighttpd/lighttpd_%.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"

# Create lighttpd user, group daemon
inherit useradd
USERADD_PACKAGES = "${PN}"
USERADD_PARAM:${PN} = "--system --no-create-home --gid daemon \
                       --shell /usr/sbin/nologin lighttpd"

# Customize lighttpd.service
SRC_URI += "file://lighttpd.service"

do_install:append() {
    install -d ${D}${systemd_system_unitdir}
    install -m 0644 ${UNPACKDIR}/lighttpd.service ${D}${systemd_system_unitdir}
}
```

Systemd Security Benefits

root commit

- Makes it easy to run services and applications with just the necessary privileges
- Also possible to limit resources to reduce Denial of Service possibilities:
  - Access to filesystems and namespaces
  - RAM
  - CPU
  - I/O
  - Access to network
- It would be costly to implement this infrastructure by yourself 😉.

See https://www.man7.org/linux/man-pages/man5/systemd.resource-control.5.html

System Hardening

Kernel and Bootloader Hardening

Kernel Hardening

root commit

Came across `kernel-hardening-checker` from Alexander Popov

- https://github.com/a13xp0p0v/kernel-hardening-checker
- Introduced only in Walnascar (`meta-oe`)
- Reports all the kernel configuration and command line settings which could be changed to harden the running kernel.
- My contributions:
  - Updated `meta-oe` to support the latest version: 0.6.10.2 (many recent changes), supporting a `-a` option and the RISC-V architecture (link)
  - Submitted a backport to Scarthgap too (link)

recipes-core/images/image-prod-test.bb

```
IMAGE_INSTALL += "kernel-hardening-checker"
```

kernel-hardening-checker example output

