CFPS — WS 2022

Seminar

Maximilian von Tschirschnitz

Lehrstuhl für Sicherheit in der Informatik / I20
Prof. Dr. Claudia Eckert
Technische Universität München

Oct 25, 2022

Paper Writing

Disclaimer: Other disciplines (than Computer Science) and even other sub-disciplines in CS may have their own style on how to structure research papers!

The following is **one possible** way out of many. Every paper has it's own dynamics and different things make sense.

Classical Research Papers

Structure

1. Abstract
2. Introduction
3. Related Work
4. Background
5. *Your stuff*
6. Evaluation
7. Discussion
8. Conclusion / Future Work

Alternative Structure

1. Abstract
2. Introduction
3. Background
4. *Your stuff*
5. Evaluation
6. Discussion
7. Related Work
8. Conclusion / Future Work

Sometimes the reader needs knowledge about your work before they can understand the related work part.

General Stuff

- Prefer *direct speech* over indirect speech.
- You may use the scientific *We* when you write
- Do not use shortened forms like: **Don't**, **We'll** or **Kinda**
- Decide if you want to write *American* or *British English*. Be consistent!

General Stuff II

- English sentences commonly do not have a deeply nested sentence structure!
- You may want to structure your paragraphs in: Statement, Explanation, Example style.
- Reference **all** figures, tables in your text!
- Try to be **precise** in your formulation!
  - this doesn't necessarily mean it has to be complicated!

Abstract

- Extent: **170 - 250 words** (my impression from ACM CCS 2019)
- Informal! Do not cite!

Structure

- Motivation - Why do we care?
- Problem Statement - What problem are we trying to solve?
- Approach - How did we go about solving?
- Result - What is the answer to the problem?
- Conclusions - What are the implications?

See also

`https://users.ece.cmu.edu/~koopman/essays/abstract.html`

Introduction

- Motivation for problem
- Extent: Usually **one** page!
- High-Level Overview of the problem and the structure of the paper.
- Clearly state your **contributions** in the end.

setups. In summary, we make the following contributions:

- We provide FridgeLock, a tool to study the impact of suspend time memory encryption on real world setups.
- FridgeLock is designed as an LKM, such that suspend time memory encryption can easily be tested on a large number of Linux distributions without the need to recompile their kernel. We achieve this using DKMS to recompile the LKM in case of security updates. This results in a solution more agnostic to kernel changes.
- We tested our module on various distributions on the x86 platform and provide performance measurements, showing a user-acceptable performance for real world usage.

Related Work

- Makes the reviewer or editors task easier: What is so novel about your work?
- Therefore you state in this section what has already been done on this topic.
- Give a wide overview only shortly introducing other approaches.

Background

Explain the foundations of the problem to the reader.

Your stuff

This is your main part! ;)

You describe here your...

- idea
- methodology
- implementation (if applicable)

Evaluation

- How did you test your work?
  - How have you made sure it works in all cases?
- How does it perform...
  - performance-wise?
  - in comparison to other approaches?

Discussion

- Defend your work against anything the reader might not like...
  - How much impact has your work?
  - What does the evaluation suggest?
  - What are the limitations?

