Enhanced Processor Defence Against Physical and Software Threats by Securing DIFT Against Fault Injection Attacks

PhD Dissertation Defense

William PENSEC

Université Bretagne Sud, UMR 6285, Lab-STICC, Lorient, France

December 19, 2024

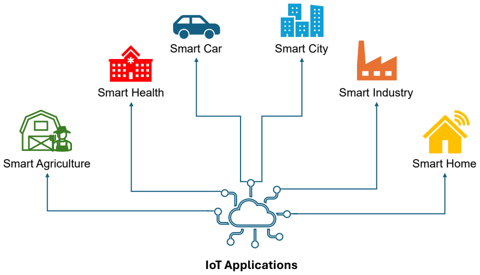
Outline

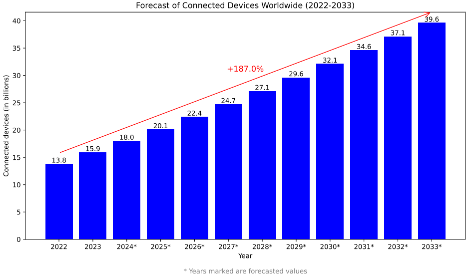
Outline

Context: Embedded Systems and IoT

Internet of Things (IoT)

- Wide range of application
- Fast growing market with exponential usage
- Rely on sensors depending on their use
- Collect and share data
- Manipulation of critical data
- Increasingly vulnerable to multiple threats





Threats

- Software threats: malwares, memory overflow attacks, SQL injection, etc
- Network threats: DDoS, Man-In-The-Middle, jamming, etc
- Hardware threats: physical attacks such as reverse engineering, Side-Channel Attacks (SCA), Fault Injection Attacks (FIA)

Motivations: IoT Under Threats

Threats

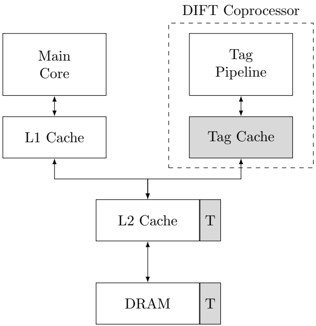
- **Software threats**: malwares, memory overflow attacks, SQL injection, etc
- Network threats: DDoS, man-in-the-middle, jamming, etc
- **Hardware threats**: physical attacks such as reverse engineering, Side-Channel Attacks (SCA), Fault Injection Attacks (FIA)

Software threats: Information Flow Tracking

- Security mechanism
- Protection against software attacks (e.g.: *buffer overflow*, *format string*, *SQL injections*, ...) [1, 2]
- Static or Dynamic
- Software, Hardware or Hybrid
- Hardware DIFT: off-core, off-loading core, in-core

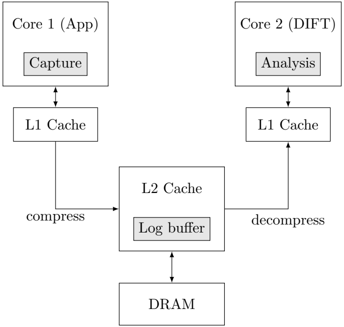
Software threats: Information Flow Tracking

- Security mechanism
- Protection against software attacks (e.g.: *buffer overflow*, *format string*, *SQL injections*, . . .) [1, 2]
- Static or Dynamic
- Software, Hardware or Hybrid
- Hardware DIFT: **off-core**, off-loading core, in-core



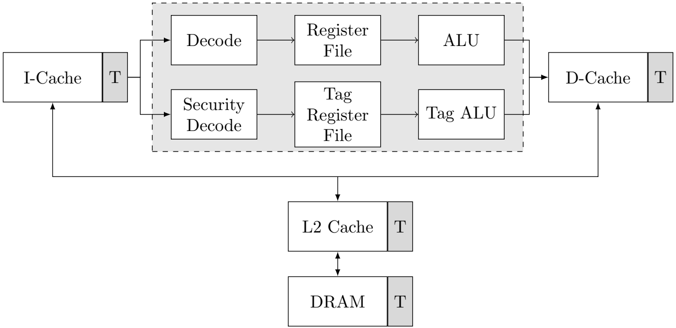
Software threats: Information Flow Tracking

