HardBlare: Towards a hardware-assisted information flow tracking ecosystem for ARM processors

Muhammad Abdul Wahab [α], Pascal Cotret [α], Mounir Nasr Allah [β], Guillaume Hiet [β], Guy Gogniat [γ], Vianney Lapôtre [γ]

[α] SCEE / IETR, CentraleSupélec, Cesson-Sévigné - FRANCE
[β] CIDRE / IRISA / INRIA, CentraleSupélec, Cesson-Sévigné - FRANCE
[γ] Lab-STICC, University of South Brittany, Lorient - FRANCE

Introduction

HardBlare proposes a software/hardware codesign methodology to ensure that security properties are preserved all along the execution of the system but also during files storage. The general context is to address **Dynamic Information Flow Tracking (DIFT)** that generally consists in attaching marks (also known as tags) to denote the type of information that are saved or generated within the system.

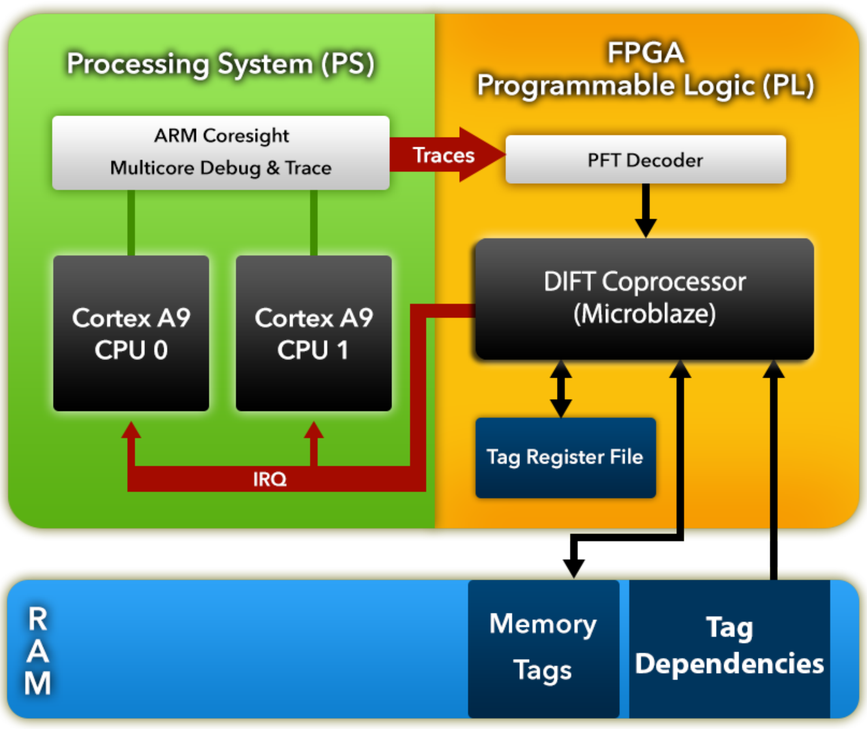
Let's suppose that "print" function, which sends output stream to stdout, is public and the tag of a variable `x` is underlined variable $\underline{x}$.

Example code	Tag initialization	Tag propagation	Tag check
`p = 3;`	$\underline{p} \leftarrow$ public		
`s = 42;`	$\underline{s} \leftarrow$ secret		
`x = p + s;`		$\underline{x} \leftarrow \underline{p} + \underline{s} = \underline{s}$	
`print(x);`			if ($\underline{x}$!= public) `raise interruption`

State of the art

		Advantages	Disadvantages
Software		Flexible security policies Multiple attacks detected	Overhead (from 300% to 3700%)
Hardware		Low overhead (<10%) Invasive modifications	Fixed Security policies
Hybrid	**In-core DIFT [1],[2]**	Low overhead (<10%) Few security policies	Invasive modifications
	Dedicated CPU for DIFT [3]	Low overhead (<10%) Few modifications to CPU	Wasting resources Energy consumption (x 2)
	Dedicated DIFT Coprocessor [4],[5]	Flexible security policies Low overhead (<10%) CPU not modified	Communication between CPU and DIFT Coprocessor

Overall Architecture



Definitions

- **Tag dependencies** block contains annotations loaded when the program is launched
- **Memory tags** block contains tags related to `Load/Store` memory addresses
- **Tag register file** contains tags related to CPU registers

DIFT step-by-step

- ARM CoreSight Components export trace, which can be done for both CPUs, towards PL in PFT **(Program Flow Trace)** protocol
- **PFT Decoder** decodes trace in usable format
- Using decoded trace, DIFT Coprocessor reads **tag dependencies** block
- DIFT Coprocessor
  - looks for the tags either in **memory tags** or **tag register file**
  - computes tags depending on propagation rules
  - updates corresponding tags either in **memory tags** or **tag register file**
  - checks for security policy violation and raise an interruption