Conclusion

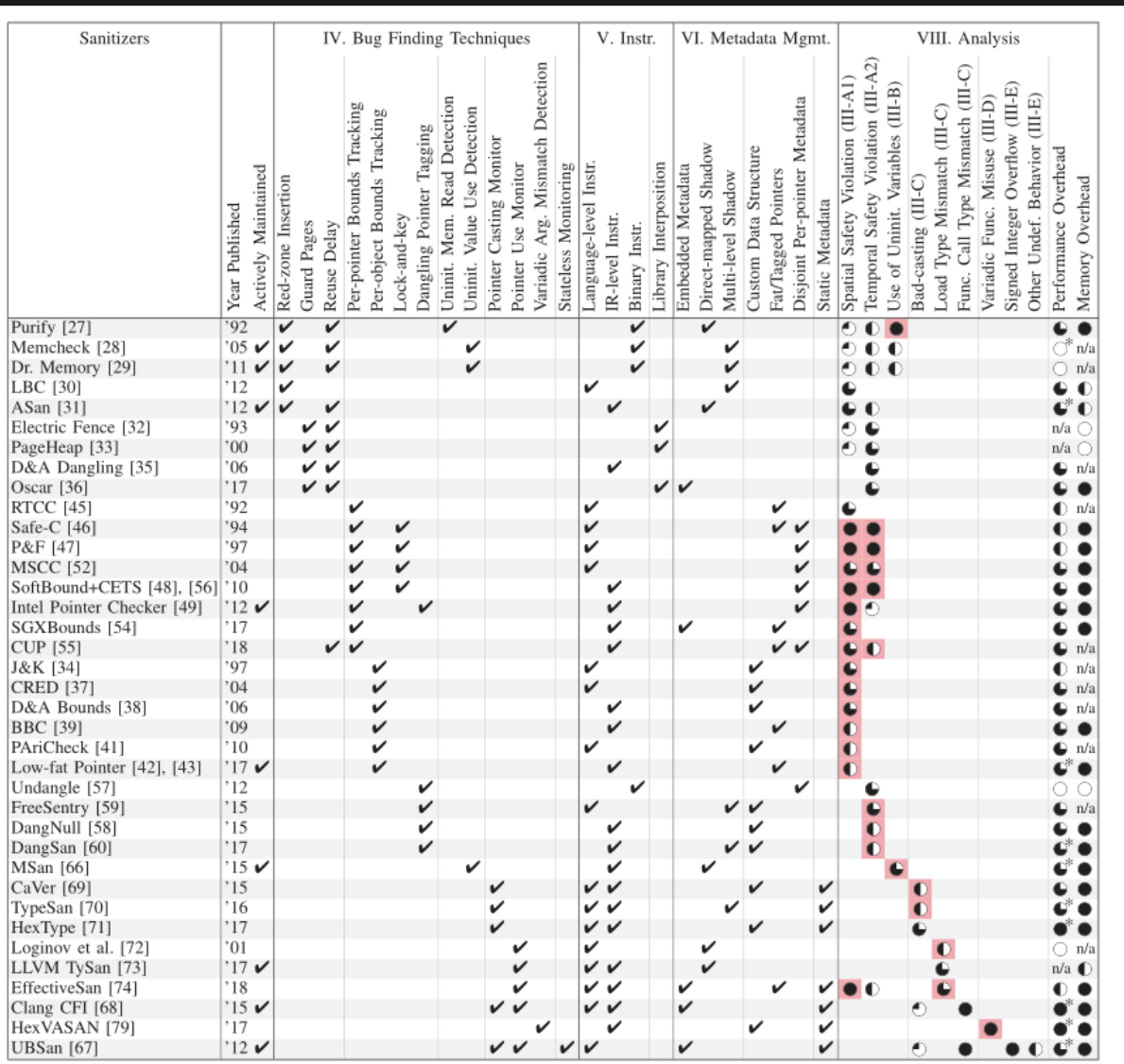
- Take a step back from your deep technical writing and summarize one last time what the reader has just learned.
- You may want to incorporate the next steps you are planning to do (Future Work).

SoK: Structure

Similar as for Classic Research Papers, but *Related Work* is skipped and *Your Stuff* is ...

- **systematization** of the research field
  - adds structure
  - identifies questions and problem statements
- **categorization** of others work
  - outlines the basic idea, limitations and implications of every surveyed paper
  - compares surveyed papers