```
root@beaglebone-yocto:~# kernel-hardening-checker -a
[+] Going to autodetect and check the security hardening options of the running kernel
[+] Detected version of the running kernel: (6, 12, 41)
[+] Detected kconfig file of the running kernel: /proc/config.gz
[+] Detected cmdline parameters of the running kernel: /proc/cmdline
ICMPv6: process `sysctl' is using deprecated sysctl (syscall) net.ipv6.neigh.default.base_reachable_time - use net.ipv6.neigh.default.base_reachable_
[!] WARNING: sysctl command returned 1
[+] Saved sysctls to a temporary file /tmp/sysctl-58k0_iyd
[+] Detected architecture: ARM
[+] Detected compiler: GCC 150200
[!] WARNING: can't check CONFIG_ARCH_MMAP_RND_COMPAT_BITS without CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MAX
[!] WARNING: can't check vm.mmap_rnd_compat_bits without CONFIG_ARCH_MMAP_RND_COMPAT_BITS_MAX
=========================================================================================================================
              option_name               | type |       reason       | decision |desired_val | check_result
=========================================================================================================================
CONFIG_BUG                              |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_SLUB_DEBUG                       |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_THREAD_INFO_IN_TASK              |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_IOMMU_DEFAULT_PASSTHROUGH        |kconfig| self_protection   |defconfig | is not set | OK: is not found
CONFIG_IOMMU_SUPPORT                    |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_STACKPROTECTOR                   |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_STACKPROTECTOR_STRONG            |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_STRICT_KERNEL_RWX                |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_STRICT_MODULE_RWX                |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_REFCOUNT_FULL                    |kconfig| self_protection   |defconfig |      y     | OK: version >= (5, 4, 208)
CONFIG_INIT_STACK_ALL_ZERO              |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_CPU_MITIGATIONS                  |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_VMAP_STACK                       |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_IOMMU_DEFAULT_DMA_STRICT         |kconfig| self_protection   |defconfig |      y     | FAIL: is not found
CONFIG_STACKPROTECTOR_PER_TASK          |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_HW_RANDOM_TPM                    |kconfig| self_protection   |defconfig |      y     | FAIL: is not found
CONFIG_CPU_SW_DOMAIN_PAN                |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_HARDEN_BRANCH_PREDICTOR          |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_HARDEN_BRANCH_HISTORY            |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_DEBUG_ALIGN_RODATA               |kconfig| self_protection   |defconfig |      y     | OK
CONFIG_RANDOM_KMALLOC_CACHES            |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SLAB_MERGE_DEFAULT               |kconfig| self_protection   |   kspp   | is not set | FAIL: "y"
CONFIG_BUG_ON_DATA_CORRUPTION           |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SLAB_FREELIST_HARDENED           |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SLAB_FREELIST_RANDOM             |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SHUFFLE_PAGE_ALLOCATOR           |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_FORTIFY_SOURCE                   |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_DEBUG_VIRTUAL                    |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_INIT_ON_ALLOC_DEFAULT_ON         |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_STATIC_USERMODEHELPER            |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SECURITY_LOCKDOWN_LSM            |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_LSM                              |kconfig| self_protection   |   kspp   | *lockdown* | OK: in "landlock,lockdown,yama,loadpin,safesetid,ipe,bpf"
CONFIG_SECURITY_LOCKDOWN_LSM_EARLY      |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_LOCK_DOWN_KERNEL_FORCE_CONFIDENTIALITY|kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_DEBUG_CREDENTIALS                |kconfig| self_protection   |   kspp   |      y     | OK: version >= (6, 6, 8)
CONFIG_DEBUG_NOTIFIERS                  |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_KFENCE                           |kconfig| self_protection   |   kspp   |      y     | OK
CONFIG_KFENCE_SAMPLE_INTERVAL           |kconfig| self_protection   |   kspp   |     100    | OK
CONFIG_RANDSTRUCT_FULL                  |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_HARDENED_USERCOPY                |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_HARDENED_USERCOPY_FALLBACK       |kconfig| self_protection   |   kspp   | is not set | FAIL: CONFIG_HARDENED_USERCOPY is not "y"
CONFIG_HARDENED_USERCOPY_PAGESPAN       |kconfig| self_protection   |   kspp   | is not set | FAIL: CONFIG_HARDENED_USERCOPY is not "y"
CONFIG_GCC_PLUGIN_LATENT_ENTROPY        |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_MODULE_SIG                       |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_MODULE_SIG_ALL                   |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_MODULE_SIG_SHA512                |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_MODULE_SIG_FORCE                 |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_INIT_ON_FREE_DEFAULT_ON          |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_EFI_DISABLE_PCI_DMA              |kconfig| self_protection   |   kspp   |      y     | OK: CONFIG_EFI is "is not set"
CONFIG_RESET_ATTACK_MITIGATION          |kconfig| self_protection   |   kspp   |      y     | OK: CONFIG_EFI is "is not set"
CONFIG_UBSAN_BOUNDS                     |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_UBSAN_LOCAL_BOUNDS               |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_UBSAN_TRAP                       |kconfig| self_protection   |   kspp   |      y     | FAIL: CONFIG_UBSAN_BOUNDS is not "y"
CONFIG_UBSAN_SANITIZE_ALL               |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_SCHED_CORE                       |kconfig| self_protection   |   kspp   |      y     | FAIL: is not found
CONFIG_DEBUG_SG                         |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_LIST_HARDENED                    |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_SCHED_STACK_END_CHECK            |kconfig| self_protection   |   kspp   |      y     | OK: CONFIG_VMAP_STACK is "y"
CONFIG_SYN_COOKIES                      |kconfig| self_protection   |   kspp   |      y     | OK
CONFIG_WERROR                           |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_DEFAULT_MMAP_MIN_ADDR            |kconfig| self_protection   |   kspp   |    32768   | FAIL: "4096"
CONFIG_ARM_DEBUG_WX                     |kconfig| self_protection   |   kspp   |      y     | FAIL: "is not set"
CONFIG_ARM_SMMU                         |kconfig| self_protection   |a13xp0p0v |      y     | FAIL: "is not set"
CONFIG_ARM_SMMU_DISABLE_BYPASS_BY_DEFAULT|kconfig| self_protection   |a13xp0p0v |      y     | FAIL: is not found
```

...

