

Chasing One-day Vulnerabilities Across Open Source Forks

Stefano Zacchioli

Polytechnic Institute of Paris

`stefano.zacchioli@telecom-paris.fr`

21 Oct 2025

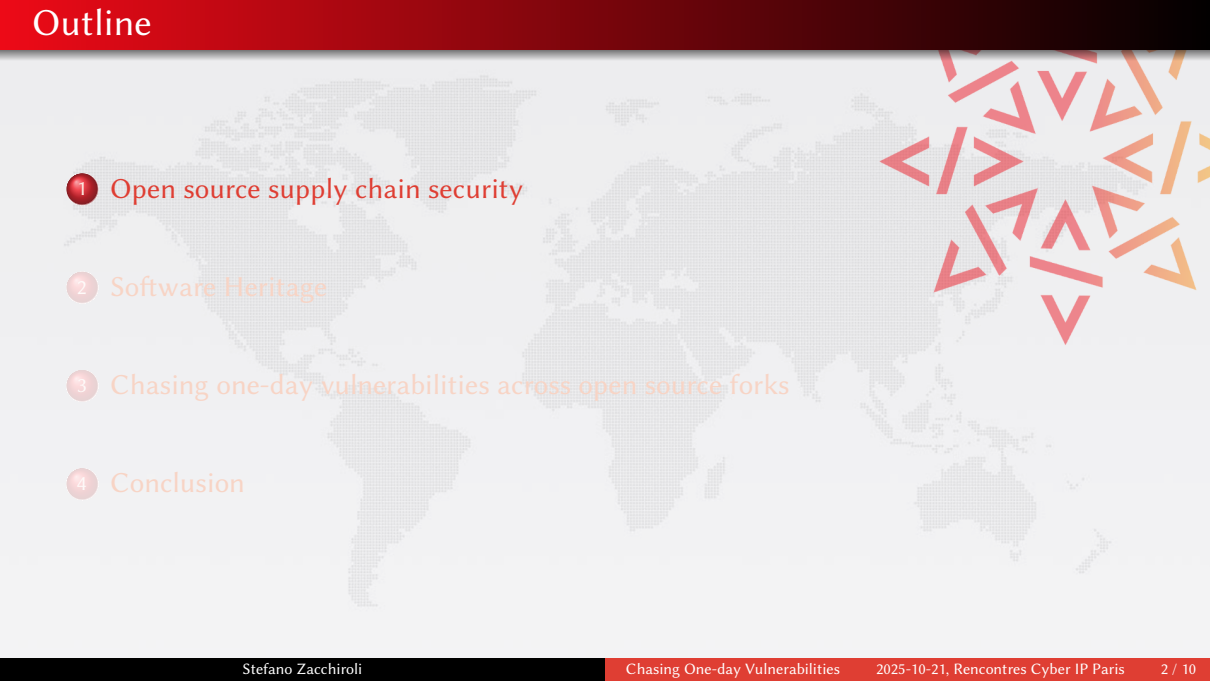
Rencontres Cyber IP Paris

École polytechnique, Palaiseau, France



Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Outline

- 
- 1 Open source supply chain security
 - 2 Software Heritage
 - 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: executive orders (Biden 2022; Trump Jan 2025)
 - EU: CRA, progressively coming into effect

TOP 10 EMERGING CYBERSECURITY THREATS FOR 2030



THREATS



2030

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 manipulate conversations for ideological reasons and for monetary gain.



3

Rise of digital surveillance authoritarianism/ loss of privacy

Facial recognition, digital surveillance, pervasive geolocation or digital identities data stores may become a target for criminal 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 growing skill shortage 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

There are also enhanced threat vectors, connected smart devices, vehicles can access infrastructure for follow-up 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 in space, the security of these new objects and technologies could 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 evolving and becoming more complex with cyberattacks due to the increase of cyber threats, cyber espionage, cyber sabotage and social engineering.



8

Skill shortage

Lack of capabilities and competencies could see cybercriminal groups target organizations with the largest technology and IT-based resource.



9

Cross border ICT service providers as a single point of failure

ICT sector connecting critical services such as transport, energy grids and industry that provides services across borders are likely to be targeted by hackers, such as backdoors, physical manipulations, and threats of service and equipment being, or being potential conflict.



