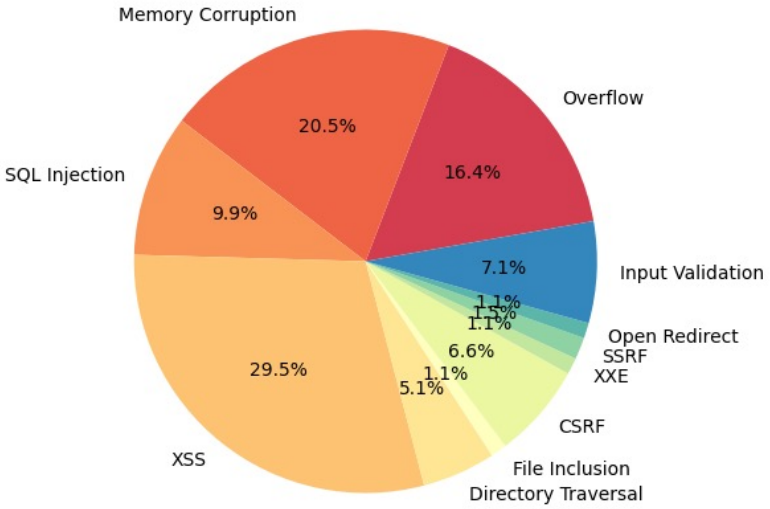
CCTAG: Configurable and Combinable Tagged Architecture

Zhanpeng Liu, Yi Rong, Chenyang Li, Wende Tan, Yuan Li, Xinhui Han, Songtao Yang, Chao Zhang

lydorazoe@gmail.com, hanxinhui@pku.edu.cn

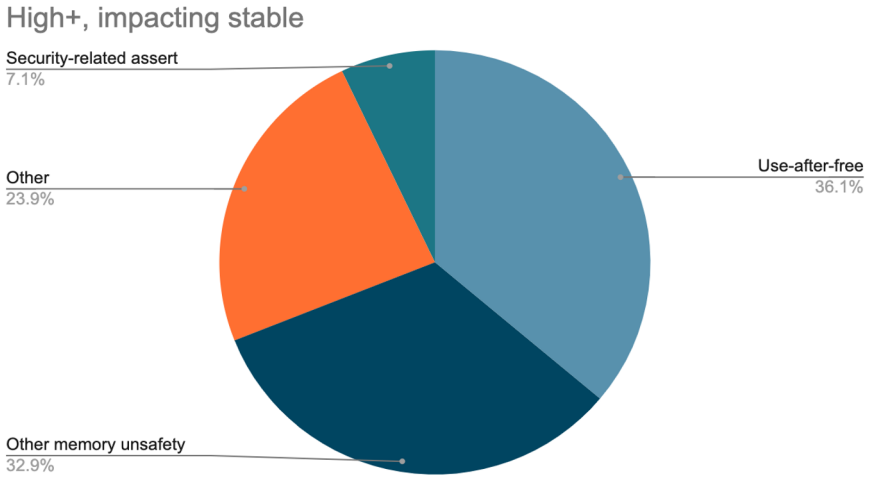
Memory Safety Vulnerabilities Are Prevalent

And they tend to be critically severe



Origin: https://www.cvedetails.com/vulnerabilities-by-types.php

CVE Distribution from CVEdetails
(From 2015 – 02/2025)



Origin: https://www.chromium.org/Home/chromium-security/memory-safety/

Image Credit: Yuan Li

Chrome's critical severity
security bugs' distribution
(#912, From 2015)

