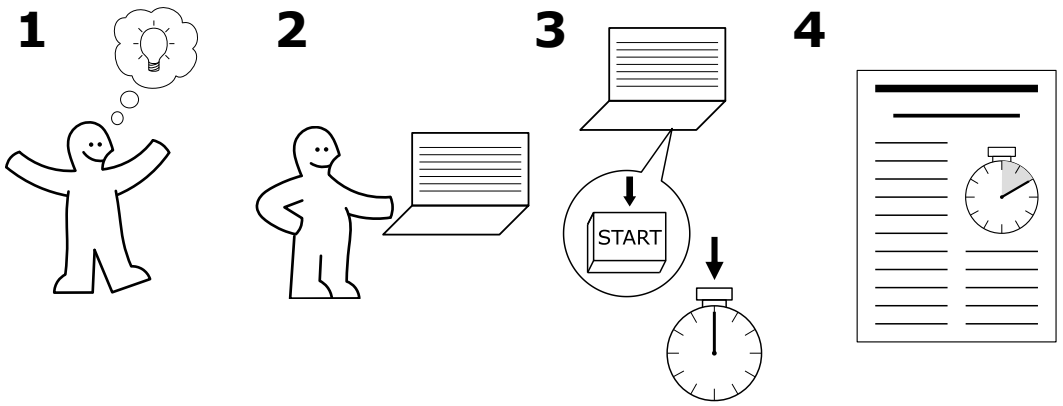
Benchmarking Flaws in Systems Security

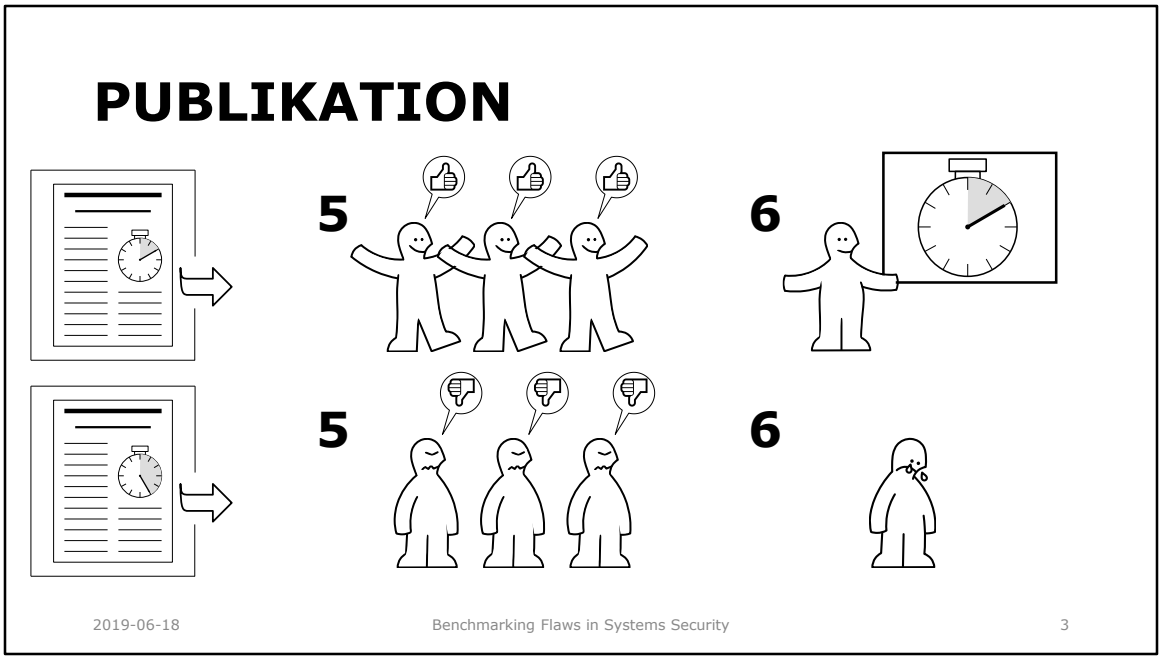
Erik van der Kouwe, Gernot Heiser, Dennis Andriesse, Herbert Bos, and Cristiano Giuffrida

