“These results must be false“: A usability evaluation of constant-time analysis tools

Marcel Fourné
Paderborn Univ., MPI-SP

Daniel De Almeida Braga
Univ Rennes, CNRS, IRISA

Jan Jancar
Masaryk Univ.

Mohamed Sabt
Univ Rennes, CNRS, IRISA

Peter Schwabe
MPI-SP, Radboud Univ.

Gilles Barthe
MPI-SP, IMDEA Software Institute

Pierre-Alain Fouque
Univ Rennes, CNRS, IRISA

Yasemin Acar
George Washington Univ., Paderborn Univ.

USENIX Security - August, 14-16th 2024

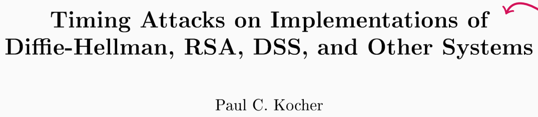
It All Started a Long Time Ago...

Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems

1996

Paul C. Kocher

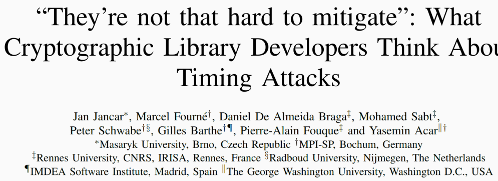
It All Started a Long Time Ago...



Timing Attacks on Implementations of Diffie-Hellman, RSA, DSS, and Other Systems

Paul C. Kocher

1996



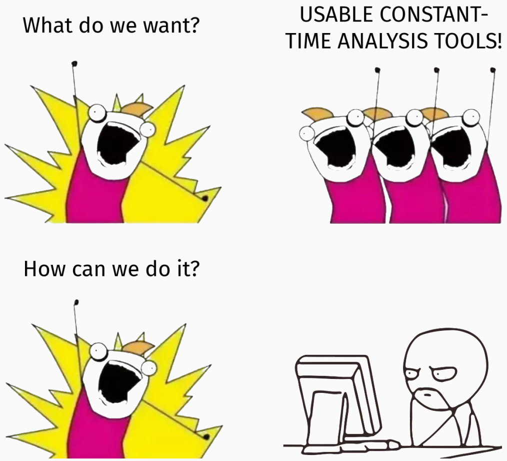
“They’re not that hard to mitigate”: What Cryptographic Library Developers Think About Timing Attacks

Jan Jancar[*], Marcel Fourné[†], Daniel De Almeida Braga[‡], Mohamed Sabt[‡], Peter Schwabe[†§], Gilles Barthe[†¶], Pierre-Alain Fouque[‡] and Yasemin Acar[∥†]
[*]Masaryk University, Brno, Czech Republic [†]MPI-SP, Bochum, Germany
[‡]Rennes University, CNRS, IRISA, Rennes, France [§]Radboud University, Nijmegen, The Netherlands
[¶]IMDEA Software Institute, Madrid, Spain [∥]The George Washington University, Washington D.C., USA

2022

Common complain: tools *“not really usable by mere mortals yet”*

Let's Make a Usable Tool!



Let’s Make a Usable Tool!

1. What are the pain points when using CT tools?
2. How effective are the tools at discovering and fixing problems?
3. How can we support potential users in using these tools?

Objectives: Assess usability of CT tools and provide recommendations.

Let's Make a Usable Tool!

1. What are the pain points when using CT tools?
2. How effective are the tools at discovering and fixing problems?
3. How can we support potential users in using these tools?

Usable according to

- Setup and familiarization
- Secret designation and target building
- Identifying and fixing issues

Objectives: Assess usability of CT tools and provide recommendations.

Methodology

- **Approach:** Two-part usability study with busy crypto developers
- **Tasks:** Participants worked on repair tasks and real-world library audits.

Methodology

- **Approach:** Two-part usability study with ~~busy crypto developers~~
- **Tasks:** Participants worked on repair tasks and real-world library audits.

Methodology

- **Approach:** Two-part usability study with 24 (post) graduate students
- **Tasks:** Participants worked on repair tasks and real-world library audits.

Methodology

- **Approach:** Two-part usability study with 24 (post) graduate students
- **Tasks:** Participants worked on repair tasks and real-world library audits.