SoK: Overview Table Example



Sanitizers	Year Published	Actively Maintained	IV. Bug Finding Techniques: Red-zone Insertion	Guard Pages	Reuse Delay	Per-pointer Bounds Tracking	Per-object Bounds Tracking	Lock-and-key	Dangling Pointer Tagging	Uninit. Mem. Read Detection	Uninit. Value Use Detection	Pointer Casting Monitor	Pointer Use Monitor	Variadic Arg. Mismatch Detection	Stateless Monitoring	V. Instr.: Language-level Instr.	IR-level Instr.	Binary Instr.	Library Interposition	VI. Metadata Mgmt.: Embedded Metadata	Direct-mapped Shadow	Multi-level Shadow	Custom Data Structure	Fat/Tagged Pointers	Disjoint Per-pointer Metadata	Static Metadata	VIII. Analysis: Spatial Safety Violation (III-A1)	Temporal Safety Violation (III-A2)	Use of Uninit. Variables (III-B)	Bad-casting (III-C)	Load Type Mismatch (III-C)	Func. Call Type Mismatch (III-C)	Variadic Func. Misuse (III-D)	Signed Integer Overflow (III-E)	Other Undef. Behavior (III-E)	Performance Overhead	Memory Overhead
Purify [27]	'92		✔		✔					✔								✔									◔	◐	●						◕	●
Memcheck [28]	'05	✔	✔		✔						✔							✔				✔					◔	◐	◐						○*	n/a
Dr. Memory [29]	'11	✔	✔		✔						✔							✔				✔					◔	◐	◐						○	n/a
LBC [30]	'12		✔													✔						✔					◕								◕	◐
ASan [31]	'12	✔	✔		✔												✔				✔						◕	◐							◕*	◐
Electric Fence [32]	'93			✔	✔														✔								◔	◕							n/a	○
PageHeap [33]	'00			✔	✔														✔								◔	◕							n/a	○
D&A Dangling [35]	'06			✔	✔												✔											◕							◕	n/a
Oscar [36]	'17			✔	✔														✔	✔								◕							◕	●
RTCC [45]	'92					✔										✔								✔			◕									◐	n/a
Safe-C [46]	'94					✔		✔								✔								✔			●	●								◐	●
P&F [47]	'97					✔		✔								✔								✔			●	●								◐	●
MSCC [52]	'04					✔		✔								✔								✔			◕	◕								◕	●
SoftBound+CETS [48], [56]	'10					✔		✔									✔							✔			●	●								◕	●
Intel Pointer Checker [49]	'12	✔				✔			✔								✔							✔			●	◔								◕	●
SGXBounds [54]	'17					✔											✔			✔				✔			◕									◕	●
CUP [55]	'18			✔		✔											✔							✔			◕	◐								◕	n/a
J&K [34]	'97						✔									✔							✔				◕									◐	n/a
CRED [37]	'04						✔									✔							✔				◕									◕	n/a
D&A Bounds [38]	'06						✔										✔						✔				◕									◕	n/a
BBC [39]	'09						✔										✔							✔			◐									◕	●
PAriCheck [41]	'10						✔									✔							✔				◐									◕	n/a
Low-fat Pointer [42], [43]	'17	✔					✔										✔							✔			◐									◕*	●
Undangle [57]	'12								✔									✔								✔										○	○
FreeSentry [59]	'15								✔							✔						✔	✔					◕								◕	n/a
DangNull [58]	'15								✔								✔					✔	✔					◕								◕	●
DangSan [60]	'17								✔								✔					✔	✔					◕								◕*	●
MSan [66]	'15	✔									✔					✔	✔				✔							◕	◐							◕*	●
CaVer [69]	'15											✔				✔	✔						✔			✔			◐							◕	●
TypeSan [70]	'16											✔				✔	✔					✔				✔			◐							◕*	●
HexType [71]	'17											✔				✔	✔						✔			✔			◐							●*	●
Loginov et al. [72]	'01												✔			✔	✔				✔								◐							○	n/a
LLVM TySan [73]	'17	✔											✔			✔	✔				✔								◐							n/a	◐
EffectiveSan [74]	'18												✔			✔	✔			✔				✔		✔		●	◐		◕					◐	●
Clang CFI [68]	'15	✔										✔	✔			✔	✔			✔						✔				◔		●				●*	●
HexVASAN [79]	'17													✔		✔	✔						✔			✔							●			●*	●
UBSan [67]	'12	✔										✔	✔		✔	✔				✔						✔				◔		●		●	◐	◕*	●

TABLE II
OVERVIEW OF SANITIZERS

SPATIAL SAFETY VIOLATION
- ◴ No stack/global var. overflow detection
- ◐ No overflow to padding detection
- ◕ No intra-object overflow detection

TEMPORAL SAFETY VIOLATION
- ◴ Dangling pointer identified at compile time
- ◐ No protection for pointers to reused memory

BAD-CASTING
- ◴ Polymorphic class support only
- ◐ Non-polymorphic class support
- ◕ Better but incomplete run-time type tracing

LOAD TYPE MISMATCH
- ◐ Scalar type granularity

PERFORMANCE OVERHEAD
- ○ Over 10x
- ◐ Up to 10x
- ◕ Up to 3x
- ● Up to 10%
- ●* Verified (see Appendix A)

Important

Every part of your paper has to have a golden thread easily recognizable.

Making fancy complicated sounding papers is easy, making easy to understand text is the skill!

What we read on this topic:
Steven Pinker's - The Sense of Style
https://www.goodreads.com/book/show/20821371-the-sense-of-style

Constance Hale's - Sin and Syntax https://www.goodreads.com/book/show/310014.Sin_and_Syntax

William Zinssner - On writing Well https://www.goodreads.com/book/show/53343.On_Writing_Well

Questions?