- Security mechanism
- Protection against software attacks (e.g.: *buffer overflow*, *format string*, *SQL injections*, . . .) [1, 2]
- Static or Dynamic
- Software, Hardware or Hybrid
- Hardware DIFT: off-core, **off-loading core**, in-core



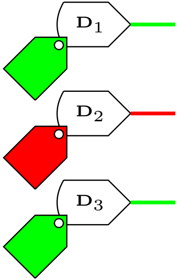
Software threats: Information Flow Tracking

- Security mechanism
- Protection against software attacks (e.g.: *buffer overflow*, *format string*, *SQL injections*, ...) [1, 2]
- Static or Dynamic
- Software, Hardware or Hybrid
- Hardware DIFT: off-core, off-loading core, **in-core**



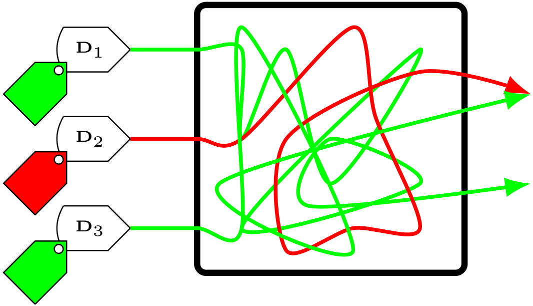
Three steps

- Tag initialisation
- Tag propagation
- Tag check



Three steps

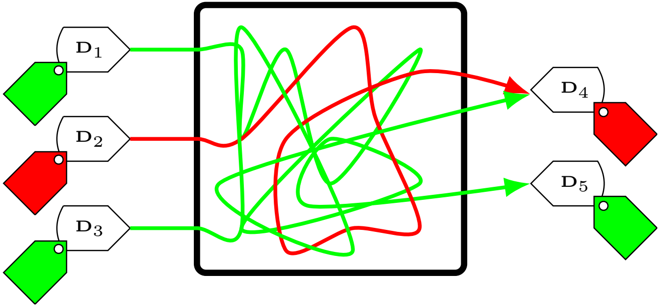
- Tag initialisation
- Tag propagation
- Tag check



Dynamic Information Flow Tracking

Three steps

- Tag initialisation
- Tag propagation
- Tag check

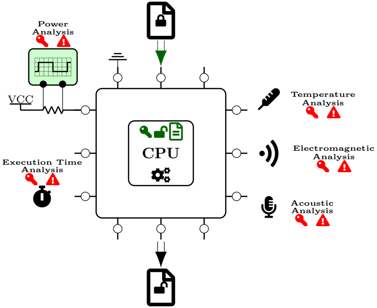


Hardware threats: Physical Attacks

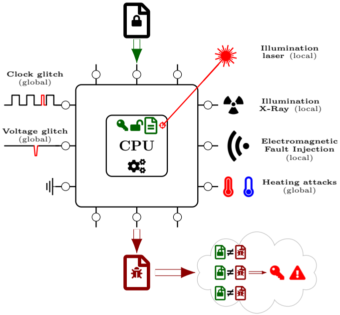
- Reverse Engineering: process of information retrieval from a product by analysing and understanding the design, functionality, and operation of existing hardware
- Side-Channel Attacks: exploit information leakages on the circuit behaviour
- Fault Injection Attacks: involve deliberately introducing one or more fault(s) into the system to observe its behaviour and identify potential vulnerabilities.