EuroS&P 2019, 18 June 2019, Stockholm, Sweden

PUBLIKATION



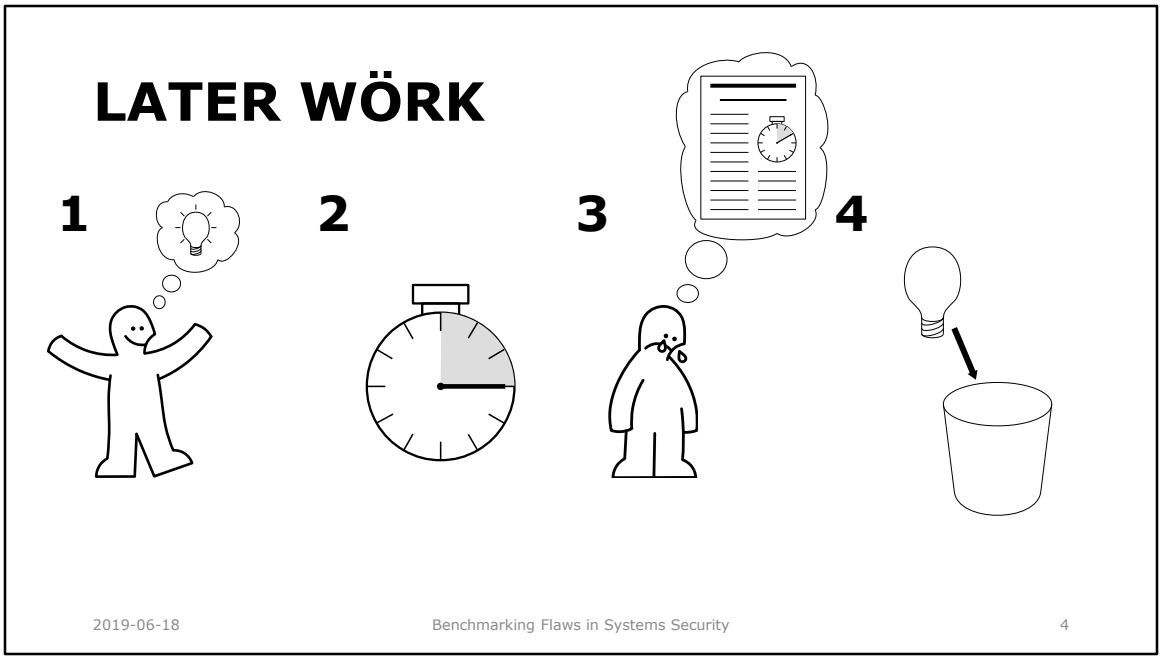
1. Researcher comes up with idea
2. Researcher implements prototype
3. Researcher benchmarks prototype
4. Researcher writes paper with benchmark result



Two options:

- Good results – reviewers approve, paper is published, researcher is happy
- Bad results – reviewers reject, paper is not published, researcher is sad

