When in-core DIFT faces fault injection attacks

Abstract

Both software and physical attacks are serious threats for Internet of Things (IoT) devices. Low-cost and low-power processors are usually the key component of these systems. They manipulate sensitive data leading to strict security needs. In this paper, we study the impact of Fault Injection Attacks (FIA) on protected RISC-V processor integrating a Dynamic Information Flow Tracking (DIFT) mechanism against software threats.

1 Introduction

Dynamic Information Flow Tracking (DIFT) techniques can detect various software attacks, for example, buffer overflow, SQL injections, or malware, by attaching and propagating tags to information containers at runtime [1]. A tag check security policy allows rising an alert when a malicious behaviour is detected. Several DIFT implementations have been studied in the literature: hardware, software, and hybrid DIFT [2]. Information containers will differ on which type of DIFT is used; these range from files to registers. In this paper, we consider hardware DIFT operating at a hardware level.

Hardware DIFT solutions can be grouped into two main categories: off-core and in-core. Off-core DIFT [3] relies on a dedicated co-processor to perform tag-related operations. This approach does not require internal processor modification and reduces the computation load on the main processor. However, the communication and synchronization between the main processor and the co-processor need to be carefully managed. In-core DIFT leads to internal modification of the processor. Tag-related operations are spread over the pipeline stages and are computed in parallel with the data treatments. Compare to the off-core approach, it does not require specific communication and synchronization management. However, significant invasive changes to the processor are required. In this paper, we consider the D-RI5CY processor implementing the in-core DIFT proposed in [4]. We analyse the impact of Fault Injection Attacks (FIA) on the efficiency of the D-RI5CY DIFT mechanism.

The rest of the paper is structured as follows. Section 2 presents the main motivations of this work. Then, Section 3 studies the impact of FIA on the D-RI5CY DIFT mechanism. Finally, Section 4 concludes the work and draws some perspectives.

2 Motivation

FIA can be performed by disturbing the power supply, or the clock, using EM pulse or laser shots [5]. The impact of injection varies depending on the type of FIA. Laser injections are the most precise in terms of spacial and temporal precision while the power supply or clock will affect the whole circuit limiting the spatial precision.

Figure 1 presents an overview of the D-RI5CY processor. DIFT-related components are highlighted in red. These components allow to store, propagate and check tags during the execution of a sensitive application. The security policy is configured through two CSR named TPR and TCR.

In this work, we propose to combine software and physical attacks to defeat the DIFT mechanism implemented in the D-RI5CY processor. We consider an attacker able to inject faults in the registers related to the DIFT-related components since several physical faults lead to a setup/hold time violation in flip-flops. We consider 3 types of injections: set to 0, set to 1, or a bit-flip at a random position of the targeted register. In the next section, we analyse the behaviour of the DIFT mechanism in the presence of such faults. The results of this analysis will help to build a robust DIFT mechanism in future works taking into account both software and physical attacks.

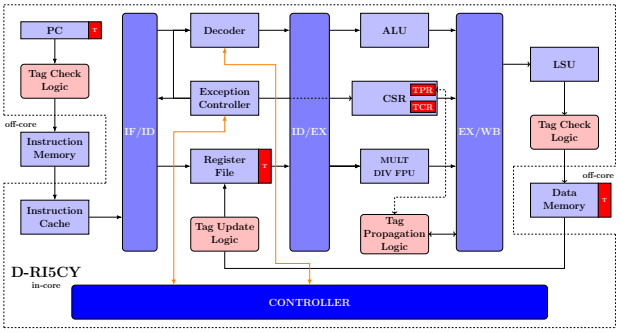

Figure 1: *Overview of the D-RI5CY processor*

3 Vulnerability assessment

To analyse the potential vulnerabilities, we have studied the case of the exploitation of a buffer overflow leading to a Return-oriented programming (ROP) attack[1] and the execution of a shellcode.

The attacker exploits a buffer overflow to reach the return address (ra) register. Due to the DIFT

[1] `https://github.com/sld-columbia/riscv-dift/blob/master/pulpino_apps_dift/wilander_testbed/`

mechanism, the tag associated with the buffer data overwrites the *ra* register tag. Since the buffer data is manipulated by the user, it is tagged as *not trusted* (tag value = 1). When returning from the called function, the corrupted *ra* register is loaded into *PC* (Program Counter) via a *jalr* instruction. The execution flow is hijacked and the first shellcode instruction is fetched from the address (*0x6fc*). These steps are presented in *Cycle 1* and *Cycle 2* of Figure 2. This figure also shows the *ra* register tag propagation. At Cycle 1, this tag is extracted from the *register file tag*. At Cycle 2, it is propagated in a register called *pc_if_o_tag*. At Cycle 3, the tag is propagated in a register called *pc_id_o_tag* and the first shellcode instruction is decoded. Since the *ra* has been tagged as not trusted, an exception is raised during the tag check process. It is worth noting that the D-RI5CY processor relies on the illegal instruction mechanism to stop the attack.