Hardware threats: Physical Attacks

- Reverse Engineering: process of information retrieval from a product by analysing and understanding the design, functionality, and operation of existing hardware
- Side-Channel Attacks: exploit information leakages on the circuit behaviour
- Fault Injection Attacks: involve deliberately introducing one or more fault(s) into the system to observe its behaviour and identify potential vulnerabilities.



- Reverse Engineering: process of information retrieval from a product by analysing and understanding the design, functionality, and operation of existing hardware
- Side-Channel Attacks: exploit information leakages on the circuit behaviour
- Fault Injection Attacks: involve deliberately introducing one or more fault(s) into the system to observe its behaviour and identify potential vulnerabilities.



Issue

How can we maintain maximum protection against software attacks in the presence of physical attacks?

Objectives of this PhD Thesis

Contributions

- Provide a robust security mechanism against software and hardware threats.
- Take into account Fault Injection Attacks
- Propose lightweight countermeasures against FIA
- Take into account constraints, such as area and performance overhead

Outline

D-RI5CY - origins

- Design [3] made by researchers at Columbia University (USA) with Politecnico di Torino (Italy)
- Based on the 32-bit RISC-V processor: RI5CY (Pulp Platform)
- Open source[1]
- 1-bit tag datapath
- Flexible security policy that can be modified at runtime

[1] https://github.com/sld-columbia/riscv-dift

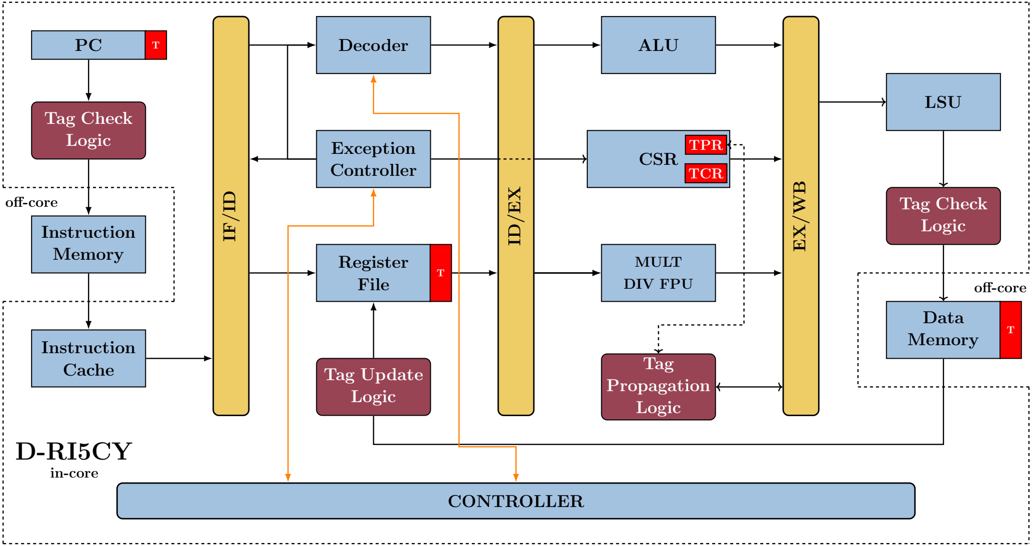


Figure 1: Architecture of the D-RI5CY.

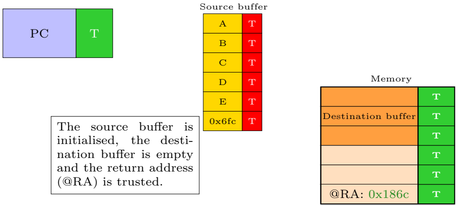
Threat model

We consider an attacker able to:

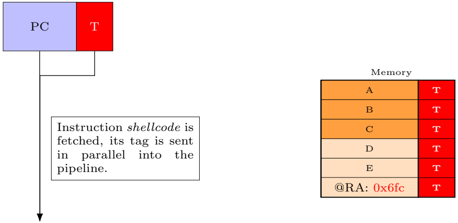
- perform a physical attack to defeat the DIFT mechanism and realise a software attack,
- inject faults in DIFT-related registers:
  - bit set,
  - bit reset,
  - bit-flip.

