Side-Channel Attacks on RISC-V Processors: Current Progress, Challenges, and Opportunities

Mahya Morid Ahmadi1, Faiq Khalid1, Muhammad Shafique2

1Technische Universität Wien (TU Wien), Vienna, Austria
2Division of Engineering, New York University Abu Dhabi (NYUAD), Abu Dhabi, United Arab Emirates

Email: {mahya.ahmadi,faiq.khalid}@tuwien.ac.at, muhammad.shafique@nyu.edu

Abstract—Side-channel attacks on microprocessors, like the RISC-V, exhibit security vulnerabilities that lead to several design challenges. Hence, it is imperative to study and analyze these security vulnerabilities comprehensively. In this paper, we present a brief yet comprehensive study of the security vulnerabilities in modern microprocessors with respect to side-channel attacks and their respective mitigation techniques. The focus of this paper is to analyze the hardware-exploitable side-channel attack using power consumption and software-exploitable side-channel attacks to manipulate cache. Towards this, we perform an in-depth analysis of the applicability and practical implications of cache attacks on RISC-V microprocessors and their associated challenges. Finally, based on the comparative study and our analysis, we highlight some key research directions to develop robust RISC-V microprocessors that are resilient to side-channel attacks.

Keywords—RISC-V; Side-channel; Secure ISA; microprocessors; cache; hardware security.

I. INTRODUCTION

The exponential increase in using advanced microprocessors for critical applications (e.g., surveillance systems) makes these systems vulnerable to several security threats, e.g., remote micro-architectural [1] and side-channel attacks [2]. These attacks may lead to system failure, information leakage, and denial-of-service. Therefore, it is imperative to study security as a fundamental parameter along with performance constraints in the early design stages of these microprocessors, especially the emerging microprocessors like in fifth generation of Reduced Instruction Set Computer (RISC-V). To address this critical issue in microprocessors, researchers have developed several defenses against these threats. However, broadly, research in the security of microprocessors has been mostly divided into independent directions of hardware and software threats [3][4]. In hardware security, researchers are focused on chip modification and physical intrusions [5], and at the software level, they are studying software stack attacks like software malware [6]. Therefore, these are not applicable to the advanced attacks, such as the software-exploitable hardware attacks. These attacks combine the software and hardware bugs to exploit the microprocessor at run-time. For example, software-exploitable timing side-channels [2][7] has been exposed only a couple of years ago. As shown in Figure 1, the execution of software leaves side-channel traces at different levels of the system. These traces can be exploited by software-exploitable side-channel attacks to remotely leak confidential information from the trusted hardware with high bandwidth microarchitectural channels. Typically, these attacks target shared resources like last-level cache, which are inevitable in the high performance emerging microprocessors [8]. Recent studies show that these vulnerabilities are not limited to current microprocessors, but the next generation of microprocessors are also vulnerable to these attacks. Hence, it is imperative to study these vulnerabilities in emerging microprocessors, like RISC-V, to make them robust against these powerful attacks.

Therefore, the main focus of this paper is to highlight the security vulnerabilities in one of the most important emerging microprocessors, i.e., RISC-V. The reason behind this is that RISC-V is predicted to be a widely-adopted architecture in the coming days, as its rapid proliferation has already been witnessed in both industrial and academic Research and Development (R&D) projects and product lines [9]. Moreover, RISC-V [10], as an open-source, with extensible Instruction Set Architecture (ISA), is rapidly becoming the mainstream architecture for emerging embedded systems. The flexibility of this ISA made it popular in lightweight embedded systems of edge devices as well as server-side complex multi-core systems. Open hardware designs are essential for security, low-power applications, and fast R&D cycles. Since the RISC-V microprocessors are in the early stage of R&D, therefore, it is the right time to investigate security solutions there, with an eye to the past mistakes to avoid them in the next-generation of processors. To protect emerging embedded systems against security vulnerabilities, first, the applicability of state-of-the-art software-exploitable attacks on RISC-V processors must be studied. Then, based on the observations of these studies, a new attack surface can be found that can exploit the unique features of RISC-V ISA (e.g., memory...
Towards the above-mentioned research question, in this paper, we made the following key contributions:

1) We also provide a brief yet comprehensive overview and categorization (Section II-A) of the different side-channel attacks for microprocessors and their respective defenses (Section II-B).

2) We first study the side-channel data leakage attack on RISC-V processors running confidential applications like RSA encryption (Sections III-A and III-B).

3) Based on the study, we demonstrated the architectural vulnerabilities in a high-performance RISC-V microprocessor by successfully implementing a cross-core timing side-channel attack on the RISC-V microprocessors (Section IV). Our results show that RISC-V is vulnerable to timing side-channel attacks; hence, a lightweight yet powerful defense mechanism is required.

