Unsafe Rust: Conscious Choice or Spiky Shortcut?

Sandra Höltervennhoff[†], Philip Klostermeyer[†], Noah Wöhler[§], Yasemin Acar[‡], Sascha Fahl[§]
[†]Leibniz University Hannover, {hoeltervennhoff, klostermeyer}@sec.uni-hannover.de
[§]CISPA Helmholtz Center for Information Security, {noah.woehler, sascha.fahl}@cispa.de
[‡]George Washington University, acar@gwu.edu

CASA Cyber Security in the Age of Large-Scale Adversaries

Leibniz Universität Hannover

Motivation

- Rust is designed such that memory safety bugs are prevented
- Unsafe Rust grants additional capabilities, but lifts safety guarantees
- Responsibility of verification on developers, raising importance of understanding developers' (mis)conceptions and their procedures

Research Questions

RQ1. How is Unsafe code perceived and utilized?

RQ2. Is Safe and Unsafe code understood correctly?

RQ3. How is Unsafe code verified for security/safety?

RQ4. What are the consequences when Unsafe code is used incorrectly?

Methodology

- Semi-structured online interviews
- Piloting with three participants
- Eligible participants for main study: GitHub users with contribution towards unsafe code fragments
- Seven interviews conducted so far
- Analysis: Open coding

Intro
Introduction to the interview and obtaining verbal consent.

1. Participant Background
Establish participant's background and their (professional) experience with (unsafe) Rust.

2. Unsafe Code
Explore mental models, understanding, and experience surrounding unsafe Rust.

3. Testing and Tooling
Identify participants' testing practices, supporting tools, and reliance on as well as vetting of third party codebases.

4. Company Security Policies and Guidelines
Identify guidelines and resources available to contributors, content and applicability of security policies related to unsafe Rust.

5. Code Review
Establish code review practices and peculiarities of reviewing unsafe Rust.

6. Unsafe Code Misuse
Explore possible incidents of unsafe code misuse and their consequences.

7. Additional Security Measures
Identify additional measures that participants might take. Ask for suggestions w.r.t. producing more secure unsafe code.

Outro
Debrief and collect feedback for the interview.

Selected Findings

Perception and Understanding:

"There are guarantees that must be upheld that the compiler cannot check"

"[It] provides you with a very limited set of mechanisms"

Assumption: "C mode in Rust"

Usage:

- Keys: Isolation of unsafe code, (safe) interface, providing documentation

"I don't think too much about running an unsafe block to call some graphics APIs"

"I wouldn't want my unsafe code to run my pacemaker"

Guidance:

- Keys: No project-related unsafe guidelines or resources

"None of the projects I work on really think too much about it"

Misuse & Vulnerabilities:

"You just fill an advisory, do a patch release [...] and you move on"