```
CONFIG_MMIOTRACE_TEST                   |kconfig|cut_attack_surface| lockdown | is not set | OK: is not found
CONFIG_KPROBES                          |kconfig|cut_attack_surface| lockdown | is not set | FAIL: "y"
CONFIG_BPF_SYSCALL                      |kconfig|cut_attack_surface| lockdown | is not set | FAIL: "y"
CONFIG_MMIOTRACE                        |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_LIVEPATCH                        |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_IP_DCCP                          |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_FTRACE                           |kconfig|cut_attack_surface|a13xp0p0v | is not set | FAIL: "y"
CONFIG_VIDEO_VIVID                      |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_INPUT_EVBUG                      |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_CORESIGHT                        |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_XFS_SUPPORT_V4                   |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_BLK_DEV_WRITE_MOUNTED            |kconfig|cut_attack_surface|a13xp0p0v | is not set | FAIL: "y"
CONFIG_FAULT_INJECTION                  |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_ARM_PTDUMP_DEBUGFS               |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_ARM_PTDUMP                       |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK: is not found
CONFIG_SECCOMP_CACHE_DEBUG              |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_LKDTM                            |kconfig|cut_attack_surface|a13xp0p0v | is not set | OK
CONFIG_TRIM_UNUSED_KSYMS                |kconfig|cut_attack_surface|a13xp0p0v |      y     | FAIL: "is not set"
CONFIG_VMSPLIT_3G                       |kconfig| harden_userspace |defconfig |      y     | OK
CONFIG_COREDUMP                         |kconfig| harden_userspace |  clipos  | is not set | FAIL: "y"
CONFIG_ARCH_MMAP_RND_BITS               |kconfig| harden_userspace |a13xp0p0v |     16     | FAIL: "8"
nosmep                                  |cmdline| self_protection  |defconfig | is not set | OK: is not found
nosmap                                  |cmdline| self_protection  |defconfig | is not set | OK: is not found
nokaslr                                 |cmdline| self_protection  |defconfig | is not set | OK: is not found
nopti                                   |cmdline| self_protection  |defconfig | is not set | OK: is not found
no_hash_pointers                        |cmdline| self_protection  |defconfig | is not set | OK: is not found
nospectre_v1                            |cmdline| self_protection  |defconfig | is not set | OK: is not found
nospectre_v2                            |cmdline| self_protection  |defconfig | is not set | OK: is not found
nospectre_bhb                           |cmdline| self_protection  |defconfig | is not set | OK: is not found
nospec_store_bypass_disable             |cmdline| self_protection  |defconfig | is not set | OK: is not found
dis_ucode_ldr                           |cmdline| self_protection  |defconfig | is not set | OK: is not found
arm64.nobti                             |cmdline| self_protection  |defconfig | is not set | OK: is not found
arm64.nopauth                           |cmdline| self_protection  |defconfig | is not set | OK: is not found
arm64.nomte                             |cmdline| self_protection  |defconfig | is not set | OK: is not found
iommu.passthrough                       |cmdline| self_protection  |defconfig |      0     | OK: CONFIG_IOMMU_DEFAULT_PASSTHROUGH is not found
rodata                                  |cmdline| self_protection  |defconfig |     on     | OK: rodata is not found
iommu.strict                            |cmdline| self_protection  |defconfig |      1     | FAIL: is not found
mitigations                             |cmdline| self_protection  |defconfig |    auto    | OK: mitigations is not found
slab_merge                              |cmdline| self_protection  |   kspp   | is not set | OK: is not found
