CHERITech'26
CONFERENCE

CHERI

12-13 Nov 2026
Munich, DE

LittleCHERI: Ultra-Fine-Grain Compartmentalization using Secure Stack Sharing

Elias Storme
PhD Candidate
DistriNet, KU Leuven, Belgium



Located at semi | SEMICON EUROPA

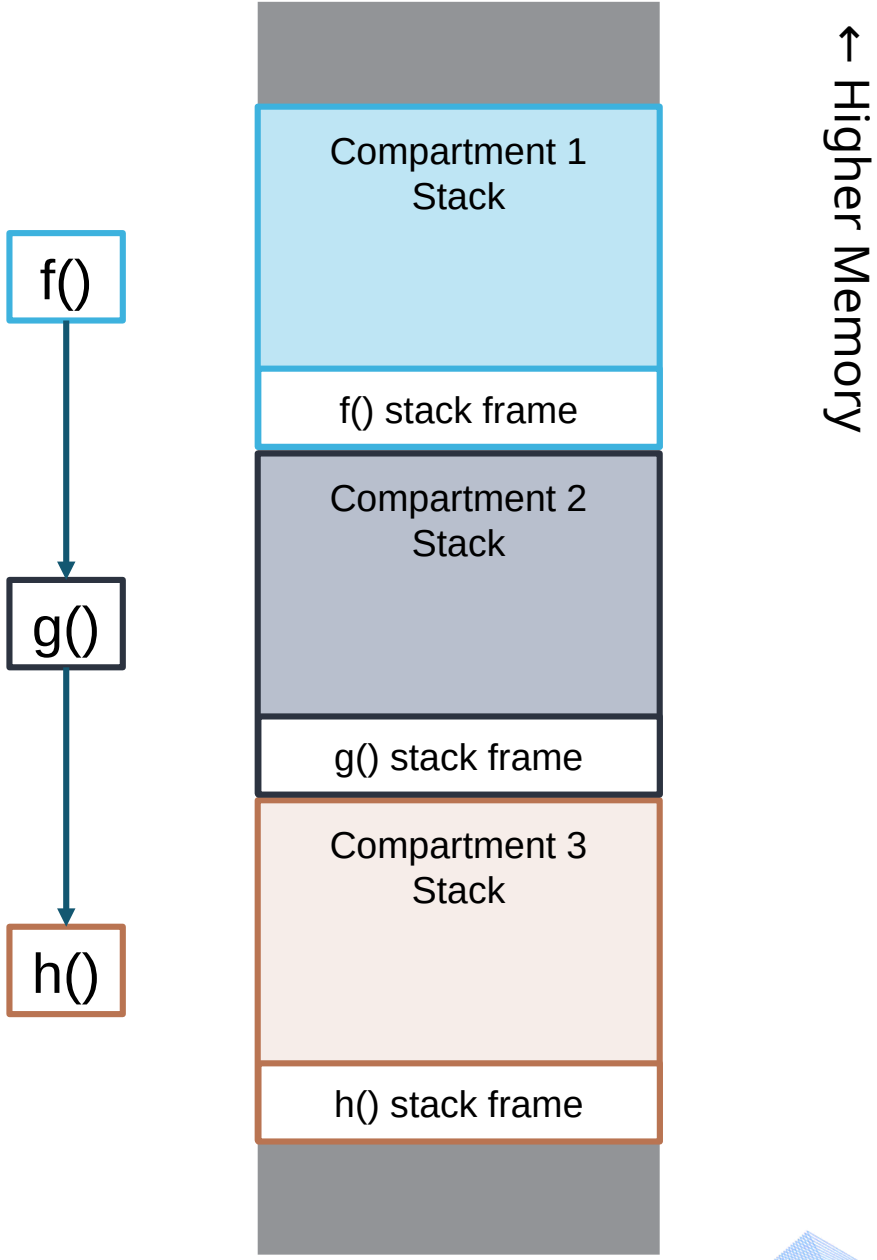
Research in progress

- At DistriNet, KU Leuven, Belgium
- By:
  - Elias Storme
  - Sander Huyghebaert
  - Thomas Van Strydonck
  - Steven Keuchel
  - Dominique Devriese



