TagGuard: End-to-End Hardware DIFT for Bare-Metal RISC-V SoCs

Ali Ait Hassou, William Pensec, Florent Bruguier, Pascal Benoit
LIRMM, Univ. Montpellier, CNRS, France

1. Motivation & Objective

Threat

Memory corruption is the leading attack vector in modern software: **≈70%** of CVEs assigned annually by Microsoft and Google are memory safety issues (CISA, 2023). Bare-metal IoT devices are especially exposed with no OS, MMU, or ASLR: any successful exploit leads to immediate, unmediated system compromise.

Objective

We aim to design a hardware protection mechanism that:

- tracks untrusted data flows **end-to-end** across the SoC: from peripheral input to execution;
- detects **known and unknown** memory corruption exploits (CWE-class and zero-day) without relying on vulnerability signatures.

2. TagGuard Hardware Architecture

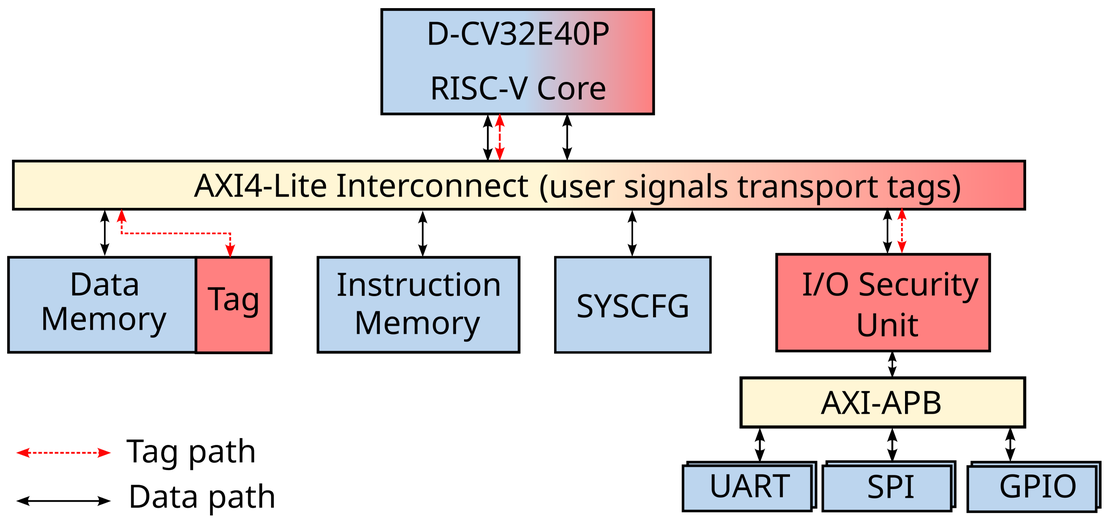

Fig. 1: End-to-end tag path in SoC (AXI4-Lite `user` signals).

- **Core**: in-CV32E40P core tag propagation and checking.
- **Memory**: per-byte tags stored alongside data.
- **Interconnect**: tags ride AXI4-Lite transactions.
- **I/O**: automatic tag creation on peripheral inputs.

