École Polytechnique Fédérale de Lausanne

Tooling and Analysis of the Scudo Allocator

by Elias Valentin Boschung

Bachelor Thesis

Approved by the Examining Committee:

Prof. Dr. sc. ETH Mathias Payer
Thesis Advisor

Expert Reviewer
External Expert

Mao Philipp Yuxiang
Thesis Supervisor

EPFL IC IINFCOM HEXHIVE
BC 160 (Bâtiment BC)
Station 14
CH-1015 Lausanne

May 30, 2023

Follow the white rabbit...
— The Matrix

Dedicated to my pet bunny.

The dedication is usually a short inspirational quote.

Define your dedication in `\dedication{...}` and show them with `\makededication`.

Acknowledgments

This is where you thank those who supported you on this journey. Good examples are your significant other, family, advisers, and other parties that inspired you during this project. Generally this section is about 1/2 page to a page.

Consider acknowledging the use and location of this thesis package.

Define your acknowledgments in `\acknowledgments{...}` and show them with `\makeacks`.

Lausanne, May 30, 2023 Elias Valentin Boschung

Abstract

The ScudoGEFPlugin tool enables inspection of the heap memory of an android app which uses native C libraries.

While there is a lot of existent tooling to debug errors related to heap memory in C code, it is only for the standard libc allocator. However Android uses its own allocator since Android 11, the scudo hardened allocator. Since scudo uses its own structures and way to allocate memory, those tools can not be used for debugging android C libraries.

In this project the goal was to analyze the way scudo allocates memory and then write some tooling to debug it. The tool developed takes the form of an extras plugin into the popular GEF plugin for GDB, which in turn is a popular debugger for C programs.

The abstract serves as an executive summary of your project. Your abstract should cover at least the following topics, 1-2 sentences for each: what area you are in, the problem you focus on, why existing work is insufficient, what the high-level intuition of your work is, maybe a neat design or implementation decision, and key results of your evaluation.

Contents

Acknowledgments 1

Abstract (English/Français) 2

1 Introduction 4

2 Background 5

3 Design 6

4 Implementation 7

5 Evaluation 8

6 Related Work 9

7 Conclusion 10

Bibliography 11

Chapter 1

Introduction

The introduction is a longer writeup that gently eases the reader into your thesis [1]. Use the first paragraph to discuss the setting. In the second paragraph you can introduce the main challenge that you see. The third paragraph lists why related work is insufficient. The fourth and fifth paragraphs discuss your approach and why it is needed. The sixth paragraph will introduce your thesis statement. Think how you can distill the essence of your thesis into a single sentence. The seventh paragraph will highlight some of your results The eights paragraph discusses your core contribution.

This section is usually 3-5 pages.

Chapter 2

Background

Most standard android apps are written in Java, which is the main development language for writing android apps. However for more advanced uses there exists support for including libraries written in standard C code, called native libraries referring to the underlying structure of android which is actually a modified version of linux. These native libraries can be used together with Java code, allowing C functions to be called from the java part of the code.

The code in these native libraries interacts directly with the underlying linux kernel, as any C program on a normal computer would, allocating data on the stack and the heap. However while normally on linux computers the heap memory allocations are handled by the standard libc malloc allocator, android uses it's own allocator since Android 11, called the scudo hardened allocator.

The background section introduces the necessary background to understand your work. This is not necessarily related work but technologies and dependencies that must be resolved to understand your design and implementation.

This section is usually 3-5 pages.

Chapter 3

Design

Explain gdb, gef and gef extra plugins and how they come together/interact.

Introduce and discuss the design decisions that you made during this project. Highlight why individual decisions are important and/or necessary. Discuss how the design fits together.

This section is usually 5-10 pages.

Chapter 4

Implementation

Explain how the plugin is specifically structured, how the commands, the data structures and the constants are defined.

The implementation covers some of the implementation details of your project. This is not intended to be a low level description of every line of code that you wrote but covers the implementation aspects of the projects.

This section is usually 3-5 pages.

Chapter 5

Evaluation

In the evaluation you convince the reader that your design works as intended. Describe the evaluation setup, the designed experiments, and how the experiments showcase the individual points you want to prove.

This section is usually 5-10 pages.

Chapter 6

Related Work

The related work section covers closely related work. Here you can highlight the related work, how it solved the problem, and why it solved a different problem. Do not play down the importance of related work, all of these systems have been published and evaluated! Say what is different and how you overcome some of the weaknesses of related work by discussing the trade-offs. Stay positive!

This section is usually 3-5 pages.

Chapter 7

Conclusion

In the conclusion you repeat the main result and finalize the discussion of your project. Mention the core results and why as well as how your system advances the status quo.

Bibliography

[1] Sushant Dinesh, Nathan Burow, Dongyan Xu, and Mathias Payer. “RetroWrite: Statically Instrumenting COTS Binaries for Fuzzing and Sanitization”. In: *IEEE International Symposium on Security and Privacy*. 2020.