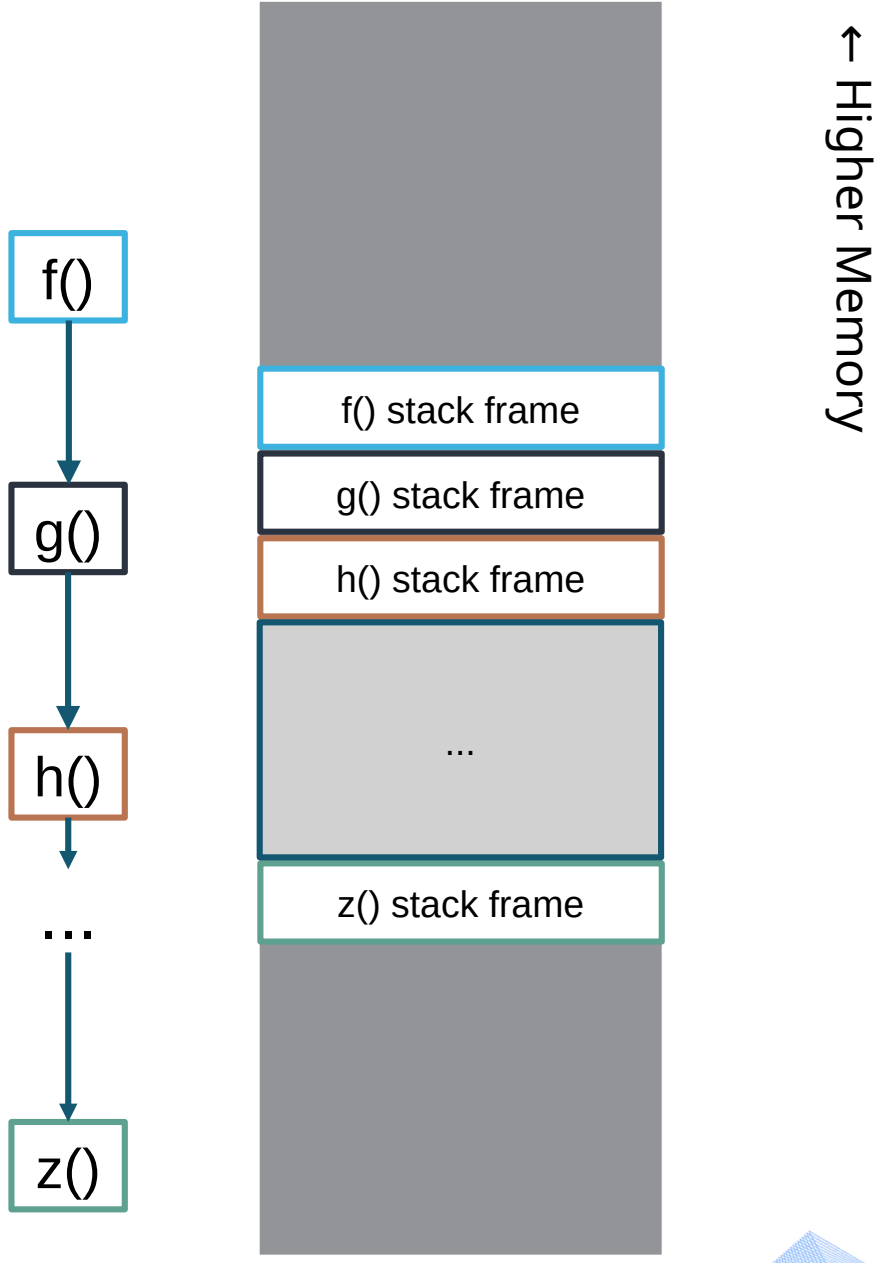
CHERI Compartmentalization Remains Relatively Expensive

- Separate stacks per compartment
- Many compartments ⇒ memory fragmentation



Stack Sharing has Potential to Lower Costs

- Lower cost ⇒
  create more compartments!



Stack Needs Temporal Safety

- Shared stack between mutually distrusting adversaries
- Theoretical literature[1]:
  - Propose secure calling convention
  - Do not elaborate beyond models

[1] Skorstengaard et al. 2020, Georges et al. 2021

What’s missing?

- Design gaps between model and real hardware
- Compiler & runtime implementation
- Performance evaluation