4) Towards the end, we briefly discuss the potential defenses for the cross-layer side-channel attacks on the RISC-V microprocessor (Section V) and highlighted research challenges and opportunities for cross-layer side-channel attacks on a RISC-V microprocessor (Section VI).

II. BACKGROUND

In this section, we provide the necessary background information for side-channel attacks on RISC-V microprocessors. First, we present the taxonomy for state-of-the-art side-channel attacks with the focus on power side-channel attacks and timing attacks. Then, we discuss the vulnerabilities in the encryption algorithms, and towards the end, we present a brief overview of the RISC-V ISA.

A. Side-Channel Attacks

There is a large body of the work on hardware attack to microprocessors [2]. However, in this paper, we focus on attacks in hardware that are exploiting an unwanted channel to leak confidential data, which are known as side-channels. The principle of a security manipulation starts with forcing the victim to transfer the critical data to the target location, then store it or consume it, which gives the opportunity to attacker to steal the data. If the leaking channel of data is built using physical parameters of the system, which is dependant on the control flow, it is called a side-channel. There are several side-channels reported in hardware, and among the most exploited ones, we can name power, timing, and temperature.

In Figure 2, we are presenting a known taxonomy of side-channel attack and their mitigation techniques in general-purpose microprocessors. Based on the attack model, attacks are divided into two categories: sourced from software and sourced from the hardware. Attacks from hardware exploit physical features, e.g., dynamic power, by implanting sensors in shared resources of the system. While the former category requires physical access to the microprocessor, attacks from software can be exploited remotely. Side-channel attacks from software leak the confidential data through microarchitectural events of shared resources, e.g., timing attack to cache [11]. As the efforts on the side-channel attack to RISC-V processors are focused on power and timing attacks, we are elaborating on these attacks in the following sections.

1) Power attacks: In cryptographic algorithms, the computation is based on a secret value $S$, which is not necessarily the key to encryption but it is derived from the key. The activity of encryption software causes data dependant signal transitions in the current surges of the hardware, which are visible in internal wires, power, and ground connection. By collecting the dynamic power traces, an adversary can find the secret value correlated to the input data pattern while considering measurement errors and environmental noise.

Power traces built on the direct current correspondence of the software operation, are analyzed as Simple Power Analysis (SPA). In this analysis, it is assumed that a change in each bit of $S$ is visible in the leakage power pattern. In attack to applications with complicated relation of input and application flow, Differential Power Analysis (DPA) can reveal the $S$ by statistical tests on the differentiation of many power traces. DPA is a very powerful technique that can easily eliminate the random noise sourced from environment or obfuscation mitigation techniques.

2) Timing attacks: Microarchitectural side-channels are typically timing-based channels, as shown in Figure 3. Due to sharing functional units among different programs, an attacker can, in general, observe the timing of the operation of the individual functional unit and its output. Since the designs of the functional units are known to the attacker, these timing leakages reveal whether the fast or slow execution path was taken [12]. In the taxonomy presented in Figure 2, Software-to-hardware side-channel attacks are categorized based on the shared unit. In [13], the attacker manipulates...
using several variants, the shared address assumption limits the threat model. Also, there is a need for an atomic instruction for flushing a specific line which is not present in all architectures (e.g., RISC-V). As another example is FLUSH+FLUSH [18], where malicious user records the loop time for *clflush* instruction in the attack-loop phase.

- **Time-Driven Attacks**: In this category of attacks, attacker attempts to measure the victim’s execution time. The execution time of applications is dependant on several parameters like execution path, data flow, and memory access time. Hence, these attacks require a large and detailed execution profile for key extraction. EVICT+RELOAD [19] is a Time-Driven cache attack, whereby evicting specific cache lines, the attacker prepares the system to reveal the victim’s control flow. By forcing the victim to run enough times, attacker gains the required number of traces to extract key-dependent memory accesses. This attack is based on the average execution time of a confidential program. Therefore, it needs a large number of sample data to recognize the key.

- **Trace-Driven Attacks**: In this category, the attacker collects the traces of encryption on a shared resource, e.g., cache, based on a known message attack. These traces can form a profile of hit and miss in the cache, which gives information about the key-dependent lookup addresses. This ability gives an adversary the opportunity to make inferences about the secret key. In [20], cache traces are determined by the power consumption in the cache for each hit and miss to break the AES encryption algorithm.

### B. Defense Mechanisms

In Figure 4, defense mechanisms are categorized into three subsets, based on the source of leakage channel they are protecting against. Microarchitectural mitigation techniques, e.g., isolation, are implemented both in software and hardware levels that each can address a specific range of attacks [29]. Also, mitigation techniques against power side-channels, are proposed both at circuit level, as resilient processors, and algorithmic level, as resilient application implementation. We will present works in these categories in Section IV.