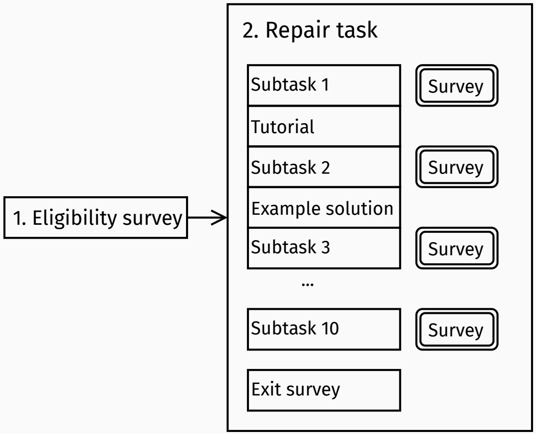
Selected tools

- MemSan
- timecop
- dudect
- ctverif
- Binsec/Rel
- haybale-pitchfork

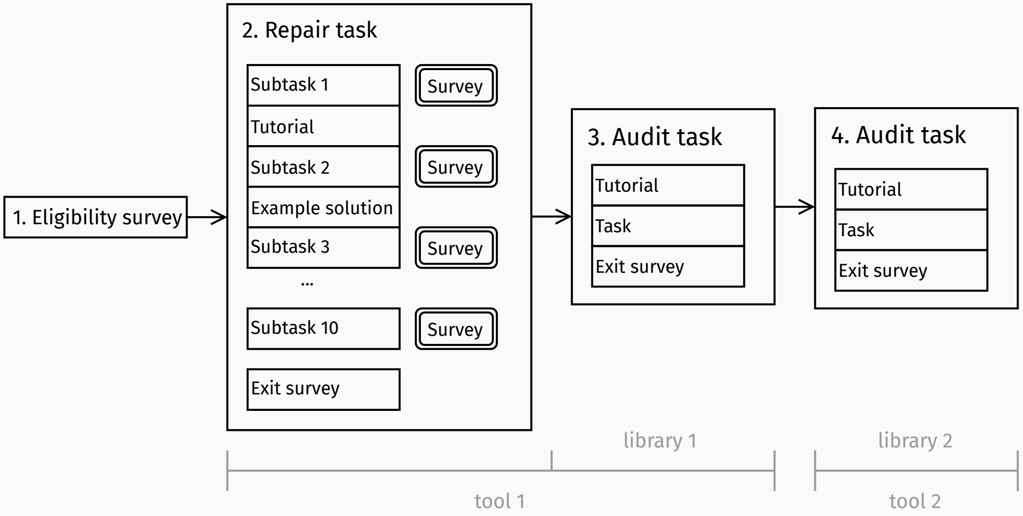
Selected libraries

- OpenSSL
- GNUTLS/Nettle
- mbedTLS

Study Workflow



Study Workflow

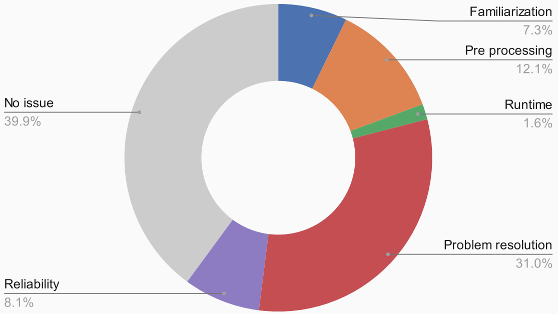


Key Findings - Repair tasks

Overall complaints:

- Detecting leakage seems doable
- Fixing leakage is tricky
- Participants were not experts

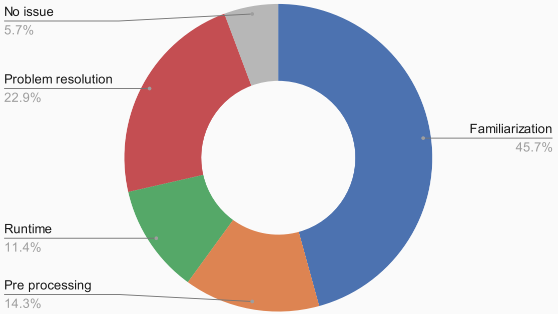
Let's look at the evolution of these complaints



Key Findings - Repair tasks

Task 1: Branch on secret

- Issues with familiarization
- Elephant in the room: bad documentation!
- What if we provide a short and effective tutorial?

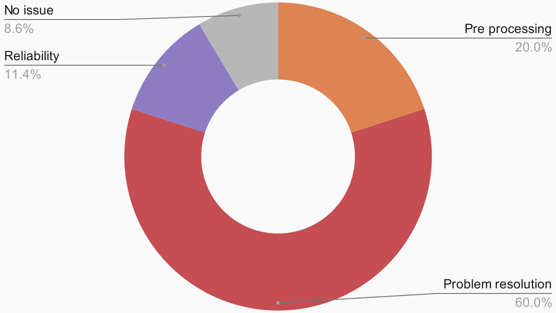


Key Findings - Repair tasks

Task 3: Memory access on secret

- Much better with our tutorial!
- Lingering pre-processing issues

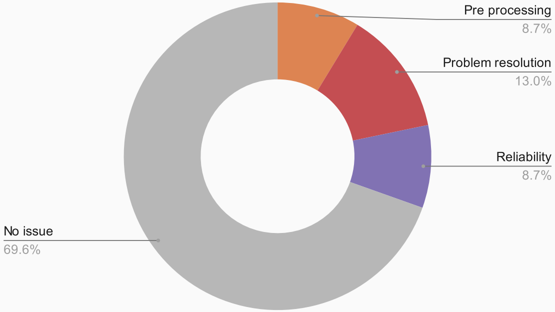
"At this point, I was using pitchfork somewhat routinely."