LittleCHERI Contributions

- Design refinements
- Implementation
- Evaluation

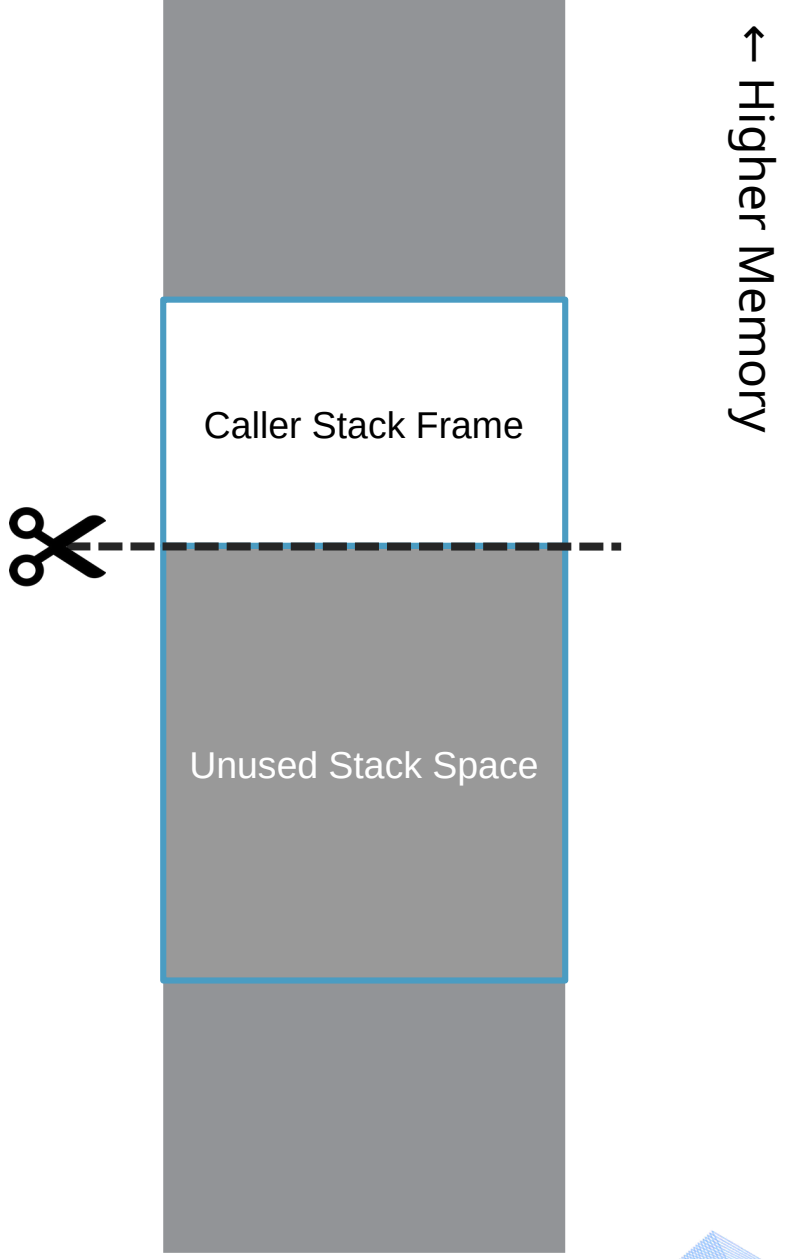
LittleCHERI Contributions

- **Design refinements**
- Implementation
- Evaluation

Design Refinements

Example: Reserve Stack

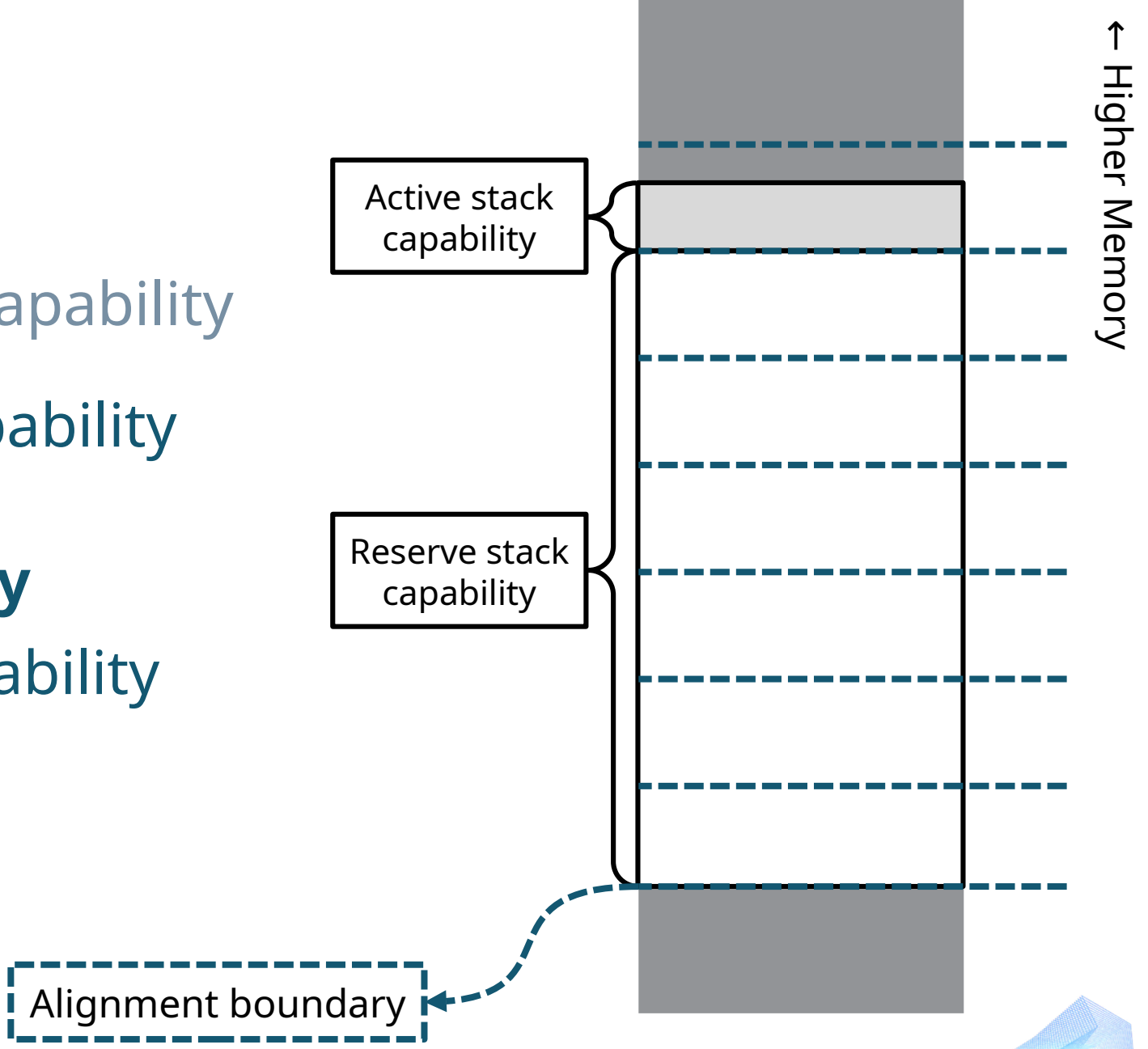
- Call frame boundary is badly aligned
- Large capability + bad alignment
  ⇒ rounding (capability compression)