Defensive Mechanisms

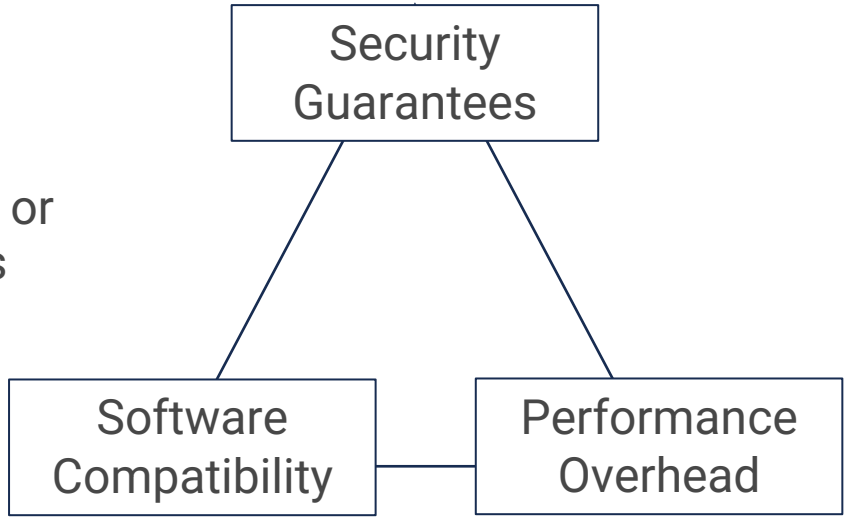
Can never find all vulnerabilities (Rice's Theorem)

Might not be able to fix vulnerabilities in time

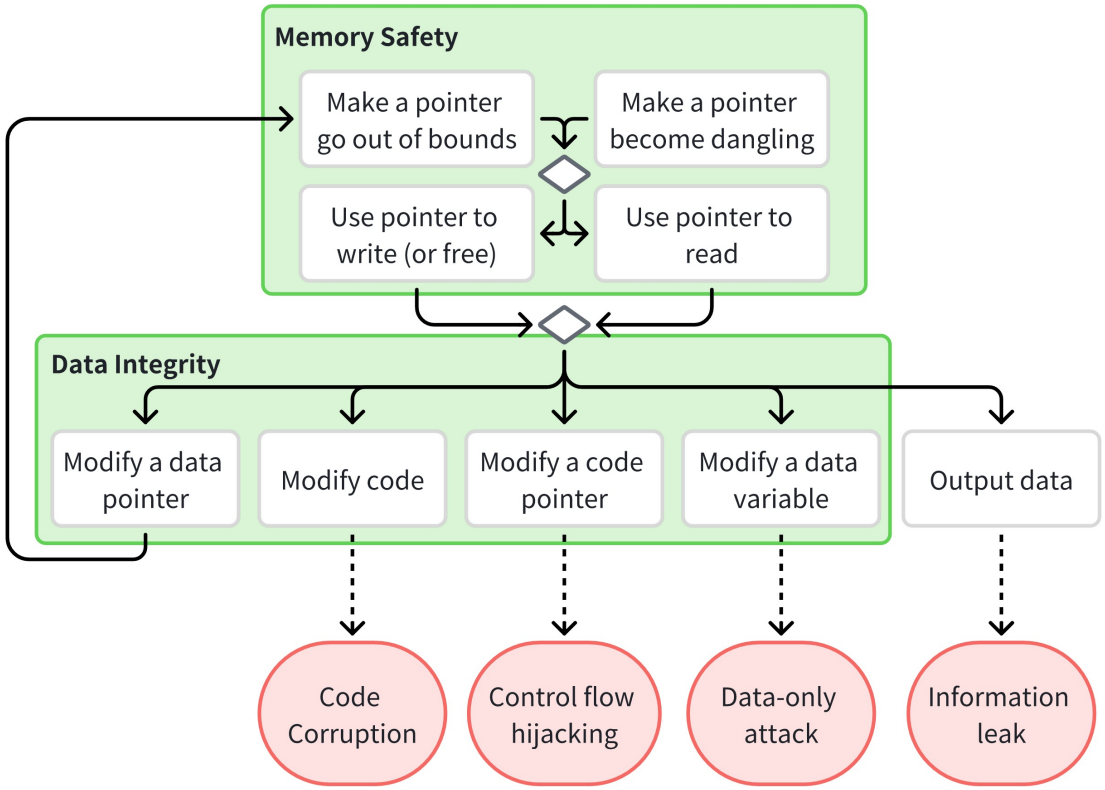
Need defensive mechanisms to mitigate **Unknown Vulnerabilities**

Have to trade-off

Most of the previous works only target specific vulnerability types or prevent certain exploitation steps



Attack Model from *SoK: Eternal War in Memory* [1]



Other defensive mechanisms:
NX, CFI, DFI, DIFT, etc.

