Improving the Design and Evaluation of Cryptographic Implementations against Leakage

Can Open Source Help and How?

François-Xavier Standaert
UCLouvain, ICTEAM Institute, Crypto Group

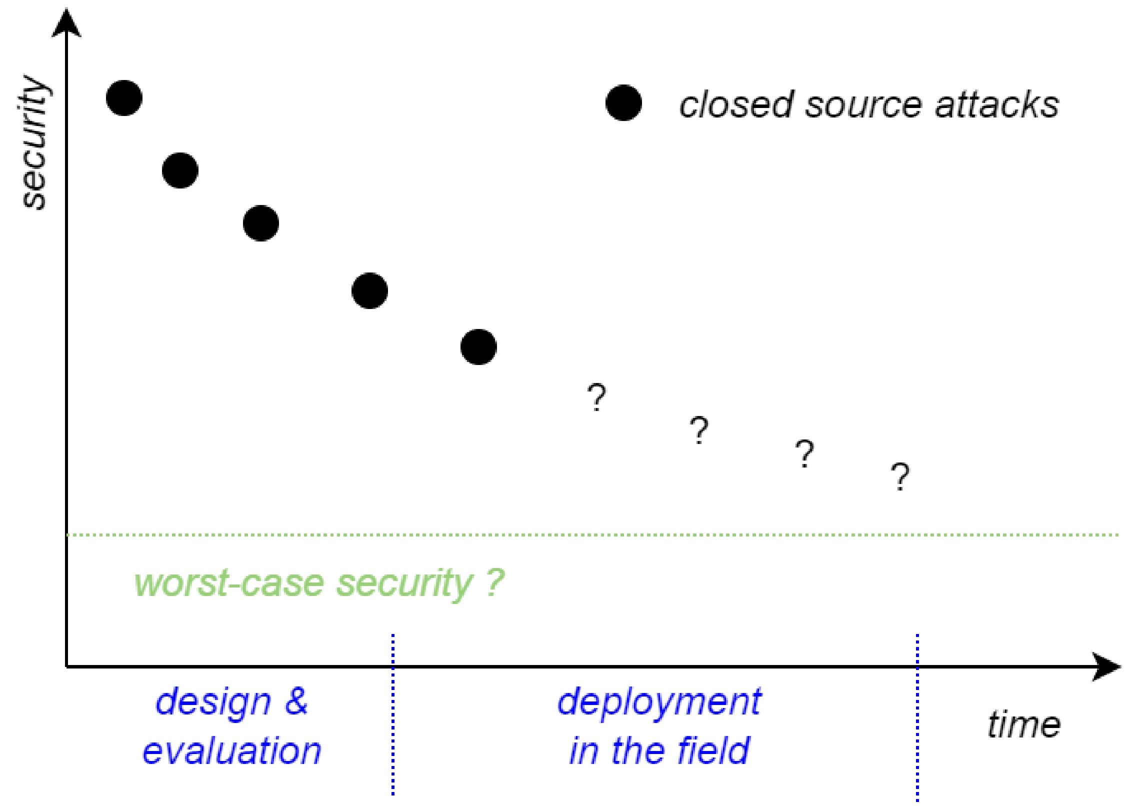
NIST Crypto-Reading Club, June 29 2022

Outline

- Vision & goals (rapidly)
- Exemplary outcomes
- Proposed organization
    - Yearly schedule
    - Projects’ lifetime
    - Licensing policy
- Timeline
- Questions