3. D-CV32E40P Core Architecture

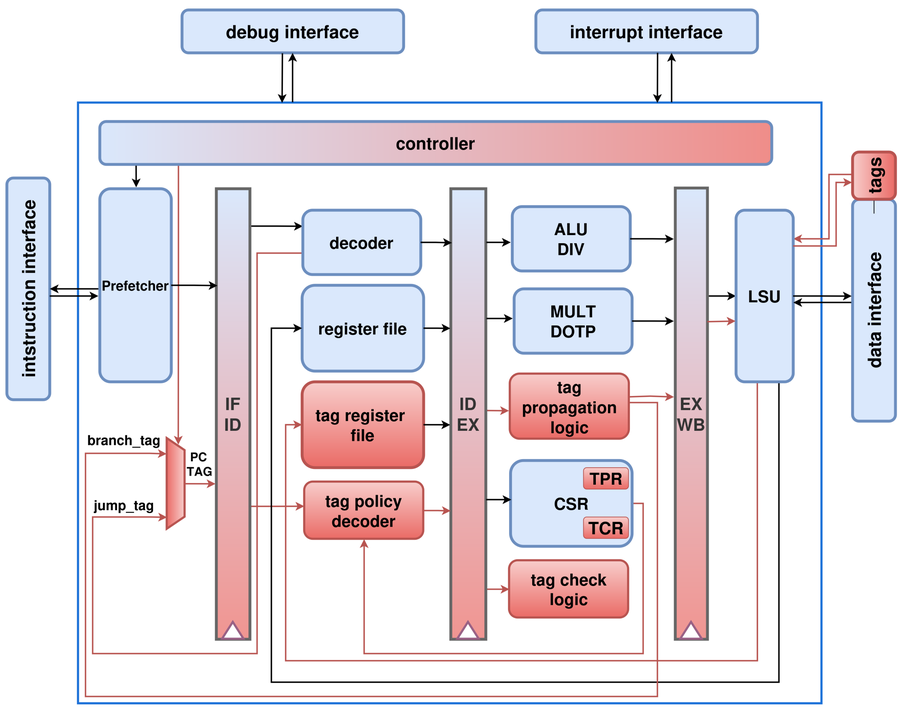

Fig. 2: D-CV32E40P pipeline with tag path (red).

4. Tag Propagation & Checking Policy (TPR/TCR)

TPR (`0x0000A8A2`)

Field	Bits	Value	Mode
Arith	1:0	10	OR
Branch	3:2	00	OLD
Jump	5:4	10	OR
Shift	7:6	10	OR
Compare	9:8	00	OLD
Logical	11:10	10	OR
L/S	13:12	10	OR

TCR (`0x00200018`)

Field	Bits	Value	State
Execute (PC)	21	1	ON
Branch S2	4	1	ON
Branch S1	3	1	ON
Arith	2:0	000	OFF
Jump	7:5	000	OFF
Shift	10:8	000	OFF
Compare	13:11	000	OFF
Logical	16:14	000	OFF
L/S	20:17	0000	OFF

5. TagGuard Software Architecture

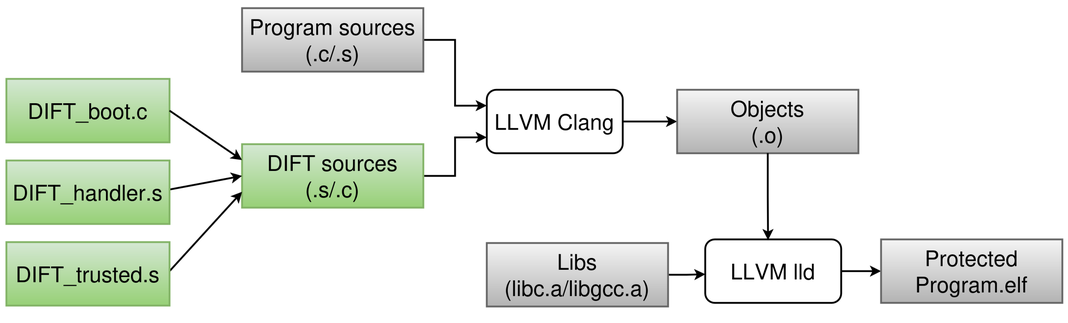

Fig. 3: TagGuard toolchain and startup flow.