Horizontal Integration:
provide comprehensive protection

Vertical Integration:
defense in depth; better trade-off

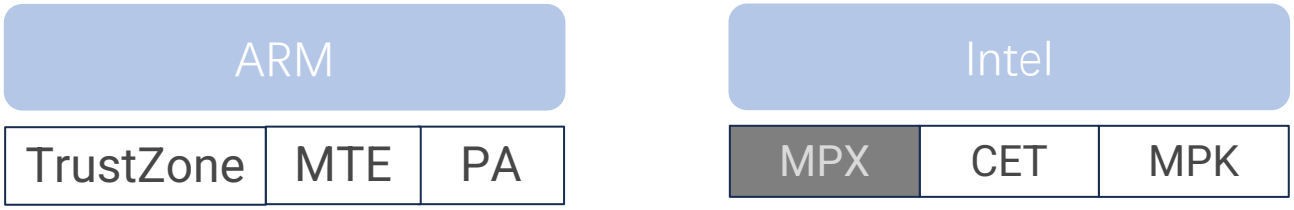
(Partial) Attack model demonstrating four exploiting types

[1] L. Szekeres, M. Payer, T. Wei, and D. Song, "**SoK: Eternal war in memory**," in Proceedings of the 2013 IEEE Symposium on Security and Privacy, ser. SP ' 13.

Challenges

Simply stacking defenses will accumulate performance penalties, especially for pure-software solutions.

Trends: **Hardware-Software Co-design**



Research works: HDFI [2], CHERI [3], PUMP [4, 5], etc.

Achieve both **configurability** and **combinability** with rigid hardware constraints

Maintain a **low-complexity** and more **practical** design

[2] C. Song, H. Moon, M. Alam, I. Yun, B. Lee, T. Kim, W. Lee, and Y. Paek, "**HDFI: Hardware-assisted data-flow isolation**," in 2016 IEEE Symposium on Security and Privacy (SP), 2016.

[3] R. N. Watson, J. Woodruff, P. G. Neumann, S. W. Moore, J. Anderson, D. Chisnall, N. Dave, B. Davis, K. Gudka, B. Laurie, S. J. Murdoch, R. Norton, M. Roe, S. Son, and M. Vadera, "**CHERI: A hybrid 15 capability-system architecture for scalable software compartmentalization**," in 2015 IEEE Symposium on Security and Privacy, 2015.