10

Artificial Intelligence Abuse

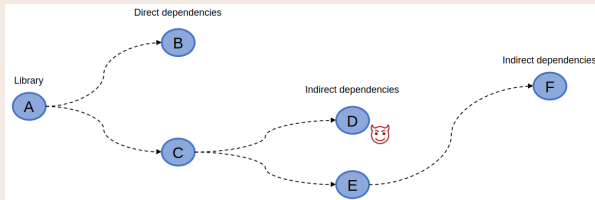
Manipulation of AI algorithms and training data can be used to enhance cyberattacks or activities such as the creation



Software supply chain attacks

Reusing OSS via dependencies

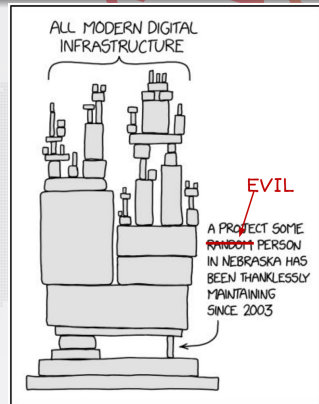
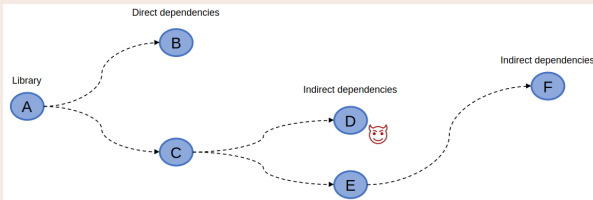
- **Software dependencies:** a popular way of reusing open source software.
- Software product *A* uses functionalities implemented in OSS product *B* ...and so on.



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.

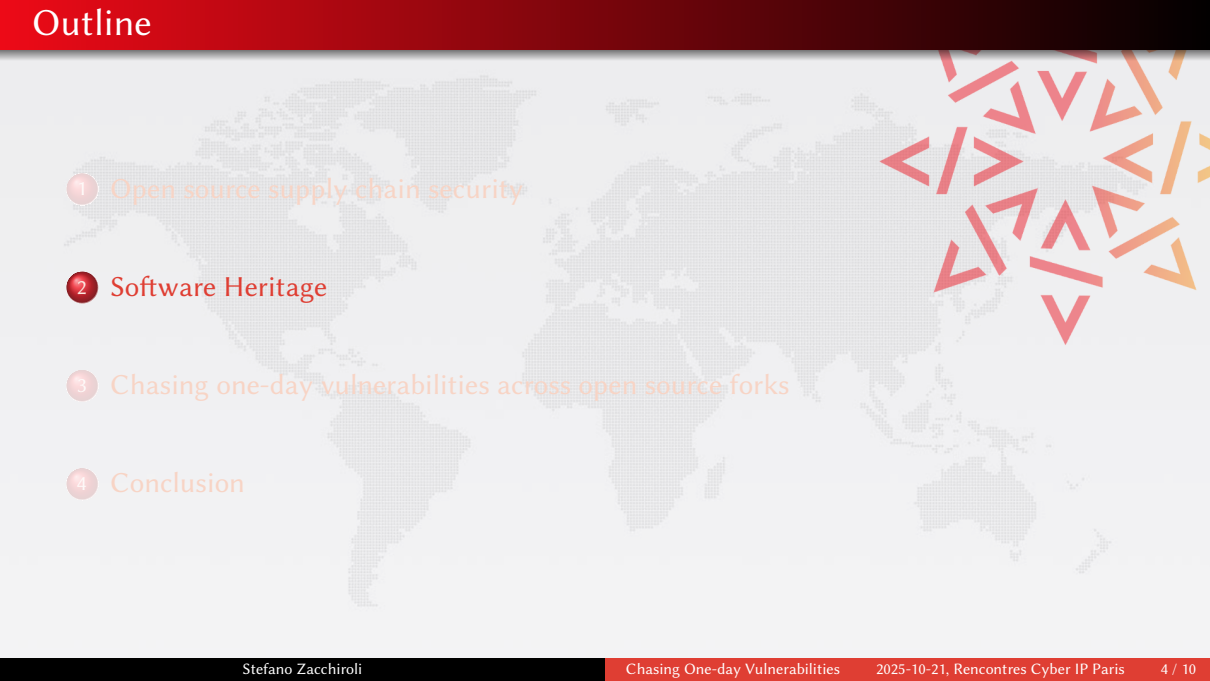


based on xkcd.com/2347

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), ...

