


# Yocto Project and the Cyber Resilience Act

Paul Barker, The Yocto Project

(C) 2026, CC-BY-SA 4.0

Source

# Who am I?

- I work full time on Yocto Project
- Act as a leverage function, help to coordinate our activities and help the project to scale effectively
- Still get my hands dirty with a bit of implementation and debugging



# Contents


## Introduction to the CRA

- What is it, and why should you care?

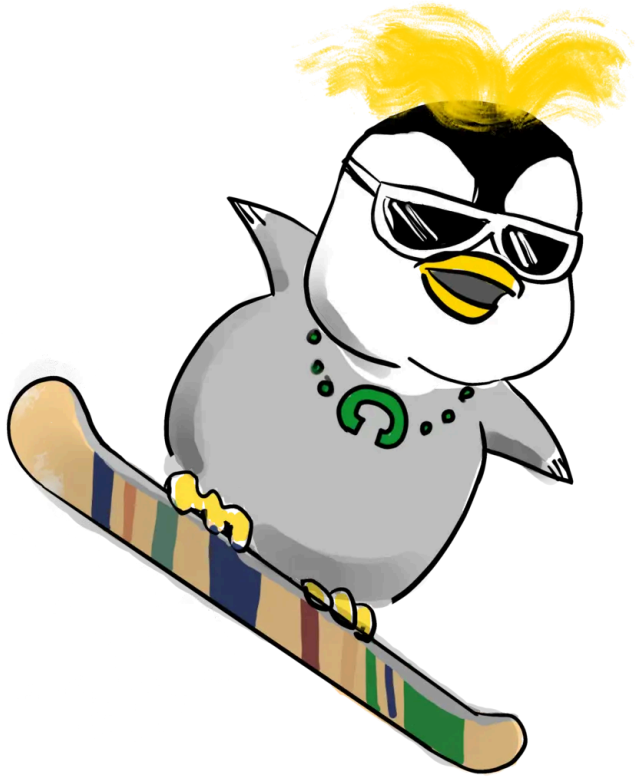
## The Yocto Project: Today

- Existing features we provide which help with CRA compliance

## The Yocto Project: Tomorrow

- How we can meet our new obligations and help you meet yours

# Introdution to the CRA



# Caveat Emptor

- This is a high-level introduction only.
- I am not a lawyer!
- I am just one person within Yocto Project, views may differ
- Consult your own legal advice for more details.

# What is the CRA

The Cyber Resilience Act (CRA) is a piece of EU legislation, regulating the cybersecurity, availability of vulnerability fixes, support period, etc for "Products with Digital Elements" (PDEs).

A PDE is essentially any product with a network connection.

Industries with existing regulations are excluded (Automotive, Medical, etc).

# Categories under the CRA

- The CRA considers two types of economic actors:
  - Manufacturers
  - Open Source Stewards
- Everyone else is out-of-scope.
- Designation of Manufacturer or Steward is per-product.

# Manufacturers

- Organisations placing products on the EU market
  - Includes many "free" products and services
- Full regulations apply (highlights, not a comprehensive list):
  - Avoid known exploitable vulnerabilities
  - Track software components and vulnerabilities
  - Reporting to relevant Computer Security Incident Response Teams (CSIRTs)
  - Provision of security updates for the useful lifetime of the product

# Manufacturers, continued

Other requirements (highlights, not a comprehensive list):

- Cybersecurity risk assessment
    - This is the starting point and must inform what actions manufacturers take to mitigate security risks
- Creation of technical documentation and SBOMs
    - Publication is not required, these may be kept internal but must be provided to relevant authorities on request

# Manufacturers, continued

For specific sensitive product classes, additional requirements and standards apply, and third-party product certification may be required:

- Operating systems
- Routers & Firewalls
- Password Managers
- Browsers
- etc

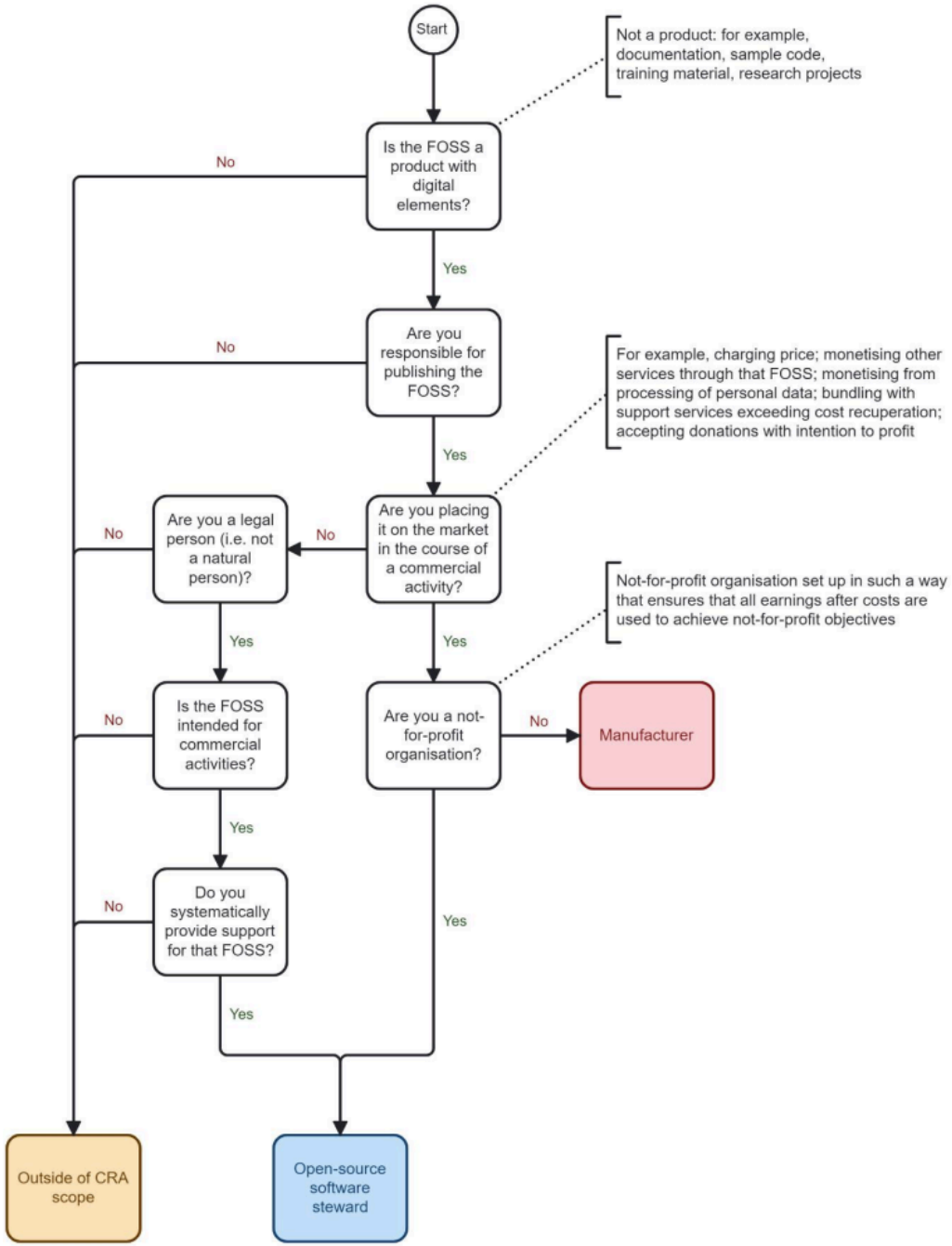
# Open Source stewards

- Organisations maintaining Free & Open Source Software (FOSS) intended for commercial use.
- Light-touch regulation:
  - Must have a compliant cybersecurity policy.
  - Must co-operate with "market surveillance authorities" (MSAs, aka regulators & CSIRTs).
  - No fines for non-compliance.