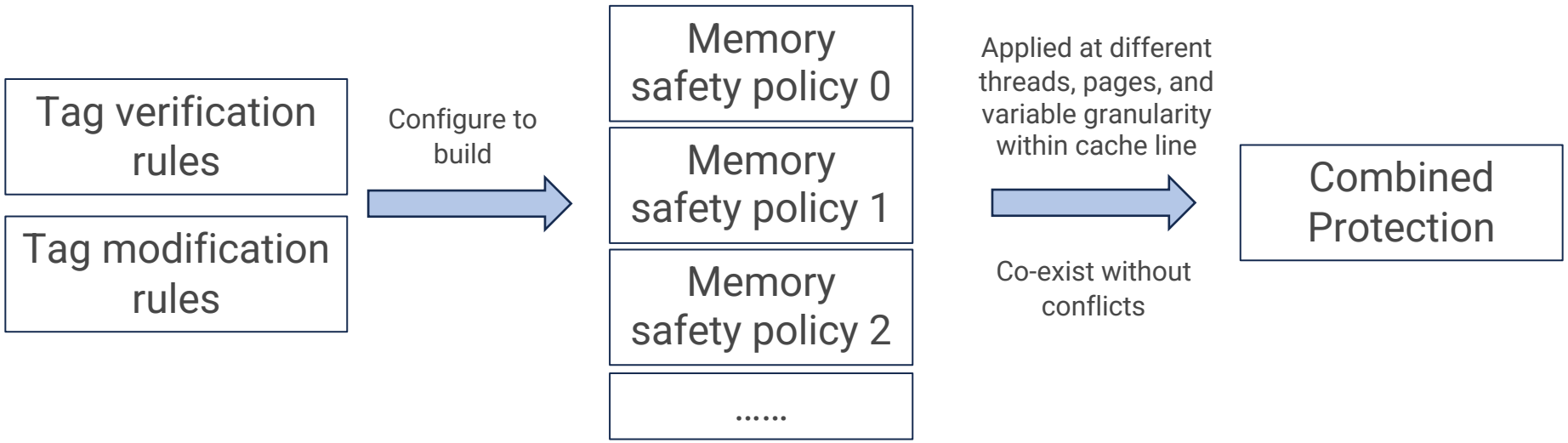
[4] U. Dhawan, C. Hritcu, R. Rubin, N. Vasilakis, S. Chiricescu, J. M. Smith, T. F. Knight, B. C. Pierce, and A. DeHon, "**Architectural support for software-defined metadata processing**," in Proceedings of the Twentieth International Conference on Architectural Support for Programming Languages and Operating Systems, ser. ASPLOS '15.

[5] U. Dhawan, N. Vasilakis, R. Rubin, S. Chiricescu, J. M. Smith, T. F. Knight, B. C. Pierce, and A. DeHon, "**Pump: A programmable unit for metadata processing**," in Proceedings of the Third Workshop on Hardware and Architectural Support for Security and Privacy, ser. HASP '14.

Our Proposal: CCTAG

Tagged Architecture

Add security metadata to registers and memory granula



Configurable: Towards Versatility

TABLE III: Supported tag checking & updating rules.

Constant	Description
MT_CHECK_NONE	On load/store, do not check
MT_CHECK_EQUAL	On load/store, check if the memory tag matches with tag in pointer
MT_CHECK_UNCOND	On load/store, check if the memory tag is 1(or 0), subject to MT_CHECK_VAL
MT_CHECK_COND	On load/store, if the tag in pointer is 1, check if the memory tag is 1 (or 0), subject to L_MT_CHECK_VAL
L_PROP	On load, propagate the memory tag to the register tag
S_NONE	On store, do not change the memory tag
S_SET	On store, set the memory tag to 1
S_UNSET	On store, set the memory tag to 0
S_PROP	On store, propagate the register tag to the memory tag