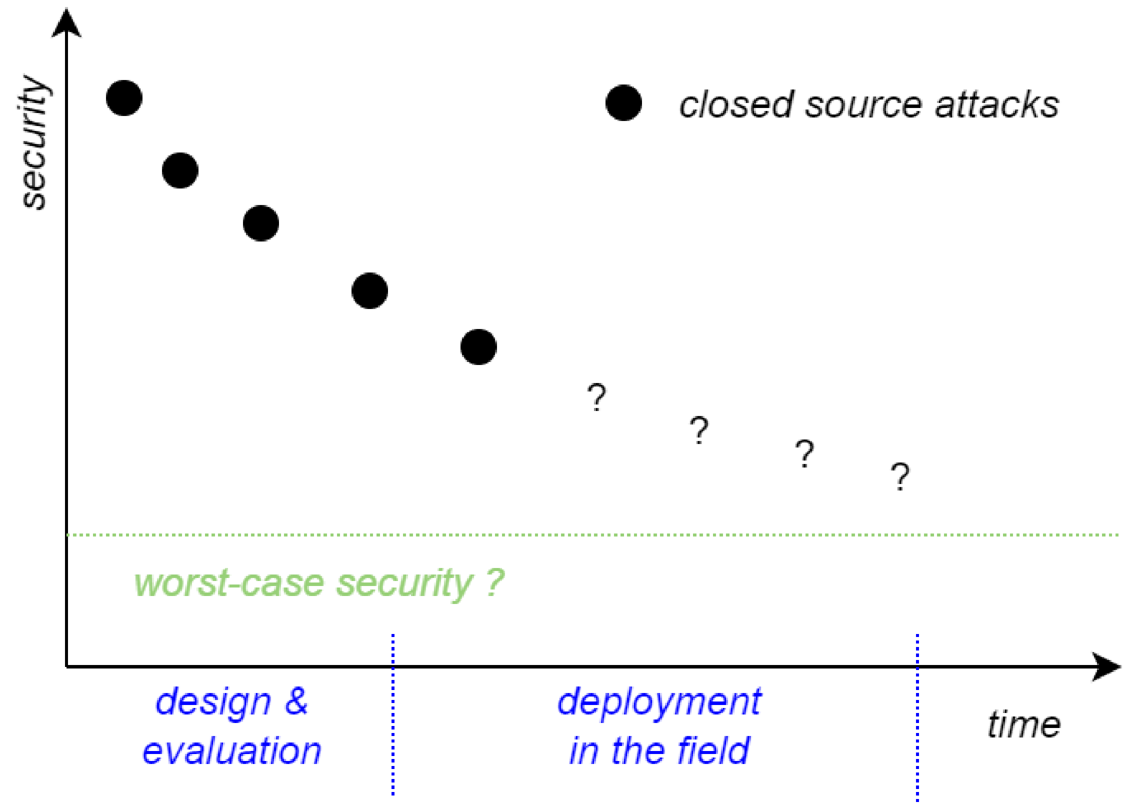
Vision I

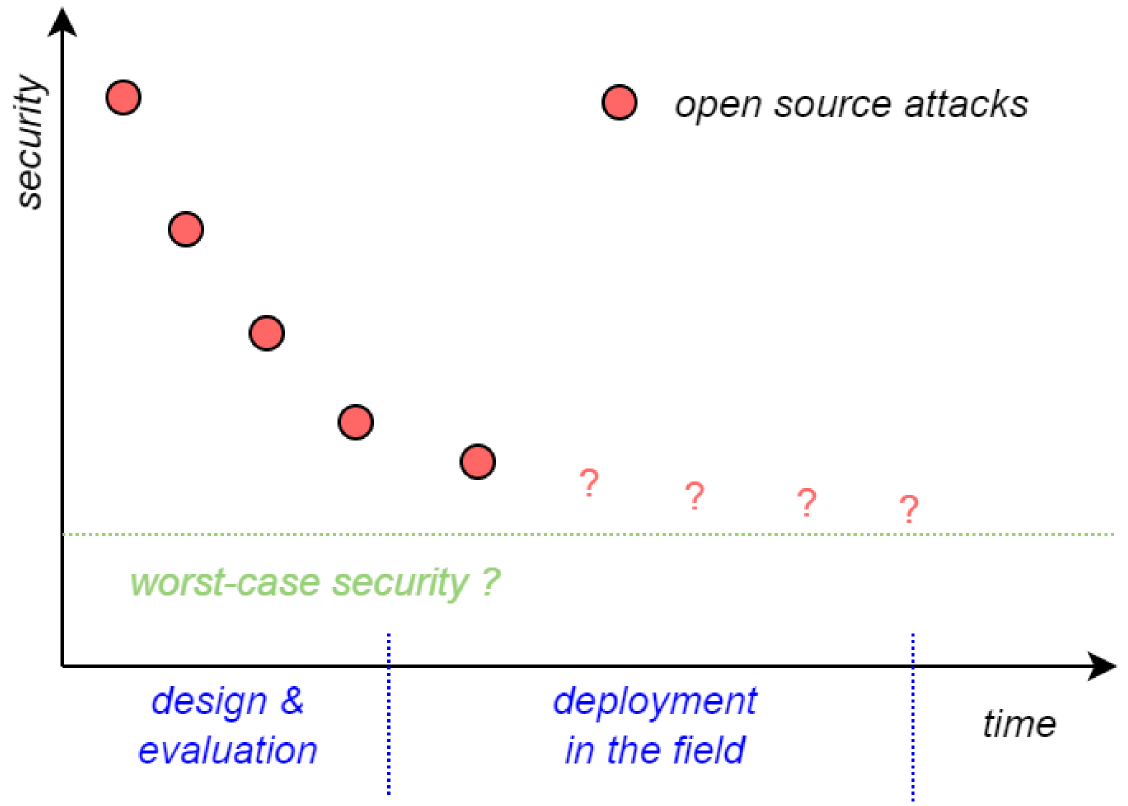
- Long-term (≈ worst case) security is hard to predict & anticipate