# Open Source stewards: criteria

You are an Open Source Steward if you answer yes to *all* these questions in the context of a particular FOSS project:

- Is the organisation a legal entity (a company or non-profit, not an individual)?
- Is the organisation responsible for publishing the FOSS?
- Is the organisation providing systematic support for the FOSS?
    - E.g. employing maintainers, hosting development services, etc
- Is the FOSS intended for commercial use?


Start
Not a product: for example, documentation, sample code, training material, research projects
Is the FOSS a product with digital elements?
No
Yes
Are you responsible for publishing the FOSS?
No
For example, charging price; monetising other services through that FOSS; monetising from processing of personal data; bundling with support services exceeding cost recuperation; accepting donations with intention to profit
Yes
Are you placing it on the market in the course of a commercial activity?
No
Are you a legal person (i.e. not a natural person)?
No
Yes
Not-for-profit organisation set up in such a way that ensures that all earnings after costs are used to achieve not-for-profit objectives
Yes
Are you a not-for-profit organisation?
No
Manufacturer
Is the FOSS intended for commercial activities?
No
Yes
Do you systematically provide support for that FOSS?
No
Yes
Yes
Outside of CRA scope
Open-source software steward

# Out-of-scope

It's worth emphasising what is *not* in scope of the CRA:

- Open source contributors (both individuals and companies)
- Non-commercial open source projects

# A shout-out: OpenSSF

The Open Source Security Foundation (OpenSSF) is doing excellent work in this area.

https://openssf.org

# The Yocto Project:

# Today



# Are we in scope?

We are clearly in scope of the CRA:

- The Yocto Project is a legal entity under LF Projects LLC
- We publish open source software
- We contract maintainers, provide repository hosting and run an autobuilder cluster
- Our software is clearly intended for commercial use
    - Presence of commercial project members makes this obvious

# Are we a manufacturer or a steward?

We are a **steward**:

- We do not charge for access to our software
- We do not collect user data in exchange for using our software
- Commercial members have influence but do not receive dedicated support or consulting in exchange for their membership
- We are a non-profit organisation

# How we see the CRA

To the Yocto Project, the CRA is a challenge...

- We will need to adapt
- Our users will have higher expectations

... and an opportunity!

- We already provide best in class tooling
- We have a strong, well-informed community
- Our build process makes it easy to add any required metadata and options

# Tooling: Software integrity

We use several approaches to ensure that you are shipping the intended software:

- Checksum and git commit hash verification during source download
- Binary reproducibility
- Generation of signed sstate and/or package feeds
- `bitbake-setup` can also verify git commit hashes of layers

# Tooling: Over-the-air (OTA) update

In most cases the ability to ship updates is now required.

We support various methods via third-party layers:

- swupdate
- RAUC
- Mender and other commercial solutions

# Tooling: SBOMs & CVE tracking

- We generate Software Bill-of-Materials (SBOM) documents in SPDX 3 format by default
- We support Package URLs (PURLs)
- Recently integrated `sbom-cve-check` tool
    - Analyses SBOM, so can be ran after the build process to discover new CVEs affecting an image

# Tooling: Source mirroring and archival

That doesn't sound like it's relevant to the CRA... but it is!

- Upstream sources do sometimes disappear off the internet
- Having your own copy of the source ensures you can investigate and patch issues

# Autonomy is a feature

With an Embdded Linux build system like Yocto Project or buildroot, you can take ownership and ensure that your obligations are met

- No single software supplier to act as a bottleneck
- No need to ask for permission
- No enforced services or data collection (which might have security vulnerabilities)

# You're part of a community

We have a CRA Working Group which meets bi-weekly via a Zoom call. Please join us!

