

Introduction Our Research

Why Rust

Methodology

Research Questions

RQ1

RQ2

RQ3

RQ4

Process Diagra

#### Results

Approaches

RQ2: Performance

RQ3: Challenge 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



#### Introduction

Rusty Linux

Introduction

Why Rus

...., .....

Methodology Research Questions RQ1

RQ3 RQ4

Process Diagra

#### Results

Approaches
RQ2: Performance
Implications
RQ3: Challenges an
Limitations
RQ4: Lessons

Conclusio

Questions?



#### Boise State University

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



#### Our Research

Rusty Linux

Introductio Our Research

vviiy ixust

Research Questions RQ1

RQ3 RQ4

Process Diagram

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges and Limitations

Conclusion

Questio

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).

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



▶ Paper Link

<sup>&</sup>lt;sup>1</sup>Al Prompt: A rusty computer with a penguin next to it



Rusty Linux

Introduction

Why Rust

vviiy ixusi

Methodology

RQ1

RQ2

RQ3

Process Diag

Results

Approaches RQ2: Performance

Implications

RQ4: Lesson

Conclusio

Questions'

■ Low-level **control** like C and C++



Rusty Linux

Why Rust

- Low-level **control** like C and C++
- Strong safety guarantees



Rusty Linux

Why Rust

- Low-level **control** like C and C++
- Strong safety guarantees
- **Modern**, functional paradigms



Rusty Linux

Introduction Our Research

Why Rust

·····y ···as

Methodology

Research Questions RQ1

RQ1

RQ3

RQ4

#### Results

RQ1: Existing
Approaches
RQ2: Performance
Implications

Limitations RQ4: Lesson

Conclusio

Questions

- Low-level **control** like C and C++
- Strong safety guarantees
- **Modern**, functional paradigms
- Industrial development and backing



Rusty Linux

Introduction Our Research

Why Rust

vviiy itust

Research Questions

RQ1

RQ3 RQ4

Process Diagra

Results

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges an Limitations

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



Our Research

Why Rus

Research Questions

RQ1 RQ2 RQ3

RQ4 Process Diagra

#### Results

Approaches

RQ2: Performance
Implications

RQ3: Challenges a
Limitations

RQ4: Lessons

Conclusion

Question

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



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

Introduction Our Research

Why Rus

Methodology Research Questions

RQ1 RQ2

RQ4

rocess Diagra

Doculto

RQ1: Existing
Approaches
RQ2: Performance

Limitations

RQ4: Lessons Learned

Conclusion

Questions

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





Introduction Our Research

Why Rus

Methodology Research Questions RQ1

RQ1 RQ2

Process Diag

\_ .

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges a

Conclusion

Questions?

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





Introduction Our Research

Why Rus

Methodology Research Questions

RQ1 RQ2

RQ3

Process Diagr

i rocess Diagn

RQ1: Existing
Approaches
RQ2: Performance

RQ3: Challen Limitations

Conclusio

Questions 7

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





## Process Diagram

Rusty Linux

Our Research

Why Rust

Research Questions

RQ1 RQ2

RQ3

Process Diagram

RQ1: Existing Approaches

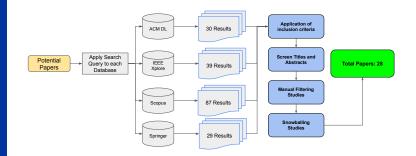
RQ2: Performant Implications

Limitations

RQ4: Lessons Learned

Conclusion

Questions'





## Results

Rusty Linux

Our Research

Why Rus

Methodology

Research Questions RQ1

RQ2

Process Diagr

Process Diagn

#### Results

Approaches
RQ2: Performanc
Implications

Limitations RQ4: Lessons

Conclusion

Questio

Our findings!





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



Introductio
Our Research

Why Rus

Methodology Research Questions RQ1 RQ2

RQ4 Process Diagram

Results

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges a Limitations

Conclusion

Questions?

#### 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

Introductio Our Research

vvny Ru

Methodology Research Questions RQ1 RQ2 RQ3

RQ4 Process Diagra

RQ1: Existing
Approaches
RQ2: Performance
Implications
RQ3: Challenges an
Limitations
RQ4: Lessons

Conclusior

Questions

#### 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 -> 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



Rusty Linux

Introduction

Why Rus

vvny Rus

Methodology

Research Questions

RQ1

RQ3

RQ4

Process Diag

Results

RQ2: Performance

RQ3: Challenges and Limitations

Learned

Conclusion

Questions?

- Rust can produce larger binaries
- lacktriangle The same issue that C++ templates have!



Rusty Linux

Introductio
Our Research

Why Rus

Methodology Research Questions

RQ1 RQ2 RQ3

RQ4 Process Diagra

Process Diagram

RQ1: Existing Approaches

Implications
RQ3: Challenges and

RQ4: Lesson:

Conclusion

Questions

- 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



Rusty Linux

Introductio

Why Rus

Methodology Research Questions RQ1

RQ4 Process Diagran

Results

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges and

Limitations RQ4: Lessons

Conclusion

Questions

- 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?



Rusty Linux

Introductio
Our Research

Why Rus

Methodology Research Questions RQ1

RQ4 Process Diagran

Results

Approaches

RQ2: Performance
Implications

Limitations RQ4: Lessons

Conclusion

Questions

- 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?



Rusty Linux

Introductio

Why Rus

Methodology Research Questions RQ1

RQ4 Process Diagram

Results

Approaches
RQ2: Performance
Implications

RQ4: Lessons

Conclusion

Questions?

- 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

RQ4: Lessons Learned

■ Impossible to use 100% rust - Same with C, some low level asm is needed to setup initial stack pointer, etc.



## **RQ4**: Lessons Learned

Rusty Linux

RQ4: Lessons Learned

- 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



## RQ4: Lessons Learned

Rusty Linux

Introductio Our Research

Why Rus

Methodology
Research Questions
RQ1
RQ2
RQ3
RO4

Process Diagran

RQ1: Existing
Approaches
RQ2: Performance
Implications
RQ3: Challenges and
Limitations
RO4: 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)



Rusty Linux

Conclusion

■ We are still in the early stages of figuring out who to do kernel dev in Rust.



Rusty Linux

Conclusion

- We are still in the early stages of figuring out who to do kernel dev in Rust.
- High potential for enhanced security and stability



Rusty Linux

Introduction

Why Rus

Methodology Research Questions

RQ1 RQ2

RQ4 Process Diagram

i rocess Diagram

Results

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges a Limitations

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)



Rusty Linux

Introductio

Why Rus

Research Questions

RQ3 RQ4

ocess Diagra

RQ1: Existing Approaches RQ2: Performance Implications RQ3: Challenges an Limitations RQ4: Lessons

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)

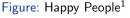


## Questions?

Rusty Linux

Questions?

Questions?





<sup>&</sup>lt;sup>1</sup>Al Prompt: People attending a conference who all want to ask a question and are really excited! 4 D > 4 P > 4 B > 4 B >