Fine-grained Permissions

Memory Coloring

Data Integrity

Information Flow Tracking

Taint Analysis

Configurable: Towards Versatility

Adaptable to different granularities

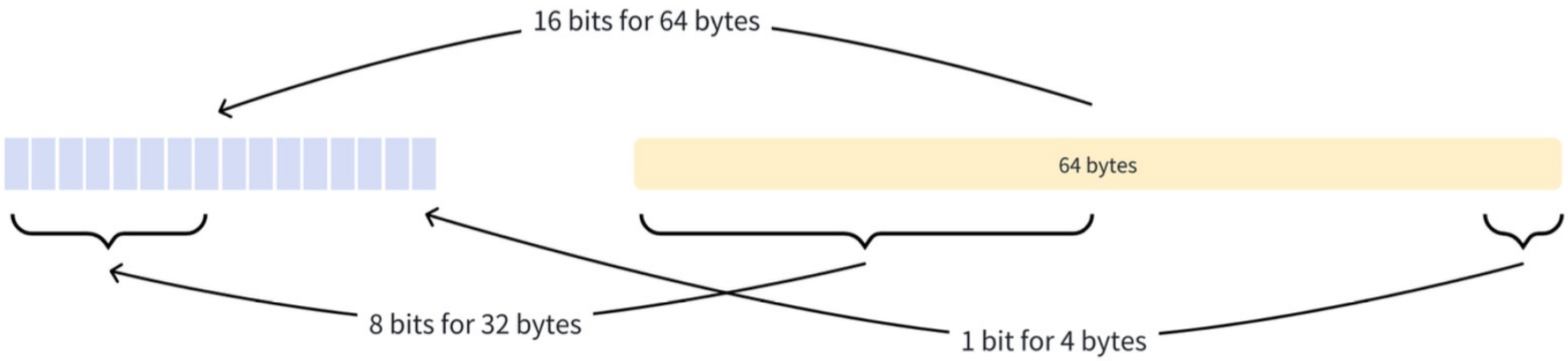


Figure 1: Memory tag partition.

Combinable: Avoid Conflicts

Policy-Centric Mask Design

Instead of per tag bit configuration

Tag Policy: **policy mask**, **granularity**, and **rules** for tag checking and updating

