RISC-V Security Model (non-normative)

RISC-V Security Model Task Group

Version 0.1, 10/2023: This document is in development. Assume everything can change. See http://riscv.org/spec-state for details.

Table of Contents

Preamble . . . 1
Copyright and license information . . . 2
Contributors . . . 3
1. Introduction . . . 4
1.1. GLOSSARY/TAXONOMY . . . 4
1.2. Guiding Principles . . . 4
1.2.1. Intrinsic Security . . . 4
1.2.2. Principles of Zero Trust [Victor Lu] . . . 4
1.3. Generic Architecture/Framework . . . 4
1.4. Security Goals . . . 4
1.5. Adversary Model . . . 4
2. Threat Model . . . 5
2.1. Platform Integrity and Protection . . . 5
2.1.1. Secure Boot . . . 5
2.1.2. Verified Boot . . . 5
2.1.3. Attestation . . . 5
2.1.4. Debug . . . 5
2.1.5. RAS, QoS and Performance Monitoring . . . 5
2.2. Software Protection . . . 5
2.2.1. Pointer/Object Safety . . . 5
2.2.2. Stack Safety (CFI) . . . 5
2.2.3. Call/ Jump Safety (CFI) . . . 5
2.2.4. Compartmentalization . . . 5
2.3. Data Protection . . . 5
2.3.1. Code/ Data Confidentiality . . . 5
2.3.2. Code/ Data Integrity . . . 5
2.3.3. Code/Data Replay Protection . . . 5
2.4. Logical Side-channels [Luis Fiolhais, Yann Loisel] . . . 5
2.4.1. Spatial timing channel Safety . . . 5
2.4.2. Temporal Side-Channel Safety . . . 5
2.4.3. Covert channels . . . 5
2.5. Logical (Software) Attacks . . . 6
2.5.1. Approaches . . . 6
Non-invasive . . . 6
Software Remote . . . 6
2.5.2. Types . . . 6
Row hammer attacks & row press . . . 6
Power, Voltage attacks [Paul Elliott] . . . 6

Glitching, Fault injection [Paul Elliott] . . . 6
Others? . . . 6
2.6. Physical (Access) Attacks . . . 6
2.6.1. Approaches . . . 6
Semi-invasive . . . 6
Full-Invasive . . . 6
2.6.2. Types . . . 6
Row hammer attacks & row press . . . 6
Power, Voltage attacks [Paul Elliott] . . . 6
Glitching, Fault injection [Paul Elliott] . . . 6
Others? . . . 6
2.7. Supply Chain Protection . . . 6
2.7.1. Hardware Supply Chain Safety . . . 6
2.7.2. Software Supply Chain Safety . . . 6
2.8. Device Data Protection . . . 6
2.8.1. Peripheral/ IP Authentication . . . 6
2.8.2. Device Data confidentiality and integrity . . . 7
3. Platform Security Model . . . 8
3.1. Platform Root-of-Trust [Paul Elliot, Yann Loisel] . . . 8
3.1.1. Platform Unique Identity . . . 8
3.1.2. Cryptographically-Secure Entropy Source (TRNG) [Markku-JS] . . . 8
3.1.3. RTM, RTR, RTU . . . 8
3.1.4. DICE . . . 8
3.1.5. Key Management . . . 8
3.1.6. Sealed Storage . . . 8
3.2. Device Lifecycle [Yann Loisel, Terry Wang] . . . 8
3.2.1. Device Provisioning . . . 8
3.2.2. Debug . . . 8
3.2.3. Ownership Transfer . . . 8
3.2.4. Authorized Firmware Execution . . . 8
Secure Boot . . . 8
Verified Boot . . . 8
3.2.5. Device Attestation [Samuel O] . . . 8
3.2.6. Firmware Provisioning and Updates . . . 8
3.2.7. Firmware Anti-rollback . . . 8
3.3. Isolation and Trusted Execution [Ravi Sahita] . . . 8
3.3.1. Risc-V Isolation Building Blocks [Nicholas Wood, Ravi, Nick] . . . 9
PMP and ePMP . . . 9
sPMP . . . 9
MTT . . . 9
Hypervisor extension . . . 9