Outline

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



Software Heritage
THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog



find and reference all
software source code



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog



find and **reference** all
software source code

Universal archive



preserve and **share** all
software source code



Software Heritage

THE GREAT LIBRARY OF SOURCE CODE

Collect, preserve and share *all* software source code

Preserving our heritage, enabling better software and better science for all

Reference catalog



find and **reference** all
software source code

Universal archive



preserve and **share** all
software source code

Research infrastructure



enable analysis of all
software source code

One infrastructure
open and shared

Cultural Heritage



Industry



Research



Public Administration



Software Heritage



One infrastructure
open and shared

Cultural Heritage



Industry



Research

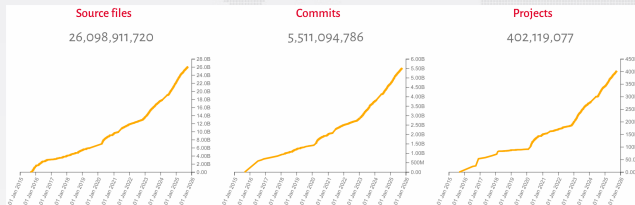


Public Administration



Software Heritage

The largest archive ever built



One infrastructure
open and shared

Cultural Heritage



Industry



Research

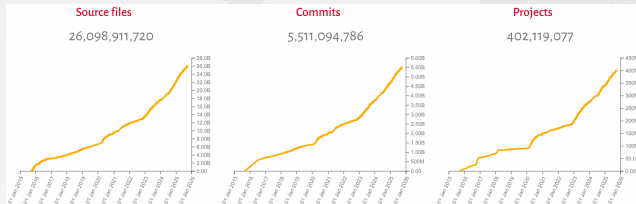


Public Administration



Software Heritage

The largest archive ever built



Bitbucket 2,818,626 origins	git 33,472 origins
R 29,046 origins	debian 142,984 origins
GitHub 266,235,919 origins	gitleaks 5,803,610 origins
git 3,824 origins	Gogs 422 origins
Guix 56,248 origins	GNU 354 origins
launchpad 654,759 origins	Maven 425,606 origins
rpm 4,070,945 origins	Phabricator 198 origins
Fedora PAGURE 72,459 origins	pub.dev 63,023 origins
purl 621,919 origins	SOURCEFORGE 382,368 origins
	stagit 343 origins

Securing open source with Software Heritage

What does Software Heritage bring to the table?

- The largest archive that guarantees the:
 - 1 **availability**
 - 2 **integrity** → see SWHID (SoftWare Hash IDentifiers), ISO 18670
 - 3 **traceability** of (OSS) source code

swhid.org

Securing open source with Software Heritage

What does Software Heritage bring to the table?

- The largest archive that guarantees the:
 - 1 availability
 - 2 integrity → see SWHID (SoftWare Hash IDentifiers), ISO 18670 swhid.org
 - 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** → see SWHID (SoftWare Hash IDentifiers), ISO 18670 swhid.org
 - 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