6. Example: Buffer Overflow Attack Walkthrough

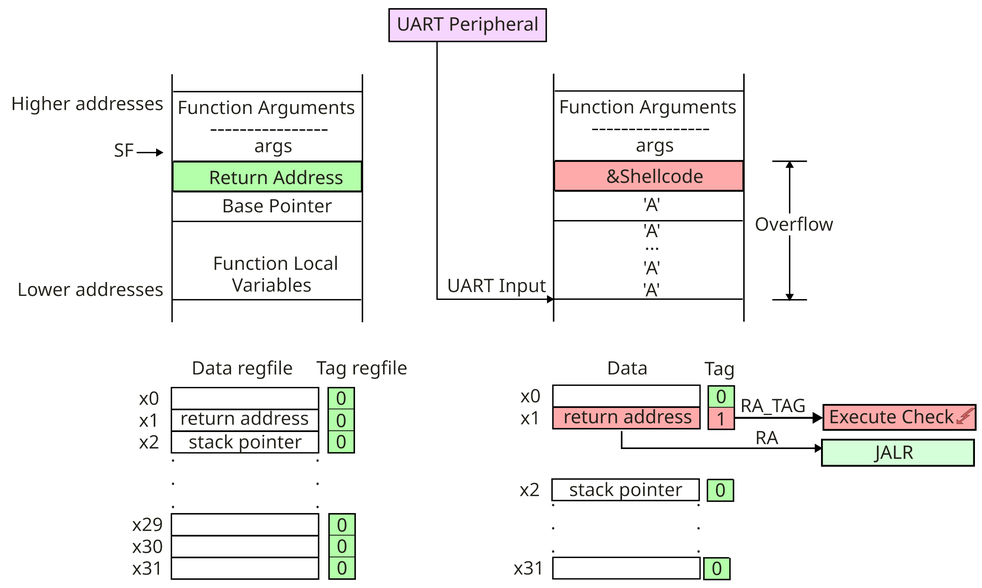

Fig. 5: CWE-121 stack overflow detection by TagGuard.

7. Experimental Results

Hardware overhead (Xilinx Artix-7 XC7A200T)

Design	LUTs	Regs	BRAM	F_{max}
ADAM SoC (base)	24 437	13 388	257	50 MHz
ADAM SoC (TagGuard)	25 054	13 699	306	50 MHz
Δ	**+2.52%**	**+2.32%**	+19.1%	**0%**

Security evaluation (16 exploits / 15 CWEs, 0 FN, 0 FP)

#	Attack	CWE	Exploit target	Detection
Control-flow hijacking				
1		120	BSS buffer overflow	Execute check
2		121	Stack RA overwrite (ROP)	Execute check
3		787	Function-pointer overwrite	Execute check
Memory corruption				
4		119	OOB index write	Branch check
5		121	Stack variable corruption	Branch check
6		122	Heap overflow	Branch check
7		125	OOB read	Branch check
8		134	Format string %n	Branch check
9		190	Integer overflow bypass	Branch check
10		193	Off-by-one error	Branch check
11		415	Double free	Branch check
12		416	Use-after-free	Branch check
13		476	Null dereference	Branch check
14		676	`strcpy` overflow	Branch check
15		824	Uninitialized pointer use	Branch check
MMIO register corruption				
16		284	Peripheral control-reg write	MMIO guard

8. Conclusion & Future Work

- TagGuard provides end-to-end protection for bare-metal RISC-V with low overhead (+2.52% LUTs, +2.32% regs, no frequency loss).
- 16 exploits across 15 distinct CWEs detected with 0 false negatives and 0 false positives.
- Future work: **COTS binary protection** and **demand-paged tag memory** to reduce BRAM overhead.

9. Acknowledgement

This work was supported by the French National Research Agency (ANR) through the government grant SCREAM (ANR-24-CE39-0199-02).

10. References

[1] F. Paiva Alencar *et al.*, "ADAM: Adaptive microcontroller platform for edge ai systems," in *Proc. 36th Int. Workshop on Rapid System Prototyping (RSP)*, 2026, pp. 29–35.

[2] G. E. Suh *et al.*, "Secure program execution via dynamic information flow tracking," in *ASPLOS*, 2004.

[3] C. Palmiero *et al.*, "Design and implementation of a DIFT architecture to secure a RISC-V core for IoT," in *IEEE HPEC*, 2018.