Lesson learned: benchmarking is important

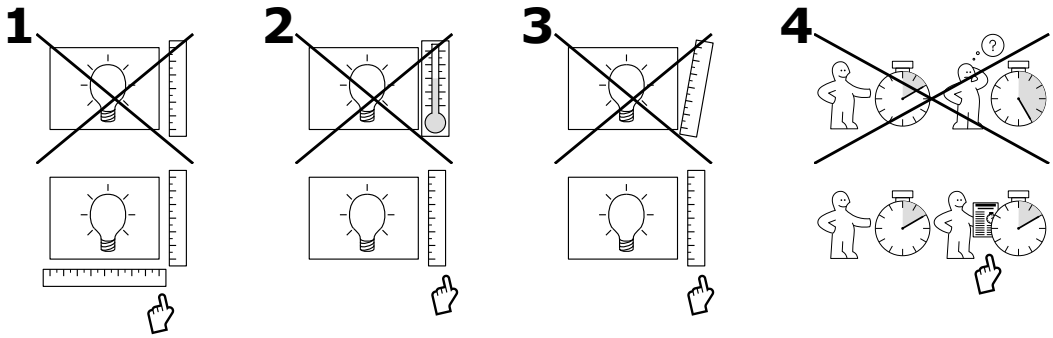


For later research:

1. Researcher comes up with idea
2. Researcher implements and benchmarks prototype
3. Researcher finds related work with better benchmarking results
4. Researcher decides not to publish

Lesson learned: benchmarking flaws can kill off good research

BENCHMÄRK



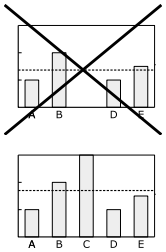
Complete Relevant Sound Reproducible

Properties needed for good benchmarking:

- Complete: verifies all claimed contributions, shows any negative impact the system may have
- Relevant: all results must be relevant in the sense that they actually tell the reader something meaningful about the system.
- Sound: all numbers measure what is intended with reasonable accuracy and repeatability.
- Reproducible: sufficient info to allow others to build the system and perform its evaluation in the same way.

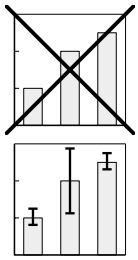
FLÅW KLASSIFIKATION

A



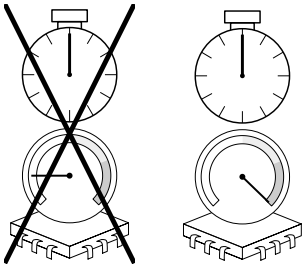
Selective benchmarking

B



Improper handling of results

C



Wrong benchmarks

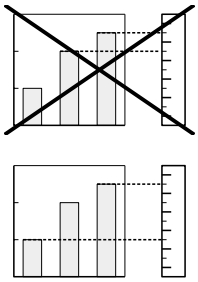
We identified 22 flaws violating requirements.

We divided them in 6 groups:

- A – selective benchmarking
  - Not measuring all contributions, considering multidimensionality of performance
  - Example: missing subbenchmarks in SPEC CPU
- B – improper handling of results
  - Interpreting and presenting benchmarking results incorrectly
  - Example: ignoring measurement inaccuracy
- C – wrong benchmarks
  - Benchmarks that misrepresents performance
  - Example: using an IO-intensive workload to measure a CPU-intensive defense