- Discuss your concerns
- Learn from other engineers in our community
- Share what you've learned

# Limitations

What the Yocto Project can do:

- Act as a responsible Open Source steward
- Provide tooling and best practices to help manufacturers

What the Yocto Project **cannot** do:

- Provide specific instructions for your particular products
- Meet your obligations for you

# The Yocto Project:

# Tomorrow



# Cybersecurity policy

We need to update our policies to be CRA compliant:

- Must encourage secure product development and effective vulnerability handling
- Must encourage voluntary reporting of vulnerabilities
- Must address how we document and remediate vulnerabilities
- Must promote the sharing of information
- Should document how we plan to handle reporting & cooperation with MSAs

# Reporting requirements

We have a few obligations around reporting, which we will also document:

- Must notify MSAs about actively exploited vulnerabilities we are aware of
  - Only *actively exploited* issues, not all CVEs
- Must notify MSAs about severe cybersecurity incidents that may impact sercurity of downstream products
  - E.g. unauthorised git push to our repositories
- Must inform impacted users, preferably in a structured machine-readable format
  - In practice this will be via posting to our mailing lists

# Other bits and pieces

We also need to:

- Define the project scope clearly for CRA purposes
- Collect evidence that we follow our security policies
- Move all security documentation into our version controlled docs
    - The wiki does not provide adequate change review processes

# Threat model and security audit

We are pursuing funding for work in the following areas and hope to have further updates soon:

- Development of a threat model template/framework for Yocto Project users
- Audit of integrity features provided by bitbake fetchers
- Audit of network accessible services we provide
    - Layer index, error reporting service, etc
    - Hash equivalence server (hashserv) & PR server (prserv)
    - Toaster

# Secure from first boot

Products should not have static/reused login credentials.

Multiple options are available to address this, we should integrate one or more:

- Provision of device-specific login credentials which can be printed on a label or card
- Prompt for creation of username and password as part of initial setup
- Integration of cloud-init, ignition, etc

# Secure boot

Secure boot implementation is very hardware specific. To make it truly effective requires secure hardware/software co-design from the early stages of product development.

However, we should be able to provide an emulated reference implementation (using QMEU).

BSP vendors should be encouraged to provide hardware-specific reference implementations.

Ideally we will have at least one reference platform with fully integrated secure boot support in the future.

# Secure boot: Common features in OE-core

We should be able to migrate common secure boot components into our openembedded-core layer, with automated testing via QEMU:

- UEFI secure boot with dynamically generated signing keys
- OP-TEE, now used by RISC-V as well as ARM
- TPM2 support

# SBOMs & CVE tracking improvments

There are various ways we could improve CVE tracking support:

- CycloneDX (input/output) and VEX (output) format support
- Additional data sources such as EU Vulnerability Database (EUVD), various OSV data sources, OpenSSF Malicious Software list, etc
    - We cannot rely on the US National Vulnerability Database (NVD) as a single source
- Work with data sources to move most `CVE_STATUS` assignments upstream and our of our metadata
    - Where upstream data has missing or inaccurate CPE

# SLSA attestation

> Supply-chain Levels for Software Artifacts, or SLSA (“salsa”), is a set of incrementally adoptable guidelines for supply chain security, established by industry consensus
> -- https://slsa.dev/spec/v1.2/about

SLSA Attestations capture details about the sources, dependencies and build process used to create an image.

- We can meet SLSA levels 1 and probably 2 on the autobuilder
- We can help users create SLSA Attestations for their own builds

# Retirement of obsolete recipes

Across our layers we have many recipes for unmaintained or obsolete software.

This is more of an issue in meta-openembedded than openembedded-core.

We want to encurage the use of well maintained software by our choices of what recipes to provide in our core layers. Recipes for unmaintained/obsolete software should be retired (either completely or to a separate legacy layer).