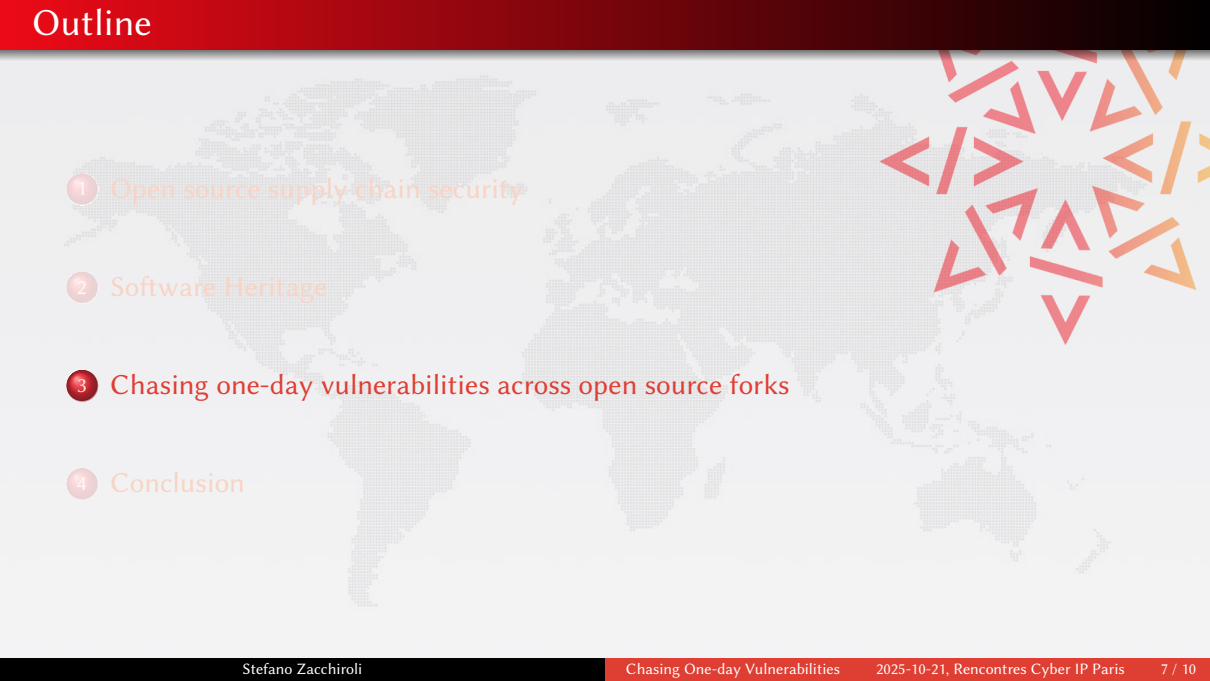
- 2023–2027 R&D project, funded by French national CampusCyber
- 8 research teams; co-led by Télécom Paris



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.

Outline

- 
- 1 Open source supply chain security
 - 2 Software Heritage
 - 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*.

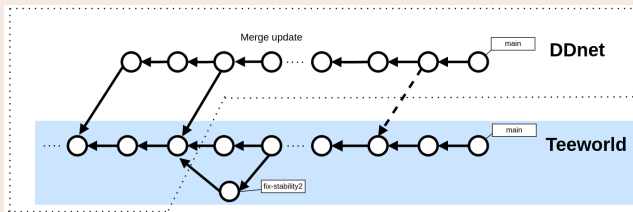
One-day vulnerabilities in open source

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*.

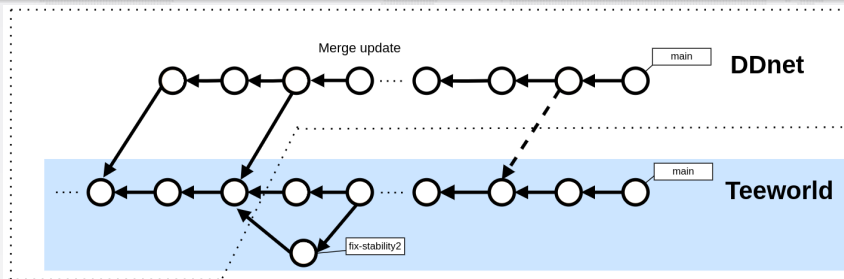
Reusing OSS via forks

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.



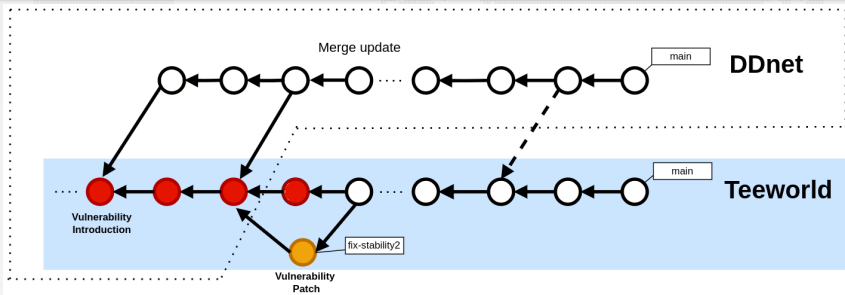
Vulnerability propagation through forks

- 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.