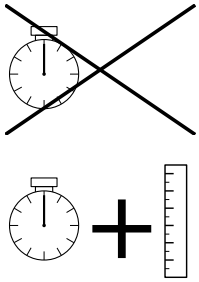
FLÅW KLASSIFIKATION

D



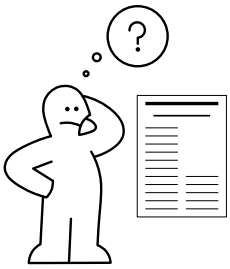
Improper comparison

E



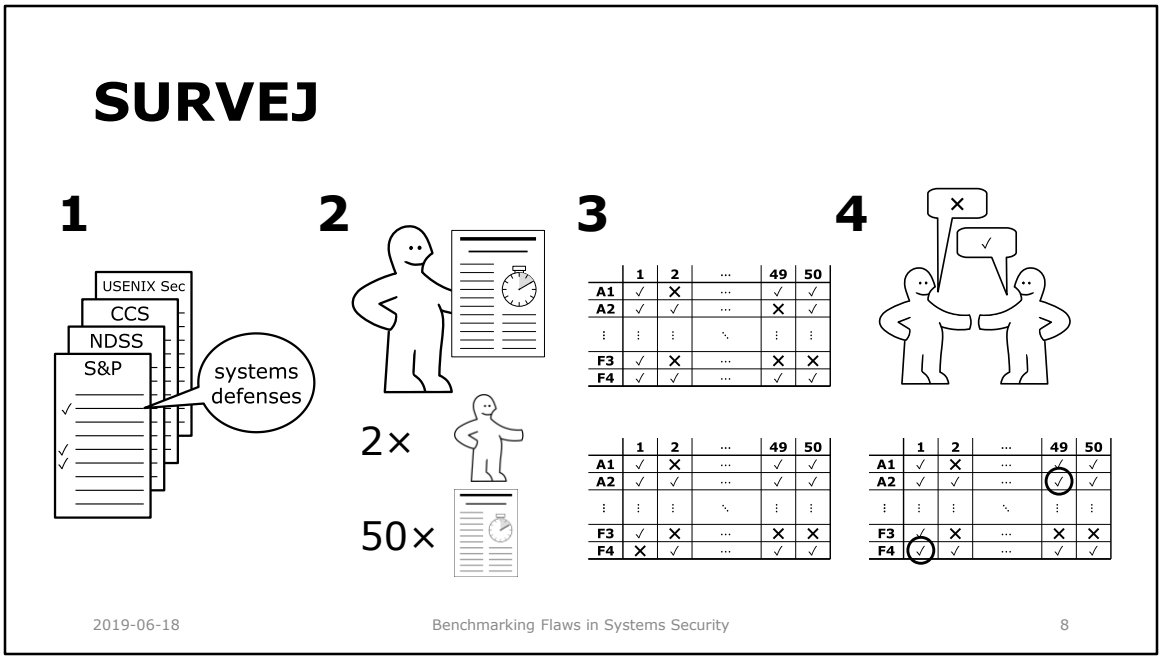
Benchmarking omissions

F



Missing information

- D – improper comparison
  - Not putting results into perspective correctly, compared to base performance and other work
  - Example: inappropriate baseline
- E – Benchmarking omissions
  - Not measuring impact outside main contributions
  - Example: ignoring memory overhead
- F – Missing information
  - Not specifying information needed for reproducibility
  - Example: not specify benchmarking platform

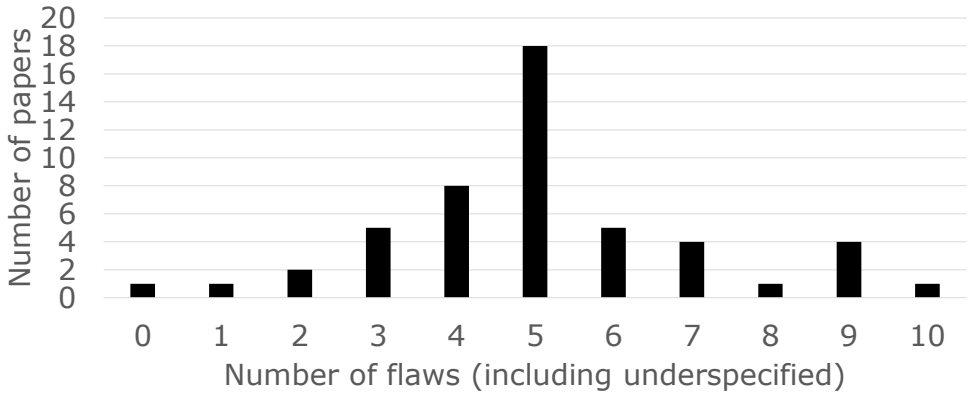


We conducted a survey to find the extent of these mistakes.