# Should we become an OSV?

**Maybe**

- We could register as a package vendor and provide our own OSV data
- We would have our own "yocto" Package URLs (PURLs)
- We would have to passthrough upstream CVEs by reporting in our OSV data
  - Pro: Ability to easily exclude upstream CVEs which are not applicable in the Yocto Project context, no more `CVE_STATUS` data in recipes at all
  - Con: Additonal ongoing effort
- This would require a significant long-term resource investment

# Should we become a CVE Numbering Authority (CNA)?

**No**

- We could issue our own CVEs directly, but it is not worth the cost
- This would only apply for software we directly publish (bitbake, metadata layers, layerindex, wic, etc), not for upstream projects for which we have recipes
    - So the scope is narrow
- The overwhelming majority of CVEs we deal with are in upstream software, not in our direct components

# Contributions needed!

The security and compliance features in Yocto Project will be determined by what our contributors choose to work on.

We have limited resources within the core team to go and implement things ourselves.

We are pursuing funding opportunities but even if successful these will not everything that we would like to achieve.

So, please help us to continue improving the Yocto Project!

# Resources & appendices

# Resources

Official EU publications:

- Full CRA text
- Draft guidance (look at the 70-page PDF)

OpenSSF resources:

- CRA Brief Guide for OSS Developers
- CRA Stewards Playbook

# Resources, continued

- LF Free Training Course: Understanding the EU Cyber Resilience Act (CRA) (LFEL1001)
- CRA Standards development
- Eclipse Open Regulatory Compliance Working Group (ORC WG)

# Appendix: CRA Article 24(1)

> Open-source software stewards shall put in place and document in a verifiable manner a cybersecurity policy to foster the development of a secure product with digital elements as well as an effective handling of vulnerabilities by the developers of that product. That policy shall also foster the voluntary reporting of vulnerabilities as laid down in Article 15 by the developers of that product and take into account the specific nature of the open-source software steward and the legal and organisational arrangements to which it is subject. That policy shall, in particular, include aspects related to documenting, addressing and remediating vulnerabilities and promote the sharing of information concerning discovered vulnerabilities within the open-source community.

# Appendix: CRA Article 24(2)

> Open-source software stewards shall cooperate with the market surveillance authorities, at their request, with a view to mitigating the cybersecurity risks posed by a product with digital elements qualifying as free and open-source software.
>
> Further to a reasoned request from a market surveillance authority, open-source software stewards shall provide that authority, in a language which can be easily understood by that authority, with the document tion referred to in paragraph 1, in paper or electronic form.

# Appendix: CRA Article 24(3)

> The obligations laid down in Article 14(1) shall apply to open-source software stewards to the extent that they are involved in the development of the products with digital elements. The obligations laid down in Article 14(3) and (8) shall apply to open-source software stewards to the extent that severe incidents having an impact on the security of products with digital elements affect network and information systems provided by the open-source software stewards for the development of such products.

# Appendix: CRA Article 14(1) & 14(3)

> A manufacturer shall notify any actively exploited vulnerability contained in the product with digital elements that it becomes aware of simultaneously to the CSIRT designated as coordinator, in accordance with paragraph 7 of this Article, and to ENISA. The manufacturer shall notify that actively exploited vulnerability via the single reporting platform established pursuant to Article 16.

> A manufacturer shall notify any severe incident having an impact on the security of the product with digital elements that it becomes aware of simultaneously to the CSIRT designated as coordinator, in accordance with paragraph 7 of this Article, and to ENISA. The manufacturer shall notify that incident via the single reporting platform established pursuant to Article 16.

# Appendix: CRA Article 14(8)

> After becoming aware of an actively exploited vulnerability or a severe incident having an impact on the security of the product with digital elements, the manufacturer shall inform the impacted users of the product with digital elements, and where appropriate all users, of that vulnerability or incident and, where necessary, of any risk mitigation and corrective measures that the users can deploy to mitigate the impact of that vulnerability or incident, where appropriate in a structured, machine-readable format that is easily automatically processable.

# Thank you!