Vision I

- Long-term (≈ worst case) security is hard to predict & anticipate

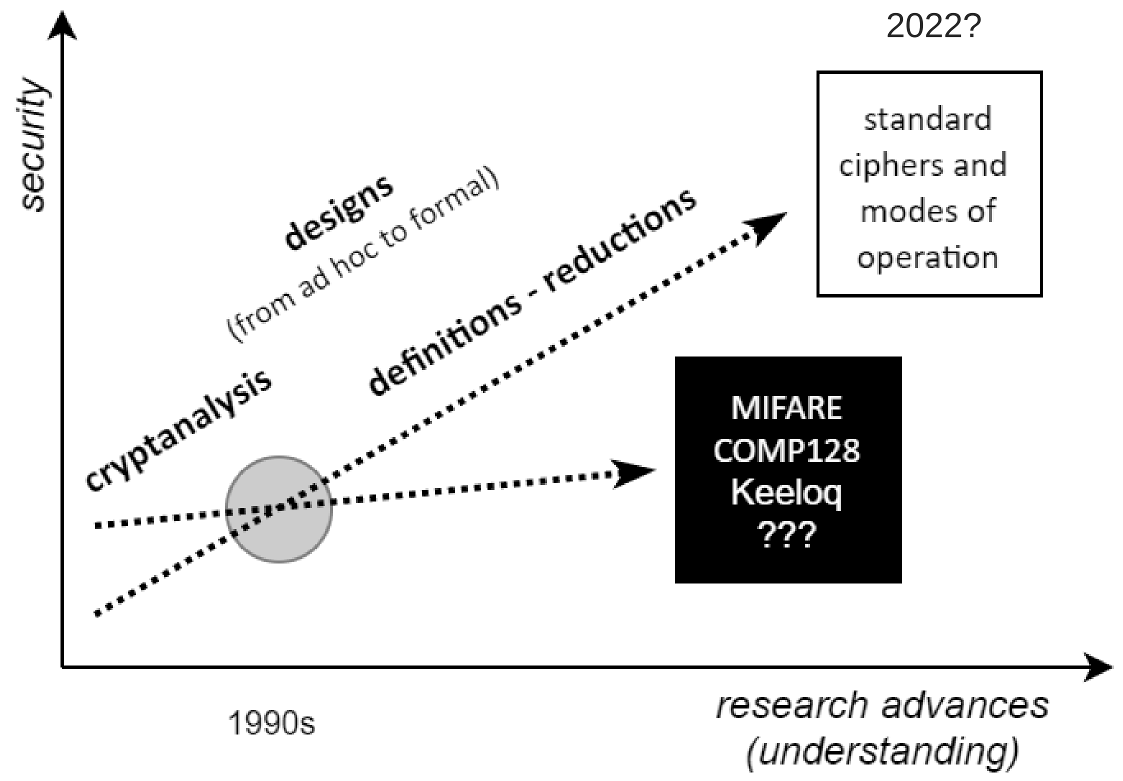




- Such an anticipation is significantly easier in an open setting

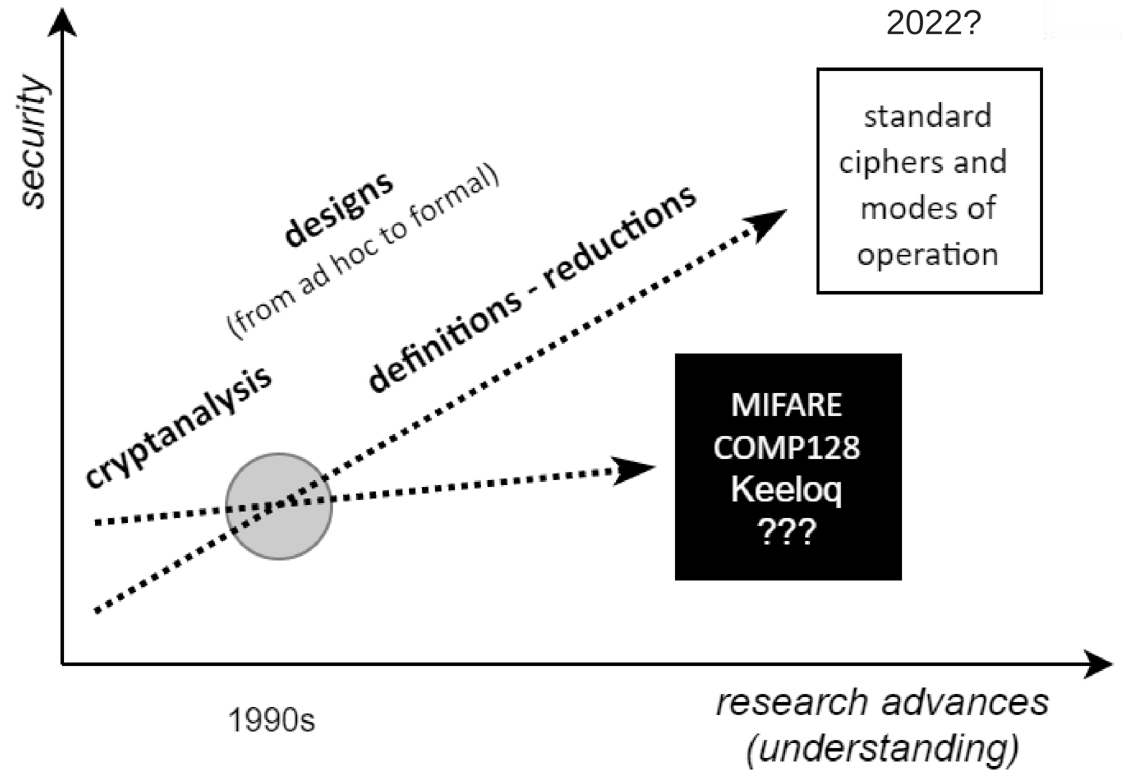
Vision II

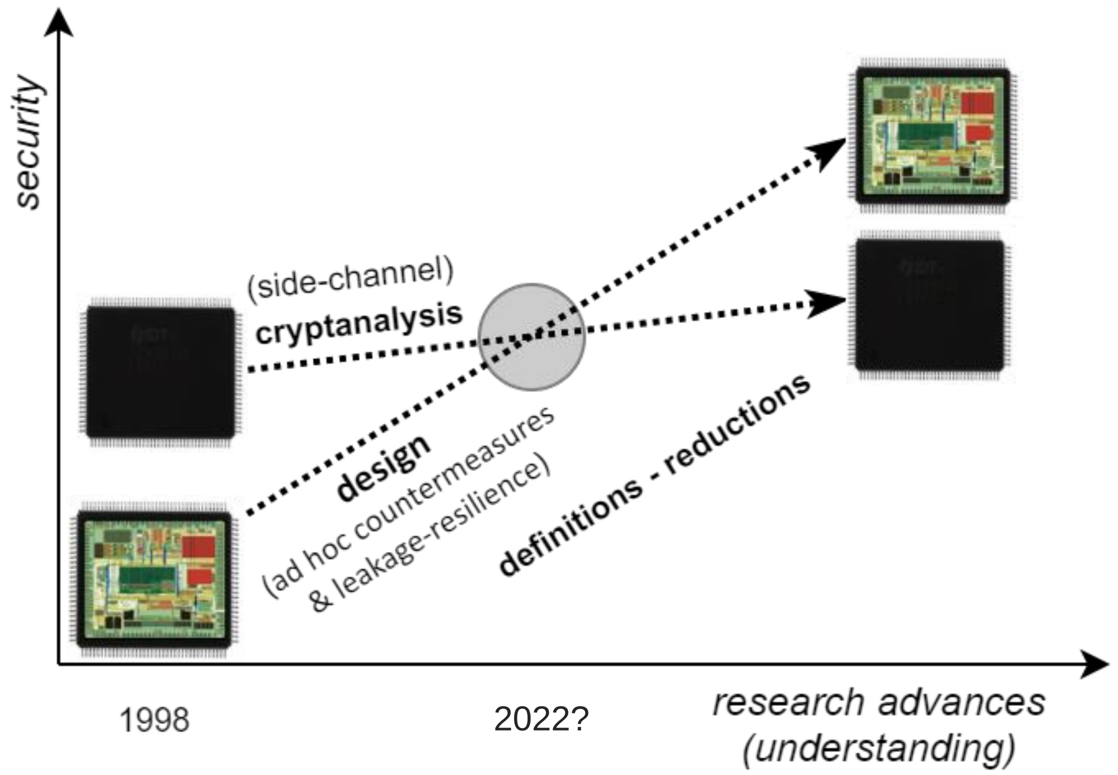
- As research advances, the advantages of security by obscurity vanish and the interest of open source solutions increases
  - Well known for algorithms



Vision II

- As research advances, the advantages of security by obscurity vanish and the interest of open source solutions increases
  - Well known for algorithms, what about implementations?




security
(side-channel)
cryptanalysis
design
(ad hoc countermeasures & leakage-resilience)
definitions - reductions
1998
2022?
research advances (understanding)

Goals (why open source can help)

1. Improving the **long-term** security of crypto. implementations
   - Make open source implementations available together with public data sets and evaluation tools that ensures reproducibility
   - Develop trainings and teaching material that support the vision
   - Maintain all this in the long-term, with continuous developments

Goals (why open source can help)

1. Improving the **long-term** security of crypto. implementations

2. Complementing the existing industrial ecosystem
   - *For design companies*: focus on high security & long term; minimize the risk of undiscovered attack paths + enable closed source integration via a dual licensing process if/when needed
     $\Rightarrow$ Leaves the question of which attack paths are realistic and the resulting cost vs. security tradeoff to existing (e.g., CC) frameworks
   - *For evaluation labs*: focus on the continuous security assessment of specific blocks - integration will always require certification
   - *For standardization*: maintain reference implementations together with open evidence of good implementation security properties

Goals (why open source can help)

1. Improving the **long-term** security of crypto. implementations
2. Complementing the existing industrial ecosystem
3. Serving as a constructive interface between academia & industry
   - Identify practically-relevant targets for security research
   - Minimize the need to target deployed product to claim relevance
     - Avoid reverse engineering efforts of limited scientific interest
     - Avoid dealing with complex responsible disclosure issues

Goals (why open source can help)