Design Refinements

Example: Reserve Stack Capability

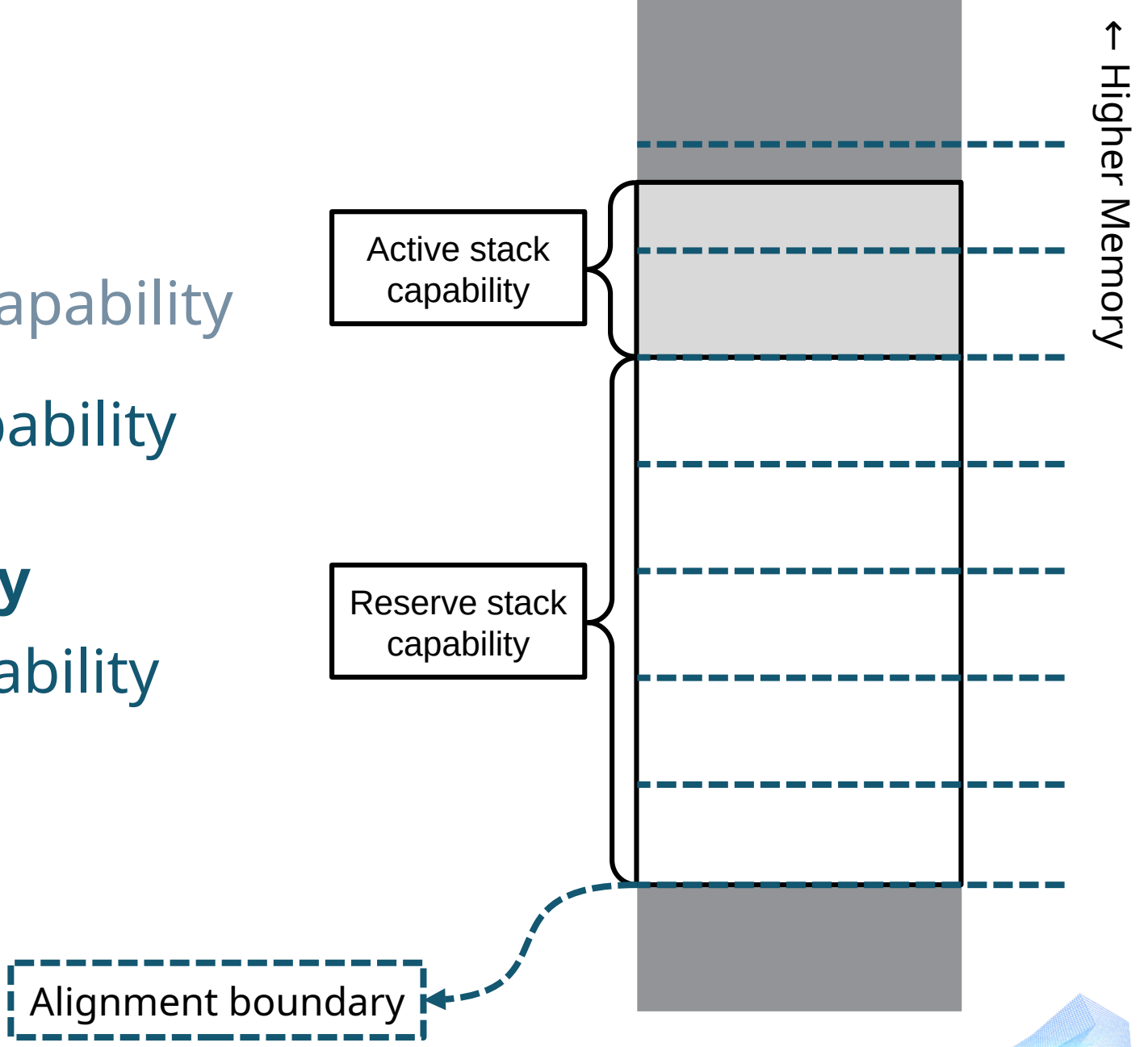
- Split authority of stack capability
  - Large, well aligned **reserve stack capability**
  - Smaller active stack capability