### C. Data Dependant Implementation of Encryption Algorithms

In an attack on cryptographic algorithms, the encryption key is the target of the attacker. Two implementation models are mostly analyzed in the literature for the security of encryption application implementation, conditional control flow, and conditional lookup table, which are respectively used in Rivest–Shamir–Adleman (RSA) and Advanced Encryption Standard (AES) encryption systems.

1) **Conditional Control Flow**: RSA is a cryptosystem for exchanging the key between two parties. To calculate the private key and the public key, modular exponentiation is implemented using a square-and-multiply algorithm that computes \( r = b^e \mod m \) dependant to bits of \( e \) in a conditional loop. As shown in Figure 5, line 5 will be executed if the condition in line 4 is fulfilled and it causes bit-dependant physical footprints like dynamic power consumption and execution time. In [21], Musthag et al. have presented a comprehensive survey in attacks and countermeasures on RSA.

2) **Conditional Table Lookup**: In block cipher cryptosystems, the secret of the system is not directly used for encryption,
but the key enables an address in the pre-computed tables of data. When the index used to access a Look-Up Table (LUT), depends on the secret data, the access time and dynamic power consumption may vary due to the behavior of memory access. Such data leakages have been exploited in various block ciphers, e.g., AES in [22] that implements S-Boxes using lookup tables to reduce hardware overhead.

Figure 5. Data-dependent implementation of modular exponentiation which is used in RSA encryption algorithm.

D. RISC-V Architecture

RISC-V [10] with open-source ISA is received as an alternative in academia and also popular in the IoT industry. Since it is a flexible, upgradeable, and optimizable architecture, it is predicted to be adopted widely in emerging embedded systems. RISC-V ISA is not limited to a specific hardware implementation of a processor core. The modular architecture implementation with variants of address space sizes makes it suitable for lightweight edge devices to the high-performance servers. The base integer instruction set is flexible to adopt the required software stack and to add additional requirements. Moreover, the simplicity of the design enables the RISC-V to be an ideal base for the application-specific design. Accelerators and co-processors can share the compiler tool-chain, operating system binaries, and control processor implementations. It speeds up the process of agile hardware processor design by the support of an active and diverse community for open-source contributions. Figure 6 shows an overview of the RISC-V ecosystem and respective tools.

III. SIDE-CHANNEL ATTACKS ON RISC-V MICROPROCESSORS

In this section, we discuss state-of-the-side-channel attacks on RISC-V microprocessors.

A. Power Side-Channel Attacks

Mulder et al. in [23] discuss power side-channel vulnerabilities of RISC-V microprocessor. In this work, they distinguish between different types of data leakage by categorizing them into direct-value leaks, data-overwrites and circuit-level leaks. Since direct-value leakage can be protected using masking techniques, they show software masking is not sufficient for mitigating power side-channel. On the other hand, since side-channel analysis is usually postponed to post-fabrication stages of hardware production, the state-of-the-art is focused on studying the software implementation of cryptographic algorithm. Therefore, a promising research direction is to apply the mitigation technique in the architectural level to prevent exceeding complexity of the microprocessor-based systems. In this work, they exploit memory access patterns of an AES encryption algorithm through a data overwrite leakage. This leakage happens when the data and the mask are accessed repetitively through memory connections. The replaced data in intermediate registers causes a peak in consumed power, which is leaked by power side-channel attack.

B. Timing Side-Channel Attacks

Recent attacks on modern processors have shown that special features of emerging architectures can be a source of the attack in lower hardware levels, as well as in the previous generation. In [24], Gonzalez et al. replicated Spectre [25] attack on BOOM (Berkeley out-of-order-machine) core and exploits in-core data cache for leaking the confidential data. Since the cflush instruction is not implemented in the RISC-V ISA, the authors implement an atomic instruction to replace the targeted lines with dummy data and evict the victim’s shared page. More recently, Le et al. in [27], showed a side-channel...
attack to physically-implemented BOOM. In this attack, they target the conditional branch, which is trained to execute an instruction that assesses the confidential data.

IV. PRIME+PROBE ATTACK ON RISC-V

To identify the vulnerabilities of RISC-V against timing side-channel attacks, we implemented a state-of-the-art cache timing attack (PRIME+PROBE) to RSA encryption algorithm running on different RISC-V hardware platforms, i.e., an out-of-order speculative RISC-V core (BOOM, implemented as SoC on the FPGA) and a commercial in-order RISC-V CPU (HiFive Unleashed).