ARM Cortex-A9 Trace mode: Coresight components

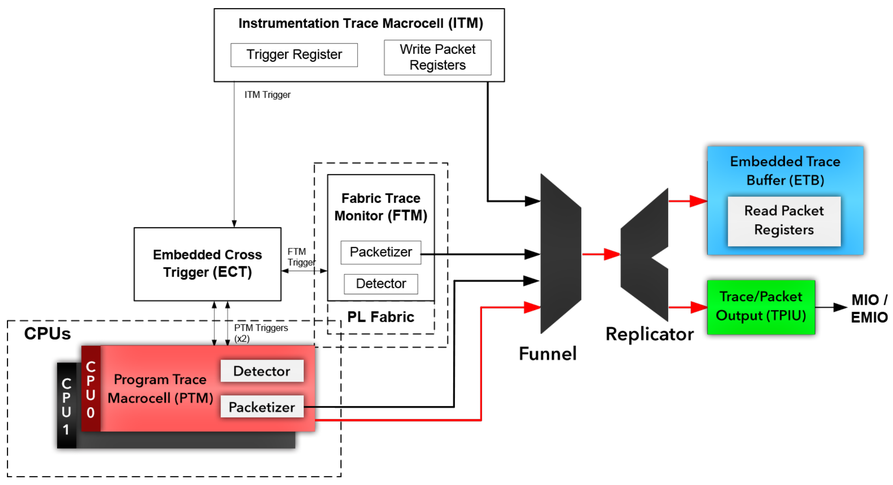


Figure: Coresight components in Zynq SoC (taken from Zynq Technical Reference Manual)

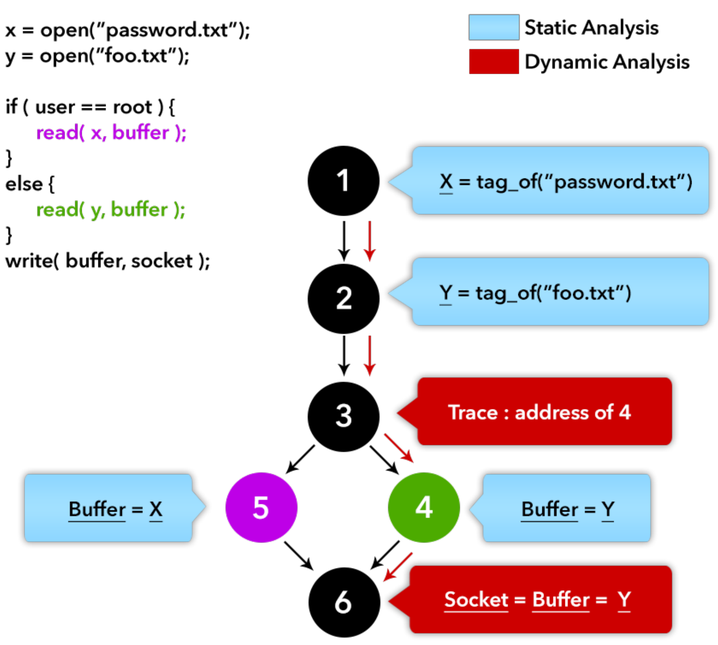
MiBench program	without Coresight components	with Coresight components
Choleski	290.418	290.469
LU	2371.905	2372.073

Table: Average time overhead (in μs) with and without Coresight components

- A program is a sequence of instructions that can be either conditional or non-conditional
- Coresight components allow to retrieve information on conditional instructions during execution without time overhead
- During compilation with LLVM, static analysis is done to obtain tag dependencies i.e. information on how to propagate tags
- GNU Linker needs to be modified to obtain tag dependencies for dynamically linked libraries

	Slice LUTs	Slice Registers	Slice	BRAM Tile
Total Design	7002	6930	2908 (21.86%)	48
Total Available	**53200**	**106400**	**13300**	**140**

Table: Post-implementation design size for Zedboard (Zynq Z-7020)



Some References

[1] G. Venkataramani, I. Doudalis, Y. Solihin, and M. Prvulovic, "Flexitaint: A programmable accelerator for dynamic taint propagation," in *In 14th International Symposium on HighPerformance Computer Architecture (HPCA-14)*, 2008.

[2] M. Dalton, H. Kannan, and C. Kozyrakis, "Raksha: A flexible information flow architecture for software security," in *In International Symposium on Computer Architecture (ISCA)*, 2007.

[3] V. Nagarajan, H.-S. Kim, Y. Wu, and R. Gupta, "Dynamic information tracking on multicores," in *12th Workshop on the Interaction between Compilers and Computer Architecture (INTERACT)*, Feb 2008.

[4] H. Kannan, M. Dalton, and C. Kozyrakis, "Decoupling dynamic information flow tracking with a dedicated coprocessor," in *Proceedings of the 2009 IEEE/IFIP International Conference on Dependable Systems and Networks, DSN 2009, Estoril, Lisbon, Portugal, June 29 - July 2, 2009*, pp. 105–114, 2009.

[5] I. Heo, M. Kim, Y. Lee, C. Choi, J. Lee, B. B. Kang, and Y. Paek, "Implementing an application-specific instruction-set processor for system-level dynamic program analysis engines," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 20, no. 4, p. 53, 2015.

Main Contributions at a Glance

- Hardware-assisted DIFT system with limited time overheads.
- Approach based on a non-modified CPU with a standard Linux and generic binaries ⇒ Could be implemented by industrial partners in medium-term.
- Hardened with hardware security mechanisms: trusted coprocessor storage and bus protection in terms of confidentiality/integrity.
- Contributions on software-related issues as well (static/dynamic IFC analysis, i.e. hybrid analysis).
- Perspectives on runtime reconfiguration and multicore/manycore systems.
- Perspectives on DIFT with other embedded processors (Intel, Texas Instruments)

http://www.hardblare.cominlabs.ueb.eu/ {muhammadabdul.wahab,mounir.nasrallah}@centralesupelec.fr