1. Improving the **long-term** security of crypto. implementations
2. Complementing the existing industrial ecosystem
3. Serving as a constructive interface between academia & industry
4. Making the model and organization flexible and open
   - Non-profit organization funded by industrial sponsors
   - Enabling new development & maintenance of external contributions
   - Projects determined by developers, sponsors and scientific committee
   - Sponsoring used to fund developers with limited overheads (developers could be attached to other research institutions)

Exemplary outcomes

- General examples
  - SCALib: an open source (efficient) SCA evaluation library
  - AES-HPC, a masked hardware AES at arbitrary order
    - Ongoing and planned to be a CHES 2023 CTF
  - Leakage-resistant AE (based on unprotected AES coprocessor)

- NIST-specific examples
  - Library of masked gadgets + evaluation datasets
  - SCA-resistant lightweight crypto standard implementation
  - PQ crypto standard implementation (SCA-resistant or not)

Organization I: typical year

1. ***Yearly report***. The developers of the association write a public report describing yearly progresses and listing potential plans (with a tentative time budget) for the next year

Organization I: typical year

1. ***Yearly report***. The developers of the association …
2. ***Sponsors' workshop***. Sponsors meet during an annual workshop to discuss the report and identify promising developments

Organization I: typical year

1. ***Yearly report***. The developers of the association …
2. ***Sponsors' workshop***. Sponsors meet during …
3. The ***scientific council*** reviews the proposed developments and establishes a priority list (+ suggests developers) based on
   - The projects' scientific maturity (theory background, PoC)
   - Their security in the worst-case evaluation setting
   - Mid/long-term relevance for emerging industrial applications
   - Genericity and portability on wide range of platforms

Organization I: typical year

1. ***Yearly report***. The developers of the association …
2. ***Sponsors' workshop***. Sponsors meet during …
3. The ***scientific council*** reviews the proposed …
4. ***Projects selection***. The association board approves the report. Developers are contacted to work on the projects of the priority list by the association's contact points and open source projects are launched for one year given the budget constraints

Organization II: projects’ lifetime

1. ***Development.*** The selected projects are developed towards prototype codes with accurate documentation and test vectors. Their (e.g., physical) security is evaluated internally

Organization II: projects’ lifetime

1. ***Development.*** The selected projects are developed …
2. ***Public evaluation.*** The prototype code is released under a copyleft licence together with its documentation and preliminary security evaluation report. A challenge is organized based on public data sets (possibly linked to conferences) and a call for contributions aiming at improving the code is open

Organization II: projects' lifetime

1. ***Development.*** The selected projects are developed …
2. ***Public evaluation.*** The prototype code is released …
3. ***Gold sponsoring.*** The code is used in industrial projects. The gold sponsoring enables integration under a non-copyleft license during the membership year, while it remains available under a copyleft license for open source developments, which we denote as dual licensing. As long as a code is gold-sponsored, the detection of security flaws goes through a separate incident response process, following standard responsible disclosure

Organization II: projects' lifetime

1. ***Development.*** The selected projects are developed …
2. ***Public evaluation.*** The prototype code is released …
3. ***Gold sponsoring.*** The code is used in industrial projects…
4. ***Final release.*** Once the code has been gold-sponsored for *X* years, it is released under a non-copyleft license for the whole community (*X to be discussed, tradeoff between sponsoring fee and competitive advantage, different for HW and SW?*)

Organization III: licensing policy (for sponsors)

1. ***Development and public evaluation***: copyleft licenses
    - Software implementations: GPLv3
    - Hardware implementations: CERN OHL-S

Organization III: licensing policy (for sponsors)

1. ***Development and public evaluation***: copyleft licenses
   - Software implementations: GPLv3
   - Hardware implementations: CERN OHL-S

2. ***Gold sponsoring***: yearly non-copyleft licenses
   - Software & hardware implementations: “yearly MIT”

Organization III: licensing policy (for sponsors)