Design Refinements

Example: Reserve Stack Capability

- Split authority of stack capability
  - Large, well aligned **reserve stack capability**
  - Smaller active stack capability



LittleCHERI Contributions

- Design refinements
- **Implementation**
- Evaluation

Implementation

Contributions

- Runtime support code
- Calling convention added to CHERI-LLVM
  - Modular implementation

Implementation

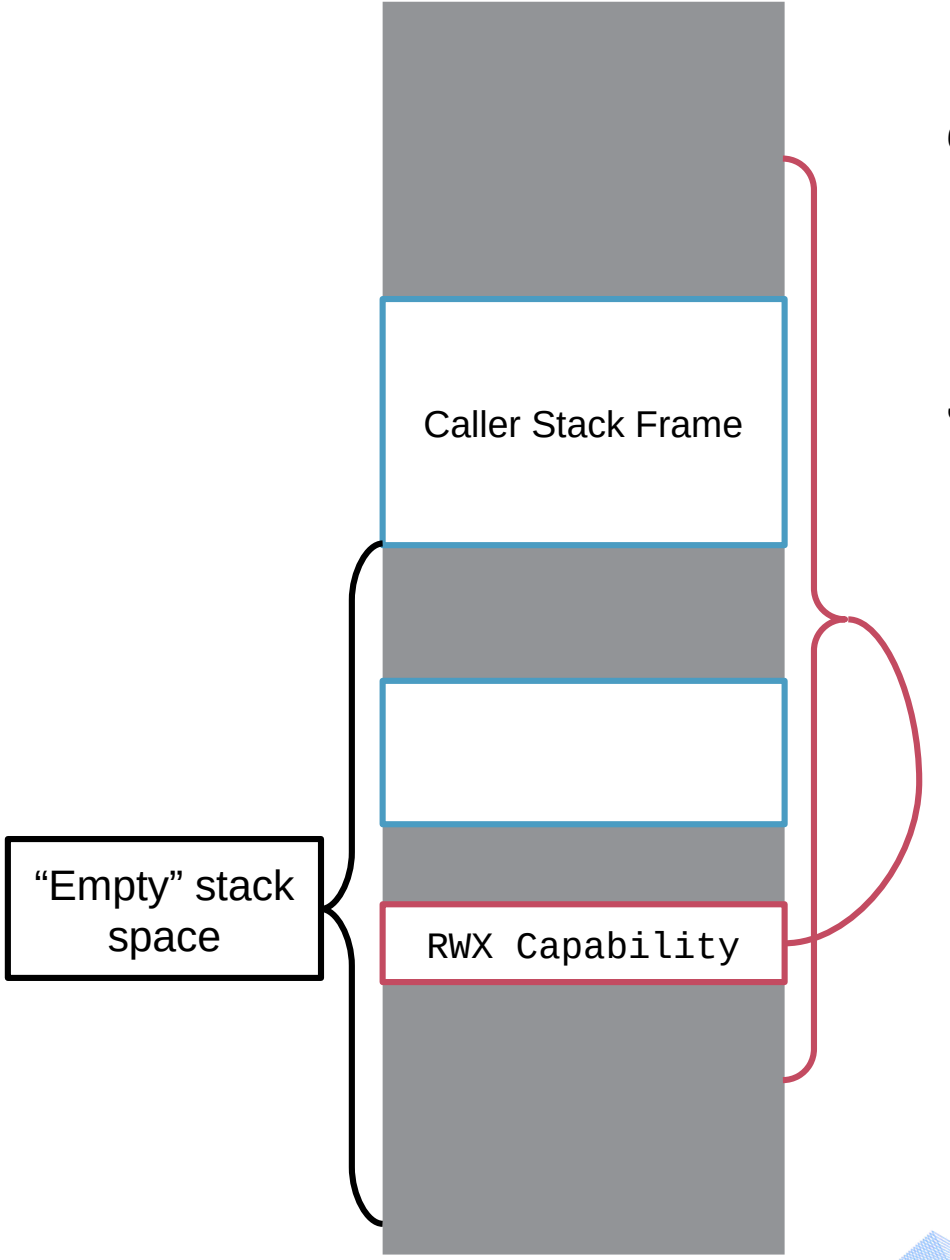
Calling Convention Components

- Stack revocation
- Stack shrinking
- Return encapsulation
- Register clearing
- Argument sanitization