Case 1: Buffer overflow

- The attacker exploits a buffer overflow to access the return address register (*RA*).



(a) Malicious buffer and *RA* trusted



(b) Overflow and overwriting of *RA* and its tag

- As the data in the source buffer is manipulated by the user, it is marked as *untrusted*.
- Thanks to DIFT, the tags associated with the source buffer data overwrite the *RA* register tag.
- When the function returns, the corrupted register *RA* is loaded into *PC* using a jalr instruction.

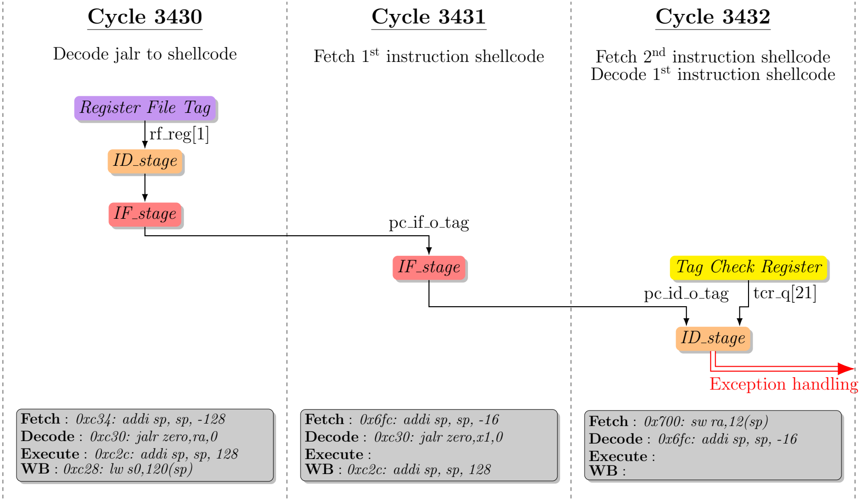


Figure 3: Temporal analysis of the tags propagation in *Buffer Overflow* attack

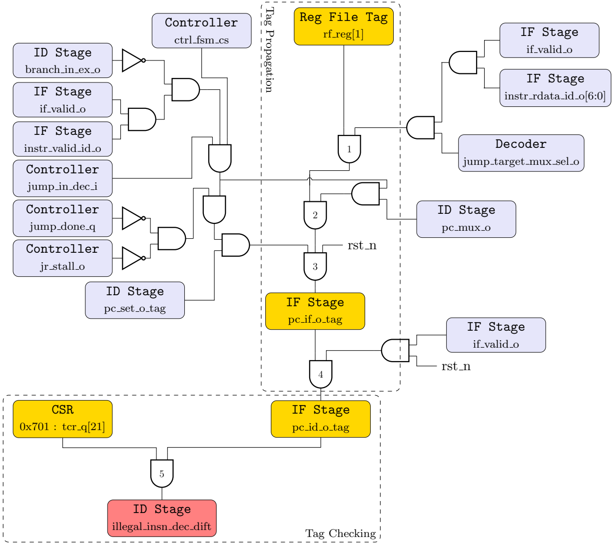


Figure 4: Logical analysis of the tags propagation in a *Buffer Overflow* attack

Case 2: WU-FTPd

- The vulnerability is the use of an unchecked user input as the format string parameter in functions that perform formatting, e.g. printf()
- An attacker can use the format tokens, to write into arbitrary locations of memory, e.g. the return address of the function.

```
void echo(){
    int a;
    register int i asm("x8");
    a = i;
    printf("%224u%n%35u%n%253u%n%n", 1, (int*) (a-4), 1, (int*) (a-3), 1, (int*) (a-2), (int*) (a-1));
}
```