slub_merge                              |cmdline| self_protection  |   kspp   | is not set | OK: is not found
page_alloc.shuffle                      |cmdline| self_protection  |   kspp   |      1     | FAIL: is not found
slab_nomerge                            |cmdline| self_protection  |   kspp   | is present | FAIL: is not present
init_on_alloc                           |cmdline| self_protection  |   kspp   |      1     | FAIL: is not found
init_on_free                            |cmdline| self_protection  |   kspp   |      1     | FAIL: is not found
hardened_usercopy                       |cmdline| self_protection  |   kspp   |      1     | FAIL: is not found
slab_common.usercopy_fallback           |cmdline| self_protection  |   kspp   | is not set | OK: is not found
kfence.sample_interval                  |cmdline| self_protection  |   kspp   |     100    | OK: CONFIG_KFENCE_SAMPLE_INTERVAL is "100"
nosmt                                   |cmdline|cut_attack_surface|   kspp   | is present | FAIL: is not present
debugfs                                 |cmdline|cut_attack_surface|  grsec   |     off    | FAIL: is not found
sysrq_always_enabled                    |cmdline|cut_attack_surface|grapheneos| is not set | OK: is not found
bdev_allow_write_mounted                |cmdline|cut_attack_surface|a13xp0p0v |      0     | FAIL: is not found
norandmaps                              |cmdline| harden_userspace |defconfig | is not set | OK: is not found
net.core.bpf_jit_harden                 |sysctl | self_protection  |   kspp   |      2     | FAIL: "0"
kernel.oops_limit                       |sysctl | self_protection  |a13xp0p0v |     100    | FAIL: "10000"
kernel.warn_limit                       |sysctl | self_protection  |a13xp0p0v |     100    | FAIL: "0"
vm.mmap_min_addr                        |sysctl | self_protection  |   kspp   |    32768   | FAIL: "4096"
kernel.dmesg_restrict                   |sysctl |cut_attack_surface|   kspp   |      1     | FAIL: "0"
kernel.perf_event_paranoid              |sysctl |cut_attack_surface|   kspp   |      3     | FAIL: "2"
dev.tty.ldisc_autoload                  |sysctl |cut_attack_surface|   kspp   |      0     | FAIL: "1"
kernel.kptr_restrict                    |sysctl |cut_attack_surface|   kspp   |      2     | FAIL: "0"
dev.tty.legacy_tiocsti                  |sysctl |cut_attack_surface|   kspp   |      0     | FAIL: "1"
user.max_user_namespaces                |sysctl |cut_attack_surface|   kspp   |      0     | FAIL: "3740"
kernel.kexec_load_disabled              |sysctl |cut_attack_surface|   kspp   |      1     | OK: CONFIG_KEXEC_CORE is not found
kernel.unprivileged_bpf_disabled        |sysctl |cut_attack_surface|   kspp   |      1     | FAIL: "2"
vm.unprivileged_userfaultfd             |sysctl |cut_attack_surface|   kspp   |      0     | OK: CONFIG_USERFAULTFD is "is not set"
kernel.modules_disabled                 |sysctl |cut_attack_surface|   kspp   |      1     | FAIL: "0"
kernel.io_uring_disabled                |sysctl |cut_attack_surface|  grsec   |      2     | FAIL: "0"
kernel.sysrq                            |sysctl |cut_attack_surface|a13xp0p0v |      0     | FAIL: "16"
fs.protected_symlinks                   |sysctl | harden_userspace |   kspp   |      1     | OK
fs.protected_hardlinks                  |sysctl | harden_userspace |   kspp   |      1     | OK
fs.protected_fifos                      |sysctl | harden_userspace |   kspp   |      2     | FAIL: "1"
fs.protected_regular                    |sysctl | harden_userspace |   kspp   |      2     | FAIL: "1"
fs.suid_dumpable                        |sysctl | harden_userspace |   kspp   |      0     | OK
kernel.randomize_va_space               |sysctl | harden_userspace |   kspp   |      2     | FAIL: "1"
kernel.yama.ptrace_scope                |sysctl | harden_userspace |   kspp   |      3     | FAIL: is not found
vm.mmap_rnd_bits                        |sysctl | harden_userspace |a13xp0p0v |     16     | FAIL: "8"