Different pages can enforce different policies

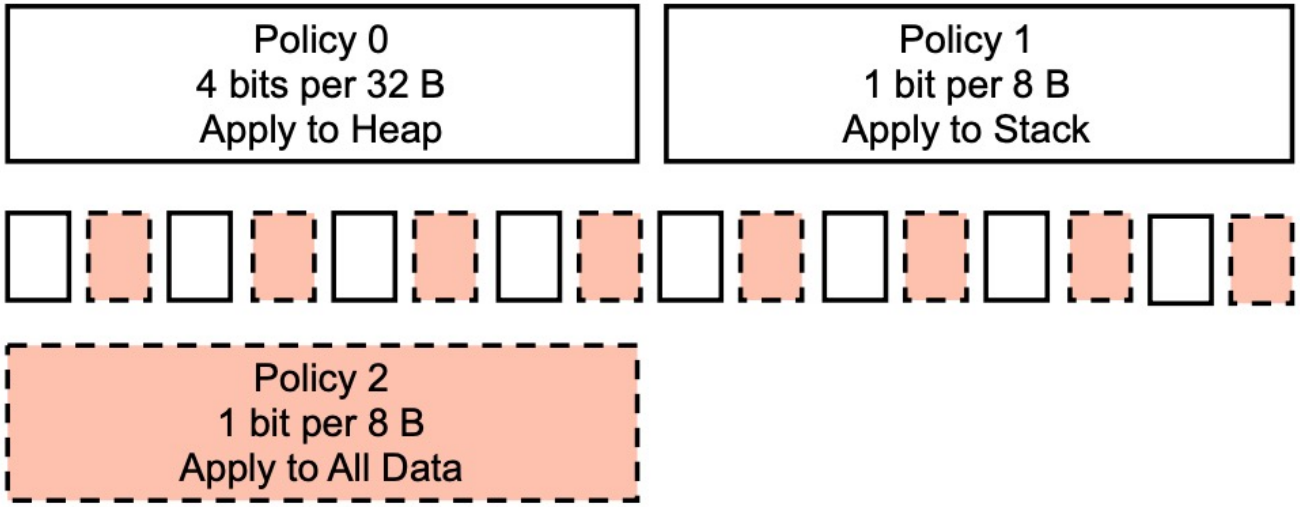
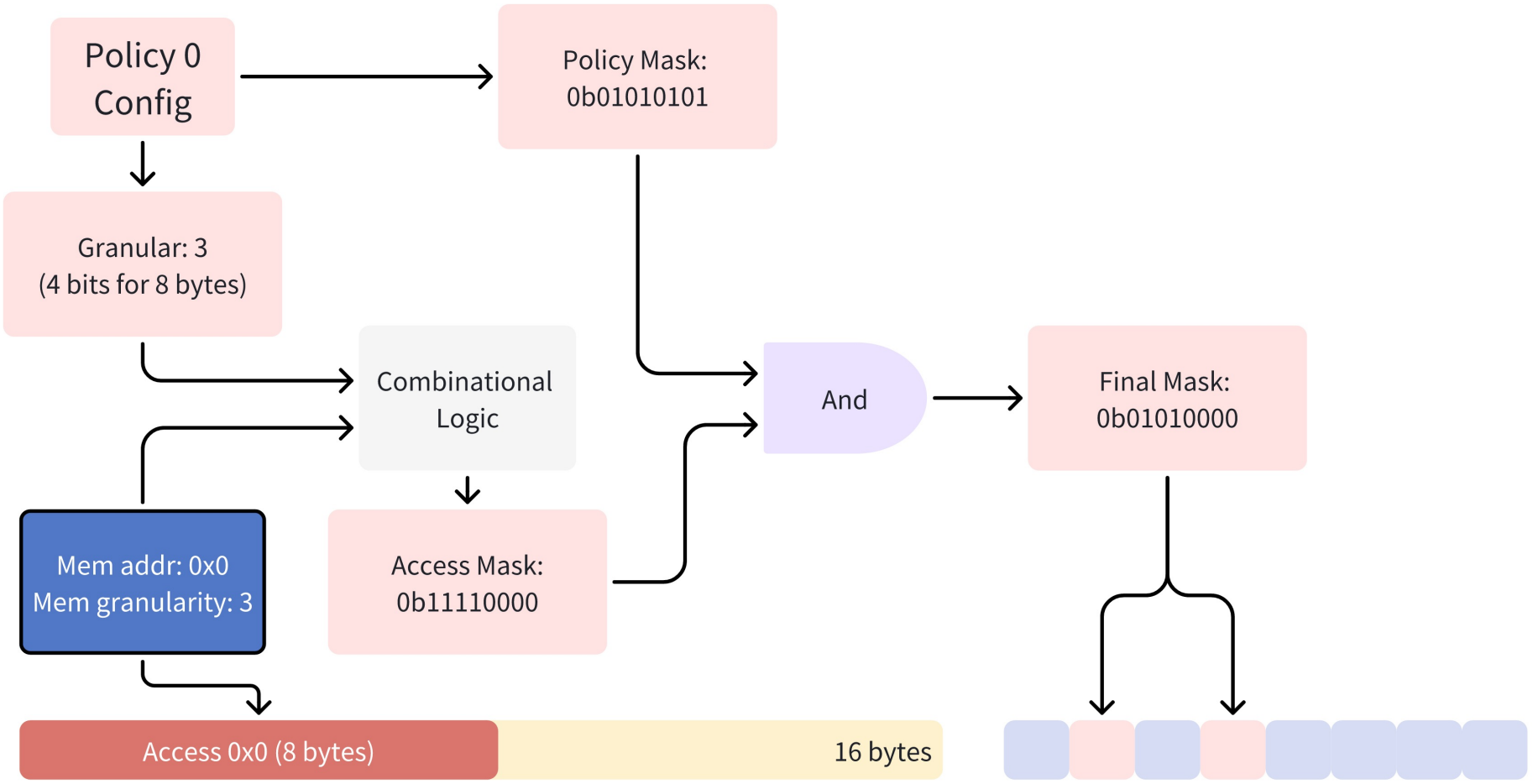


Figure 3: Distinct policies coexist without interference.