Implementation

Calling Convention Components

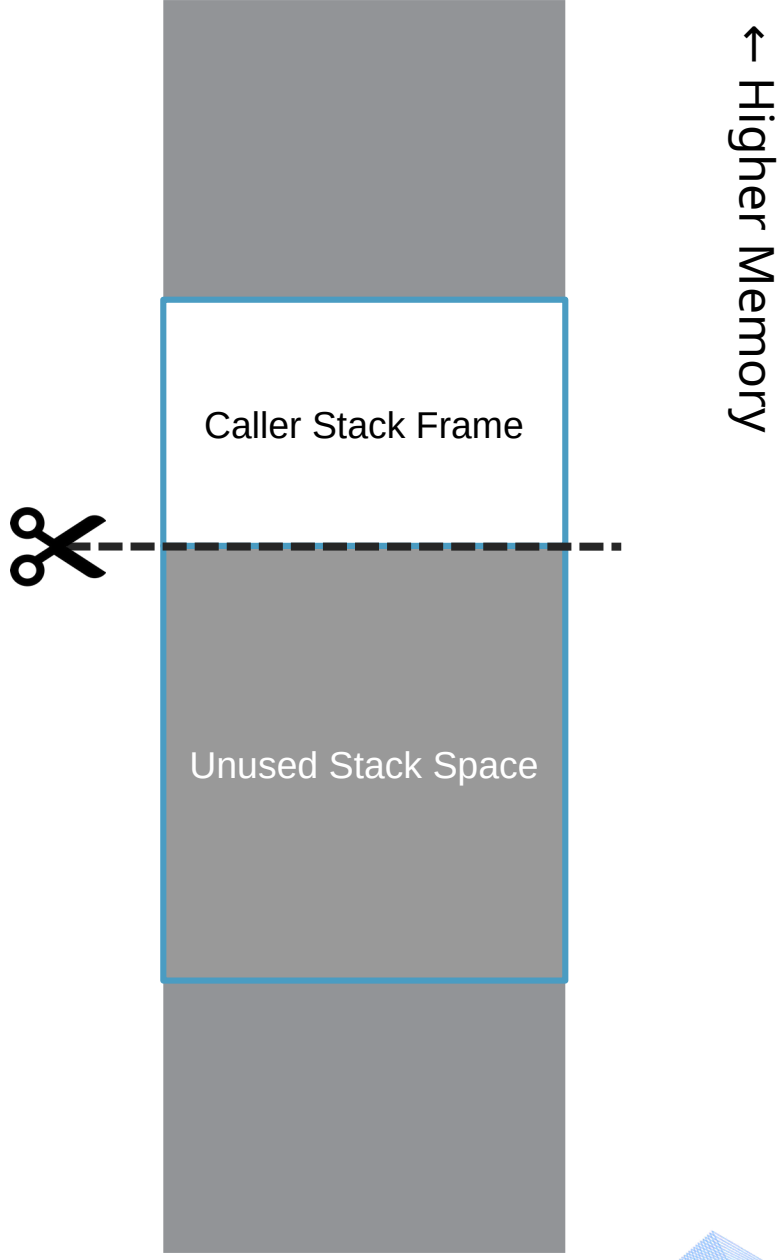
- **Stack revocation**
  - Enforce temporal stack safety
- Stack shrinking
- Return encapsulation
- Register clearing
- Argument sanitization



Implementation

Calling Convention Components

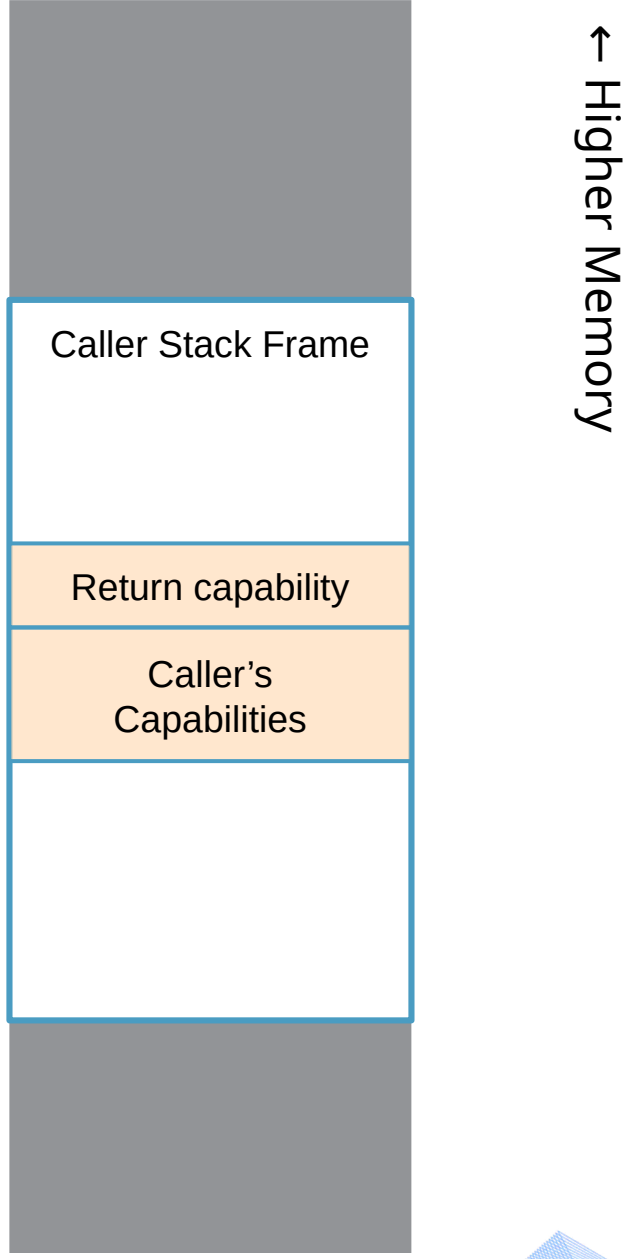
- Stack revocation
- **Stack shrinking**
  - Enabled by reserve stack cap.
- Return encapsulation
- Register clearing
- Argument sanitization