Key Findings - Repair tasks

Task 7:

Participants doing task 7
You know, I'm something of a constant-time developer myself



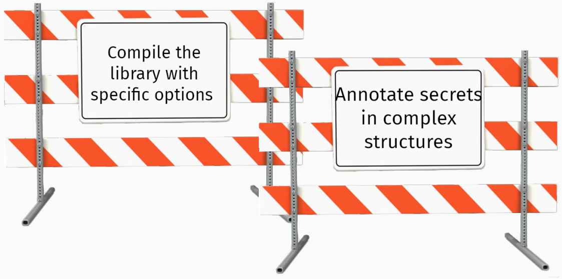
Key Findings - Audit tasks

Blocking points at each step of the road

Compile the library with specific options

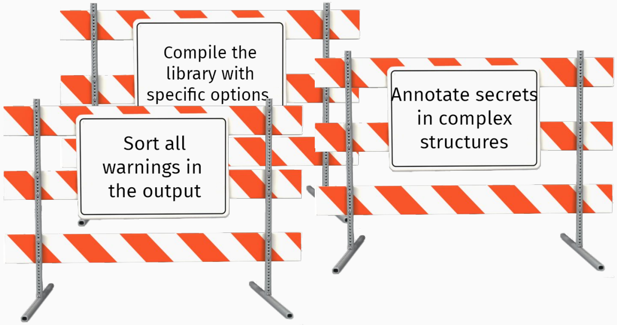
Key Findings - Audit tasks

Blocking points at each step of the road


Compile the library with specific options
Annotate secrets in complex structures

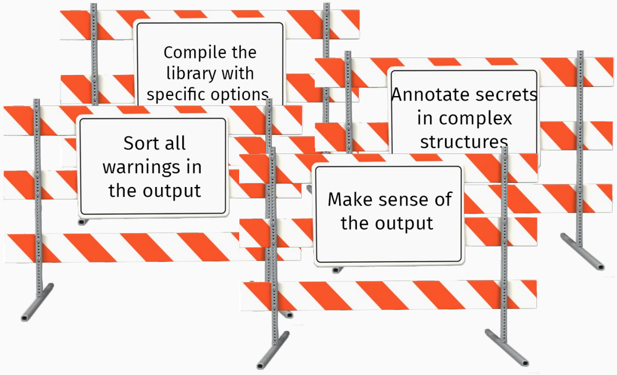
Key Findings - Audit tasks

Blocking points at each step of the road


Compile the library with specific options
Sort all warnings in the output
Annotate secrets in complex structures

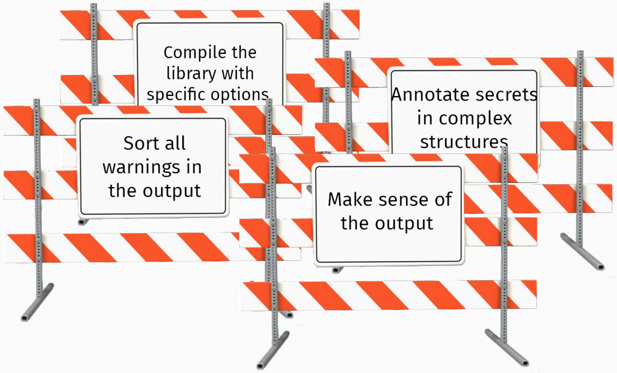
Key Findings - Audit tasks

Blocking points at each step of the road


Compile the library with specific options
Annotate secrets in complex structures
Sort all warnings in the output
Make sense of the output

Key Findings - Audit tasks

Blocking points at each step of the road


Compile the library with specific options
Annotate secrets in complex structures
Sort all warnings in the output
Make sense of the output

If they manage to run the tool, they have low confidence in the results

Recommendations

Setup and familiarization

- Straightforward installation
- Documentation **covering all** aspects
- Prioritize ease of use

Recommendations

Setup and familiarization

- Straightforward installation
- Documentation **covering all** aspects
- Prioritize ease of use

Secret designation

- Avoid significant alteration to the code
- Consider local variables

Recommendations

Setup and familiarization

- Straightforward installation
- Documentation **covering all** aspects
- Prioritize ease of use

Secret designation

- Avoid significant alteration to the code
- Consider local variables

Identifying and fixing issues

- Clear and easy to understand output
- Leakage origin tracking

Recommendations

Setup and familiarization

- Straightforward installation
- Documentation **covering all** aspects
- Prioritize ease of use

Secret designation

- Avoid significant alteration to the code
- Consider local variables

Identifying and fixing issues

- Clear and easy to understand output
- Leakage origin tracking

Integrating to developement pipeline

- Integration in CI
- Output extraction to bug-tracking tools
- Δ-mode for bug fixing

Talk to the people you are trying to help

Full paper

Artifact

Codebook, installation scripts, tutorials and more!

A huge thanks to all participants, tool developers and collaborators who made it possible