

# SWHSec

## Leveraging Software Heritage to Enhance Cybersecurity

R. Lefeuvre, C. Reux, **Stefano Zacchioli**, O. Barais, B. Combemale

Polytechnic Institute of Paris

`stefano.zacchioli@telecom-paris.fr`

29 Jan 2025

Software Heritage Symposium

UNESCO, Paris



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

- 1 Open source supply chain security
- 2 SWHSec
- 3 Chasing one-day vulnerabilities across open source forks
- 4 Conclusion

# Open source security

**Open source** software can be freely used, copied, and modified.

Open Source Software (OSS) is everywhere

- Huge boost for **innovation!** (e.g., reduced time to market)
- **96%** of (non-open) software products **depend on open source** (2022).
- Open source is at the heart of the **global digital infrastructure**.

# Open source security

Open source software can be freely used, copied, and modified.

Open Source Software (OSS) is everywhere

- Huge boost for **innovation!** (e.g., reduced time to market)
- **96%** of (non-open) software products **depend on open source** (2022).
- Open source is at the heart of the **global digital infrastructure**.

With great exposure comes great scrutiny...

- ...by both good and bad actors.
- OSS is more and more **targeted by attackers**.
- Increased **policy attention** to secure OSS, e.g.:
  - US: Biden's executive orders (2022, Jan 2025!)
  - EU: CRA, progressively coming into effect

## TOP 10 EMERGING CYBERSECURITY THREATS FOR 2030



2030

THREATS

1

Supply chain compromise of software dependencies

More integrated components and services from the supply chain and partners could lead to more all-encompassing vulnerabilities with consequences on the supply and customer side.



2

Advanced disinformation campaigns

Disinformation can enable more effective operations for geopolitical reasons and for military gains.



3

Rise of digital surveillance authoritarianism/loss of privacy

Facial recognition, digital surveillance and internet glassness or digital identities, data stores may become a target for various groups.



4

Human error and exploited legacy systems within cyber-physical ecosystems

The fast adoption of IoT, the need to retrofit legacy systems and the rigging with shortcuts could lead to a lack of knowledge, training and understanding of the cyber-physical ecosystem, which can lead to security issues.



5

Targeted attacks enhanced by smart device data

Third-party data used from internet-connected smart devices, cellular data can expose information for tailored and more sophisticated attacks.



6

Lack of analysis and control of space-based infrastructure and objects

Due to the interaction between private and public infrastructure to space, the security of these new soft architectures and technologies need to be investigated as a lack of understanding, oversight and control of space-based infrastructure can make it vulnerable to attacks and sabotage.



7

Rise of advanced hybrid threats

Physical or cyber attacks are enabling new breeding zones connected with cyberattacks due to the increase of critical devices, cloud usage, online selection, and social profiles.



8

Skill shortage

Lack of capabilities and competencies could see cybercriminal groups target organizations with the largest technology and IT talent shortage.



9

Cross border ICT service providers as a single point of failure

ICT service connecting critical services such as transport, energy, grids and military that provides services across borders are likely to be targeted by the growing skills of hackers, physical intrusions, and threats of service and management being, where potential conflict.



10

Artificial Intelligence Abuse

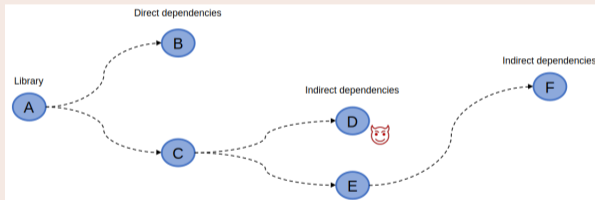
Manipulation of AI algorithms and training data can be used to enhance malicious activities such as the creation



# Software supply chain attacks

## Reusing OSS via dependencies

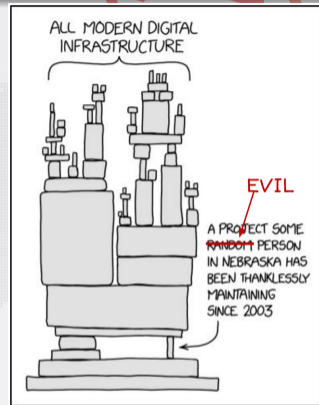
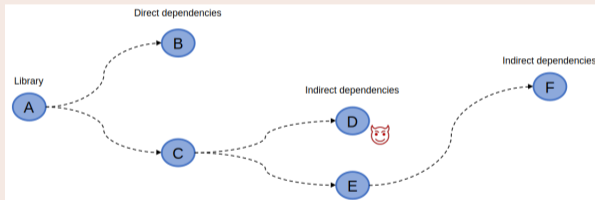
- **Software dependencies:** a popular way of reusing open source software.
- Software product *A* uses functionalities implemented in OSS product *B* ... and so on.



