



BOISE STATE  
UNIVERSITY

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

# Rusty Linux: Advances in Rust for Linux Kernel Development

Shane K. Panter<sup>1</sup>   Nasir Eisty<sup>2</sup>

<sup>1</sup>Clinical Assistant Professor  
Boise State University

<sup>2</sup>Assistant Professor  
Boise State University

International Symposium on Empirical Software Engineering  
and Measurement, October 2024



BOISE STATE  
UNIVERSITY

# Introduction

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

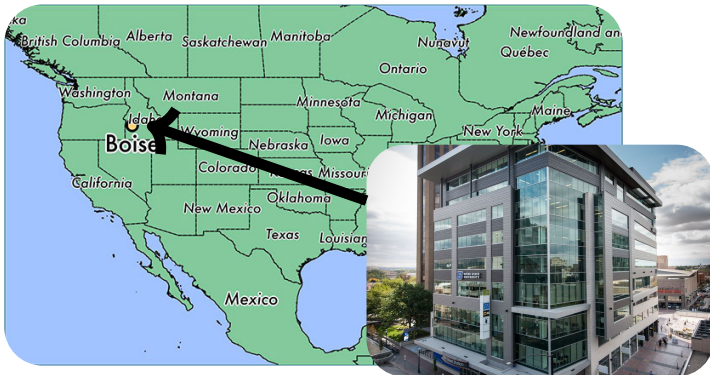
Limitations

RQ4: Lessons

Learned

Conclusion

Questions?



## Boise State University

The Computer Science Department is located in Beautiful downtown Boise Idaho, United States!



BOISE STATE  
UNIVERSITY

# Our Research

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

We aim to find the current advances in using Rust in Kernel development to reduce the number of memory safety vulnerabilities in one of the most critical pieces of software that underpins all modern applications (SLR).

► [Paper Link](#)

Figure: A rusty computer<sup>1</sup>



1

---

<sup>1</sup>AI Prompt: A rusty computer with a penguin next to it



# Why Rust

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

- Low-level **control** like C and C++
- Strong **safety** guarantees
- **Modern**, functional paradigms
- Industrial development and backing
- No garbage collector needed! All checks are performed at compile time



## Rusty Linux

### Introduction

Our Research

### Why Rust

### Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

### Results

RQ1: Existing  
Approaches

RQ2: Performance  
Implications

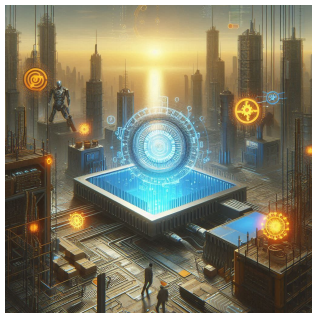
RQ3: Challenges and  
Limitations

RQ4: Lessons  
Learned

### Conclusion

### Questions?

**RQ1:** What are the existing approaches for implementing operating system kernels in Rust?<sup>2</sup>



---

<sup>2</sup>Fun side note: What happens if we put our research questions into an AI image generator?



