Exhaustive Security Verification of CHERI Processors

Anna Lena Duque Antón[1] *, Johannes Müller[1], Philipp Schmitz[1], Tobias Jauch[1], Alex Wezel[1], Lucas Deutschmann[1], Mohammad Rahmani Fadiheh[2], Dominik Stoffel[1], Wolfgang Kunz[1]

[1]Rheinland-Pfälzische Technische Universität Kaiserslautern-Landau (RPTU), Germany, [2]Stanford University, USA

Abstract

CHERI is a promising approach to safeguarding data in memory by providing and enforcing fine-grained memory protection directly in hardware. The recently published VeriCHERI verification flow [1] can verify global confidentiality and integrity properties for CHERI systems "spec-free", i.e., without relying on a golden ISA model. We present a case study to demonstrate the effectiveness and scalability of VeriCHERI on the CHERIoT Ibex RISC-V processor [2].

Introduction

Robust and trustworthy security mechanisms are essential in modern computing systems. Capability Hardware Enhanced RISC Instructions (CHERI) [3] offer a promising solution by implementing fine-grained memory protection directly at the hardware level. Establishing trust in the hardened hardware, however, requires comprehensive verification. Existing verification approaches focus largely on abstract ISA-level models [4, 5] and functional correctness of the RTL implementation [6]. These approaches, however, rely on the CHERI-specific, timing-abstract SAIL ISA model and can therefore miss non-functional vulnerabilities such as timing side channels in a concrete RTL design.

The recently proposed VeriCHERI verification flow [1] addresses these challenges by adopting a fundamentally different paradigm. Rather than relying on abstract ISA models, VeriCHERI derives RTL properties directly from the global security objectives of confidentiality and integrity. This results in a "*spec-free*" formal verification methodology capturing vulnerabilities that are visible only in timing-accurate models of the hardware, while avoiding the high costs of complete functional formal verification. The derived RTL properties are scalable for realistic designs by leveraging symbolic representations of requested memory addresses and capability configurations, and permitting verification of single execution traces over a finite number of clock cycles.

This work demonstrates the feasibility of the VeriCHERI flow by applying it to the CHERIoT Ibex RISC-V processor [2]. We detected several bugs in the design, including a Meltdown-style timing attack, which is not detectable by other verification approaches. This underscores the necessity for exhaustive security verification.

VeriCHERI

VeriCHERI assumes an attacker executing a task on a single-threaded single-core processor that implements CHERI. The attacker task is one among several mutually distrusting tasks that are isolated from each other by CHERI *capabilities*. Context switches between the tasks are managed securely by a trusted entity, such as an operating system. In this environment, the attacker attempts to breach CHERI's memory isolation by either accessing protected data, i.e., violating *confidentiality*, or modifying it, i.e., violating *integrity*. VeriCHERI formally expresses these security objectives as global end-to-end properties using the notion of *non-interference* [7]. This principle ensures that interactions between high (protected) and low (public) locations are impossible. From these global end-to-end properties, a set of four scalable interval properties is derived that provide the same security guarantees for confidentiality and integrity. The property set contains two 1-safety properties addressing the functional aspects of confidentiality and integrity, a 2-safety property covering non-functional security violations (side channels) and an inductive monotonicity property used to detect locations where capabilities are stored. The properties are presented in Fig. 1. The four properties are integrated into an iterative verification flow that systematically detects all security violations covered by the security objectives or, if there are no security vulnerabilities, guarantees their absence. In the following, we report on a case study to illustrate the VeriCHERI flow using a practical example.

Case Study on CHERIoT Ibex

The CHERIoT Ibex processor [2] implements a CHERI variant designed for memory safety in embedded IoT devices. Applying the VeriCHERI flow, we started with verifying the 1-safety integrity property (cf. Fig. 1b). In this step of the flow, we detected two security vulnerabilities: a misconfigured protection enable pin and a capability store that allows out-of-bounds access.

The macro *cheri_protected*(*symbolic_addr*) is assumed in all four VeriCHERI properties and ensures that no registers or buffers holding capabilities allow access to protected memory regions. To establish global validity of the proof, we check the inductive monotonicity property (cf. Fig. 1c) to ensure that *cheri_protected* remains true throughout any attacker task execution. In other words, if a memory region is protected from the attacker task, the capabilities available to the attacker cannot be modified to make the memory region unprotected. This corresponds to the concept of *capability monotonicity*, which is a central concept employed in CHERI. In addition, the VeriCHERI flow utilizes the monotonicity and 1-safety integrity properties to iteratively detect all registers or buffers holing capabilities.