Methology:

1. Selected systems defenses from top conferences
2. Two reviewers read all 50 papers, pretending to be reviewers; if information is missing we mark it as such rather than ask the authors
3. Mark for each (paper, flaw) pair whether there is a problem
4. Discuss cases of disagreement (8 in total: 2 missed flaws, 6 only extent of the flaw) and reach a consensus

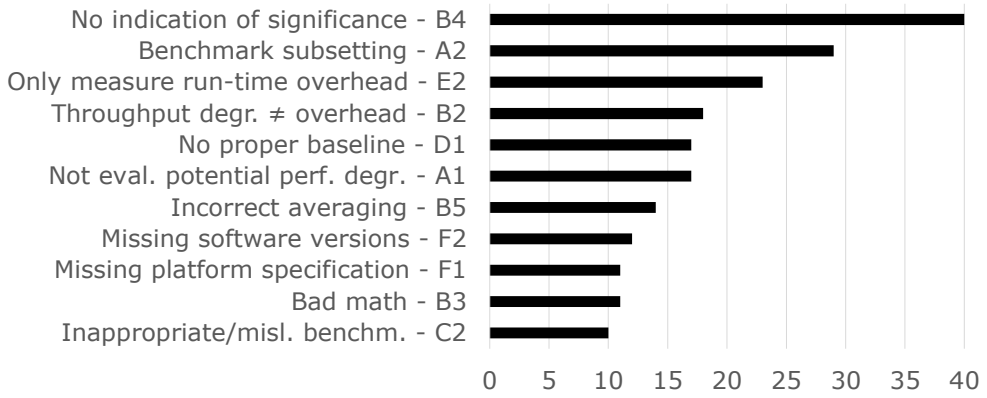
RESYLT



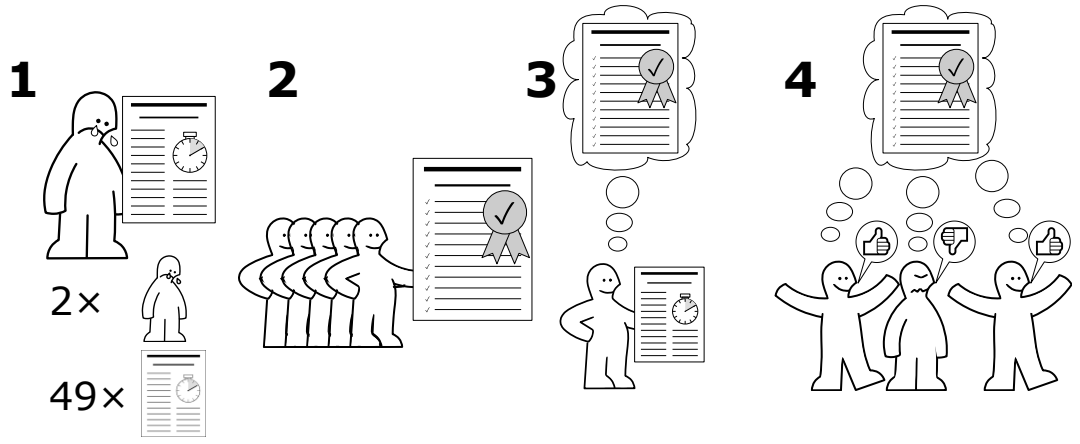
The graph shows the number of papers with *n* flaws:

- Only one paper with no flaws
- Average of 5 flaws
- Flaws are pervasive even in top conference

KOMMON FLÅWS

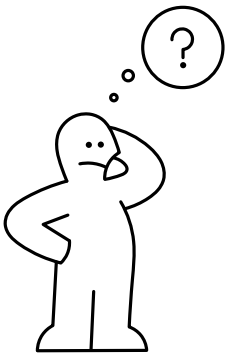


DISKUSSION



1. We have a problem, almost all papers in top conferences have benchmarking flaws
2. Solution: agree on best practices with community and build open source benchmarking tools that simplify proper benchmarking
3. Authors should consider best practices when writing
4. Program committees should consider best practices when reviewing, and require that benchmarking flaws be fixed

KVESTIONS?



Acknowledgement: human figures from https://idea-instructions.com/ (CC by-nc-sa 4.0)