MMU . . . 9
IOPMP . . . 9
IOMMU . . . 9
Supervisor domains . . . 10
3.3.2. Example use case: Basic Non-virtualized system . . . 10
3.3.3. Example use case: Basic virtualized system . . . 10
3.3.4. Example use case: Global Platform TEE [Nicholas Wood] . . . 10
3.3.5. Example use case: Confidential Computing (Cove) [Nicholas Wood, Ravi Sahita] . . . 11
3.3.6. Additional examples . . . 11
3.4. Runtime Integrity [Deepak Gupta?] . . . 11
3.4.1. Control Flow Integrity [Deepak] . . . 11
3.4.2. Memory Safety . . . 11
Memory Tagging . . . 11
Capabilities/CHERI SIG [Carl Shaw] . . . 11
3.5. Cryptographic ISA Extensions/ Accelerators [Markku-JS] . . . 11
3.5.1. Zkr/bit-manip/scalar/vector/PQC . . . 11
3.5.2. HE schemes? . . . 11
3.6. Side-channel Attack Resistance [Luis Fiolhais] . . . 12
3.6.1. Entropy defenses (LWC, PQC) . . . 12
3.6.2. Flushing defenses (fences) . . . 12
3.7. Physical Adversary Attack Resistance [Paul Elliott] . . . 12
3.8. Supply-chain Attack Resistance [Paul Elliott] . . . 12
3.9. Discovery & Config Schema . . . 12
4. Security Ecosystem . . . 13
Appendix A: References . . . 14
Bibliography . . . 15

Preamble

This document is in the Development state

Assume everything can change. This draft specification will change before being accepted as informative, so implementations made to this draft specification will likely not follow the future informative specification.

Copyright and license information

Contributors

This RISC-V specification has been contributed to directly or indirectly by (in alphabetical order): Ali Zhang, Andy Dellow, Carl Shaw, Colin O'Flynn, Dean Liberty, Dong Du, Deepak Gupta, Colin O'Flynn, Guerney Hunt, Luis Fiolhais, Manuel Offenberg, Markku Juhani Saarinen, Munir Geden, Mark Hill, Nicholas Wood, Paul Elliott, Ravi Sahita, Samuel Ortiz, Steve Wallach, Suresh Sugumar, Terry Wang, Victor Lu, Ved Shanbhogue, Yann Loisel

Chapter 1. Introduction

1.1. GLOSSARY/TAXONOMY

1.2. Guiding Principles

1.2.1. Intrinsic Security

1.2.2. Principles of Zero Trust [Victor Lu]

1.3. Generic Architecture/Framework

1.4. Security Goals

1.5. Adversary Model

Chapter 2. Threat Model

We want the threat model to be complete even if RVI may not propose mitigations for all cases

2.1. Platform Integrity and Protection

2.1.1. Secure Boot

2.1.2. Verified Boot

2.1.3. Attestation

2.1.4. Debug

2.1.5. RAS, QoS and Performance Monitoring

2.2. Software Protection

2.2.1. Pointer/Object Safety

2.2.2. Stack Safety (CFI)

2.2.3. Call/ Jump Safety (CFI)

2.2.4. Compartmentalization

2.3. Data Protection

2.3.1. Code/ Data Confidentiality

2.3.2. Code/ Data Integrity

2.3.3. Code/Data Replay Protection

2.4. Logical Side-channels [Luis Fiolhais, Yann Loisel]

2.4.1. Spatial timing channel Safety

2.4.2. Temporal Side-Channel Safety

2.4.3. Covert channels

2.5. Logical (Software) Attacks

2.5.1. Approaches

Non-invasive

Software Remote

2.5.2. Types

Row hammer attacks & row press

Power, Voltage attacks [Paul Elliott]

Glitching, Fault injection [Paul Elliott]

Others?