## Rusty Linux

### Introduction

Our Research

### Why Rust

### Methodology

Research Questions

RQ1

**RQ2**

RQ3

RQ4

Process Diagram

### Results

RQ1: Existing  
Approaches

RQ2: Performance  
Implications

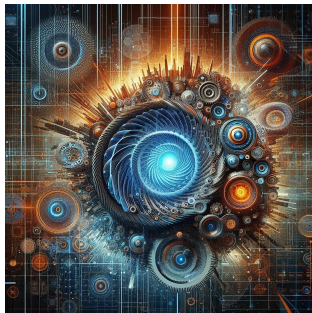
RQ3: Challenges and  
Limitations

RQ4: Lessons  
Learned

### Conclusion

### Questions?

**RQ2:** What are the performance implications of using Rust for operating system kernel development?





## Rusty Linux

### Introduction

Our Research

### Why Rust

### Methodology

Research Questions

RQ1

RQ2

**RQ3**

RQ4

Process Diagram

### Results

RQ1: Existing  
Approaches

RQ2: Performance  
Implications

RQ3: Challenges and  
Limitations

RQ4: Lessons  
Learned

### Conclusion

### Questions?

**RQ3:** What are the major challenges and limitations when developing operating system kernels in Rust?





**RQ4:** What are the lessons learned when developing operating systems kernels in Rust?







BOISE STATE  
UNIVERSITY

# Process Diagram

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

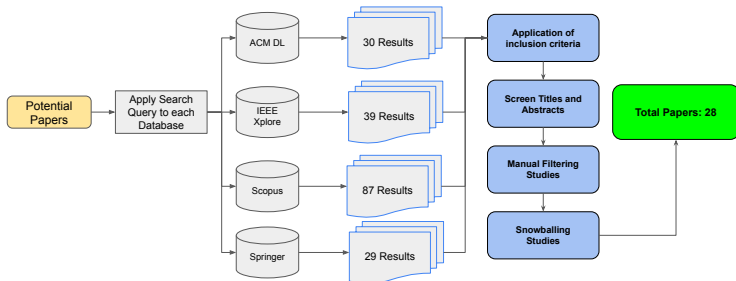
Limitations

RQ4: Lessons

Learned

Conclusion

Questions?





Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

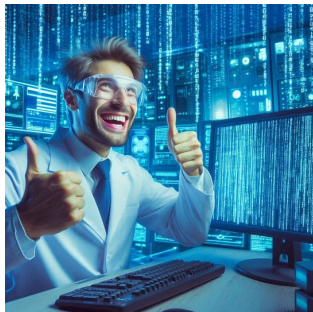
Learned

Conclusion

Questions?

Our findings!

Figure: Super happy researcher!<sup>1</sup>



1

---

<sup>1</sup>AI Prompt: scientist getting research results and is super happy in a cyberpunk universe with lots of computers showing matrix code on them



**Table:** Approaches and Methodologies for Rust in the Kernel

Approach	Papers	Operating System in Rust
Monolithic	4	Linux Kernel v6.1+
Micro-kernel	5	Atmosphere, Redox, Redleaf
Embedded	2	Tock, Hubris, Drone, Bern, HarSaRK
Unikernel	4	RustyHermit, Theseus



**Table:** Performance Implications of Rust in the Kernel

No.	Implication	Studies that Reported the challenge
1	Performance	3
2	Throughput	1
3	Latency	1

- Performance issues - Caused by the safe  $\rightarrow$  unsafe transition layer
- Throughput issues - Caused by immature and or missing bindings within the FFI layer
- Latency issues - Caused by the interrupt layer written in Rust



# RQ3: Challenges and Limitations

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and  
Limitations

RQ4: Lessons  
Learned

Conclusion

Questions?

- Binary Size
  - Rust can produce larger binaries
  - The same issue that C++ templates have!
- Missing Features
  - Rust still evolving and adding features
  - Makes it difficult to integrate into the classroom due to the rapid evolution
- Soundness - How to deal with raw memory without sacrificing safety?
- Panics - What happens when things go wrong?
- C Interop - Specific to mixed language kernels



# RQ4: Lessons Learned

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

- Impossible to use 100% rust - Same with C, some low level asm is needed to setup initial stack pointer, etc.
- Rust is not as expressive as other formal verification techniques
- Ownership root - An OS provides memory to rust so if the OS is itself written in rust who is the root owner?
  - Open research question if this can even be done in software
  - Researchers looking at hardware support (CHERI)



# Conclusion

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

- We are still in the early stages of figuring out who to do kernel dev in Rust
- High potential for enhanced security and stability
- Need to address integration issues (FFI)
- Need to expand the body of empirical evidence on Rust's impact! (Or more generally low level memory safe languages)



BOISE STATE  
UNIVERSITY

# Questions?

Rusty Linux

Introduction

Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagram

Results

RQ1: Existing

Approaches

RQ2: Performance

Implications

RQ3: Challenges and

Limitations

RQ4: Lessons

Learned

Conclusion

Questions?

## Questions?

Figure: Happy People<sup>1</sup>



---

<sup>1</sup>AI Prompt: People attending a conference who all want to ask a question and are really excited!