# Software supply chain attacks

## Reusing OSS via dependencies

- **Software dependencies**: a popular way of reusing open source software.
- Software product *A* uses functionalities implemented in OSS product *B* ... and so on.



based on [xkcd.com/2347](https://xkcd.com/2347)

## Attacking the software supply chain

- Attacking **undermaintained "leaf" packages** (e.g., D) → efficient attack strategy
- Many documented attacks: event-stream (2018), node-ipc (2022), XZ utils (2024), ...

- 
- 1 Open source supply chain security
  - 2 SWHSec
  - 3 Chasing one-day vulnerabilities across open source forks
  - 4 Conclusion

## What does Software Heritage bring to the table?

- The largest archive that guarantees the:
  - 1 availability,
  - 2 integrity, and
  - 3 traceability of (OSS) source code.



## What does Software Heritage bring to the table?

- The largest archive that guarantees the:
  - 1 availability,
  - 2 integrity, and
  - 3 traceability of (OSS) source code.
- A **universal, open knowledge base** of *facts* about open source software...
- ...that can be leveraged by everyone (not only the big players) to secure OSS.

# Securing open source with Software Heritage

## What does Software Heritage bring to the table?

- The largest archive that guarantees the:
  - 1 availability,
  - 2 integrity, and
  - 3 traceability of (OSS) source code.
- A **universal, open knowledge base** of *facts* about open source software...
- ...that can be leveraged by everyone (not only the big players) to secure OSS.

## SWHSec project

[swhsec.github.io](https://swhsec.github.io)

- 2023–2027 R&D project, funded by French national CampusCyber
- 8 research teams, including SWH core



Software Heritage  
THE GREAT LIBRARY OF SOURCE CODE

Axes: (1) extending SWH with security info + (2) code analysis, dependency analysis, **vulnerability tracking**, automatic vulnerability fixing, ... at SWH scale.

- 
- 1 Open source supply chain security
  - 2 SWHSec
  - 3 Chasing one-day vulnerabilities across open source forks**
  - 4 Conclusion

# One-day vulnerabilities in open source

## One-day vulnerabilities

- Def.: vulnerabilities that are **publicly known, but not fixed yet** in software you use.
- Challenge: **identify them quickly and exhaustively**, then apply countermeasures.
- Many tools available to detect one-day vulnerabilities *via declared dependencies*.

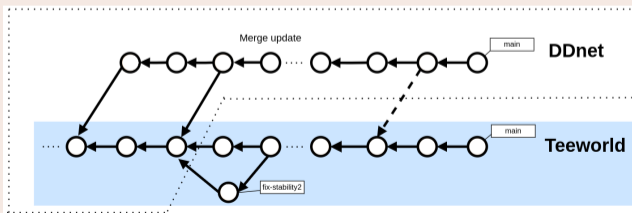
# One-day vulnerabilities in open source

## One-day vulnerabilities

- Def.: vulnerabilities that are **publicly known, but not fixed yet** in software you use.
- Challenge: **identify them quickly and exhaustively**, then apply countermeasures.
- Many tools available to detect one-day vulnerabilities *via declared dependencies*.

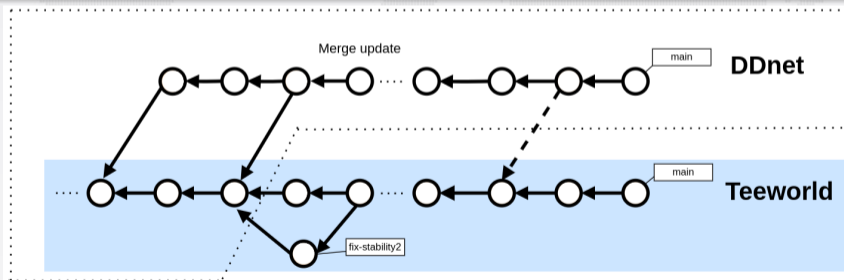
## Reusing OSS via forks

Open source is also reused via **forking**: (1) start from existing OSS (e.g., Teeworlds game), (2) create your own (e.g., DDnet), (3) periodically integrate changes.