2.6. Physical (Access) Attacks

2.6.1. Approaches

Semi-invasive

Full-Invasive

2.6.2. Types

Row hammer attacks & row press

Power, Voltage attacks [Paul Elliott]

Glitching, Fault injection [Paul Elliott]

Others?

2.7. Supply Chain Protection

2.7.1. Hardware Supply Chain Safety

2.7.2. Software Supply Chain Safety

2.8. Device Data Protection

2.8.1. Peripheral/ IP Authentication

2.8.2. Device Data confidentiality and integrity

Chapter 3. Platform Security Model

3.1. Platform Root-of-Trust [Paul Elliot, Yann Loisel]

3.1.1. Platform Unique Identity

3.1.2. Cryptographically-Secure Entropy Source (TRNG) [Markku-JS]

3.1.3. RTM, RTR, RTU

3.1.4. DICE

3.1.5. Key Management

3.1.6. Sealed Storage

3.2. Device Lifecycle [Yann Loisel, Terry Wang]

3.2.1. Device Provisioning

3.2.2. Debug

3.2.3. Ownership Transfer

3.2.4. Authorized Firmware Execution

Secure Boot

Verified Boot

3.2.5. Device Attestation [Samuel O]

3.2.6. Firmware Provisioning and Updates

3.2.7. Firmware Anti-rollback

3.3. Isolation and Trusted Execution [Ravi Sahita]

Complex systems include software components from different supply chains, and complex integration chains with different roles and actors. Isolation models enable separation of mutually distrusting software ecosystems executing on a hart so that they can be developed, certified, deployed and attested independently of each other with only a minimal shared trusted base.

3.3.1. Risc-V Isolation Building Blocks [Nicholas Wood, Ravi, Nick]

PMP and ePMP

Privileged ISA

PMP provides physical memory protection for machine mode. Physical memory can be divided into regions, and each region assigned access criteria for user mode. This enables M-mode to control physical memory access from supervisor mode. Machine mode by default can access all memory. Access rules can be locked until the next system reset to create temporal isolation boundaries, such as protecting immutable boot code. Or access rules can be dynamic, allowing machine mode to manage supervisor mode access rules at runtime.

ePMP extends PMP protection by allowing machine mode to restrict its own access to memory allocated to supervisor mode. In particular preventing machine mode from executing from or accessing memory allocated to supervisor mode. This can be used to mitigate against privilege escalation attacks, for example.

ID#	Requirement
CAT_NNN	If PMP is supported then ePMP MUST be supported.

sPMP

sPMP provides physical memory protection for supervisor mode. Supervisor physical memory, as defined by PMP access rule, can be divided into regions and each region assigned access criteria for user mode. Supervisor mode by default can access any supervisor physical memory, but supervisor mode can restrict its own access to memory assigned to user mode (similar to ePMP).

sPMP can also be used on systems supporting the H-extension. In this case a hypervisor (HS mode) controls a base sPMP configuration controlling guest physical memory accesses (VS mode). Guests can in turn control their own virtual sPMP configurations for lower privilege levels within the guest. This can support, for example, static partition hypervisors.

MTT

Hypervisor extension

Privileged ISA

The hypervisor extension adds new privilege levels in support of hypervisor based systems. Supervisor mode is split into an HS mode hosting a hypervisor, and a VS mode (virtual supervisor mode) hosting guests.

MMU

IOPMP

IOMMU

Supervisor domains

Overview

SDID

Interrupts

Performance counters

External debug

Self-hosted debug

3.3.2. Example use case: Basic Non-virtualized system

Overview

Isolation model

Device access control

Sealing

Attestation

3.3.3. Example use case: Basic virtualized system

Overview

Isolation model

Device access control

Sealing

Attestation

3.3.4. Example use case: Global Platform TEE [Nicholas Wood]

Overview

[img 3 c non virtualised tee] | *img_3_c_non-virtualised-tee.png*

Figure 1: Non-virtualised TEE example.