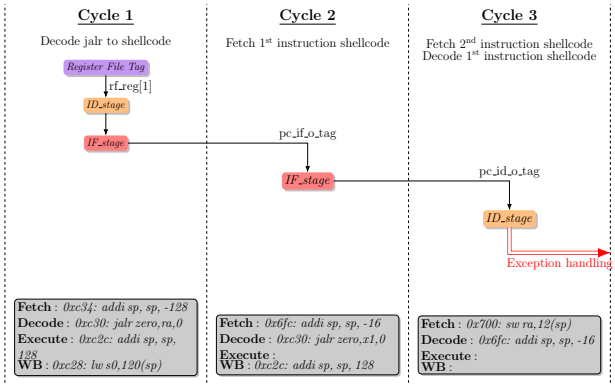

Figure 2: *Tag propagation in a buffer overflow attack*

Figure 2 shows that the DIFT mechanism can be impacted for 3 cycles before the exception is raised. To further study the sensitivity of this mechanism against FIA, Figure 3 details the logic relations between processor registers (yellow boxes) and control signals (blue boxes) driving the illegal instruction exception signal (red box). An attacker performing fault injections would like to drive the exception signal to '0' to defeat the D-RI5CY DIFT solution. Figure 3 shows that a single fault could lead to a successful injection since all paths are built with *AND* gates. For example, if we set register *rf_reg[1]* to 0, the tag will be propagated from *gate 1* up to *gate 4*. Finally, in *gate 5* there will be a comparison between 1 (from the tcr_q value) and 0 (from pc_id_o_tag) so it will result in a 0 value from this gate. Table 1 shows the locations and fault types leading to a successful injection for the considered ROP attack.

4 Conclusion and perspectives

This paper studies the impact of FIA on the D-RI5CY DIFT solution. It shows that a single fault could defeat the DIFT security policy. We plan to extend this work by studying the impact of FIA on the entire

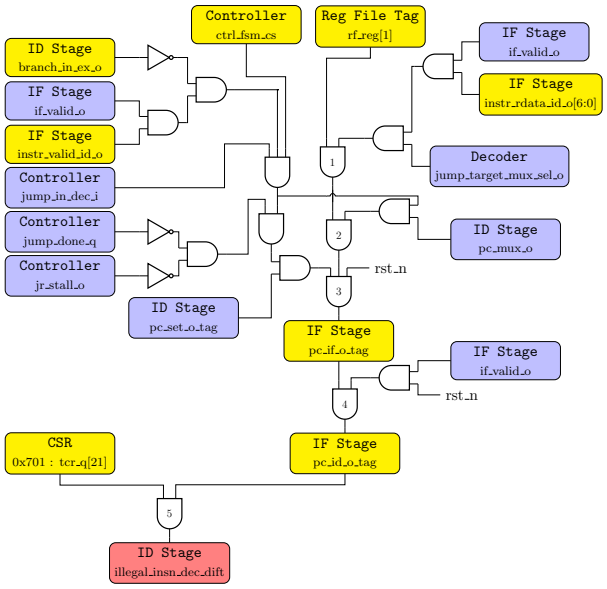

Figure 3: *Logic representation of tag propagation in a buffer overflow attack*

	set at 0	set at 1	bitflip
pc_if_o_tag	✓		✓
rf_reg[1]	✓		✓
tcr_q	✓		✓
tpr_q	✓	✓	

Table 1: *Location and fault types for successful FIA*

D-RI5CY DIFT solution. We will perform intensive fault injection emulation and actual injection targeting FPGA implementation. The obtained results will be analysed in order to build a DIFT solution robust against FIAs.

References

[1] Wei Hu et al. "Hardware Information Flow Tracking". In: *ACM Computing Surveys* (2021). DOI: 10.1145/3447867.

[2] Kejun Chen et al. "Dynamic Information Flow Tracking: Taxonomy, Challenges, and Opportunities". In: *Micromachines* (2021). DOI: 10.3390/mi12080898.

[3] Hari Kannan et al. "Decoupling Dynamic Information Flow Tracking with a dedicated coprocessor". In: *International Conference on Dependable Systems & Networks*. IEEE, 2009. DOI: 10.1109/DSN.2009.5270347.

[4] 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.

[5] Duško Karaklajić et al. "Hardware Designer's Guide to Fault Attacks". In: *IEEE Transactions on Very Large Scale Integration Systems* (2013). DOI: 10.1109/TVLSI.2012.2231707.