Implementation

Calling Convention Components

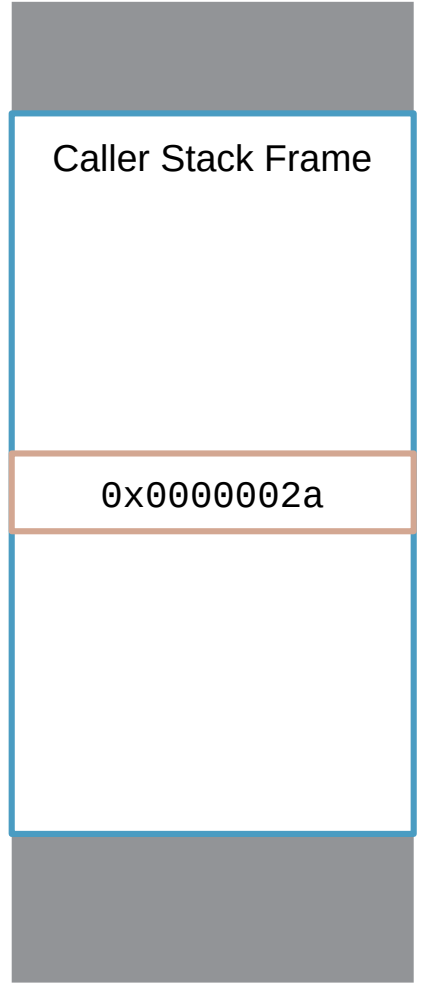
- Stack revocation
- Stack shrinking
- **Return encapsulation**
- Register clearing
- Argument sanitization



Implementation

Calling Convention Components

- Stack revocation
- Stack shrinking
- Return encapsulation
- **Register clearing**
- Argument sanitization