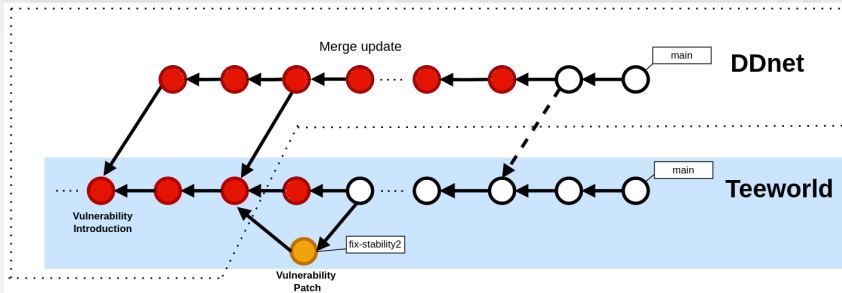
Vulnerability propagation through forks

- 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.



sw-h-vuln: chasing one-day vulnerabilities across forks... at SWH scale

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.)

swh-vuln: chasing one-day vulnerabilities across forks... at SWH scale

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.)

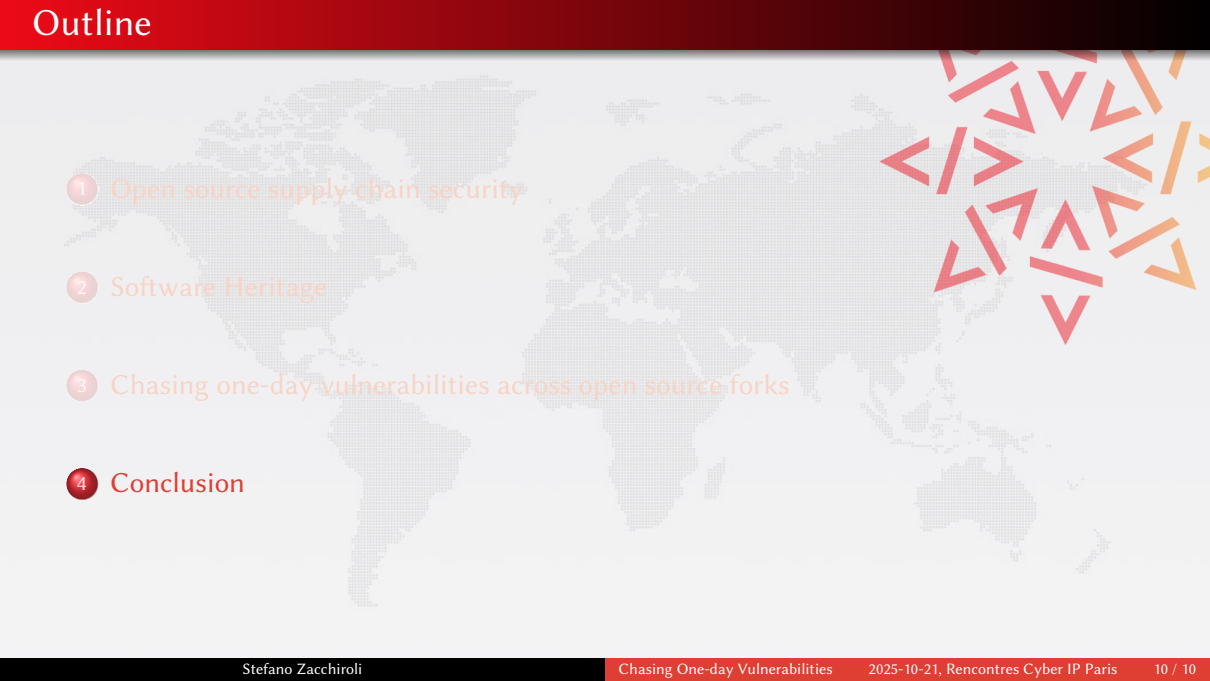
Early results

- 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.

Outline

- 
- 1 Open source supply chain security
 - 2 Software Heritage
 - 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 / stefano.zacchioli@telecom-paris.fr / [@zacchiro@mastodon.xyz](https://mastodon.xyz/@zacchiro)