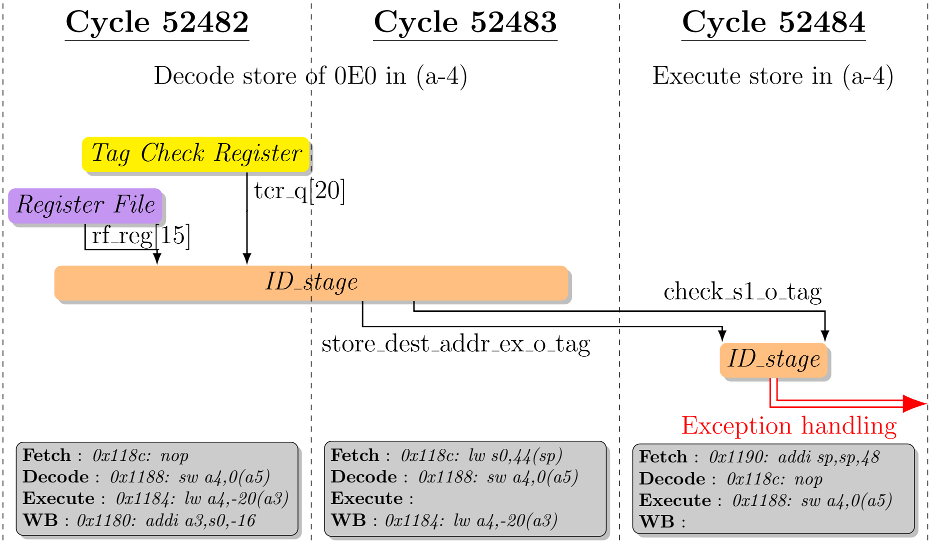


Figure 5: Temporal analysis of the tags propagation in a *format string* attack

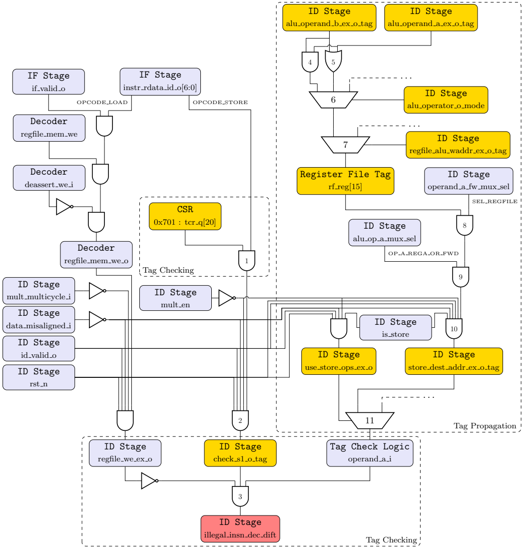


Figure 6: Logical analysis of the tags propagation in a *format string* attack

Experimental Setup - Simulation fault injections campaign

- Logical fault injection simulation is used for preliminary evaluations
  - faults are injected in the HDL code at cycle accurate and bit accurate level
  - a set of 55 DIFT-related registers are targeted
  - a reference simulation is done without fault
  - results are classed in four groups
    - crash: reference cycle count exceeded,
    - silent: current faulted simulation is the same as the reference simulation
    - delay: illegal instruction is delayed
    - success: DIFT has been bypassed
- Simulations with QuestaSim 10.6e.

Main results : 3 cases

Table 1: End of simulation status

	Crash	NSTR	Delay	Success	Total
Buffer overflow	0	1380	20	22 (1.55%)	1422
WU-FTPd	0	1767	77	52 (2.74%)	1896
Compare/Compute	0	917	12	19 (2.00%)	948

Buffer overflow

Table 2: Buffer overflow : Register sensitivity as determined by fault model and simulation time