A. Design Challenges

Most of the timing side-channel attacks are applicable to traditional microprocessors' architecture. However, the implementation of these attacks on the RISC-V microprocessors exhibits the following design challenges:

1) The cache addressing is not determined by RISC-V ISA, and it differs with implementation. For instance, the last-level cache addressing for Rocket SoC is designed to be virtually-indexed, which is the one that helps to build a shared set between the victim and the attacker.

2) In this work, we replicate the attack on a high-performance commercial processor to find out the impact of branch prediction and out-of-order units on the timing of application execution. For instance, if the attacker accesses to one set in the cache, the prefetch unit can learn the access pattern and predict the execution in pause times, while shuffling the lines in this attack prevents this.

3) The RISC-V ISA does not support the clflush instruction that is exploited by cache timing attacks in Intel microprocessors (FLUSH+RELOAD). Therefore, we choose another attack that is independent of special instruction or shared addresses in the victim's software space. We observe that RISC-V microprocessors reveal secrets of the system with the same trend in bandwidth as commercial microprocessors.

B. Experimental analysis on hardware platforms

We performed the PRIME+PROBE attack on a Rocket SoC with BOOM RISC-V multi-core implemented on the Zedboard, Xilinx Zynq-7000 FPGA board, and we show the successful implementation of the cache timing attack (PRIME+PROBE) to the RSA encryption Algorithm on RISC-V ISA in Figure 7(a). For the comprehensive analysis, we also performed the PRIME+PROBE attack on HiFive Unleashed, i.e., a commercial RISC-V multi-core CPU (see Figure 7(b)). Since, in the PRIME+PROBE attack, the attacker makes consecutive accesses to targeted cache lines in each determined time slot. By measuring the access time, the attacker can decide whether the content of the requested address is present in the cache or it has been replaced by the victim’s access. By analyzing the results in Figures 7(a) and (b), we made the following key observations:

1) The difference between cache access time for key-bit 0 (see labels A0 and B0 in Figure 7) and key-bit 1 (see labels A1 and B1 in Figure 7) is distinguishable even when other processes are running on the shared platform. In our attack implementation, we have exploited this timing differences to extract the key.

2) The identified timing vulnerabilities exist in the both FPGA implementation and HiFive Unleashed board that can be exploited by the PRIME+PROBE attack. Hence, these vulnerabilities are independent of the hardware platform. This observation leads us to a conclusion that mitigation technique for FPGA implementations can also be deployed ASIC implementation of RISV-V microprocessors.

3) These timing vulnerabilities can be exploited to perform other software-exploitable attacks on the RISC-V hardware, for example, Spectre [25] and Meltdown [26].

V. DEFENSE MECHANISMS FOR SIDE-CHANNEL ATTACKS ON RISC-V MICROPROCESSORS

The usage of high-performance processors, e.g., commercial RISC-V CPUs [24], in Internet-of-Things (IoT) devices, have already been shown to be vulnerable to microarchitectural attacks like Spectre [25] and Meltdown [26]. Efforts in defense techniques are focused on protecting sensitive data in memory locations [30][31], and protecting the software by providing an isolated execution environment [32][33]. Yu et al in [34] discuss a compiler-based solution for securing data oblivious code generation for preventing the side-channel attack, which is not applicable at software level. To protect the RISC-V CPU against power side-channel attacks, Mulder et al. [23] have proposed a masking solution at the architecture level.
VI. Open Research Challenges and Road Ahead

Currently, the focus of research in the security of RISC-V is on the security of the software. While it is crucial to protect software integrity, hardware security is a significant threat that needs to be addressed accordingly. In the following, we mention two main challenges in this research direction.

1) New features in ISA and lack of stable toolchain: Earlier studies assume that ISA is independent of hardware implementation, but the data leakage analysis shows that ISA can be exploited as side-channels. The first step is to examine the applicability of current attack vectors in RISC-V processors and then study the new features of RISC-V ISA, which introduces new variables to SCAs. To this aim, a reliable and stable toolchain is required.

2) Need for generic defense mechanisms: RISC-V, as a new open-source ISA, has a high potential to be the platform for state-of-the-art defense mechanisms. However, our observations show that most of the proposed techniques utilize special features of processors, such as Intel’s x86 and ARM, which are not available in RISC-V processors. It will be worthy of proposing generic methodologies for addressing the gap of security studies between software and hardware by utilizing RISC-V capabilities. These defense mechanisms in the early stages of design can be adopted even for lightweight processors and are not limited to complicated high-performance multi-cores.

ACKNOWLEDGMENT

This work was partially supported by Doctoral College Resilient Embedded Systems which is run jointly by TU Wien’s Faculty of Informatics and FH-Technikum Wien.

REFERENCES