Combinable: Avoid Conflicts



The checking and updating values of different policies can be **masked** and then **ORed** together

Prototype Implementation

Based on Rocket Core (RISC-V ISA)

16 bits tag / 64 bytes
(the same **ratio** as ARM MTE)

Support 4 policies

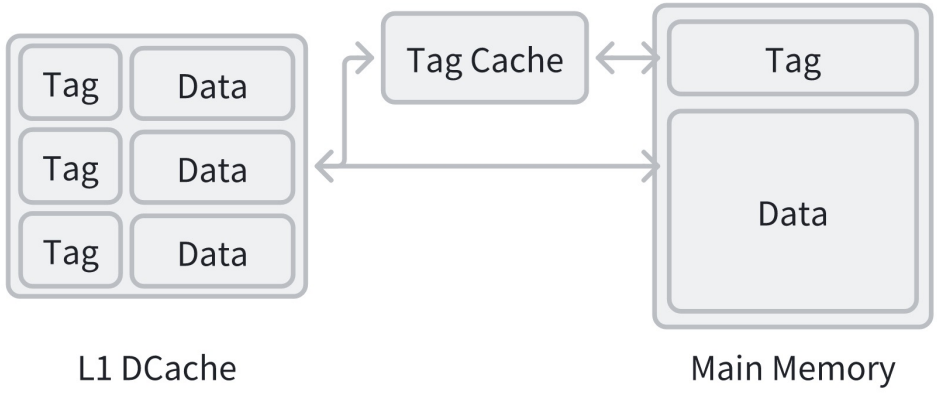
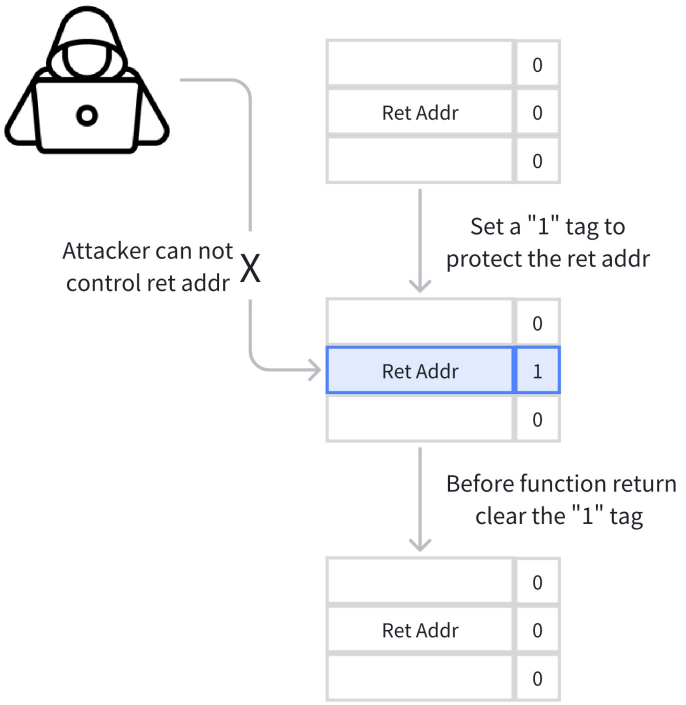


TABLE IV: Hardware resource cost of the baseline and CCTAG when synthesized on an FPGA.

	RISC-V Rocket Cores					Whole Systems					
	#LUT	%	#FF	%	#BRAM	#LUT	%	#FF	%	#BRAM	Worst Neg Slack (ns)
baseline	34,039	_	14,939	_	20	57,298	_	48,448	_	115	0.50
CCTAG	36,342	+6.77%	16,137	+8.02%	30	59,616	+4.05%	49,701	+2.59%	125	0.53

The required **hardware resources are minimal (~8%)**
compared to CHERI (> 50%) and PUMP (> 100%)

Ported Defense Applications



Listing 1: Return address protection.

```
1    1090c: 01 11          addi    sp, sp, -32
2    1090e: 06 ec          sd      ra, 24(sp)
3    10910: 22 e8          sd      s0, 16(sp)
4    10912: 00 10          addi    s0, sp, 32
5    10914: 09 45          li      a0, 2
6    10916: 2b 6c a1 00    mtsd    a0, 24(sp)
7    ...
8    10950: 09 45          li      a0, 2
9    10952: 2b 78 a1 00    mtcd    a0, 24(sp)
10   10956: e2 60          ld      ra, 24(sp)
11   10958: 42 64          ld      s0, 16(sp)
12   1095a: 05 61          addi    sp, sp, 32
13   1095c: 82 80          ret
```