[img 3 c virtualised tee] | *img_3_c_virtualised-tee.png*

Figure 2: Virtualised TEE example.

Isolation model

Device access control

Sealing

Attestation

3.3.5. Example use case: Confidential Computing (Cove) [Nicholas Wood, Ravi Sahita]

Overview

[img 3 c cove] | *img_3_c_cove.png*
Figure 3: Cove example.

Isolation model

Device access control

IOMTT IOMMU (IOPMP)

Trusted device assignment TDPS APTEE-IO

Attestation

Layered attestation (6.2.2) **Sealing**

Layered sealing (local/remote provisioning)

3.3.6. Additional examples

For reference, not detailed here (variations of the above).

- Android pKVM

3.4. Runtime Integrity [Deepak Gupta?]

3.4.1. Control Flow Integrity [Deepak]

3.4.2. Memory Safety

Memory Tagging

Capabilities/CHERI SIG [Carl Shaw]

3.5. Cryptographic ISA Extensions/ Accelerators [Markku-JS]

3.5.1. Zkr/bit-manip/scalar/vector/PQC

3.5.2. HE schemes?

3.6. Side-channel Attack Resistance [Luis Fiolhais]

3.6.1. Entropy defenses (LWC, PQC)

3.6.2. Flushing defenses (fences)

3.7. Physical Adversary Attack Resistance [Paul Elliott]

3.8. Supply-chain Attack Resistance [Paul Elliott]

3.9. Discovery & Config Schema

Chapter 4. Security Ecosystem

Appendix A: References

1. https://www.intel.com/content/www/us/en/newsroom/opinion/zero-trust-approach-architecting-silicon.html
2. https://www.forrester.com/blogs/tag/zero-trust/
3. https://docs.microsoft.com/en-us/security/zero-trust/
4. https://github.com/riscv/riscv-crypto/releases
5. https://github.com/riscv/riscv-platform-specs/blob/main/riscv-platform-spec.adoc
6. https://www.commoncriteriaportal.org/files/ppfiles/pp0084b_pdf.pdf
7. https://docs.opentitan.org/doc/security/specs/device_life_cycle/
8. https://nvlpubs.nist.gov/nistpubs/ir/2021/NIST.IR.8320-draft.pdf
9. https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-193.pdf
10. https://www.rambus.com/security/root-of-trust/rt-630/
11. https://docs.opentitan.org/doc/security/specs/
12. https://trustedcomputinggroup.org/work-groups/dice-architectures/
13. https://ieeexplore.ieee.org/iel7/8168766/8203442/08203496.pdf
14. https://dl.acm.org/doi/10.1145/168619.168635
15. https://dl.acm.org/doi/abs/10.1145/3342195.3387532
16. https://github.com/riscv/riscv-debug-spec/blob/master/riscv-debug-stable.pdf
17. https://csrc.nist.gov/csrc/media/events/non-invasive-attack-testing-workshop/documents/08_goodwill.pdf
18. https://www.iso.org/standard/60612.html
19. https://ieeexplore.ieee.org/document/6176671
20. https://tches.iacr.org/index.php/TCHES/article/view/8988
21. https://ieeexplore.ieee.org/abstract/document/1401864
22. https://www.electronicspecifier.com/products/design-automation/increasingly-connected-world-needs-greater-security
23. https://www.samsungknox.com/es-419/blog/knox-e-fota-and-sequential-updates
24. https://docs.microsoft.com/en-us/windows/security/threat-protection/intelligence/supply-chain-malware
25. https://dl.acm.org/doi/10.1145/3466752.3480068
26. https://arxiv.org/abs/2111.01421
27. https://www.nap.edu/catalog/24676/foundational-cybersecurity-research-improving-science-engineering-and-institutions
28. https://trustedcomputinggroup.org/work-groups/dice-architectures/

Bibliography

- [1] The RISC-V Instruction Set Manual Volume II: Privileged Architecture Document Version 20211203 (link)