[+] Config check is finished: 'OK' - 145 / 'FAIL' - 109
root@beaglebone-yocto:~#
```

Kernel Hardening Notes

- Kernel configuration: remove unnecessary features
  - This reduces the attack surface, and boot time too!
  - Yocto stores kernel modules in separate packages. This helps to identify unnecessary ones!
- See `security/self-protection` (kernel documentation) for details about kernel protection techniques.

Bootloader Hardening

root commit

A wide topic too!

- Make sure your bootloader is well maintained. Use mainline U-Boot if possible.
- Implement secure boot or verified boot if possible.
- No documentation available in in U-Boot to increase security.
- Lock down your serial console. Pretty easy to attack or brick a device if you have U-Boot shell access.
  - Completely disable the interactive shell: `CONFIG_BOOTDELAY=-2`
  - Or require a specific string to unlock it: `CONFIG_AUTOBOOT_ENCRYPTION=y`
- Disable all unnecessary features (network, filesystems, usb-serial...) and devices (USB, SD card...)

System Hardening

Other Hardening Techniques

Read-Only Filesystem

root commit

Very easy to implement with Yocto, whatever the init manager

- Add this image feature:

recipes-core/images/image-prod.bb

```
IMAGE_FEATURES += "read-only-rootfs"
```

- Just make sure that the mount point(s) for your read-write partition(s) are created at root filesystem creation time. The mounts points are created on demand by `systemd` when you have a read-write root filesystem.

recipes-core/images/image-common.bb

```
# Create mount point for data partition
do_rootfs[postfuncs] += "image_create_data_dir"

image_create_data_dir() {
    install -d ${IMAGE_ROOTFS}/data
}
```

Don't write, this is read-only!

overlayfs.bbclass and overlayfs-etc.bbclass

root commit

- Had to use the `overlayfs-etc` to allow to modify `/etc/shadow` and `/etc/passwd` for setting the root password at first boot.
- Except for this special need, `systemd` started the system without a single issue when the whole root filesystem is read-only 👏.
  This works as important "dynamic" files (such as `/etc/resolv.conf` are links to volatile mount points.
- Had no need for the `overlayfs` class (for the whole root filesystem), but this may be needed to allow applications to make local changes to the root filesystem, while the base system remains read-only.

recipes-core/images/image-prod.bb

```
IMAGE_FEATURES += "overlayfs-etc"
OVERLAYFS_ETC_MOUNT_POINT = "/data"
OVERLAYFS_ETC_DEVICE = "/dev/mmcblk0p3"
OVERLAYFS_ETC_FSTYPE ?= "ext4"
```

recipes-kernel/linux/linux-yocto_%.bbappend

```
FILESEXTRAPATHS:prepend := "${THISDIR}/files:"
SRC_URI += "file://overlayfs.cfg"
```

recipes-kernel/linux/files/overlayfs.cfg

```
CONFIG_OVERLAY_FS=y
```

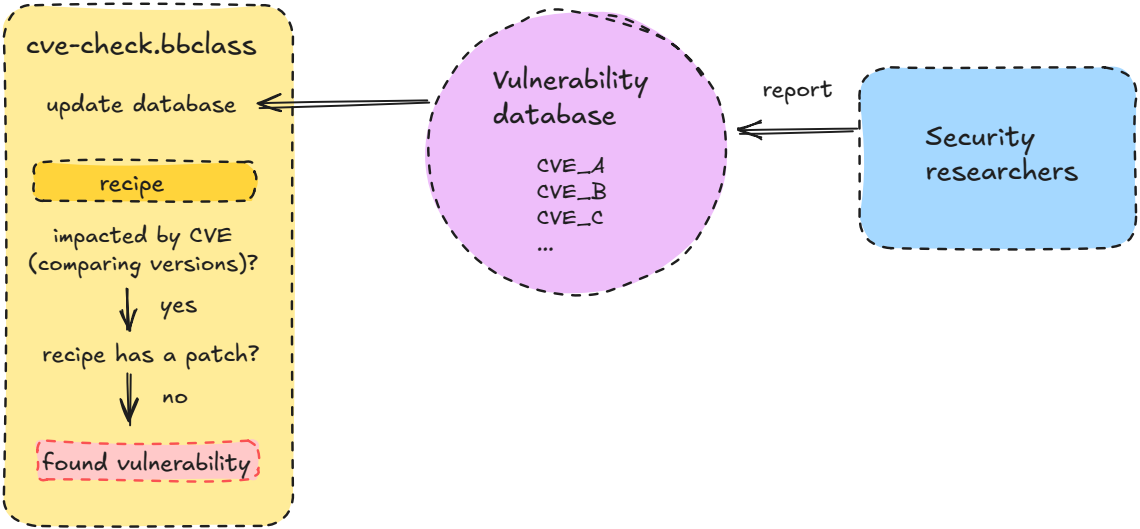
Maintaining your system

Maintaining your system

Addressing Vulnerabilities

How vulnerability checking works

root commit



Enable vulnerability checking

root commit

- Add this to `conf/local.conf`:
  ```
  INHERIT += "cve-check"
  ```
- This will add a CVE task to the recipes you're building
- You may also want to ignore CVEs that are irrelevant to Poky and OE-core:
  ```
  include conf/distro/include/cve-extra-exclusions.inc
  ```
- To speed up NVD database dowloads, request a unique key (`NVDCVE_API_KEY`)
  ```
  NVDCVE_API_KEY = "xxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxx"
  ```
- Then, `bitbake` your regular image,
  and the checks will be run (without running the other tasks if not necessary)
- You can also run checks on specific recipes:
  ```
  $ bitbake -c cve_check linux-yocto
  ```

Exploiting vulnerability report

root commit

- Simple command:

```
$ grep Unpatched build/tmp/log/cve/cve-summary.json | wc -l
39
```

- VulnScout (https://github.com/savoirfairelinux/vulnscout) is a new tool to process the vulnerability report and the project SPDX.
  - At last, a tool to consume Yocto SPDX output and generate a vulnerability assessment for it!
  - Supports SPDX 2.3, SPDX 3.0, CycloneDX, Yocto JSON vulnerability output
  - All you need to deliver to the customer is Yocto's initial vulnerability report and the SPDX description of the system.
  - The customer then doesn't need Yocto to scout for new vulnerabilities and make her/his own assessment at any time.
  - Contributed to it through bug reports and small documentation patches. Even had a private demo from the developers.

VulnScout — How to Use

- VulnScout can directly be used on Yocto's SPDX and `cve-check` outputs, but you have to create a YAML configuration file.
- However, it's easier to use with the `meta-vulnscout` layer
  - https://github.com/savoirfairelinux/meta-vulnscout
  - Supports: Yocto 5.0 (Scarthgap), 5.2 (Walnascar) and 5.3 (Whinlatter)
  - Just inherit the `vulnscout` class in an image:

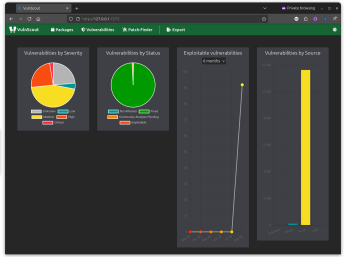
recipes-core/images/image-prod.bb

```
inherit vulnscout
```

  - Building the image generates the YAML file for Vulnscout and a command line to start the tool:

```
$ docker rm -f vulnscout
$ docker-compose -f ".vulnscout/image-prod-test-beaglebone-
yocto/docker-compose.yml" up
```

- You can keep the `.vulnscout` directory for future use outside of Yocto.



VulnScout's web interface

Addressing vulnerabilities in your products

root commit

If a fix is found (typically upstream), add the patch to your recipe

- Include the CVE identifier in the patch file name (recommended)
- Add a `CVE:<id>` line to the patch
- Also set an `Upstream-Status:` field. https://docs.yoctoproject.org/contributor-guide/recipe-style-guide.html#patch-upstream-status
- Share your patch with the Yocto community!

Of course, another option is to upgrade to a newer version of upstream (if available).

meta/recipes-bsp/grub/files/CVE-2025-0622-01.patch

```
From 2123c5bca7e21fbeb0263df4597ddd7054700726 Mon Sep 17 00:00:00 2001
From: B Horn <b@horn.uk>
Date: Fri, 1 Nov 2024 19:24:29 +0000
Subject: [PATCH 1/3] commands/pgp: Unregister the "check_signatures" hooks on
 module unload

If the hooks are not removed they can be called after the module has
been unloaded leading to an use-after-free.