↑ Higher Memory

Register file

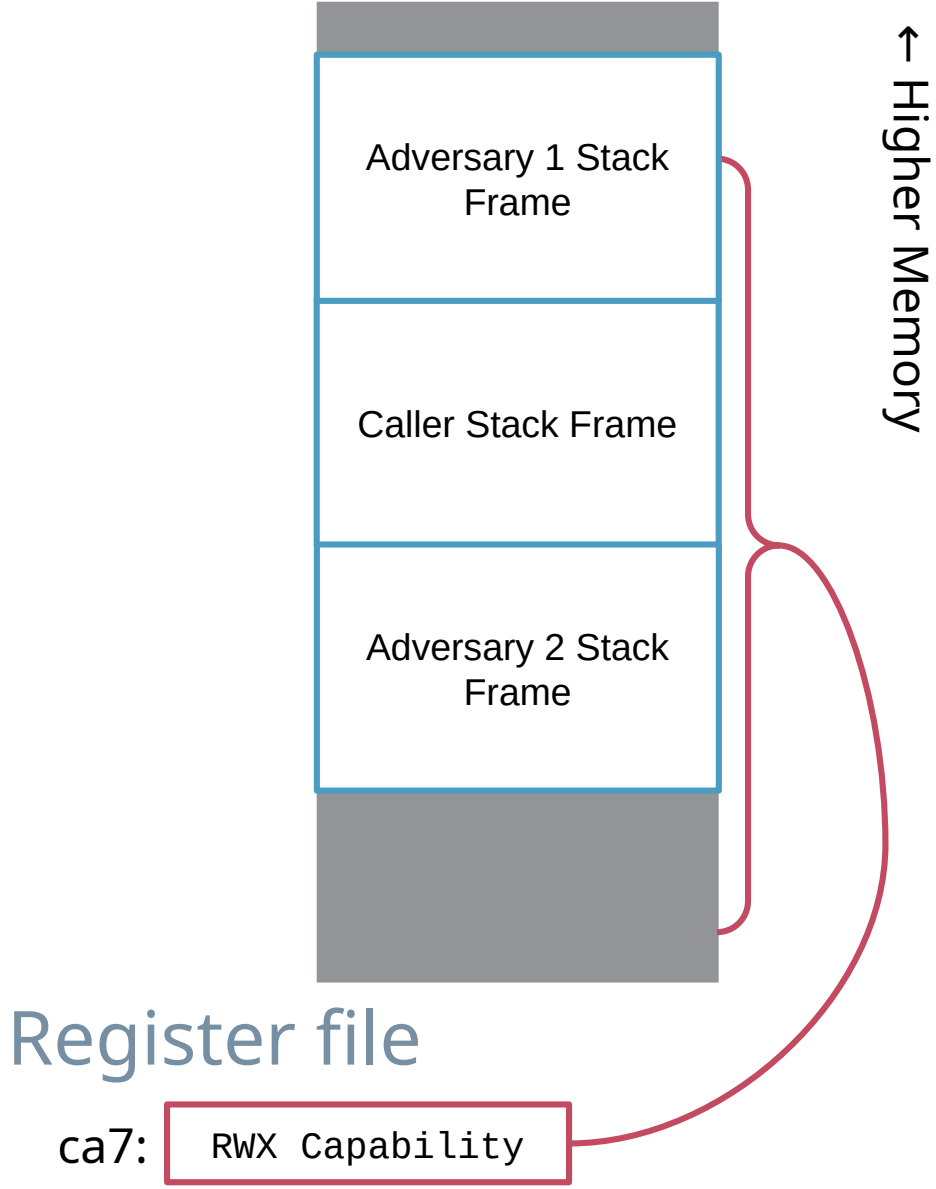
ca0: 0xdeadbeef

cs0: 0x00000000

Implementation

Calling Convention Components

- Stack revocation
- Stack shrinking
- Return encapsulation
- Register clearing
- **Argument sanitization**



LittleCHERI Contributions

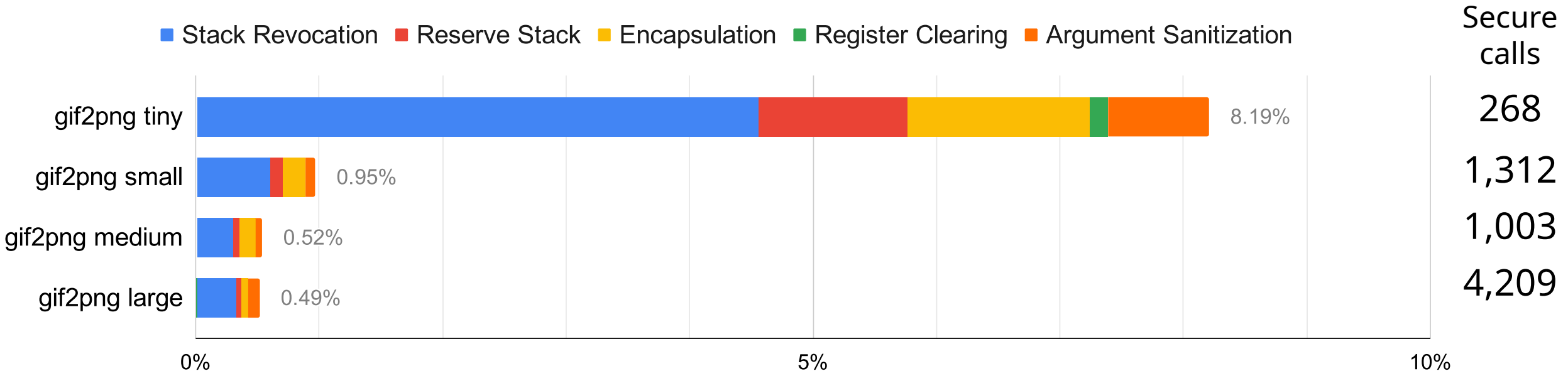
- Design refinements
- Implementation
- **Evaluation**

Evaluation

- Measure performance*, stack use and binary size
  *functional emulator coupled to gem5 memory model
- Run benchmarks:
  - CoreMark
  - Libc-bench
  - Random compartment micro-benchmark
  - **gif2png**

Evaluation

Case study: gif2png execution speed



Compartment depth: up to 8

Conclusion: LittleCHERI

- Enables ultra-fine-grained compartmentalization
- Has a CHERI-LLVM implementation
- Incurs only moderate overheads

Interested? Please get in touch!

- elias.storme@kuleuven.be
- https://people.cs.kuleuven.be/~elias.storme/littlecheri-preprint.pdf





THANKYOU

Contact contact@cheri-alliance.org

Web www.cheri-alliance.org

Located at semi | SEMICON EUROPA

CHERI™ and the CHERI logo are protected trademarks. Use of these marks is subject to the CHERI Alliance Trademark Policy – https://cheri-alliance.org/legal-and-policies/trademark/

This work © 2026 by CHERI Alliance is licensed under CC BY-SA 4.0 (Creative Commons Attribution-ShareAlike 4.0 International) – https://creativecommons.org/licenses/by-sa/4.0/