	Cycle 3428			Cycle 3429			Cycle 3430			Cycle 3431			Cycle 3432		
	set0	set1	bitflip	set0	set1	bitflip	set0	set1	bitflip	set0	set1	bitflip	set0	set1	bitflip
pc_if_o_tag										✓		✓			
rf_reg[1]							✓		✓						
tcr_q	✓			✓			✓			✓			✓		
tcr_q[21]			✓			✓			✓			✓			✓
tpr_q	✓	✓		✓	✓										
tpr_q[12]			✓			✓									
tpr_q[15]			✓			✓									

Discussion

- 4212 simulations have been performed,
- 93 successes (2.21%),
- We have shown that the D-RI5CY DIFT is vulnerable to FIA
- Propagation of faults is facilitated by paths fully made of *AND* gates

Outline

Simple Parity

Hamming Code

SECDED

Threat model

Outline

Experimental setup

Outline

Conclusion

Perspectives

Publications

Publications

International peer-reviewed conferences

1. **William Pensec**, Vianney Lapôtre, and Guy Gogniat. 2023. Another Break in the Wall: Harnessing Fault Injection Attacks to Penetrate Software Fortresses. In Proceedings of the First International Workshop on Security and Privacy of Sensing Systems (SensorsS&P), 2023. [4]
2. **William Pensec**, Francesco Regazzoni, Vianney Lapôtre, and Guy Gogniat. Defending the Citadel: Fault Injection Attacks Against Dynamic Information Flow Tracking and Related Countermeasures. 2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI), 2024, pp. 180-185. [5]
3. **William Pensec**, Vianney Lapôtre, and Guy Gogniat. Scripting the Unpredictable: Automate Fault Injection in RTL Simulation for Vulnerability Assessment. 2024 27th Euromicro Conference on Digital System Design (DSD), 2024. [6]

Enhanced Processor Defence Against Physical and Software Threats by Securing DIFT Against Fault Injection Attacks

PhD Dissertation Defense

William PENSEC

Thank you for your attention.

References

References

[1] Christopher Brant et al. "Challenges and Opportunities for Practical and Effective Dynamic Information Flow Tracking". In: *ACM Computing Surveys* 55.1 (Nov. 2021). ISSN: 0360-0300. DOI: 10.1145/3483790.

[2] Wei Hu, Armaiti Ardeshiricham, and Ryan Kastner. "Hardware Information Flow Tracking". In: *ACM Computing Surveys* (2021). DOI: 10.1145/3447867.

[3] Christian Palmiero et al. "Design and Implementation of a Dynamic Information Flow Tracking Architecture to Secure a RISC-V Core for IoT Applications". In: *High Performance Extreme Computing*. 2018. DOI: 10.1109/HPEC.2018.8547578.

[4] William Pensec, Vianney Lapôtre, and Guy Gogniat. "Another Break in the Wall: Harnessing Fault Injection Attacks to Penetrate Software Fortresses". In: *Proceedings of the First International Workshop on Security and Privacy of Sensing Systems*. SensorsS&P. Istanbul, Turkiye: Association for Computing Machinery, 2023, pp. 8–14. DOI: 10.1145/3628356.3630116.

[5] William PENSEC et al. "Defending the Citadel: Fault Injection Attacks Against Dynamic Information Flow Tracking and Related Countermeasures". In: *2024 IEEE Computer Society Annual Symposium on VLSI (ISVLSI)*. Knoxville, United States, July 2024, pp. 180–185. DOI: 10.1109/ISVLSI61997.2024.00042.

[6] William Pensec, Vianney Lapôtre, and Guy Gogniat. "Scripting the Unpredictable: Automate Fault Injection in RTL Simulation for Vulnerability Assessment". In: *2024 27th Euromicro Conference on Digital System Design (DSD)*. Paris, France, Aug. 2024, pp. 369–376. DOI: 10.1109/DSD64264.2024.00056.

[7] Transforma Insights; Exploding Topics. *Number of Internet of Things (IoT) connections worldwide from 2022 to 2023, with forecasts from 2024 to 2033*. Online. Accessed 13 August 2024. 2024. URL: https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/.