Ported three other defense applications

- Function Pointer & VTable Pointer Integrity
- ARM MTE-like Heap Memory Coloring
- Dangling Pointer Sweeping

Verified with real world CVEs

Performance Evaluation

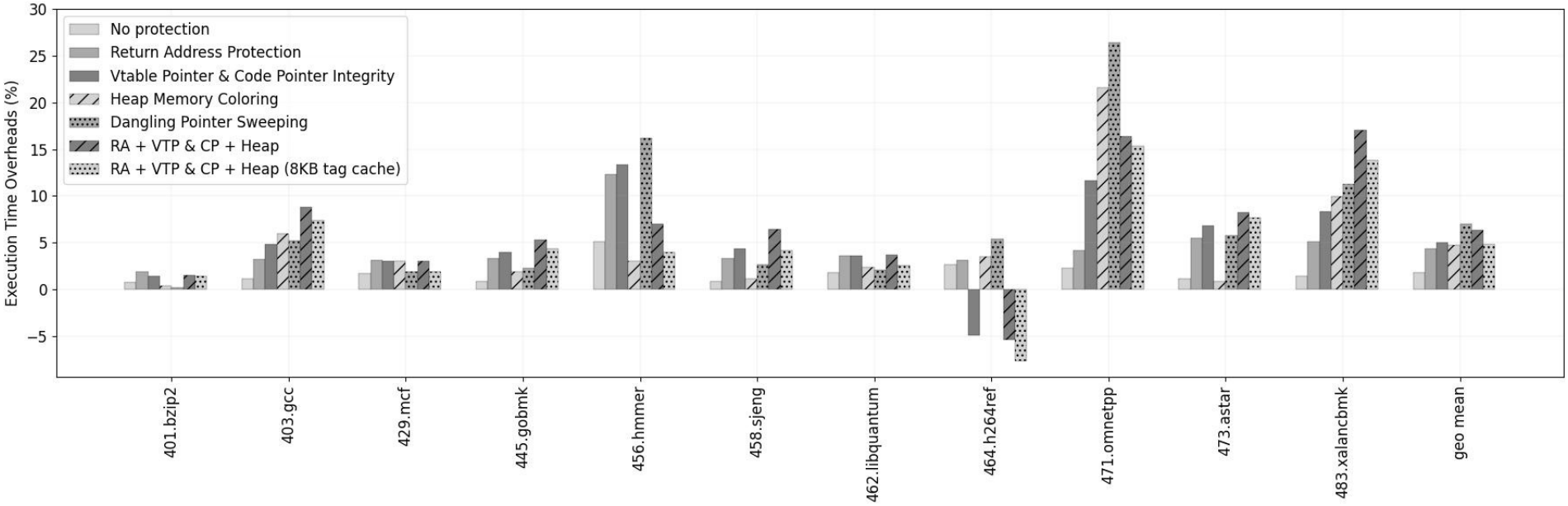


Fig. 4: Relative runtime overhead of CCTAG on SPEC CINT2006

Overhead of integrated protection (6.68%) < the sum of the individuals (13.98%)

Drops to ~4.8% with 8KB cache, lower than PUMP (~8.8%) and CHERI (~30.5%)

7.93% on SPEC CINT2017

Conclusion

A Configurable and Combinable Tagged Architecture

Support various memory safety policies

and allow them to co-exist without conflicts

Evaluated with FPGA prototype and four ported defense applications

Lightweight and of low complexity, comparable to ARM MTE

yet much more versatile and powerful

Slightly weaker capabilities compared to PUMP and CHERI, but

Easier and more cost-effective to implement in hardware

Fully compatible with existing software

More efficient at runtime

Thanks for listening!