*Corresponding author: anna.duqueanton@rptu.de

(a) *1-safety interval property for confidentiality*

```
assume:
  at t: cheri_protected(symbolic_addr);
prove:
  at t: read_access → mem_addr ≠
    symbolic_addr;
```

(b) *1-safety interval property for integrity*

```
assume:
  at t: cheri_protected(symbolic_addr);
prove:
  at t: write_access → mem_addr ≠
    symbolic_addr;
```

(c) *inductive monotonicity property*

```
assume:
at t:   cheri_protected(symbolic_addr);
prove:
at t+1: cheri_protected(symbolic_addr);
```

(d) *UPEC-CHERI property*

```
assume:
at t:   cheri_protected(symbolic_addr);
at t:   public_data_1 = public_data_2;
prove:
at t+k: arch_state_1 = arch_state_2;
```

Figure 1: *The four interval properties used in VeriCHERI. The 2-safety property UPEC-CHERI [1] is shown in an abstract form for clarity and brevity.*

If one of the properties fails, the counterexample either reveals a security vulnerability or it fails due to a register or buffer holding a capability that is not yet constrained by the *cheri_protected* macro. After resolving all issues, the integrity property is re-checked. This iterative procedure is repeated until a vulnerability is detected or the 1-safety integrity property holds successfully. We applied the procedure and detected 40 such registers and buffers in the processor.

The next step in the VeriCHERI flow is to check the 1-safety-confidentiality property (cf. Fig. 1b). If the property holds, no access to protect memory locations is possible in the design is secure w.r.t. the confidentiality objective. If the property fails, however, this can mean two things: After a read access to a protected memory region, the data either propagates to an internal buffer, where it is subsequently overwritten or flushed and does not influence the attacker-visible state, or the data impacts the attacker-visible state. The latter is possible either through direct functional leakage, e.g., the data propagates to the register file, or through a side channel, e.g., the data affects the timing of subsequent instructions in the processor. For distinguishing between the two cases, VeriCHERI employs the UPEC-CHERI property for further analysis, presented in an abstract form in Fig. 1d. This property considers two instances of the processor differing only in protected data at time t. It verifies that no propagation of protected data into architecturally visible registers can occur. For details, the reader is referred to [1].

Using the 1-safety confidentiality property, we discovered an instruction fetch vulnerability that could potentially violate the confidentiality of the system. Further analysis with UPEC-CHERI identified this vulnerability as a Meltdown-style side channel. The issue, caused by delayed pipeline flushes leaking two bits of data, was resolved by preventing illegal instructions from entering the core.

After removing the vulnerability by fixing the RTL design, all properties were successfully verified, with proof runtimes ranging from seconds to a maximum of 31 minutes. The case study required two person-months, including invariant development. Reusing invariants will significantly reduce effort for future applications.

Conclusion

VeriCHERI detects confidentiality and integrity vulnerabilities in CHERI processors exhaustively by employing global security objectives based on non-interference. In a case study on the CHERIoT Ibex core, we revealed three security bugs, including a Meltdown-style timing side channel attack, which previous ISA-based verification methods cannot detect. Additionally, we demonstrated VeriCHERI's scalability and feasibility w.r.t. manual effort.

References

[1] Anna Lena Duque Antón, Johannes Müller, Phillip Schmitz, Tobias Jauch, Alex Wezel, Lucas Deutschmann, Mohammad R. Fadiheh, Dominik Stoffel, and Wolfgang Kunz. "VeriCHERI: Exhaustive Formal Security Verification of CHERI at the RTL". In: *2024 IEEE/ACM International Conference on Computer Aided Design (ICCAD)*. 2024.

[2] Saar Amar, David Chisnall, Tony Chen, Nathaniel Wesley Filardo, Ben Laurie, Kunyan Liu, Robert Norton, Simon W. Moore, Yucong Tao, Robert N. M. Watson, et al. "CHERIoT: Complete Memory Safety for Embedded Devices". In: *Proceedings of the 56th Annual IEEE/ACM International Symposium on Microarchitecture*. 2023, pp. 641–653.

[3] Jonathan Woodruff, Robert N. M. Watson, David Chisnall, Simon W. Moore, Jonathan Anderson, Brooks Davis, Ben Laurie, Peter G. Neumann, Robert Norton, and Michael Roe. "The CHERI capability model: Revisiting RISC in an age of risk". In: *ACM SIGARCH Computer Architecture News* 42.3 (2014), pp. 457–468.

[4] Kyndylan Nienhuis, Alexandre Joannou, Thomas Bauereiss, Anthony Fox, Michael Roe, Brian Campbell, Matthew Naylor, Robert N. M. Norton, Simon W. Moore, Peter G. Neumann, et al. "Rigorous engineering for hardware security: Formal modelling and proof in the CHERI design and implementation process". In: *IEEE Symp. on Security and Privacy*. IEEE. 2020, pp. 1003–1020.

[5] Richard Grisenthwaite, Graeme Barnes, Robert NM Watson, Simon W Moore, Peter Sewell, and Jonathan Woodruff. "The Arm Morello Evaluation Platform–Validating CHERI-Based Security in a High-Performance System". In: *IEEE Micro* 43.3 (2023), pp. 50–57.

[6] Dapeng Gao and Tom Melham. "End-to-end formal verification of a RISC-V processor extended with capability pointers". In: *Formal Methods in Comp. Aided Design (FMCAD)*. IEEE. 2021, pp. 24–33.

[7] J. A. Goguen and J. Meseguer. "Security Policies and Security Models". In: *IEEE Symp. on Sec. and Privacy*. 1982, pp. 11–20.