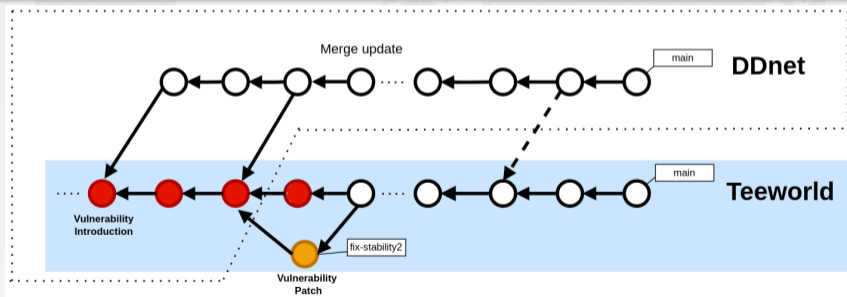
# Vulnerability propagation through forks

- Any change to a piece of software (*commit*) can **introduce a new vulnerability**.
- Or it can **fix an existing vulnerability**.
- What happens if a project is forked **between introduction and fix** of a vulnerability?
- It inherits the vulnerability, ... until the change with the fix is integrated.



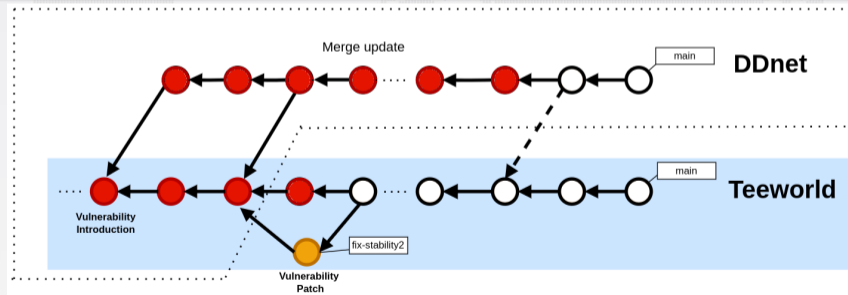
# Vulnerability propagation through forks

- Any change to a piece of software (*commit*) can **introduce a new vulnerability**.
- Or it can **fix an existing vulnerability**.
- What happens if a project is forked **between introduction and fix** of a vulnerability?
- It inherits the vulnerability, ... until the change with the fix is integrated.



# Vulnerability propagation through forks

- Any change to a piece of software (*commit*) can **introduce a new vulnerability**.
- Or it can **fix an existing vulnerability**.
- What happens if a project is forked **between introduction and fix** of a vulnerability?
- It inherits the vulnerability, ... until the change with the fix is integrated.





## Approach

- 1 Start from a **public DB of vuln. introduced/fixed** in public commits (e.g., [OSV.dev](https://osv.dev)).
- 2 "**Color**" the entire graph of public code development history **with vulnerability info.**
  - Software Heritage is the only place where this can be done at the scale of all forks, across all public code, independently of specific development platforms.
- 3 **Inform maintainers** of vulnerable forks. (After validation.)

## Approach

- 1 Start from a **public DB of vuln. introduced/fixed** in public commits (e.g., [OSV.dev](#)).
- 2 "**Color**" the **entire graph** of public code development history **with vulnerability info.**
  - Software Heritage is the only place where this can be done at the scale of all forks, across all public code, independently of specific development platforms.
- 3 **Inform maintainers** of vulnerable forks. (After validation.)

## Early results

- Identified 2.2 M (million) forks of repositories referenced by OSV.dev, containing vulnerable commits; **1.3 M forks vulnerable** in their most recent commit.
- 86.6 M vulnerable commits were specific to forks, **not findable with current tools.**
- Among 66 manually vetted cases, **5 confirmed vulnerabilities (1 critical).**



Romain Lefeuvre, Charly Reux, Stefano Zacchiroli, Olivier Barais, Benoit Combemale  
Chasing One-day Vulnerabilities Across Open Source Forks  
To appear, 2025.

- 
- 1 Open source supply chain security
  - 2 SWHSec
  - 3 Chasing one-day vulnerabilities across open source forks
  - 4 Conclusion



## Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

### Takeaways

- **Open source** software is everywhere and **increasingly targeted by attackers**.
- State-of-the-art tooling for identifying known vulnerability is limited in scope (specific platforms, specific ways of reusing code).
- We can leverage Software Heritage to **discover unfixed vulnerabilities** and improve open source **security for everyone**. The SWHSec project is working on this.
- Next steps: integration with the Software Heritage archive, public API.

### Contact

[Stefano Zacchioli](mailto:stefano.zacchioli@telecom-paris.fr) / [stefano.zacchioli@telecom-paris.fr](mailto:stefano.zacchioli@telecom-paris.fr) / [@zacchiro@mastodon.xyz](https://mstdn.social/@zacchiro)