
WHY RUST?
Introducing the Rust programming language
by István Szmozsánszky "Flaki"

ABOUT FLAKI

- Szmozsánszky István “Flaki”
- JS dev, HW hacker, Inline Skater
- Skylark.ee, Mozilla, Tessel Project, Trainer @ DPC
- `@slsoftworks`

WHAT IS RUST?

- The future of systems programming

ALIEN TECHNOLOGY

Let's talk about stories

- This is a story about history
- History can be divided up into 'epochs'
- Epochs are determined by the predominant paradigm of the time
- As things change, the paradigm (and hence the epoch) changes

- Developed first as a side project (as all world-changing projects usually are)
- Created by Graydon Hoare

MOST VULNERABILITIES IN FIREFOX

Mozilla Fixes Firefox Vulnerabilities Disclosed at Pwn2Own 2015

By Eduard Kovacs on March 24, 2015

Mozilla has released Firefox 36 updates to address the vulnerabilities presented by white-hat hackers at Pwn2Own 2015, the competition that took place last week at the CanSecWest security conference.

The first issue has been described by Mozilla as "code execution through incorrect JavaScript bounds checking elimination" (CVE-2015-0817). The security researcher who uses the online moniker "ilxu1a" leveraged this critical vulnerability at Pwn2Own to achieve medium-integrity code execution. The expert was awarded $15,000 for his accomplishment.

"Security researcher ilxu1a reported, through HP Zero Day Initiative's Pwn2Own contest, a flaw in Mozilla's implementation of typed array bounds checking in JavaScript just-in-time compilation (JIT) and its management of bounds checking for heap access. This flaw can be leveraged into the reading and writing of memory allowing for arbitrary code execution on the local system," Mozilla said in an advisory.

The second issue is a critical privilege escalation vulnerability (CVE-2015-0818) identified and reported by Mariusz Mlynski.

"[The researcher reported] a method to run arbitrary scripts in a privileged context. This bypassed the same-origin policy protections by using a flaw in the processing of SVG format [illegible] navigation," Mozilla said in a separate advisory.

Mlynski received $30,000 for this security hole and an additional $25,000 for achieving [illegible] 542 seconds.

- Pwn2own
- Spatial/temporal (off-by-one error, use-after-free)

THE FIX: RUST'S OWNERSHIP MODEL

1. Introduction
2. Getting Started
3. Tutorial: Guessing Game
4. Syntax and Semantics
   - 4.1. Variable Bindings
   - 4.2. Functions
   - 4.3. Primitive Types
   - 4.4. Comments
   - 4.5. if
   - 4.6. Loops
   - 4.7. Vectors
   - 4.8. Ownership
   - 4.9. References and Borrowing
   - 4.10. Lifetimes
   - 4.11. Mutability
   - 4.12. Structs
   - 4.13. Enums
   - 4.14. Match
   - 4.15. Patterns

The Rust Programming Language

Welcome! This book will teach you about the Rust Programming Language. Rust is a systems programming language focused on three goals: safety, speed, and concurrency. It maintains these goals without having a garbage collector, making it a useful language for a number of use cases other languages aren't good at: embedding in other languages, programs with specific space and time requirements, and writing low-level code, like device drivers and operating systems. It improves on current languages targeting this space by having a number of compile-time safety checks that produce no runtime overhead, while eliminating all data races. Rust also aims to achieve 'zero-cost abstractions' even though some of these abstractions feel like those of a high-level language. Even then, Rust still allows precise control like a low-level language would.

"The Rust Programming Language" is split into chapters. This introduction is the first. After this:

- Getting started - Set up your computer for Rust development.
- Tutorial: Guessing Game - Learn some Rust with a small project.
- Syntax and Semantics - Each bit of Rust, broken down into small chunks.
- Effective Rust - Higher-level concepts for writing excellent Rust code.
- Nightly Rust - Cutting-edge features that aren't in stable builds yet.
- Glossary - A reference of terms used in the book.
- Bibliography - Background on Rust's influences, papers about Rust.

- compile-time integrity checking
- compile-time abstraction unwrapping
- High level–but in a low level way
- zero cost abstractions
- Restricting `unsafe` code

Contributing

The source files from which this book is generated can be found on GitHub.

2. Getting Started

FEARLESS CONCURRENCY

Fearless Concurrency with Rust

Apr 10, 2015 • Aaron Turon

The Rust project was initiated to solve two thorny problems:

- How do you do safe systems programming?
- How do you make concurrency painless?

Initially these problems seemed orthogonal, but to our amazement, the solution turned out to be identical: **the same tools that make Rust safe also help you tackle concurrency head-on**.

Memory safety bugs and concurrency bugs often come down to code accessing data when it shouldn't. Rust's secret weapon is *ownership*, a discipline for access control that systems programmers try to follow, but that Rust's compiler checks statically for you.

For memory safety, this means you can program without a garbage collector *and* without fear of segfaults, because Rust will catch your mistakes.

For concurrency, this means you can choose from a wide variety of paradigms (message passing, shared state, lock-free, purely functional), and Rust will help you avoid common pitfalls.

Here's a taste of concurrency in Rust:

- A channel transfers ownership of the messages sent along it, so you can send a pointer from one thread to another without fear of the threads later racing for access through that pointer. **Rust's channels enforce thread isolation.**
- A lock knows what data it protects, and Rust guarantees that the data can only be accessed when

- Concurrency in software is HARD
- no shared mutable state

YOU CAN FIND RUST IN...

Documentation Install Community Contribute

Fork me on GitHub

Friends of Rust

(Organizations running Rust in production)

craft ai

360 DIALOG

navitia

- Browser
- Dropbox
- npm
- Toy OSs
- Next-gen OSs
- Web backends AND frontends
- Hardware
- Servo

ACADEMIA

AGILDATA

ALGORITHMIA
Open Marketplace for Algorithms

RustFest 2017

Let's meet again at a conference dedicated to the Rust programming language

TALKS

Sunday,
April 30th, 2017

LOCATION

Cosmopolite New Congress Hall
Kyiv, Ukraine

TIMELINE

✓~~Feb 16th: Open CfP~~
✓~~Feb 24th: Open Supporter Ticket~~
✓~~Mar 3rd: Open Regular Ticket Sales~~
✓~~Mar 5th: Close CfP~~
➡ Mar 19th: Confirm Speakers
- Apr 23rd: Close Ticket Sales
- Apr 30th: Conference

Get your Ticket now

✨ Supporter ✨

THANK YOU!

@SLSOFTWORKS

TALK.FLAK.IS/RUST