Fixes: CVE-2025-0622

Reported-by: B Horn <b@horn.uk>
Signed-off-by: B Horn <b@horn.uk>
Reviewed-by: Daniel Kiper <daniel.kiper@oracle.com>

CVE: CVE-2025-0622
Upstream-Status: Backport [https://git.savannah.gnu.org/cgit/grub.git/commit/?id=2123c5bca7e21fbeb...]
Signed-off-by: Peter Marko <peter.marko@siemens.com>
---
 grub-core/commands/pgp.c | 2 ++
 1 file changed, 2 insertions(+)

diff --git a/grub-core/commands/pgp.c b/grub-core/commands/pgp.c
index c6766f044..5fadc33c4 100644
--- a/grub-core/commands/pgp.c
+++ b/grub-core/commands/pgp.c
@@ -1010,6 +1010,8 @@ GRUB_MOD_INIT(pgp)

 GRUB_MOD_FINI(pgp)
 {
+  grub_register_variable_hook ("check_signatures", NULL, NULL);
+  grub_env_unset ("check_signatures");
   grub_verifier_unregister (&grub_pubkey_verifier);
   grub_unregister_extcmd (cmd);
   grub_unregister_extcmd (cmd_trust);
```

Ignoring vulnerabilities in your products

root commit

You can also modify the recipe to mark some vulnerabilities as irrelevant:

meta/recipes-devtools/rust/rust-source.inc

```
CVE_STATUS[CVE-2024-24576] = "not-applicable-platform: Issue only applies on Windows"
```

You can also group vulnerabilities that can be ignored in the same way:

meta/recipes-extended/logrotate/logrotate_3.22.0.bb

```
CVE_STATUS_GROUPS = "CVE_STATUS_RECIPE"
CVE_STATUS_RECIPE = "CVE-2011-1548 CVE-2011-1549 CVE-2011-1550"
CVE_STATUS_RECIPE[status] = "not-applicable-platform: CVE is debian, gentoo or SUSE specific on the way logrotate was installed/used"
```

See "Checking for Vulnerabilities" in the Yocto Manual:
https://docs.yoctoproject.org/dev-manual/vulnerabilities.html

Closing

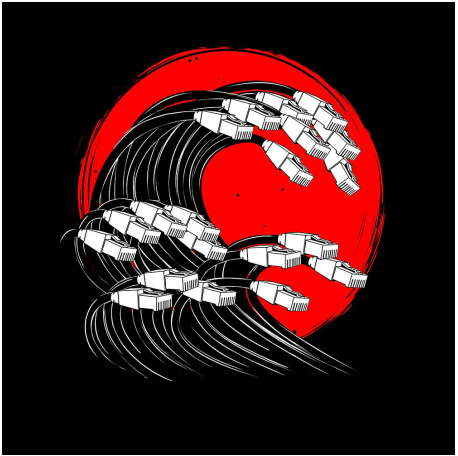
Key Takeaways

- Use LTS versions of Yocto and kernel
- Need 3 images: `dev`, `prod` and `prod-test`
- Remove unnessary packages and features
- No shared passwords between devices
- Read-only root filesystems are easy
- Use `systemd` to restrict priviledges and resources of applications and services.

- `kernel-hardening-checker`: great tool for kernel hardening
- Looking for tools for checking userspace hardening
- More hardening needed: secure / verified boot, TPM, firewall, encryption, extended file attributes, Linux Security Modules...
- VulnScout: new tool to run vulnerability assessments from Yocto SPDX output
- You are going to be busy 😉

Useful Resources

- My `meta-security-demo` layer containing the code shown here (supports QEMU x86-64 and BeagleBone):
  https://gitlab.com/rootcommit/meta-security-demo
- Esa Jääskelä — Yocto Hardening
  Series of blog posts about many aspects of the topic
  Fun and exhaustive!
  https://ejaaskel.dev/yocto-hardening/
- Esa Jääskelä — Sulka Secure Yocto Distribution
  A good example!
  https://codeberg.org/altidSec/meta-sulka-distro
- Darknet Diaries
  My favorite English speaking podcast about IT (in)security
  https://darknetdiaries.com/



https://shop.darknetdiaries.com

Thank you

root commit

Questions? Comments?

- mo@rootcommit.com
- https://fosstodon.org/@MichaelOpdenacker
- XMPP: omichael@conversations.im
- Signal: rootcommit.01

- Slides available under the CC-By-SA 4.0 license
  https://rootcommit.com/pub/conferences/2025/elce/yocto-security/
- Sources (LaTeX):
  https://gitlab.com/rootcommit/yocto-security/