1. ***Development and public evaluation***: copyleft licenses
   - Software implementations: GPLv3
   - Hardware implementations: CERN OHL-S
2. ***Gold sponsoring***: yearly non-copyleft licenses
   - Software & hardware implementations: “yearly MIT”
3. ***Final release***: perpetual permissive (non-copyleft) licenses
   - Software & hardware implementations: MIT-like

Organization III: licensing policy (for sponsors)

1. ***Development and public evaluation***: copyleft licenses
   - Software implementations: GPLv3
   - Hardware implementations: CERN OHL-S
2. ***Gold sponsoring***: yearly non-copyleft licenses
   - Software & hardware implementations: “yearly MIT”
3. ***Final release***: perpetual permissive (non-copyleft) licenses
   - Software & hardware implementations: MIT-like
4. ***Special case***: evaluation tools (more continuous dev.)
   - Copyleft license: AGPLv3
   - Non-copyleft license: “yearly MIT” (no final release)

Organization III: licensing policy (for contributors)

- Contributor License Agreement requests perpetual, worldwide, non-exclusive, no charge, royalty-free and irrevocable license
  - Ensures that SIMPLE-Crypto can relicense the works;
  - Enables the dual-licensing model;
  - Ensures that SIMPLE-Crypto can enforce the license clauses;
  - Ensures that no contributor will sue any user of the code for patent infringement caused by the contribution

Organization III: licensing policy (for contributors)

- Contributor License Agreement requests perpetual, worldwide, non-exclusive, no charge, royalty-free and irrevocable license
  - Ensures that SIMPLE-Crypto can relicense the works;
  - Enables the dual-licensing model;
  - Ensures that SIMPLE-Crypto can enforce the license clauses;
  - Ensures that no contributor will sue any user of the code for patent infringement caused by the contribution
- The contributor is guaranteed that:
  - He/she retains copyright on his/her contributions
  - The contributions will be licensed according project stage
  - The CLA is with SIMPLE, a non-profit organization whose statutory goal is to develop open source cryptographic implementations

Types of sponsors (tentative)

- **Bronze** (5kE): support the vision and goals of the association, access to the yearly sponsors’ workshop and access to the associations’ general assembly as non-voting member

Types of sponsors (tentative)

- **Bronze** (5kE): support the vision and goals of the association, access to the yearly sponsors’ workshop and access to the associations’ general assembly as non-voting member
- **Silver** (10kE): same as bronze + two free tickets to (one of) the annual training(s) organized by the association and non-copyleft access to the evaluation tools during the membership year

Types of sponsors (tentative)

- **Bronze** (5kE): support the vision and goals of the association, access to the yearly sponsors' workshop and access to the associations' general assembly as non-voting member
- **Silver** (10kE): same as bronze + two free tickets to (one of) the annual training(s) organized by the association and non-copyleft access to the evaluation tools during the membership year
- **Gold** (25kE per project): same as silver + non-copyleft license for one code project during the membership year

Types of sponsors (tentative)

- **Bronze** (5kE): support the vision and goals of the association, access to the yearly sponsors’ workshop and access to the associations’ general assembly as non-voting member
- **Silver** (10kE): same as bronze + two free tickets to (one of) the annual training(s) organized by the association and non-copyleft access to the evaluation tools during the membership year
- **Gold** (25kE per project): same as silver + non-copyleft license for one code project during the membership year

Sponsoring levels + amortization for multiple projects to be discussed

Timeline

- ***Dry run*** (2022-2023)
  - Develop training, tools & first code project (based on ERC results)
  - Collect (for now bronze) sponsors and set up the community (≈ *JHAS-like work group to discuss open source developments*)
- ***Phase 1*** (project expansion)
  - Identify relevant contributions (not only UCLouvain)
  - Try to integrate them via internships (e.g., senior PhDs, post-docs)
- ***Phase 2*** (in regime, e.g., after gold sponsoring)
  - Possibly fund researcher with longer-term contract (e.g., to ensure faster response in case of incident with the sponsored code)

QUESTIONS

https://www.simple-crypto.dev/